Module 3

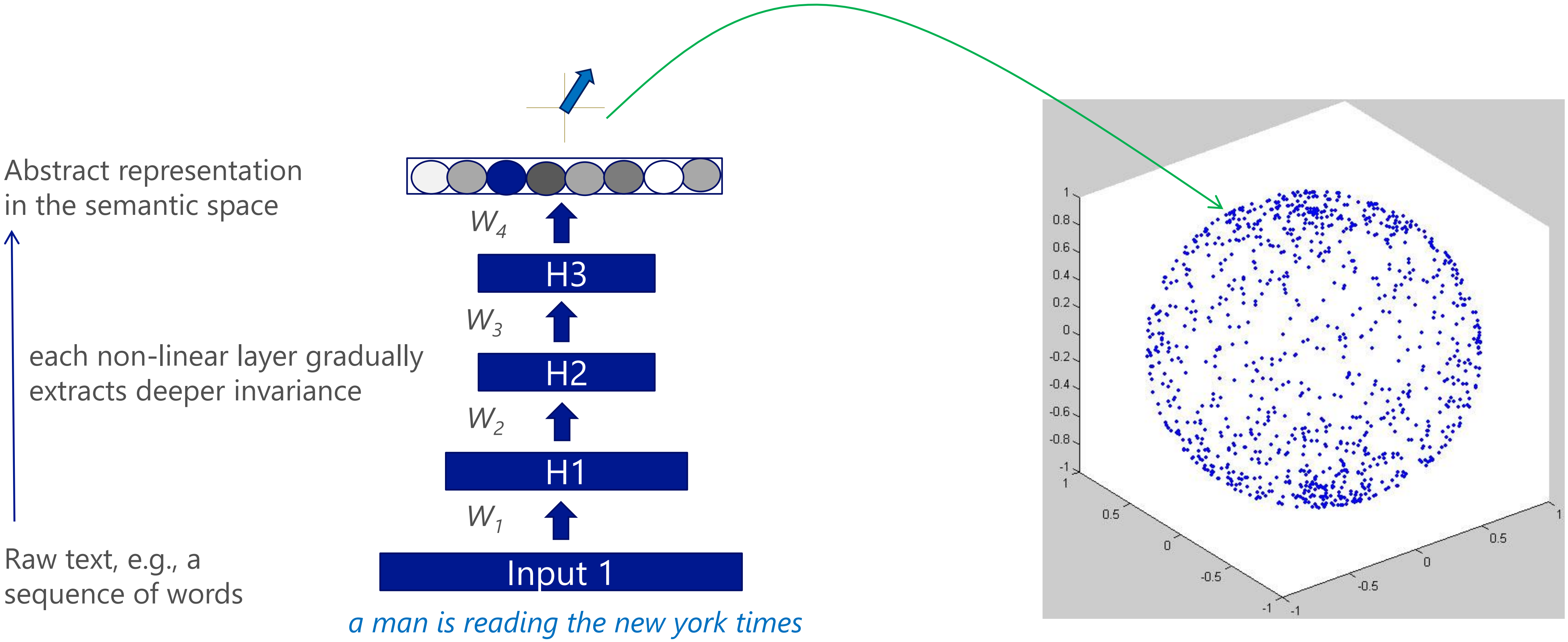
Deep Semantic Similarity Model and its Applications

Module 3 Overview

- Deep semantic similarity models (DSSM)
- DSSM for Information Retrieval
- DSSM for entity ranking

Learning continuous semantic representations for natural language

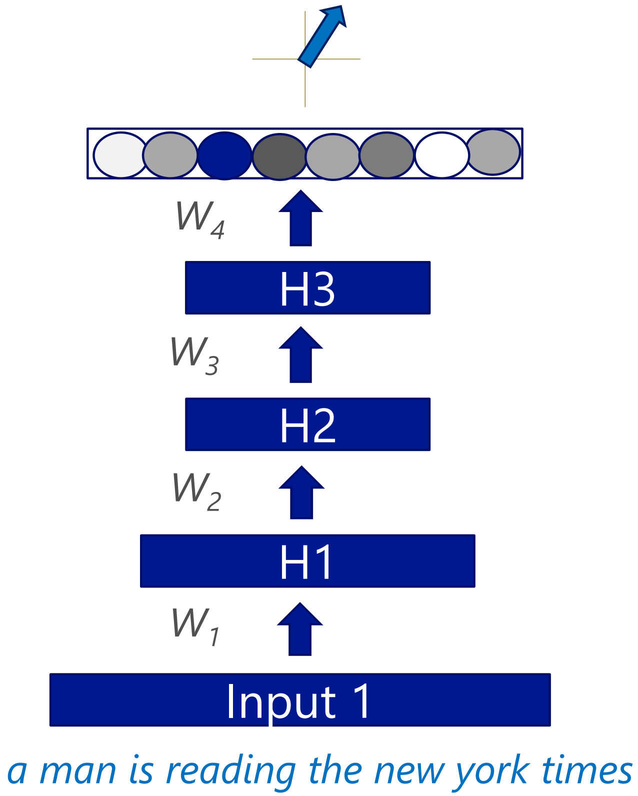
e.g., from a raw sentence to an abstract semantic vector (Sent2Vec)



Sent2Vec is crucial in many NLP tasks

Tasks	*Source*	*Target*
Web search	*search query*	*web documents*
Ad selection	*search query*	*ad keywords*
Contextual entity ranking	*mention (highlighted)*	*entities*
Online recommendation	*doc in reading*	*interesting things / other docs*
Machine translation	*phrases in language S*	*phrases in language T*
Knowledge-base construction	*entity*	*entity*
Question answering	*pattern | mention*	*relation | entity*
Personalized recommendation	*user*	*app, movie, etc.*
Image search	*query*	*image*
Image captioning	*image*	*text*
…		

The supervision problem:



However

- the semantic meaning of texts – to be learned – is latent
- no clear target for the model to learn
- How to do back-propagation?

Fortunately

- we usually know if two texts are "similar" or not.
- That's the signal for semantic representation learning.

Deep Structured Semantic Model

Deep Structured Semantic Model/Deep Semantic Similarity Model (**DSSM**)

project the whole sentence to a continuous semantic space – e.g., ***Sentence to Vector***.

The DSSM is built upon **characters** (rather than words) for scalability and generalizability

The DSSM is trained by optimizing an **similarity-driven** objective

Huang, He, Gao, Deng, Acero, Heck, "Learning deep structured semantic models for web search using clickthrough data," CIKM, October, 2013

Character-level coding (a.k.a. word hashing)

- E.g., character-trigram based *Word Hashing* of "cat"
  - -> #cat#
  - Tri-characters: #-c-a, c-a-t, a-t-#.

$$x\,(cat) = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \\ \vdots \\ \vdots \\ \vdots \\ 0 \end{bmatrix}$$

The index of word *cat* in the vocabulary

$$\Downarrow$$

$$f\,(cat) = \begin{bmatrix} \vdots \\ 1 \\ \vdots \\ 1 \\ 1 \\ \vdots \end{bmatrix}$$

Indices of #-*c*-*a*, *c*-*a*-*t*, *a*-*t*-# in the letter-tri-gram list, respectively.

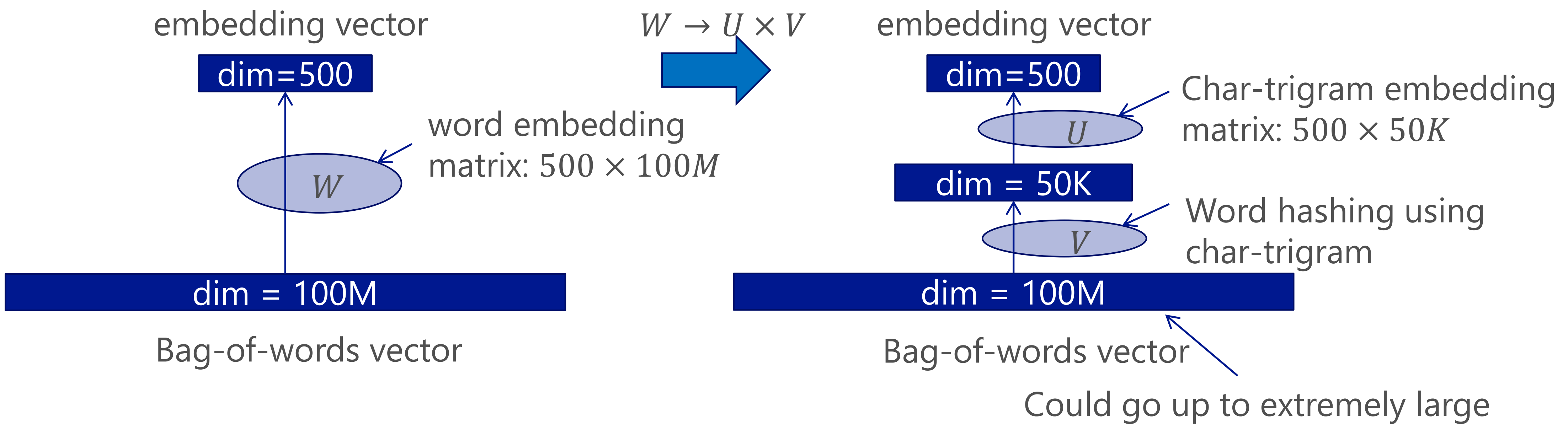
- Compact representation
  - |Voc| (500K) → |Char-trigram| (30K)
- Generalize to unseen words
- Robust to misspelling, inflection, etc.

