CIPS Summer School
July 29, 2018 Beijing

Deep and Reinforcement Learning for Information Retrieval

Jun Xu, Liang Pang

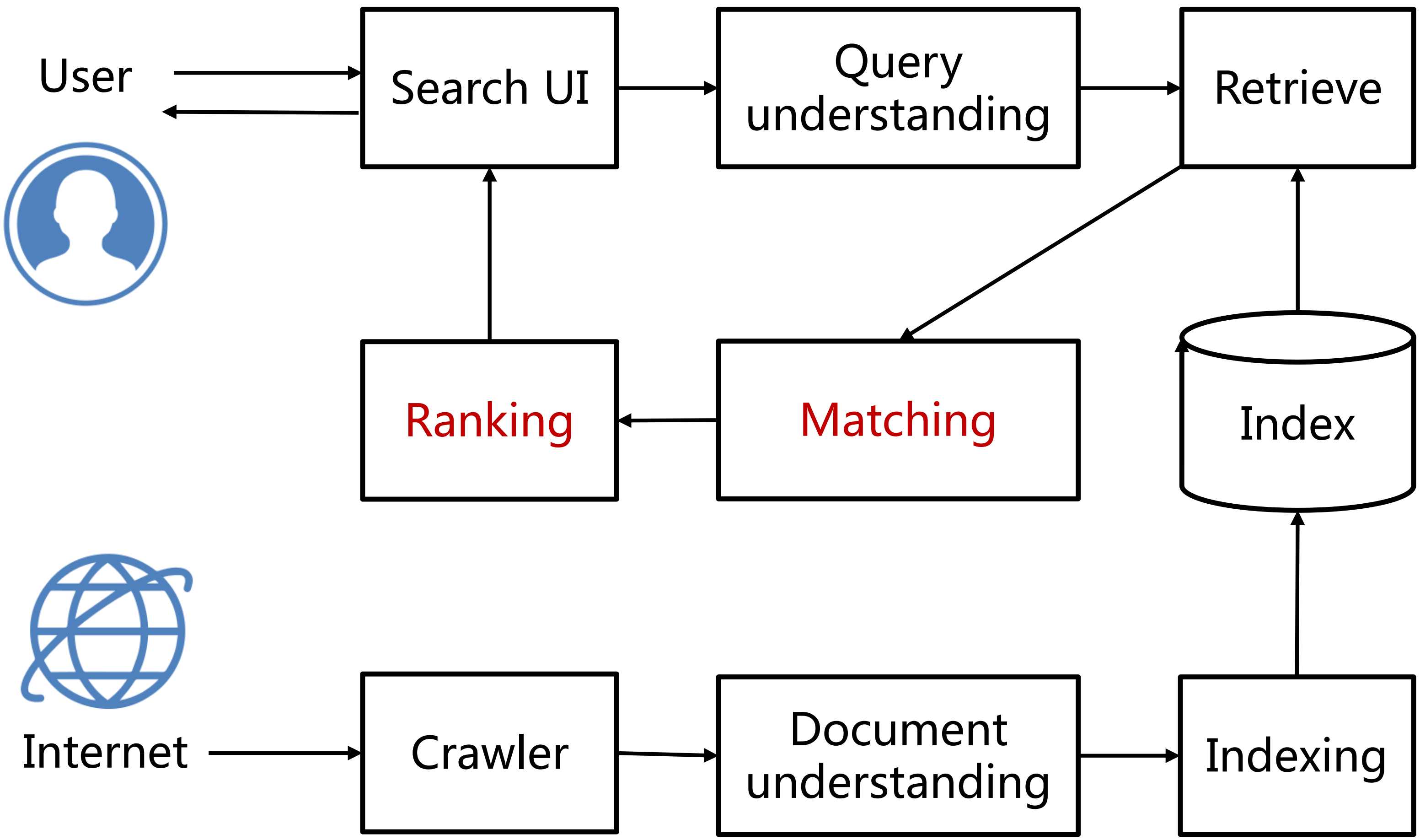
Institute of Computing Technology
Chinese Academy of Sciences

中国科学院计算技术研究所
INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES

Outline

- Introduction
- Deep Semantic Matching (Liang Pang)
- Reinforcement Learning to Rank (Jun Xu)
- Summary

Overview of Web Search Engine



Semantic Gap
the **Biggest** Challenge in Matching

- Same intent can be represented by different queries (representations)

- Search is still mainly based on term level matching

- Query document mismatch occurs, when searcher and author use different representations

Same Search Intent Different Query Representations Example: "Youtube"

yutube	yuotube	yuo tube
ytube	youtubr	yu tube
youtubo	youtuber	youtubecom
youtube om	youtube music videos	youtube videos
youtube	youtube com	youtube co
youtub com	you tube music videos	yout tube
youtub	you tube com yourtube	your tube
you tube	you tub	you tube video clips
you tube videos	www you tube com	wwww youtube com
www youtube	www youtube com	www youtube co
yotube	www you tube	www utube com
ww youtube com	www utube	www u tube
utube videos	utube com	utube
u tube com	utub	u tube videos
u tube	my tube	toutube
outube	our tube	toutube

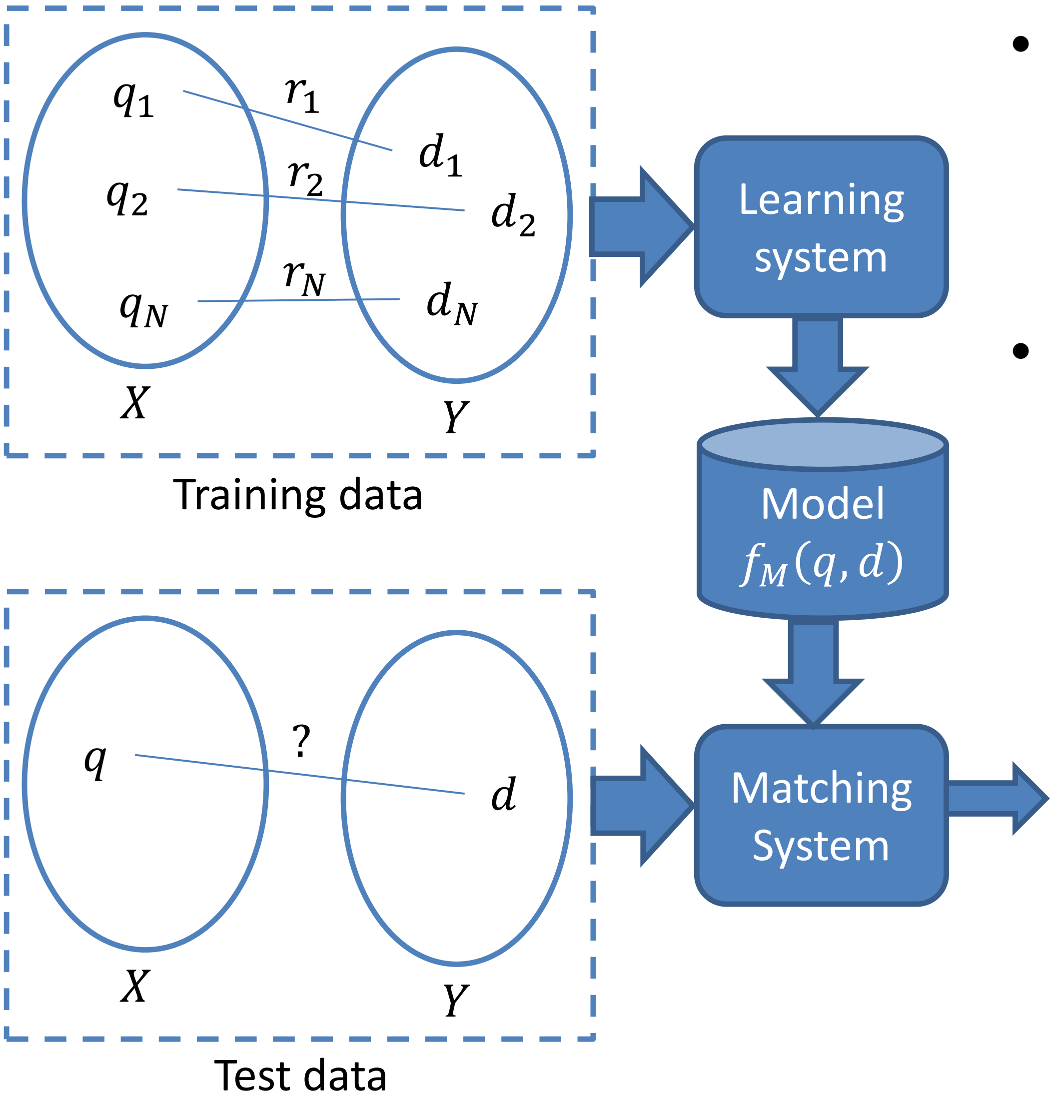
Same Search Intent Different Query Representations Example: "Distance between Sun and Earth"

"how far" earth sun	average distance from the earth to the sun
"how far" sun	how far away is the sun from earth
average distance earth sun	average distance from earth to sun
how far from earth to sun	distance from earth to the sun
distance from sun to earth	distance between earth and the sun
distance between earth & sun	distance between earth and sun
how far earth is from the sun	distance from the earth to the sun
distance between earth sun	distance from the sun to the earth
distance of earth from sun	distance from the sun to earth
"how far" sun earth	how far away is the sun from the earth
how far earth from sun	distance between sun and earth
how far from earth is the sun	how far from the earth to the sun
distance from sun to the earth	

Example of Query-Document Mismatch

Query	Document	Term Matching	Semantic Matching
seattle best hotel	seattle best hotels	partial	Yes
pool schedule	swimming pool schedule	partial	Yes
natural logarithm transformation	logarithm transformation	partial	Yes
china **kong**	china **hong kong**	partial	No
why are **windows** so expensive	why are **macs** so expensive	partial	No

Machine Learning for Matching

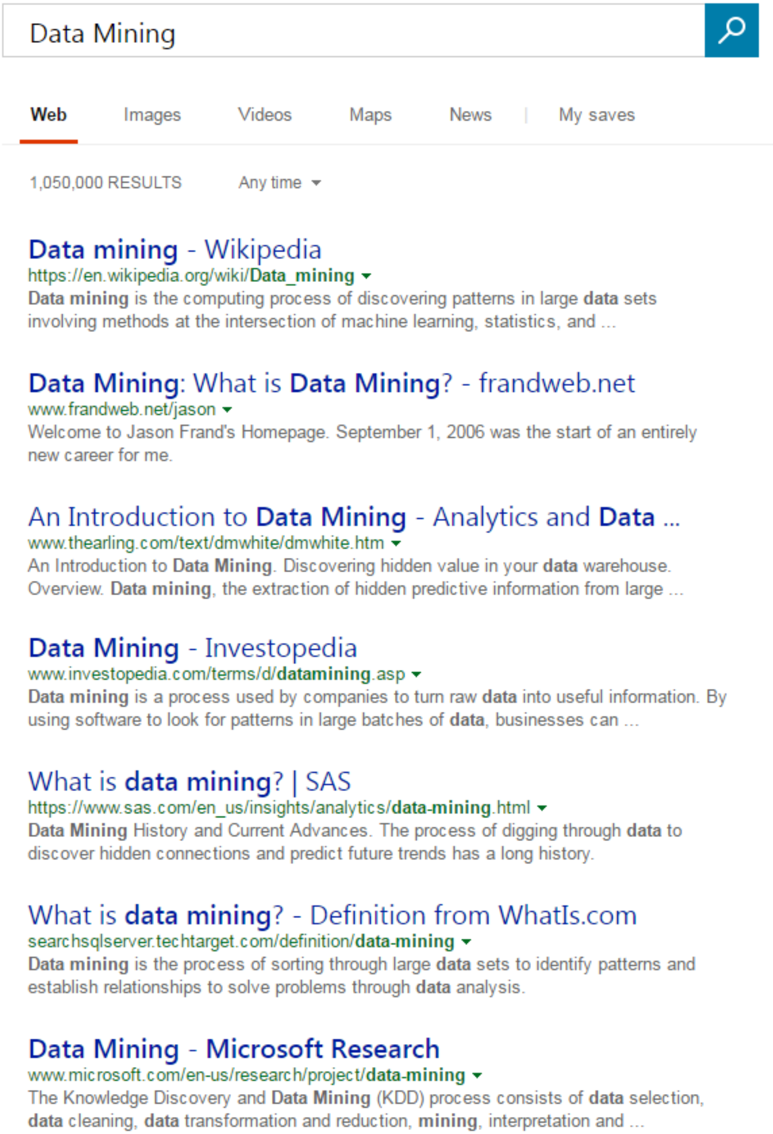


$f_M(q,d)$

- Using relations in data for learning the matching function $f_M(q,d)$ or $P(r|q,d)$
- Training data $\{(q_i, d_i, r_i)\}_{i=1}^{N}$
  - Queries and documents (users and items) represented with feature vectors or ID's
  - Target can be binary or numerical values