What if different words have the same word hashing code (collision)?

Vocabulary size	Unique letter-tg observed in voc	Number of Collisions
40K	10306	2 (0.005%)
500K	30621	22 (0.004%)

DSSM: built at the character-level

Decompose *any* word into set of context-dependent characters



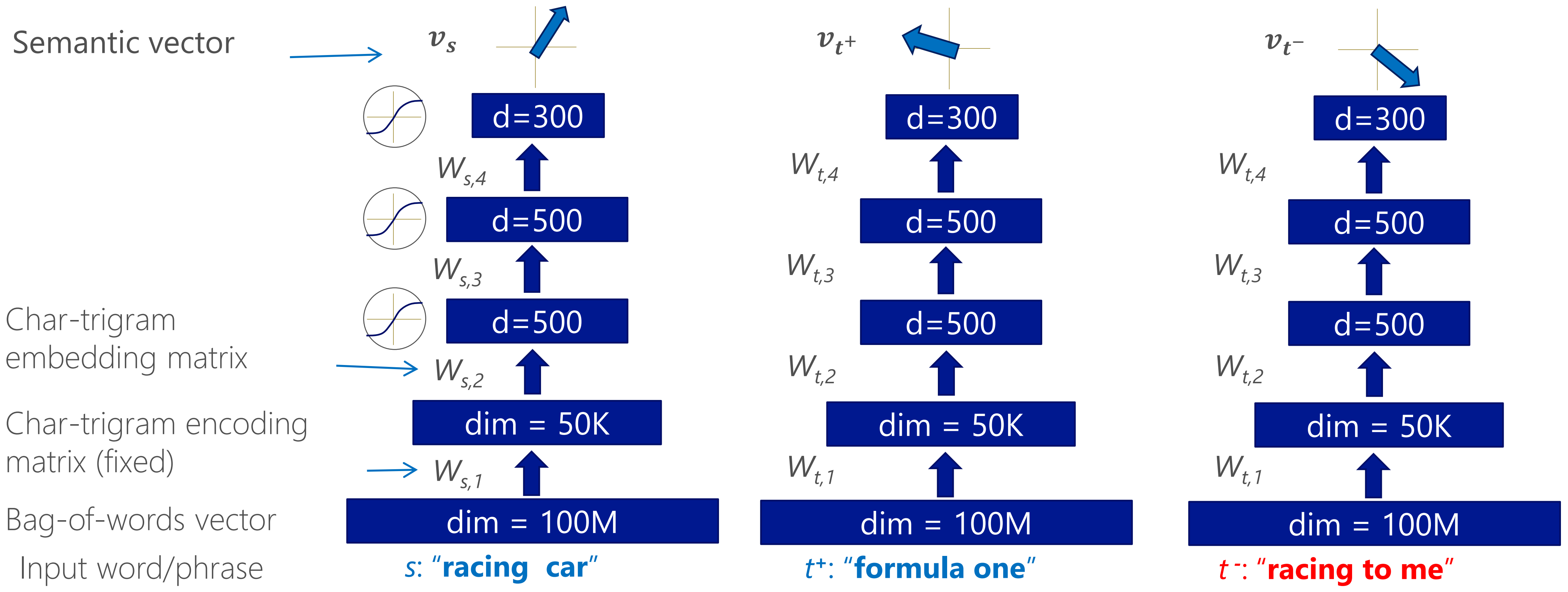
Preferable for large scale NL tasks

- Arbitrary size of vocabulary (*scalability*)
- Misspellings, word fragments, new words, etc. (*generalizability*)

DSSM: a similarity-driven Sent2Vec model

Initialization:

Neural networks are initialized with random weights



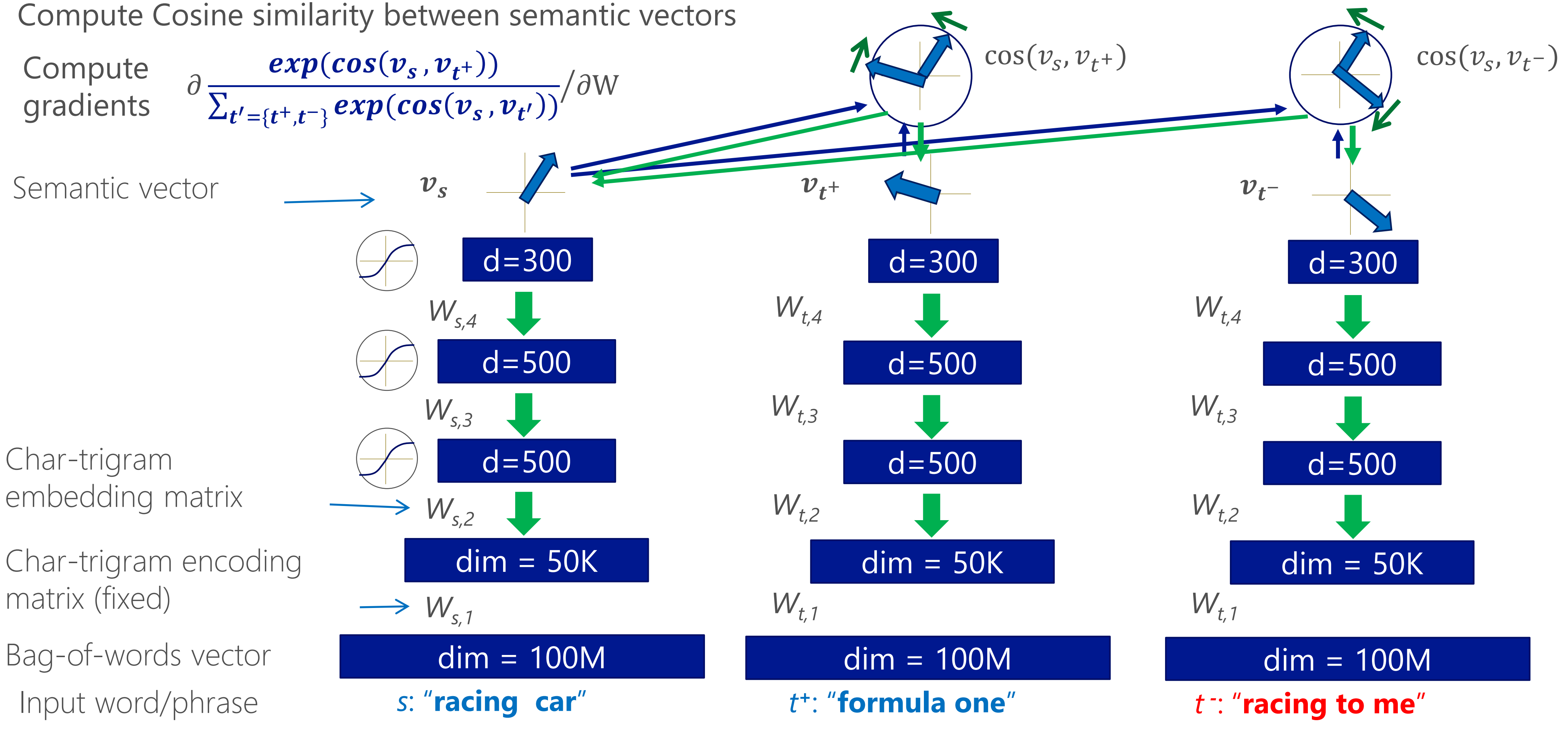
[Huang, He, Gao, Deng, Acero, Heck, "Learning DSSM for web search using clickthrough data," CIKM, 2013]

DSSM: a similarity-driven Sent2Vec model

Training:

Compute Cosine similarity between semantic vectors

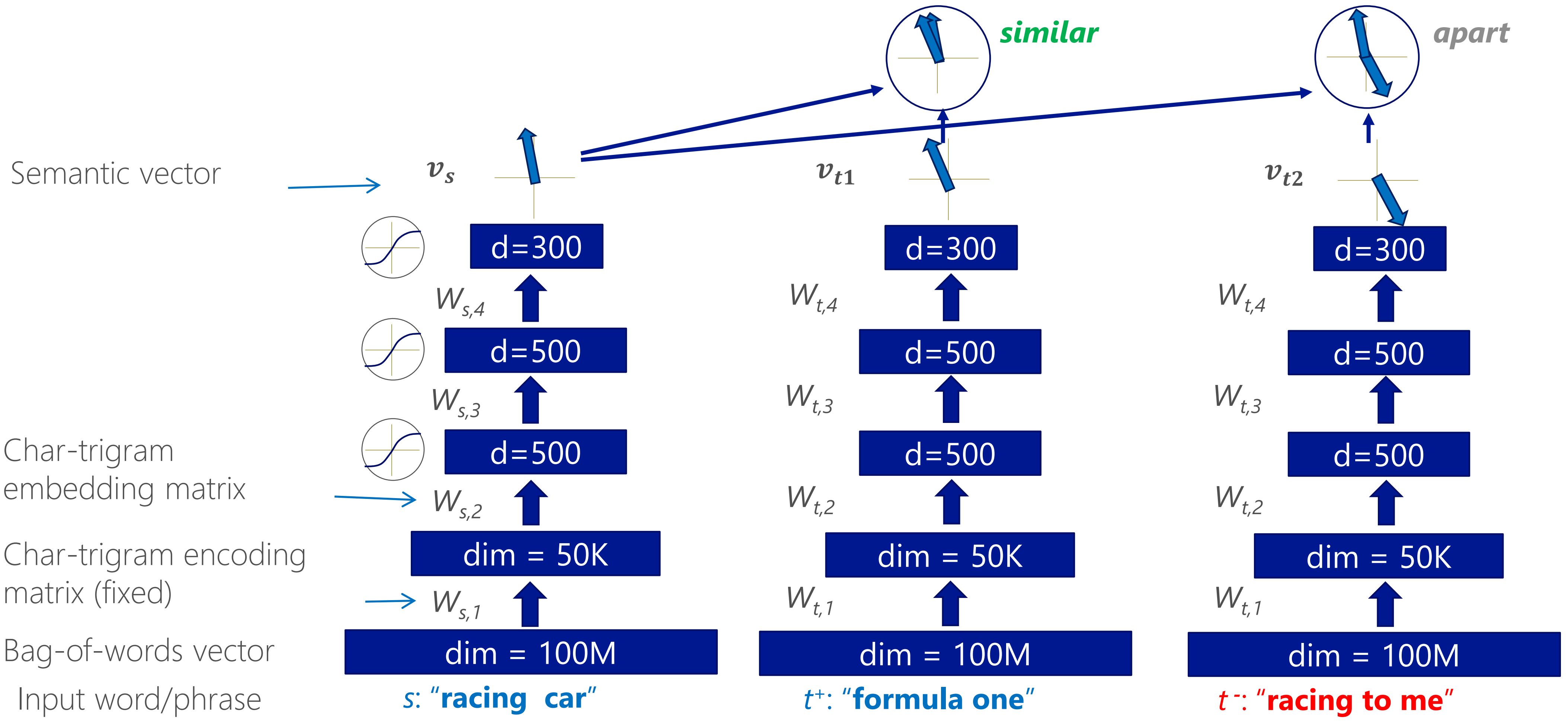
Compute gradients $\partial \dfrac{exp(cos(v_s, v_{t^+}))}{\sum_{t'=\{t^+,t^-\}} exp(cos(v_s, v_{t'}))} / \partial W$



[Huang, He, Gao, Deng, Acero, Heck, "Learning DSSM for web search using clickthrough data," CIKM, 2013]

DSSM: a similarity-driven Sent2Vec model

Runtime:



[Huang, He, Gao, Deng, Acero, Heck, "Learning DSSM for web search using clickthrough data," CIKM, 2013]

Training objectives

Objective: cosine similarity based loss

Using web search as an example:

- a query q and a list of docs $D = \{d^+, d_1^-, \dots d_K^-\}$
  - d^+ positive doc; $d_1^-, \dots d_K^-$ are negative docs to q (e.g., sampled from not clicked docs)
- Objective: the posterior probability of the clicked doc given the query

$$P_\theta(d^+|q) = \frac{\exp\left(\gamma\, cos(v_{\boldsymbol{\theta}}(q), v_{\boldsymbol{\theta}}(d^+))\right)}{\sum_{d \in \boldsymbol{D}} \exp\left(\gamma\, cos(v_{\boldsymbol{\theta}}(q), v_{\boldsymbol{\theta}}(d))\right)}$$

e.g., $v_{\boldsymbol{\theta}}(q) = \sigma(W_{s,4} \times \sigma(W_{s,3} \times \sigma(W_{s,2} \times ltg(q)))$

$v_{\boldsymbol{\theta}}(d) = \sigma(W_{t,4} \times \sigma(W_{t,3} \times \sigma(W_{t,2} \times ltg(d)))$

where $\theta = \{W_{s,2\sim4}, W_{t,2\sim4}\}$, $\sigma(\,)$ is a tanh function.

Using Convolutional Neural Net in DSSM

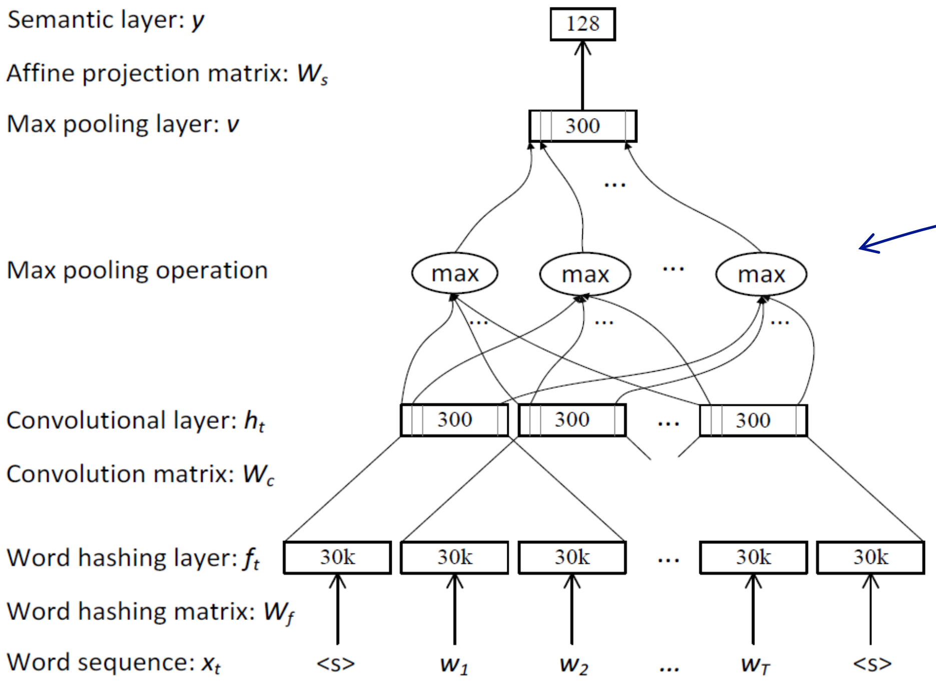


Figure 1: Illustration of the C-DSSM. A convolutional layer with the window size of three is illustrated.

Model local context at the convolutional layer

Model global context at the pooling layer

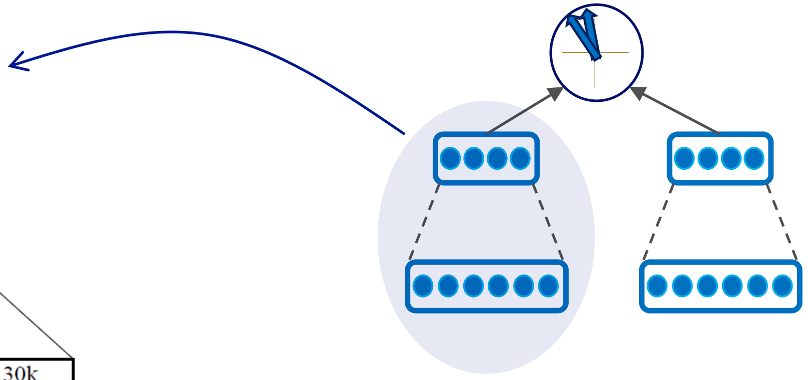


Figure credit [Shen, He, Gao, Deng, Mesnil, WWW2014]

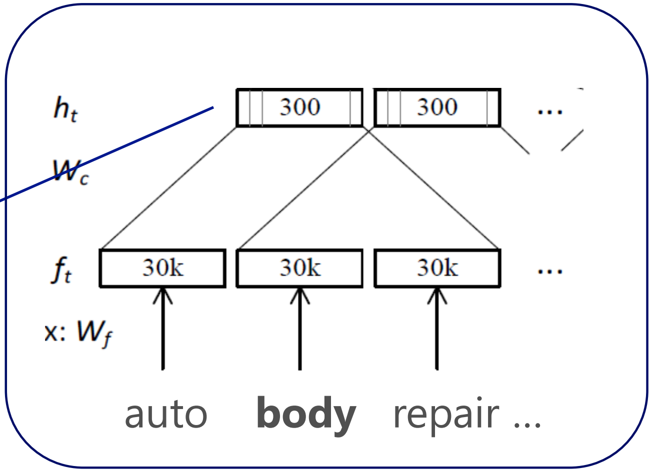
Strong performance on many NLP tasks

Information Retrieval: [Shen, He, Gao, Deng, Mesnil, WWW2014 & CIKM2014], Entity Ranking: [Gao, Pantel, Gamon, He, Deng, Shen, EMNLP2014], Question answering: [Yih, He, Meek, ACL2014; Yih, Chang, He, Gao, ACL2015], Recommendation [Elkahky, Song, He, WWW2015], Spoken language understanding [Chen, Hakkani-Tür, He, ICASSP2016]…