Ranking is Important for Web Search

Query



Doc1

Doc2

Doc3

Doc4

Doc5

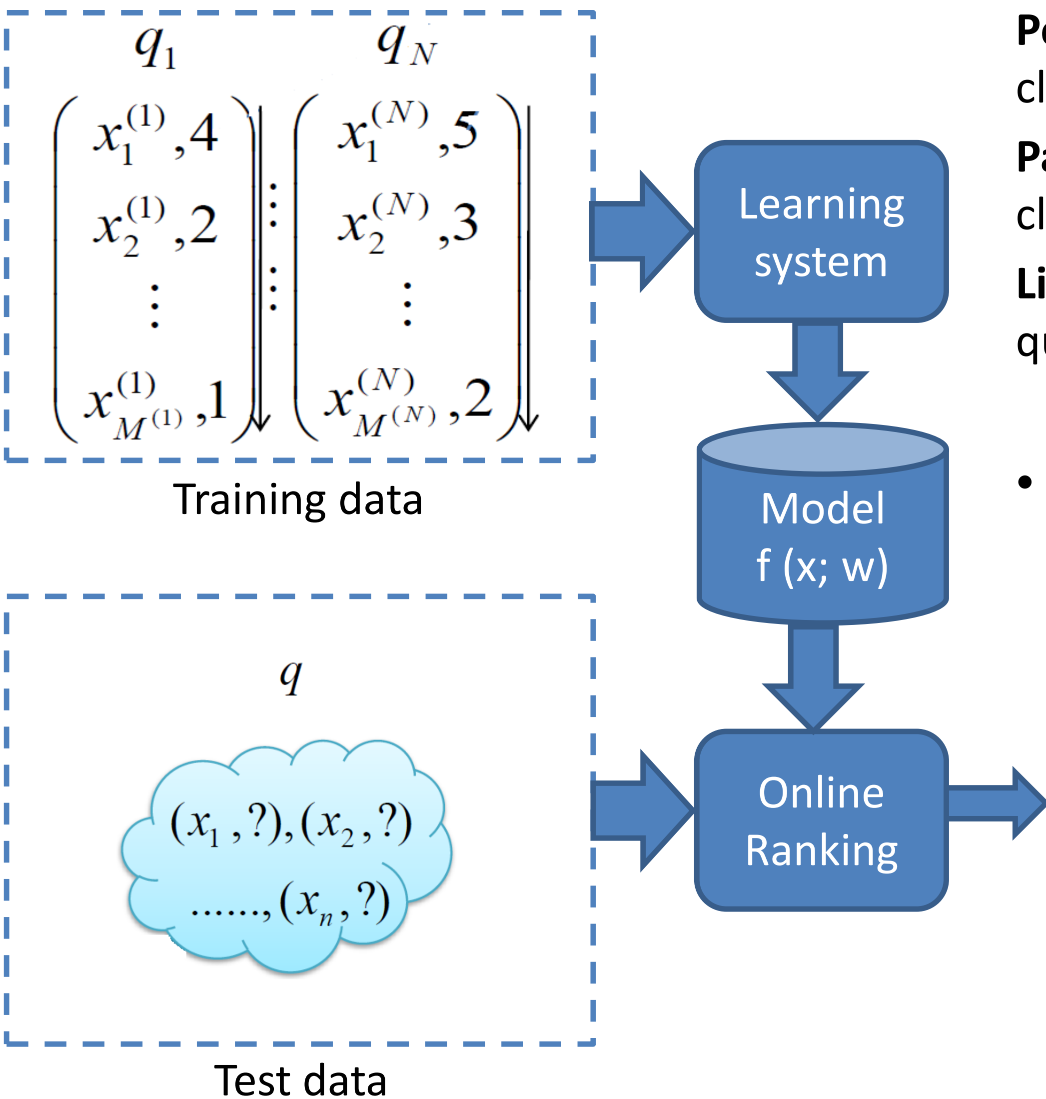
Doc6

……

- Criteria
  - Relevance
  - Diversity
  - Freshness

  ……
- Ranking model
  - Heuristic
    - Relevance: BM25, LMIR
    - Diversity: MMR, xQuAD
  - Learning to rank

Machine Learning for Ranking

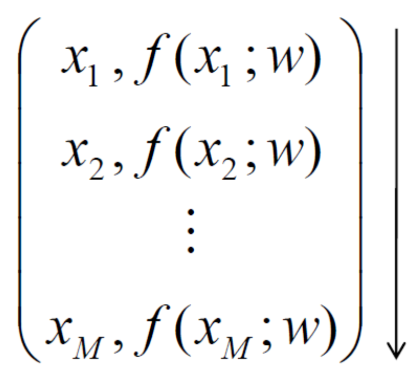


Point-wise: ranking as regression or classification over query-documents

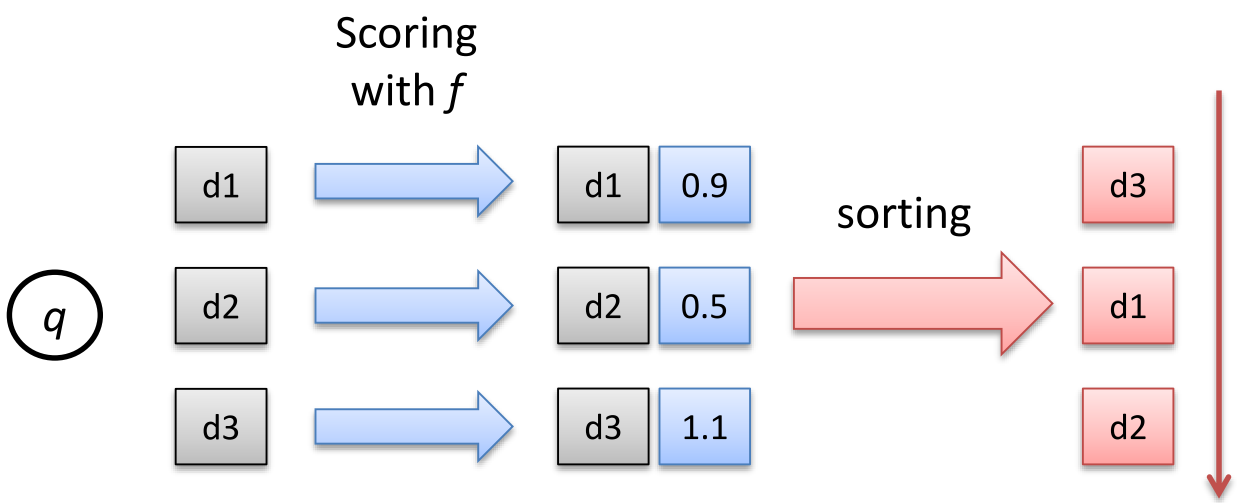
Pair-wise: ranking as binary classification over preference pairs

List-wise: training/predicting ranking at query (document list) level

- Using document partial ordering relations in data for learning the ranking function



Independent Relevance Assumption



- Utility of a doc is independent of other docs
- Ranking as scoring & sorting
  - Each documents can be scored independently
  - Scores are independent of the rank

Beyond Independent Relevance

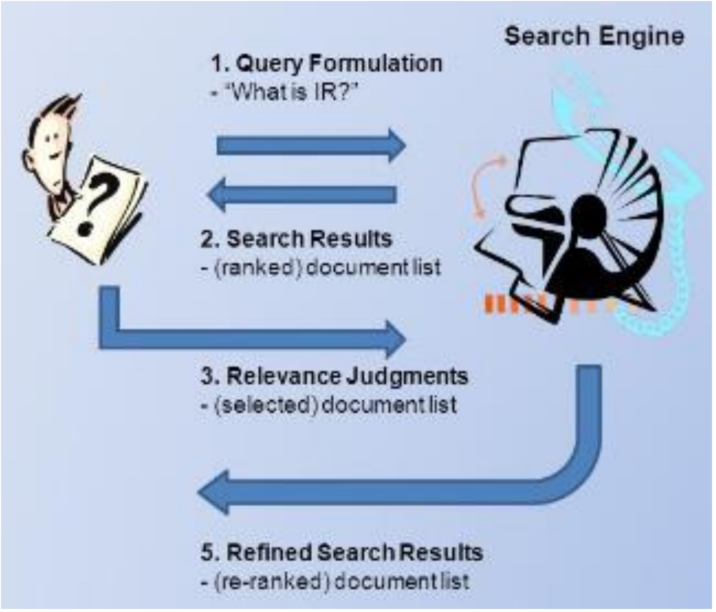
Query: Programming language

Good	Bad
Java	Java
C++	Java
Python	Java

- More ranking criteria

  e.g., search result diversification
  - Covering as much subtopics as possible with a few documents
  - Need consider novelty of a document given preceding documents
- Complex application environment

  e.g., Interactive IR
  - Human interacts with the system during the ranking process
  - User feedback is helpful for improving the remaining results



Need more powerful ranking mechanism!

Outline

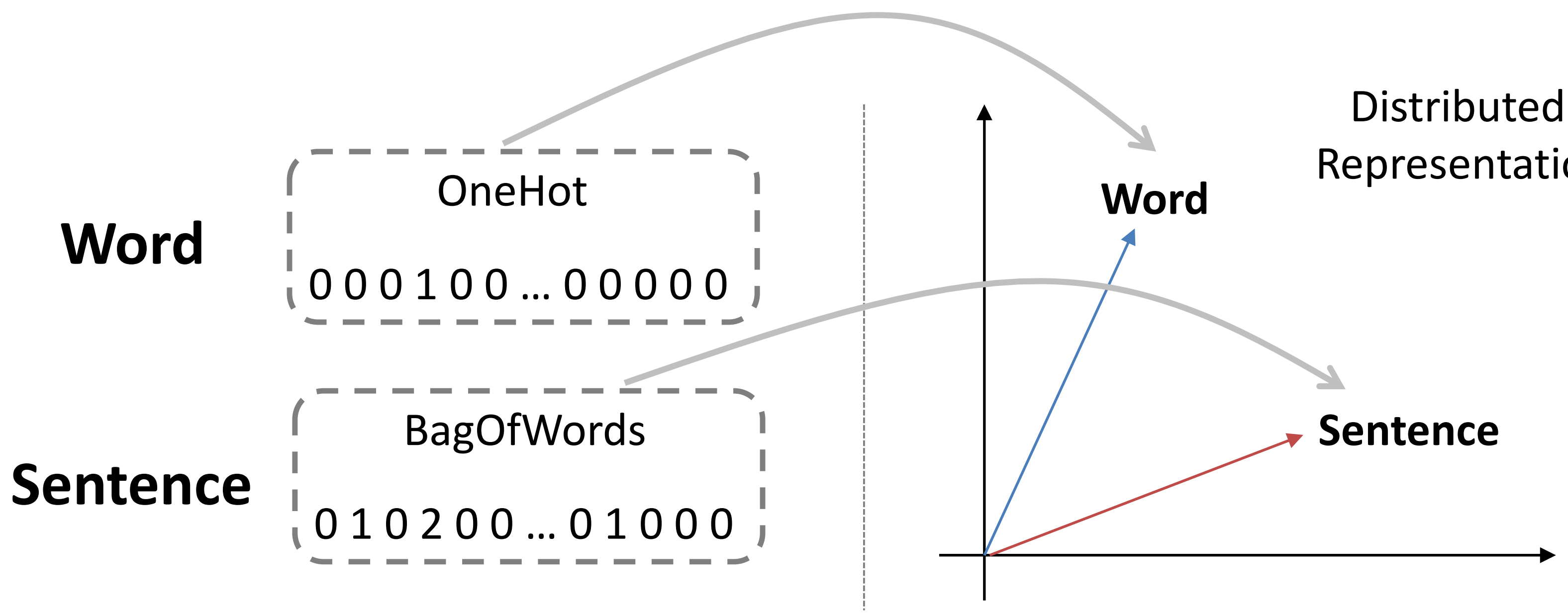
- Introduction
- Deep Semantic Matching
  – Methods of Representation Learning
  – Methods of Matching Function Learning
- Reinforcement Learning to Rank
  – Formulation IR Ranking with RL
  – Approaches
- Summary

Growing Interests in “Deep Matching”

- Success of deep learning in other fields
  - Speech recognition, computer vision, and natural language processing
- Growing presence of deep learning in IR research
  - SIGIR keynote, Tutorial, and Neu-IR workshop
- Adopted by industry
  - ACM News: *Google Turning its Lucrative Web Search Over to AI Machines* (Oct. 26, 2015)
  - WIRED: *AI is Transforming Google Search. The Rest of the Web is Next* (April 2, 2016)
- Chris Manning (Stanford)’s SIGIR 2016 keynote:
  “I’m certain that ***deep learning*** *will come to dominate SIGIR over the next couple of years* … just like speech, vision, and NLP before it.”