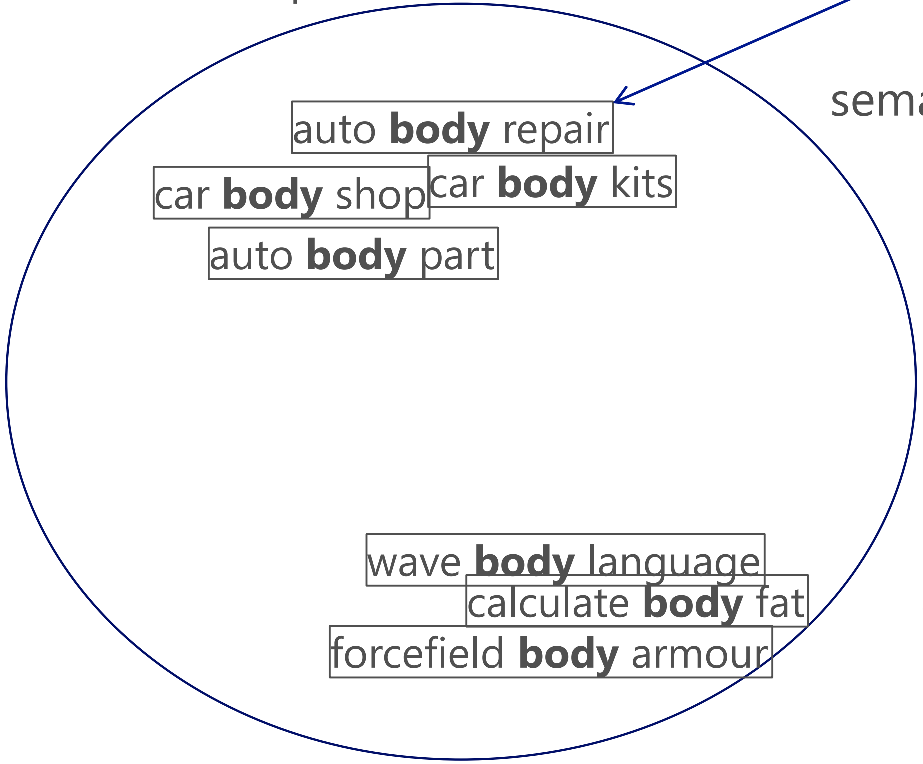
– What does the model learn at the convolutional layer?

Capture the local context dependent word sense

- Learn one embedding vector for each local context-dependent word



$$h_t = W_c \times [f_{t-1}, f_t, f_{t+1}]$$



semantic space

The similarity between different "***body***" within contexts

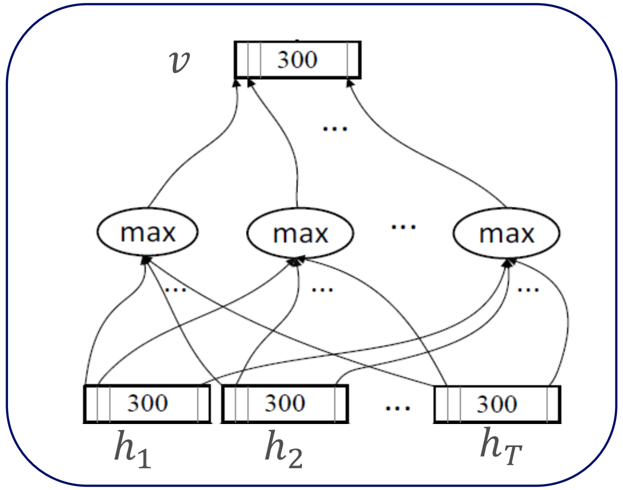
car **body** shop	**cosine similarity**
car **body** kits	0.698
auto **body** repair	0.578
auto **body** parts	0.555
wave **body** language	0.301
calculate **body** fat	0.220
forcefield **body** armour	0.165

high similarity

low similarity

CDSSM: What happens at the max-pooling layer?

- Aggregate *local topics* to form the *global intent*
- Identify salient words/phrase at the max-pooling layer



$$v(i) = \max_{t=1,\ldots,T} \{h_t(i)\}$$

where $i = 1, \ldots, 300$

Words that win the most active neurons at the **max-pooling layers:**

auto body repair cost calculator software

Usually, those are salient words containing clear intents/topics

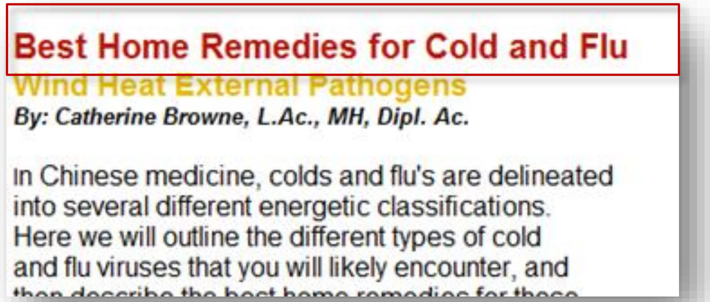
DSSM for Information Retrieval

- Training Dataset
  - Mine semantically-similar text pairs from Search Logs, e.g., 30 Million (Query, Document) Click Pairs

how to deal with stuffy nose?

stuffy nose treatment

cold home remedies

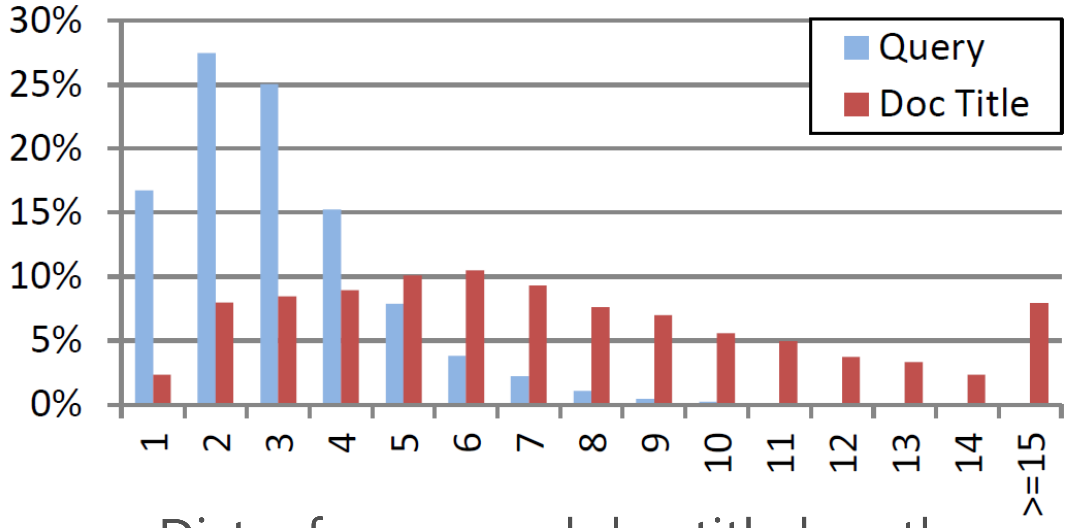


QUERY (Q)	Clicked Doc Title (T)
how to deal with stuffy nose	best home remedies for cold and flu
stuffy nose treatment	best home remedies for cold and flu
cold home remedies	best home remedies for cold and flu
… …	… …
skate at wholesale at pr	wholesale skates southeastern skate supply

[Gao, He, Nie, CIKM2010]