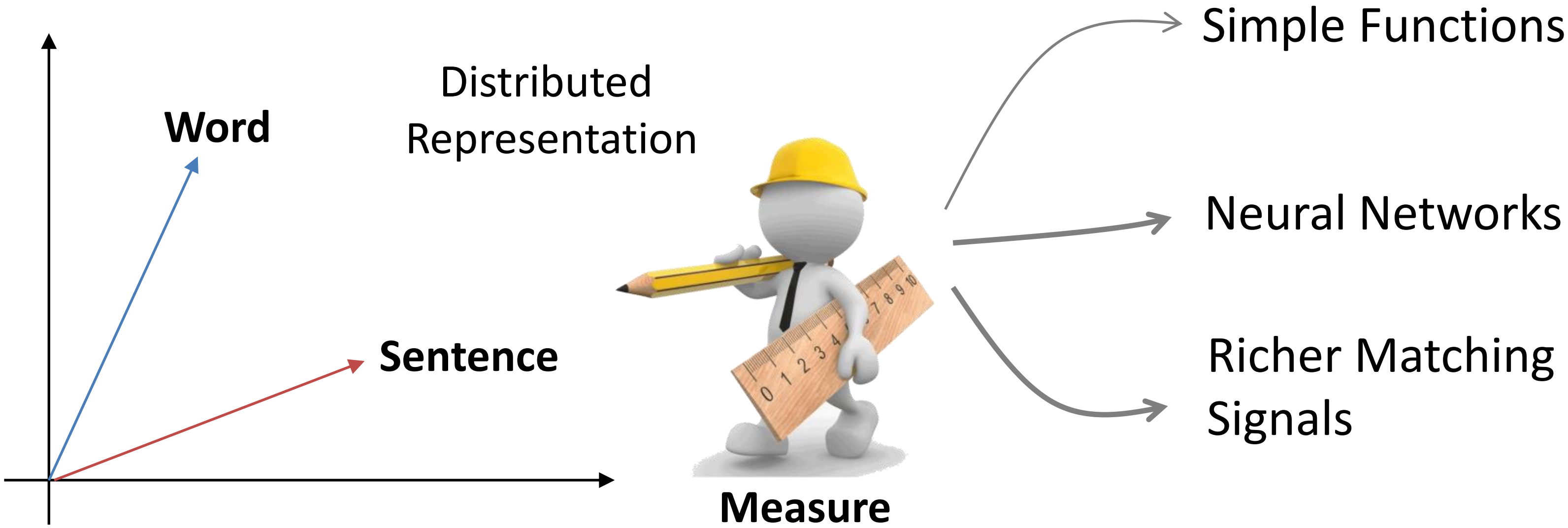
“Deep” Semantic Matching

- Representation
  - Word: one hot —> distributed
  - Sentence: bag-of-words —> distributed representation
  - Better representation ability, better generalization ability



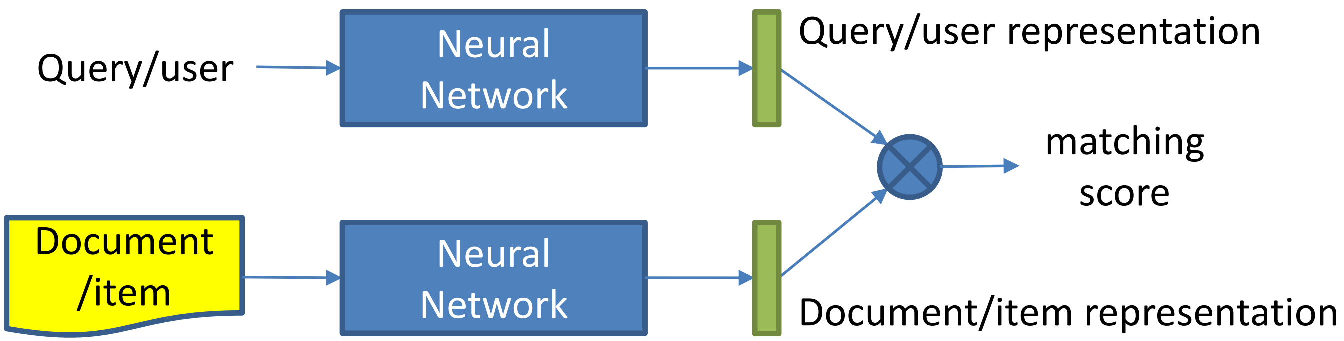
"Deep" Semantic Matching

- Matching function
  - Inputs (features): handcrafted —> automatically learned
  - Function: simple functions (e.g., cosine, dot product) —> neural networks (e.g., MLP, neural tensor networks)
  - Involving richer matching signals
  - Considering soft matching patterns

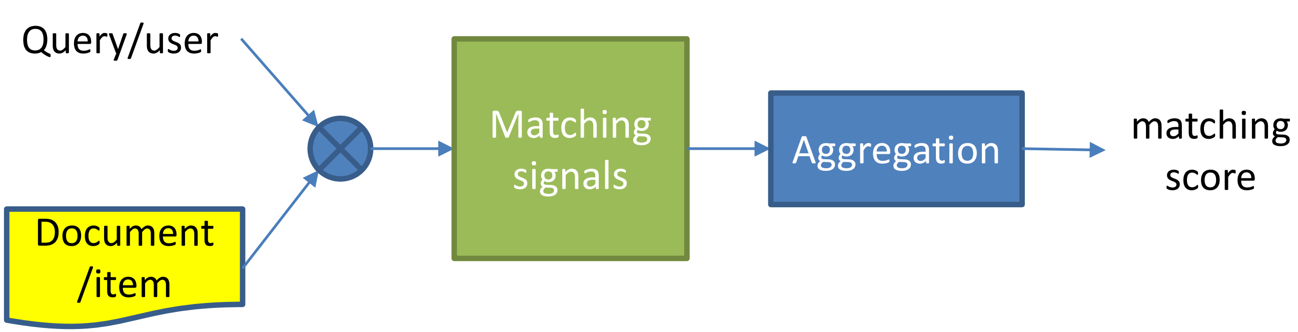


Deep Learning Paradigms for Matching

- Methods of representation learning

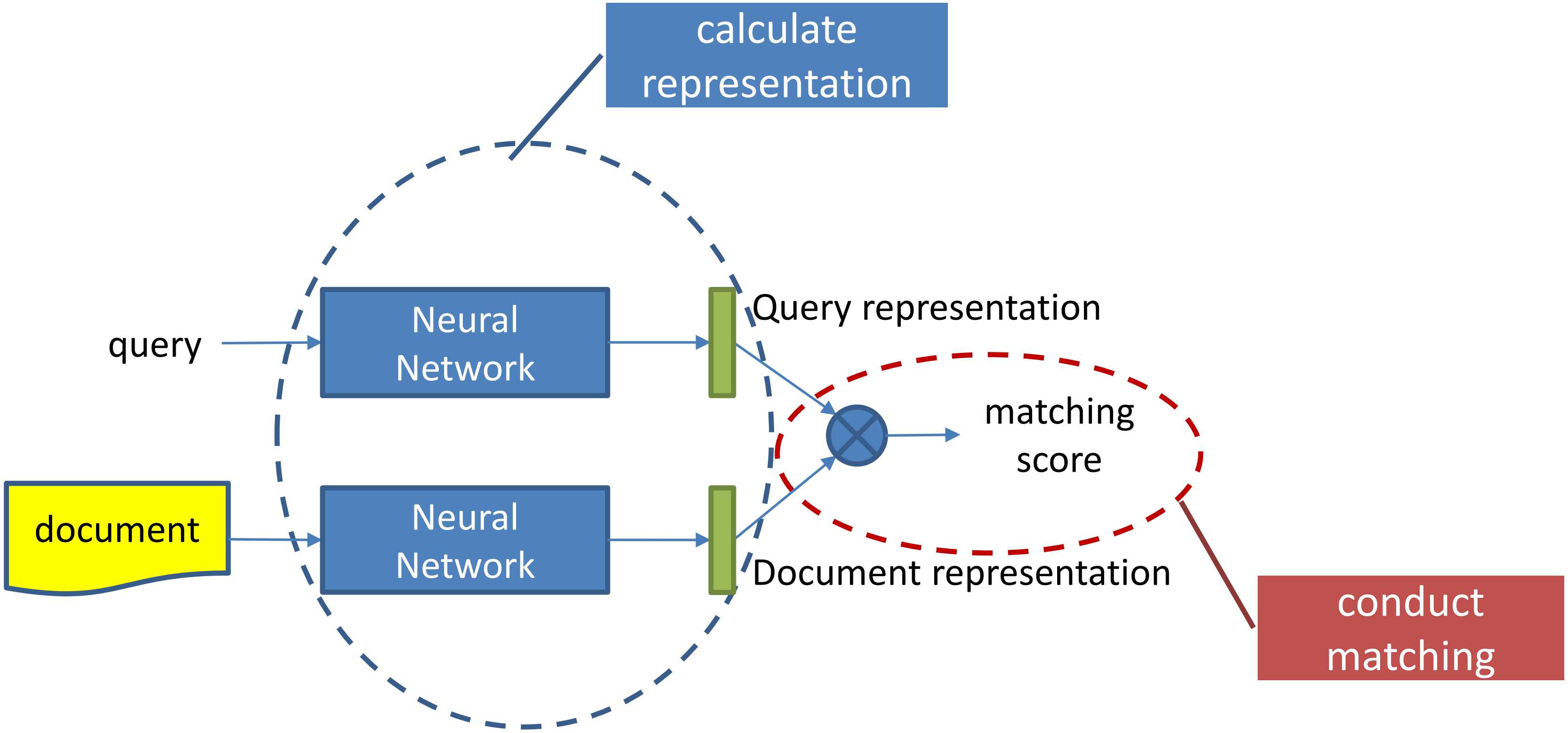


- Methods of matching function learning



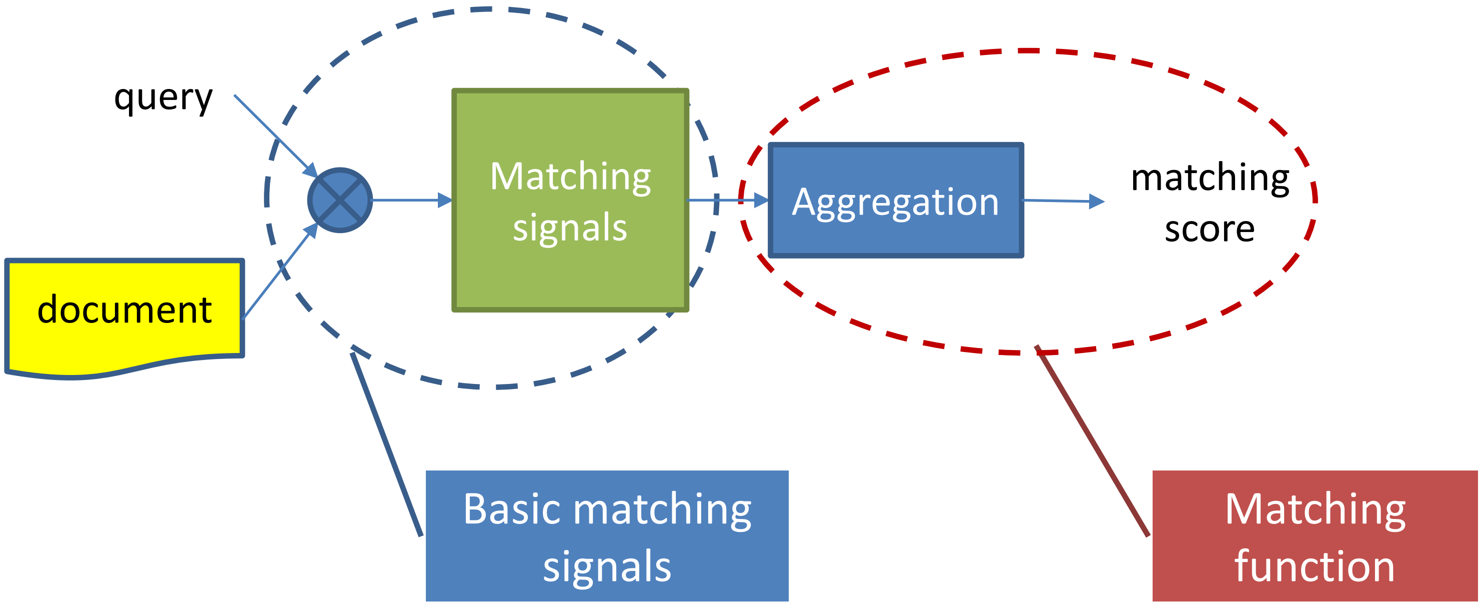
Methods of Representation Learning

- Step 1: calculate representation $\phi(x)$
- Step 2: conduct matching $F\big(\phi(x), \phi(y)\big)$



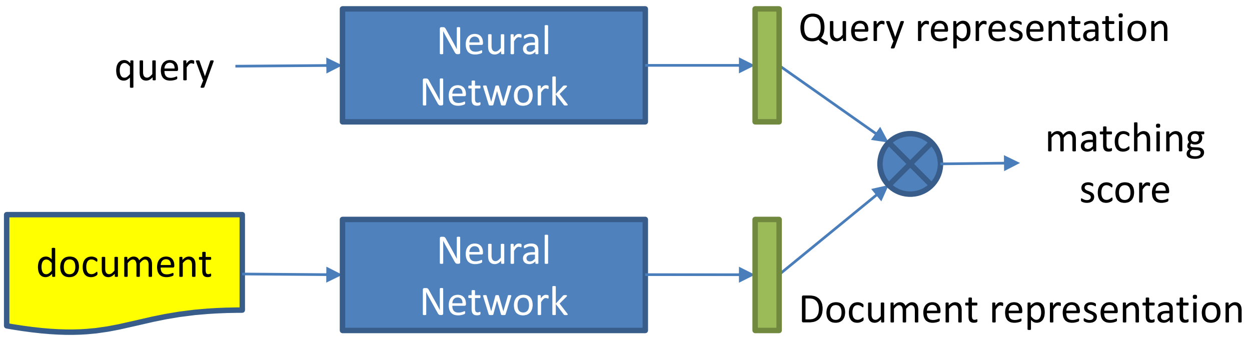
Methods of Matching Function Learning

- Step 1: construct basic low-level matching signals
- Step 2: aggregate matching patterns