Experimental Setting

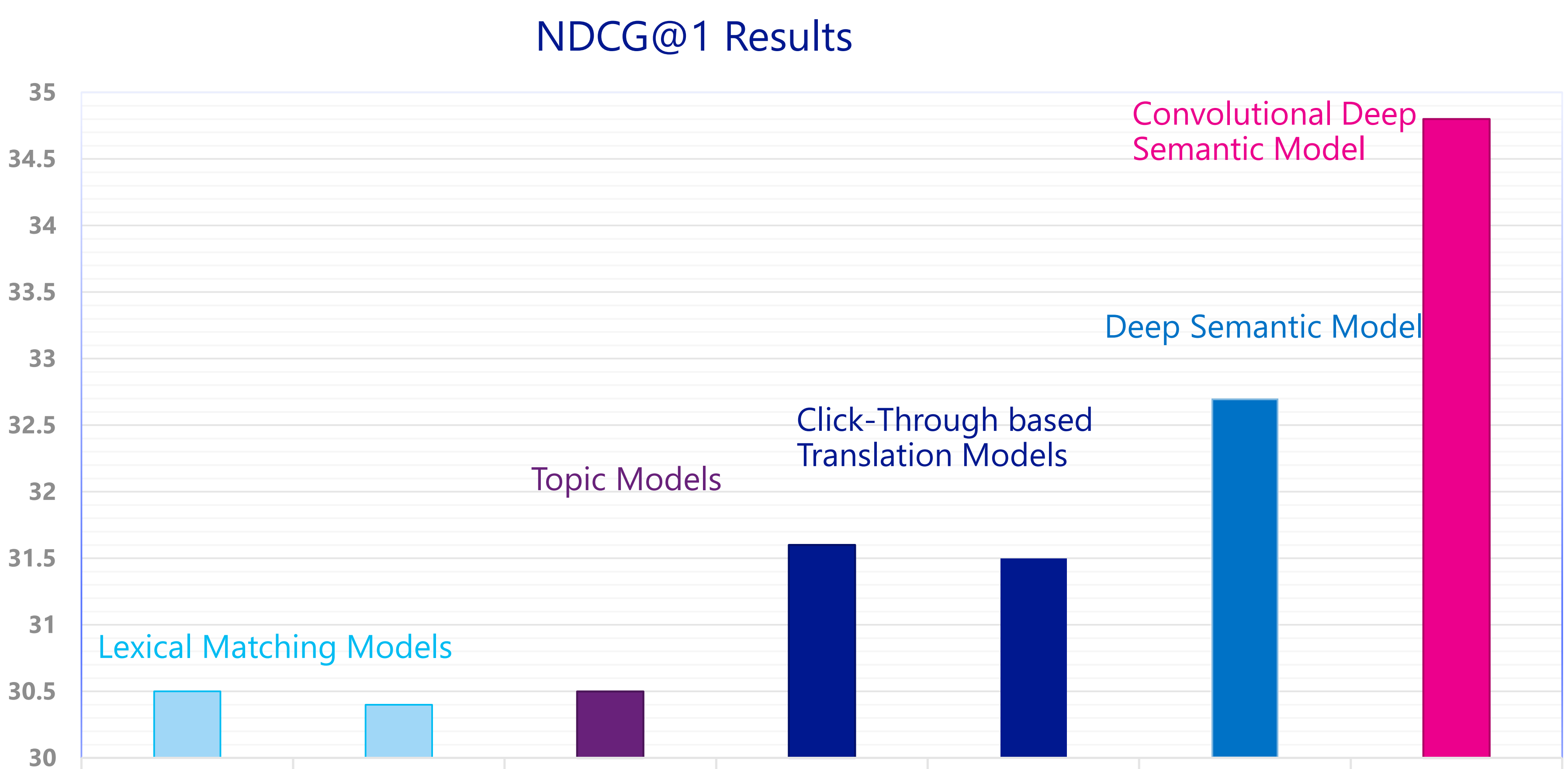
- Testing Dataset
  - **12,071** English queries
  - around 65 web document associated to each query in average
  - Human gives each <query, doc> pair the label, with range **0 to 4**
  - 0: Bad 1: Fair 2: Good 3: Perfect 4: Excellent
- Evaluation Metric: (higher the better)
  - NDCG
- Using NVidia GPU K40 for training



Dist. of query and doc title length

Results

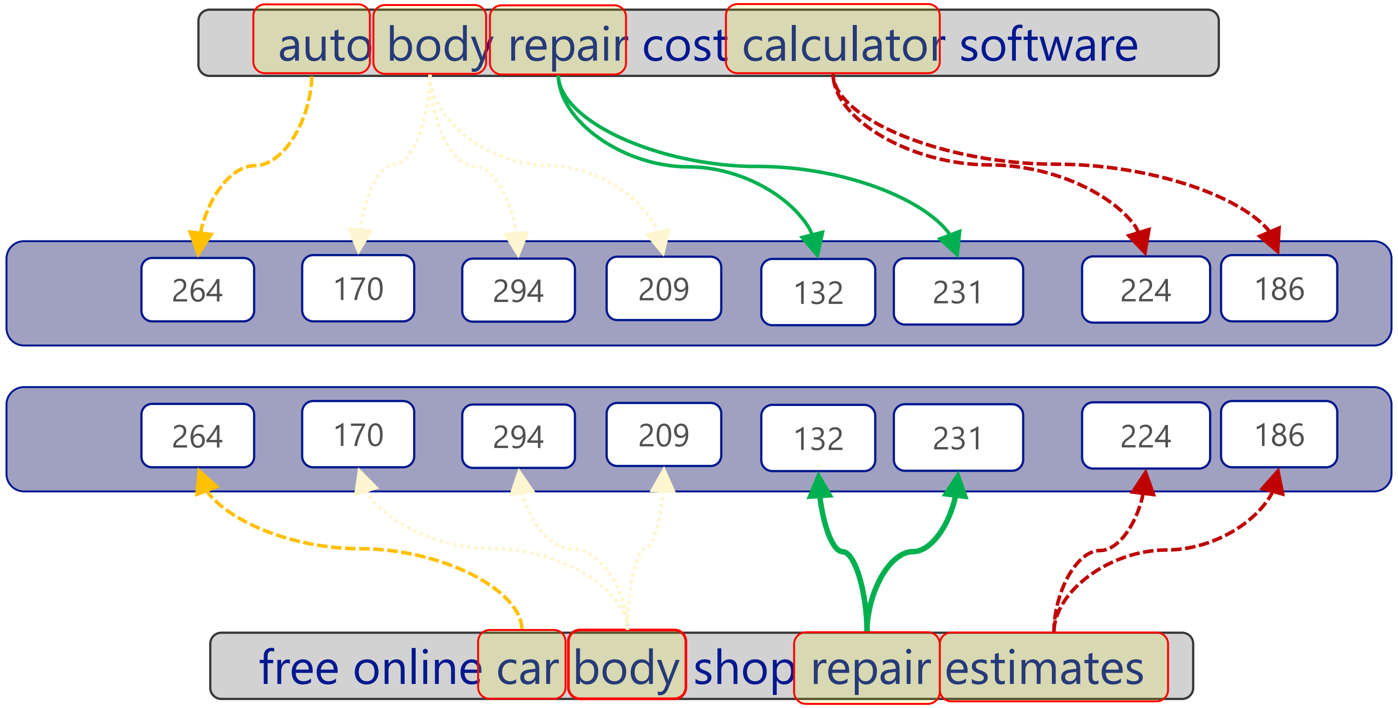
[Shen et al. CIKM2014]



	BM25	ULM	PLSA	BLTM	WTM	DSSM	CDSSM
NDCG@1	30.5	30.4	30.5	31.6	31.5	32.7	34.8

Example: semantic matching

- Semantic matching of query and document

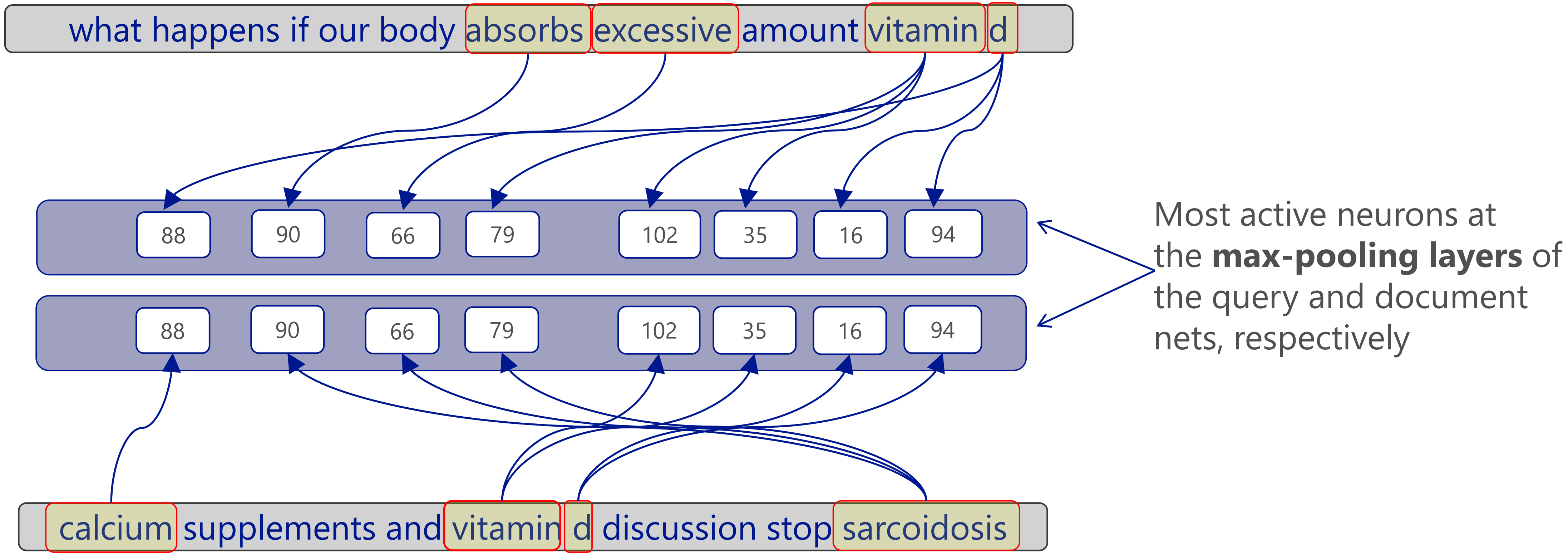


Most active neurons at the **max-pooling layers** of the query and document nets, respectively

More complex semantic matching example

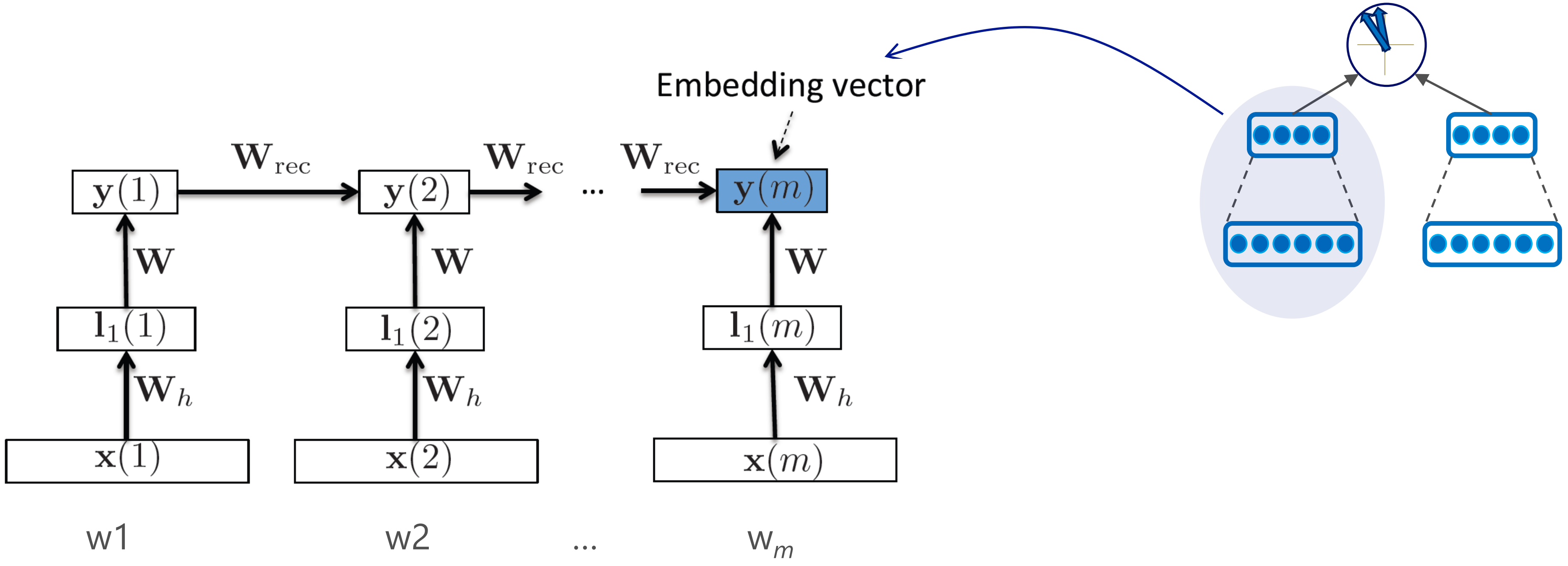
sarcoidosis is a disease, a symptom is excessive amount of calcium in one's urine and blood. So medicines that increase the absorbing of calcium should be avoid. While Vitamin d is closely associated to calcium absorbing.

We observed that "sarcoidosis" in the document title and "absorbs" "excessive" and "vitamin (d)" in the query have high activations at neurons 90, 66, 79, indicating that the model knows that "sarcoidosis" share similar semantic meaning with "absorbs" "excessive" "vitamin (d)", collectively.



Recurrent DSSM

- Encode the word one by one in the recurrent hidden layer
- The hidden layer at the last word codes the semantics of the full sentence
- Model is trained by a cosine similarity driven objective

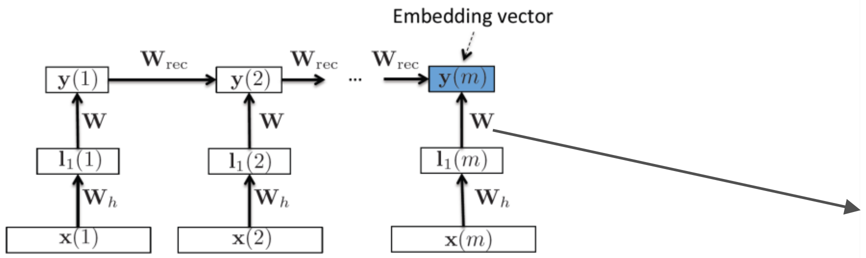


[Palangi, Deng, Shen, Gao, He, Chen, Song, Ward, Deep Sentence Embedding Using the LSTM network: Analysis and Application to IR, IEEE TASL, 2016]

Using LSTM cells

LSTM (long short term memory) uses special cells in RNN

[Hochreiter and J. Schmidhuber, 1997]



$$\mathbf{y}_g(t) = g(\mathbf{W}_4\mathbf{l}_1(t) + \mathbf{W}_{\text{rec}4}\mathbf{y}(t-1) + \mathbf{b}_4)$$
$$\mathbf{i}(t) = \sigma(\mathbf{W}_3\mathbf{l}_1(t) + \mathbf{W}_{\text{rec}3}\mathbf{y}(t-1) + \mathbf{W}_{p3}\mathbf{c}(t-1) + \mathbf{b}_3)$$
$$\mathbf{f}(t) = \sigma(\mathbf{W}_2\mathbf{l}_1(t) + \mathbf{W}_{\text{rec}2}\mathbf{y}(t-1) + \mathbf{W}_{p2}\mathbf{c}(t-1) + \mathbf{b}_2)$$
$$\mathbf{c}(t) = \mathbf{f}(t) \circ \mathbf{c}(t-1) + \mathbf{i}(t) \circ \mathbf{y}_g(t)$$
$$\mathbf{o}(t) = \sigma(\mathbf{W}_1\mathbf{l}_1(t) + \mathbf{W}_{\text{rec}1}\mathbf{y}(t-1) + \mathbf{W}_{p1}\mathbf{c}(t) + \mathbf{b}_1)$$
$$\mathbf{y}(t) = \mathbf{o}(t) \circ h(\mathbf{c}(t)) \tag{2}$$

where $\circ$ denotes Hadamard (element-wise) product.