Outline

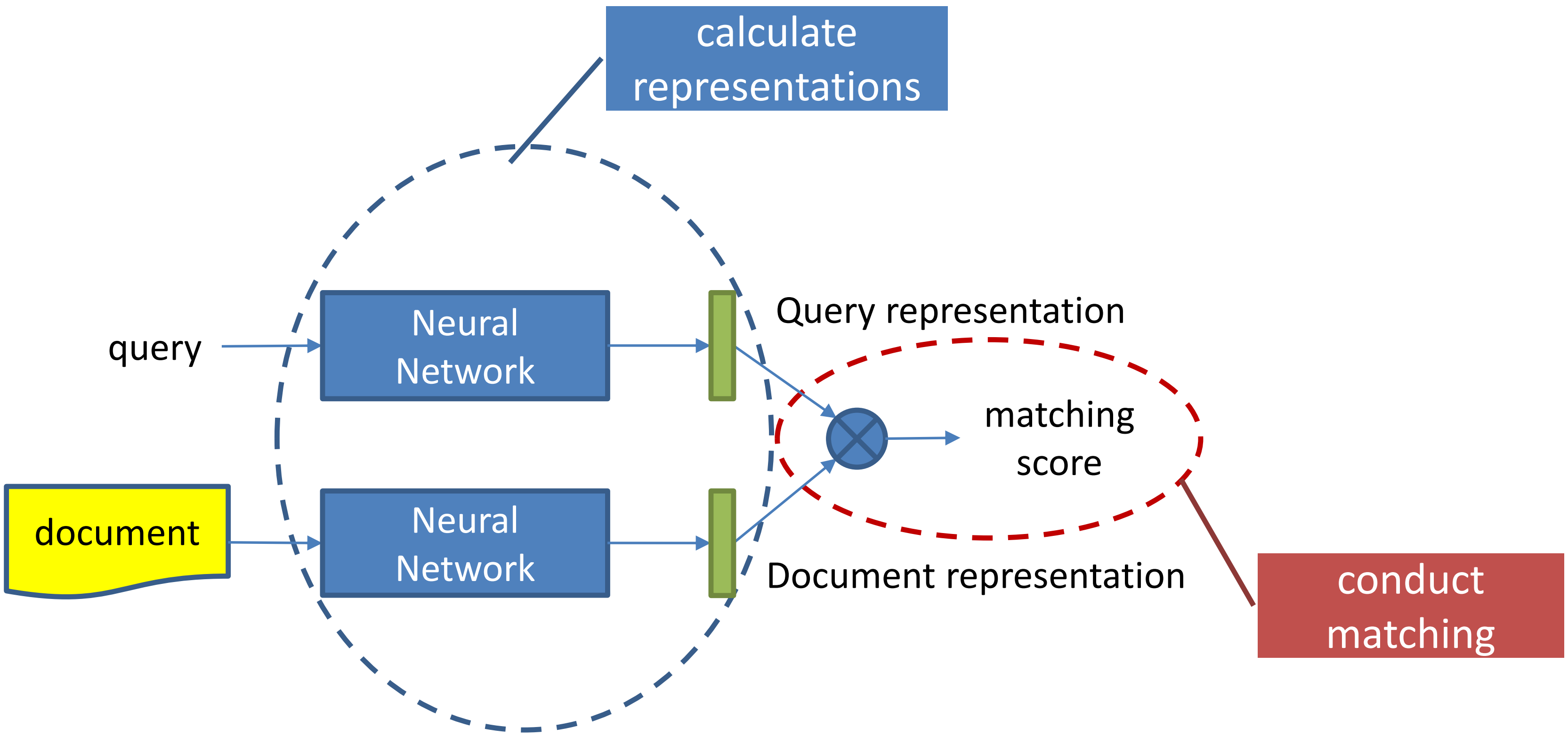
- Introduction
- Deep Semantic Matching
  - Methods of Representation Learning
  - Methods of Matching Function Learning
- Reinforcement Learning to Rank
  - Formulation IR Ranking with RL
  - Approaches
- Summary



METHODS OF REPRESENTATION LEARNING

Representation Learning for Query-Document Matching

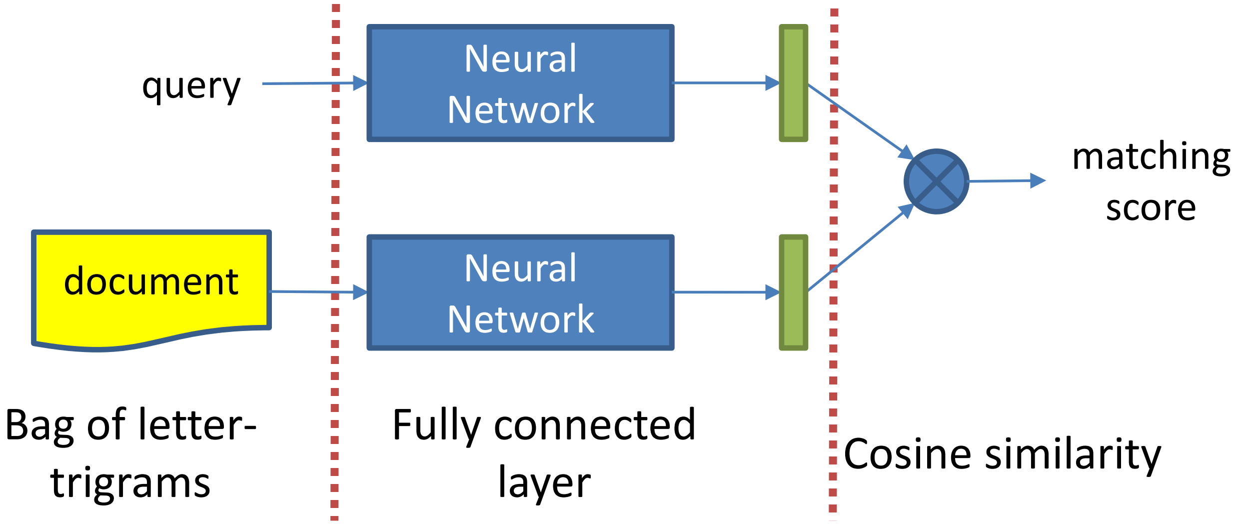
- Step 1: calculate query and document representation
  Step 2: conduct query-document matching



Typical Methods of Representation Learning for Matching

- 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 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 and Huang, 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)



- Bag-of-words representation
  - "candy store": [0, 0, 1, 0, …, 1, 0, 0]
- Bag of letter-trigrams representation
  - "#candy# #store#" --> #ca can and ndy dy# #st sto tor ore re#
  - Representation: [0, 1, 0, 0, 1, 1, 0, …, 1]
- Advantages of using bag of letter-trigrams
  - Reduce vocabulary: #words 500K → # letter-trigram: 30K
  - Generalize to unseen words
  - Robust to misspelling, inflection etc.