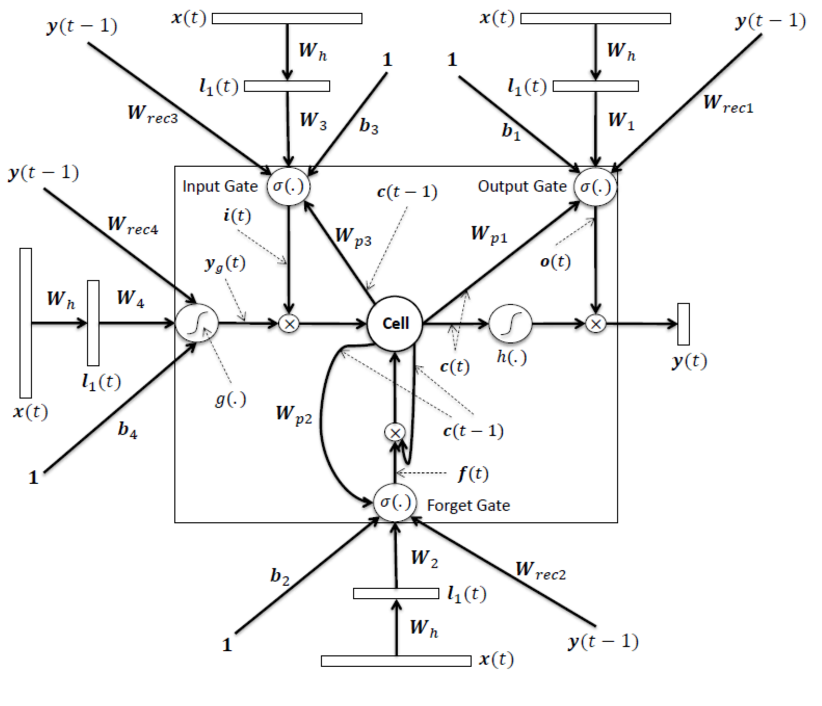


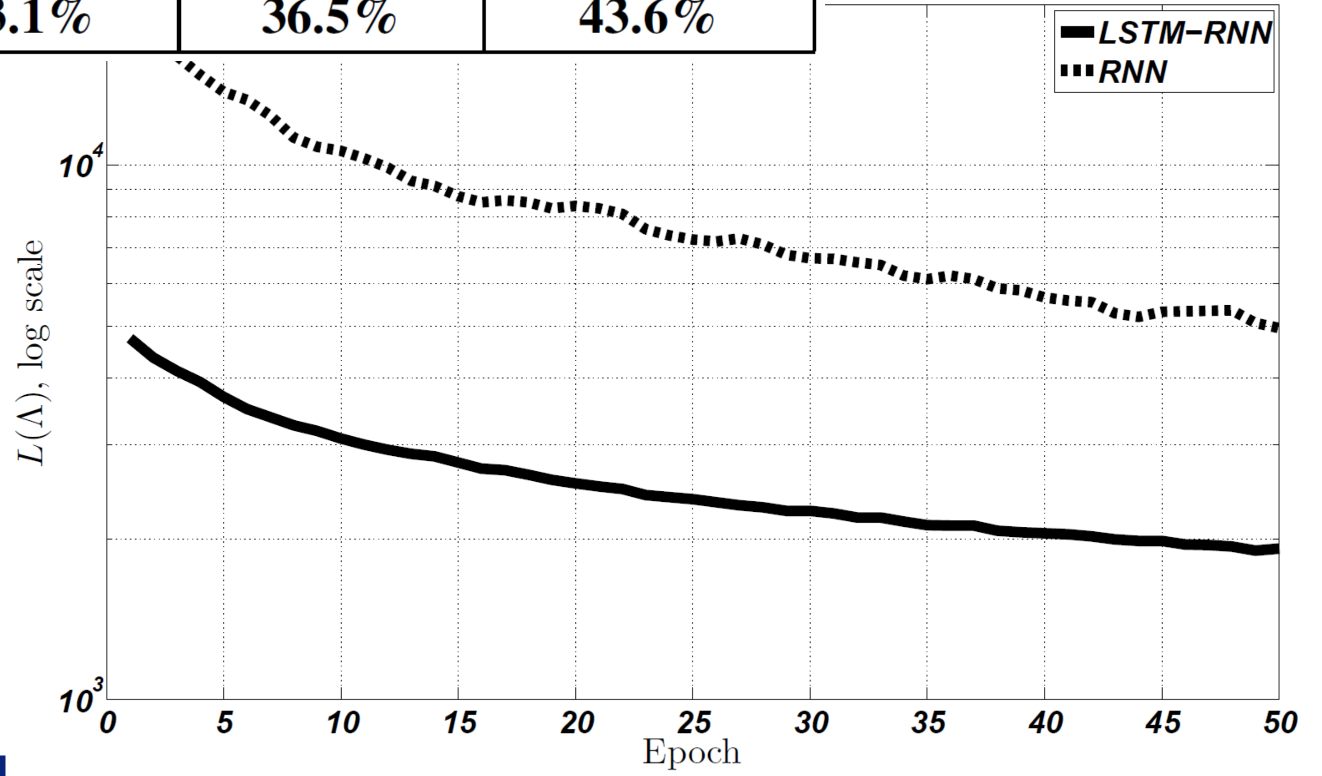
Figure 2. The basic LSTM architecture used for sentence embedding

Results

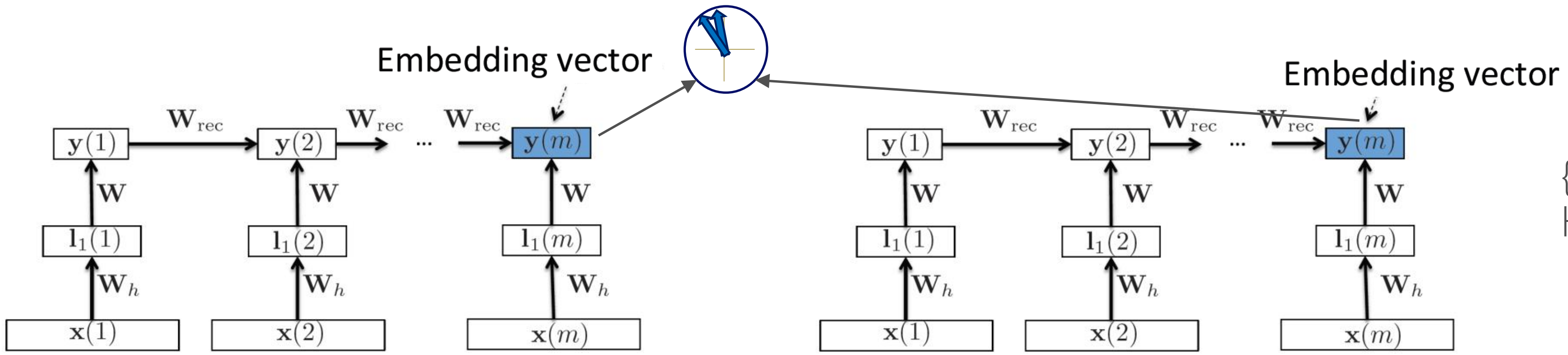
Model	NDCG@1	NDCG@3	NDCG@10
BM25	30.5%	32.8%	38.8%
PLSA (T=500)	30.8%	33.7%	40.2%
DSSM (nhid = 288/96), 2 Layers	31.0%	34.4%	41.7%
CLSM (nhid = 288/96), 2 Layers	31.8%	35.1%	42.6%
RNN (nhid = 288), 1 Layer	31.7%	35.0%	42.3%
LSTM-RNN (ncell = 96), 1 Layer	**33.1%**	**36.5%**	**43.6%**

LSTM learns much faster than regular RNN

LSTM effectively represents the semantic information of a sentence using a vector



Related work: DSSM vs. Seq2Seq



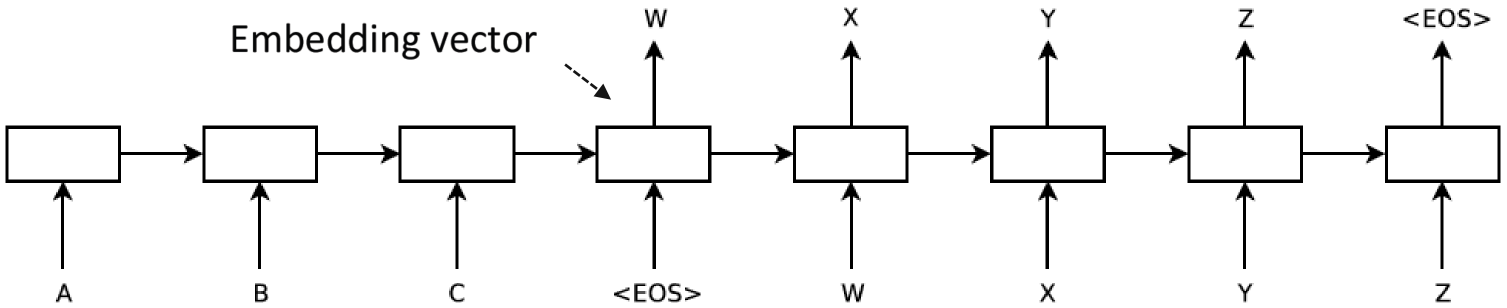
Source sentence

Target sentence

{Palangi, Deng, Shen, Gao, He, Chen, Song, Ward, 2016]

DSSM optimizes *sentence-level* semantic similarity

vs.



Source sentence

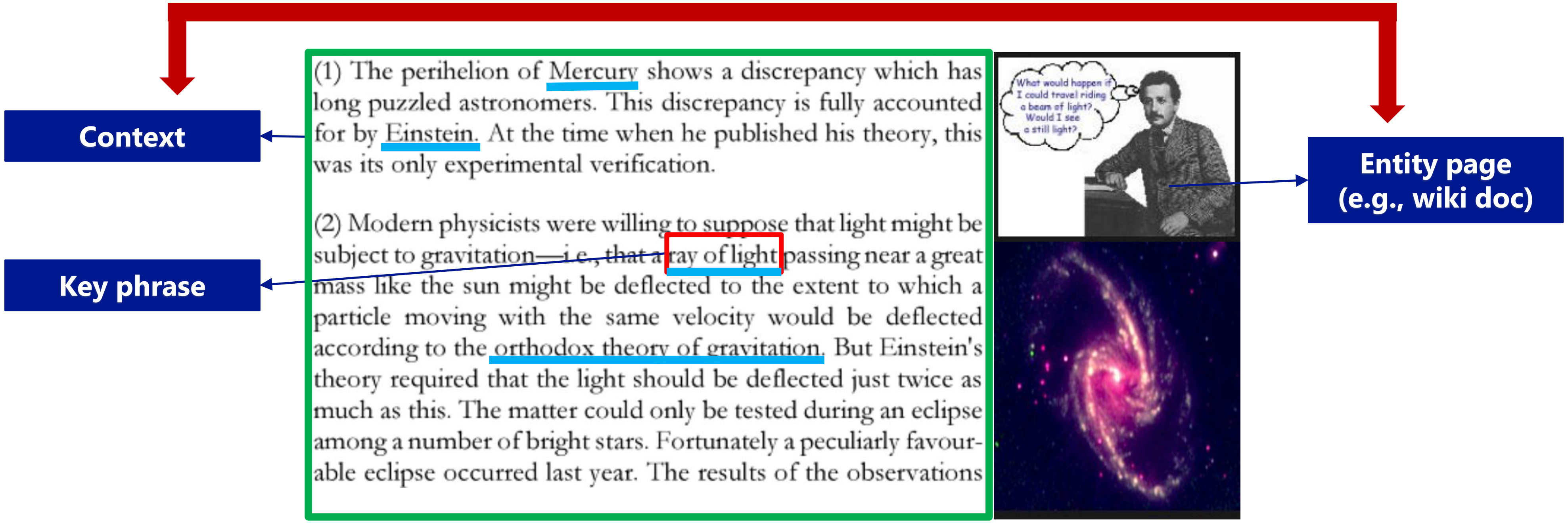
Target sentence

Seq2Seq optimizes *word-level* cross-entropy

[Sutskever, Vinyals, Le, 2014. Sequence to Sequence Learning with Neural Networks]

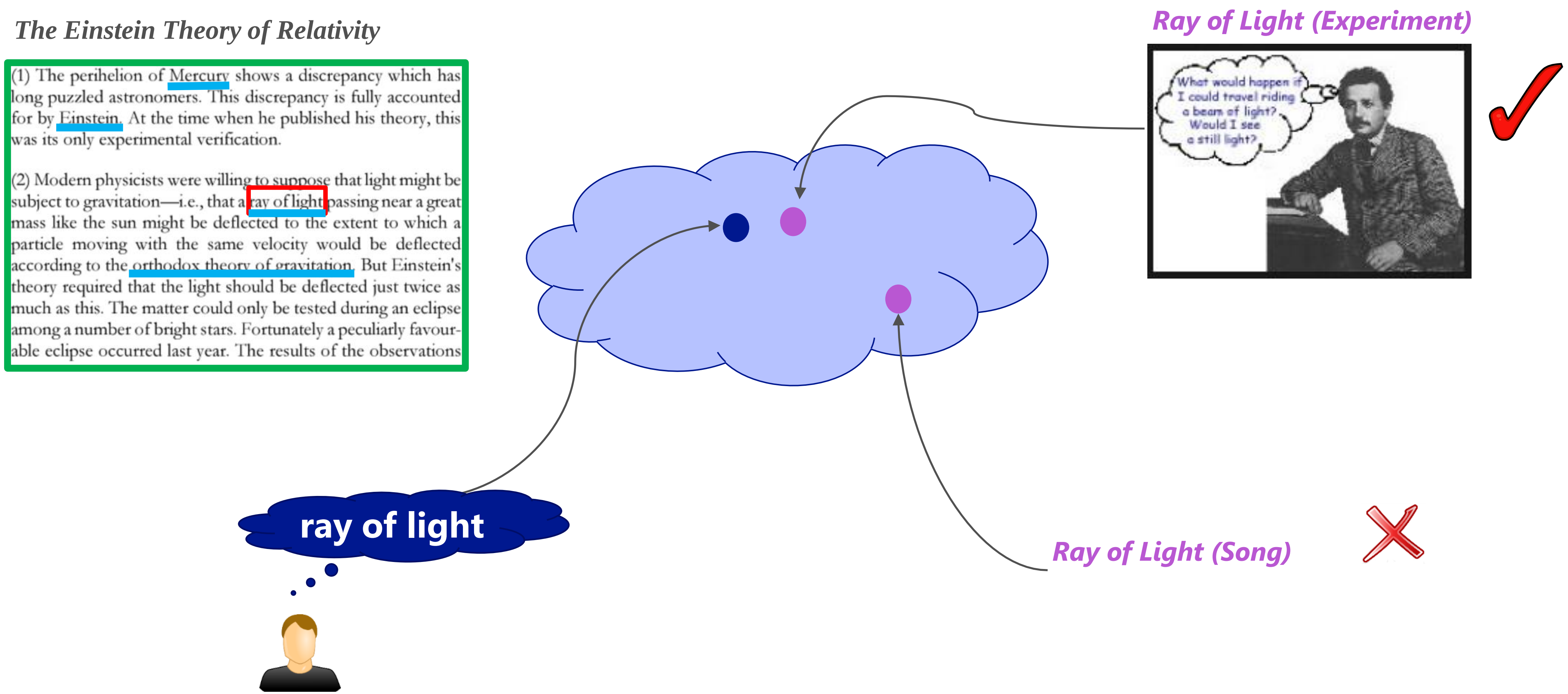
Contextual Entity Ranking

Given a user-highlighted text span representing an entity of interest, search for supplementary document for the entity



Gao, Pantel, Gamon, He, Deng, Shen, "Modeling interestingness with deep neural networks." EMNLP2014

Learning DSSM for contextual entity ranking



Xiaodong He

Extract Labeled Pairs from Web Browsing Logs
Contextual Entity Search

- When a hyperlink H points to a Wikipedia P'

...
I spent a lot of time finding music that was motivating and that I'd also want to listen to through my phone. I could find none. None! I wound up downloading three Metallica songs, a Judas Priest song and one from Bush.
...

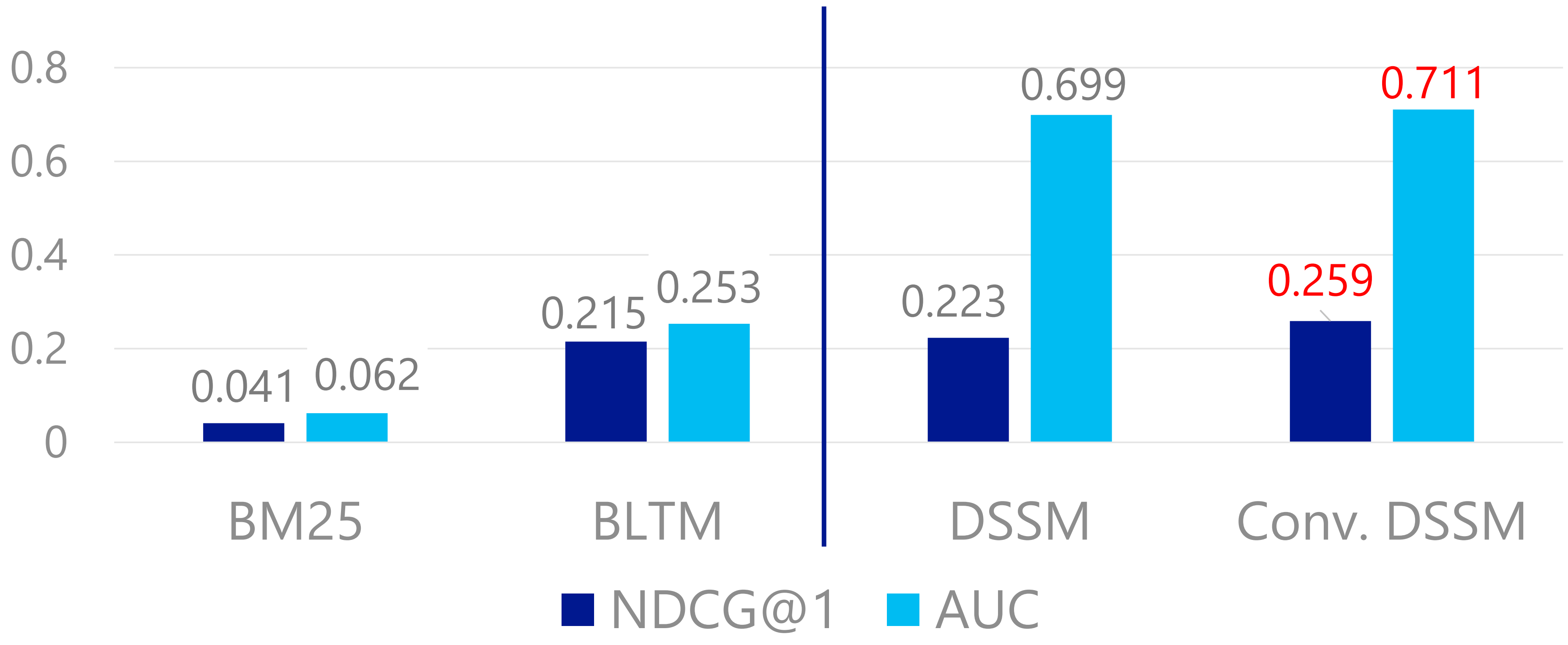
http://en.wikipedia.org/wiki/Bush_(band)

- (anchor text of H & surrounding words, text in P')

Contextual Entity Search: Experimental Settings

- Training/validation data: 18M of user clicks in wiki pages
- Evaluation data
  - Sample 10k Web documents as the source documents
  - Use named entities in the doc as query; retain up to 100 returned documents as target documents
  - Manually label whether each target document is a good page describing the entity
  - 870k labeled pairs in total
- Evaluation metric: NDCG and AUC

Contextual Entity Search Results: DSSM



- DSSM: bag-of-words input
- Conv. DSSM: convolutional DSSM

Some related work

Deep CNN for text input
Mainly classification tasks in the paper

[Kalchbrenner, Grefenstette, Blunsom, A Convolutional Neural Network for Modelling Sentences, ACL2014]

Sequence to sequence learning

[Sutskever, Vinyals, Le, 2014. Sequence to Sequence Learning with Neural Networks]

Paragraph Vector
Learn a vector for a paragraph

Quoc Le, Tomas Mikolov, Distributed Representations of Sentences and Documents, in ICML 2014

Recursive NN (ReNN)
Tree structure, e.g., for parsing

[Socher, Lin, Ng, Manning, "Parsing natural scenes and natural language with recursive neural networks", 2011]

Tensor product representation (TPR)
Tree representation

[Smolensky and Legendre: The Harmonic Mind, From Neural Computation to Optimality-Theoretic Grammar, MIT Press, 2006]

Tree-structured LSTM Network
Tree structure LSTM

[Tai, Socher, Manning. 2015. Improved Semantic Representations From Tree-Structured LSTM Networks.]