DSSM Query/Doc Representation: DNN

- Model: DNN (auto-encoder) to capture the compositional sentence representations

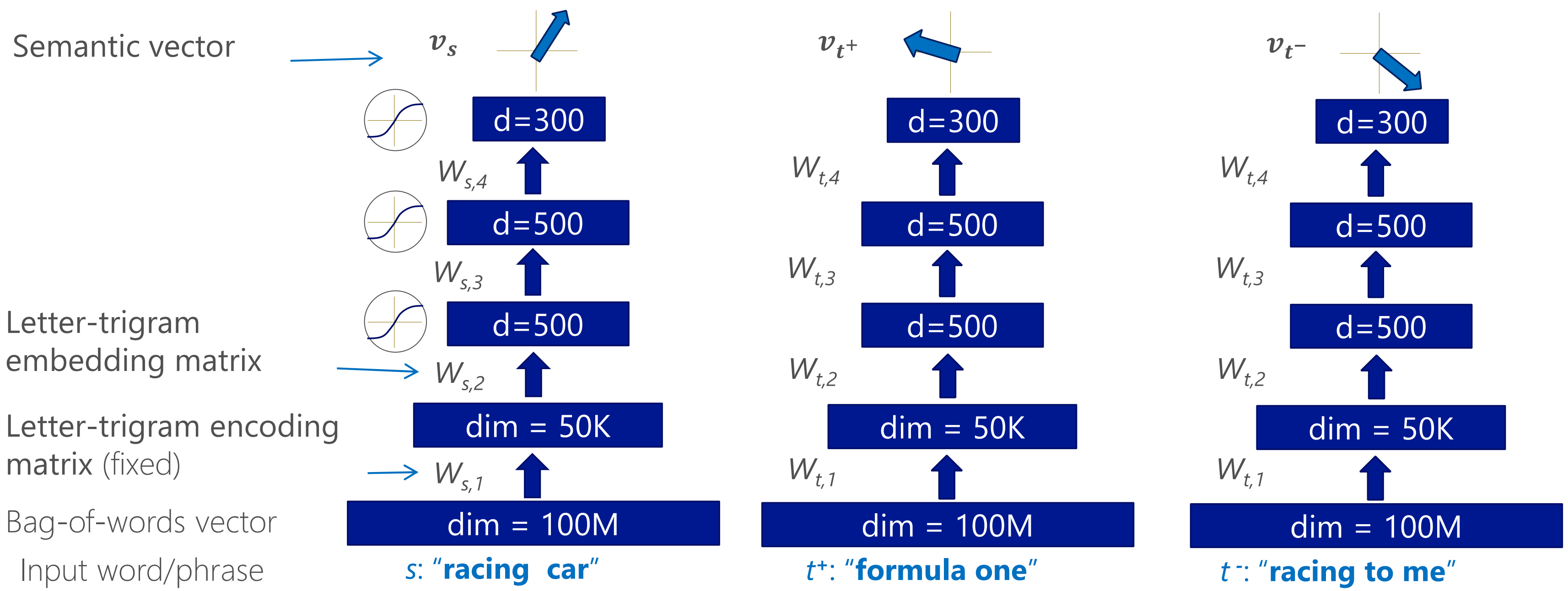


Figure courtesy of He et al., CIKM '14 tutorial

DSSM Matching Function

- Cosine similarity between semantic vectors

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

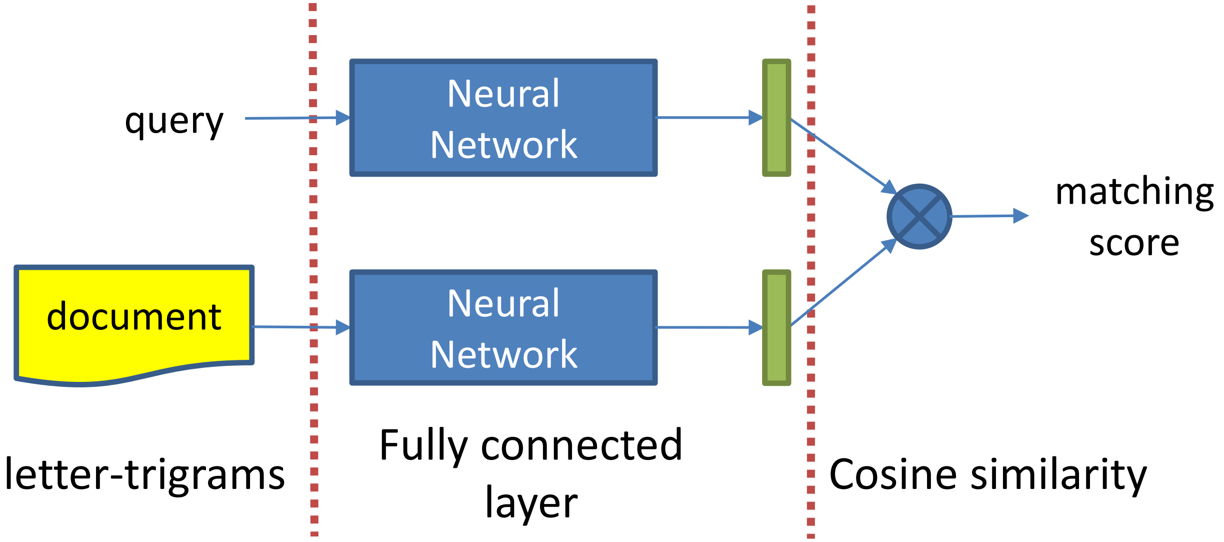
- Training
  - A query q and a list of docs $D = \{d^+, d_1^-, \cdots, d_k^-\}$
  - d^+ positive doc, $d_1^-, \cdots, d_k^-$ negative docs to query
  - Objective:

$$P(d^+|q) = \frac{\exp(\gamma \cos(q, d^+))}{\sum_{d \in D} \exp(\gamma \cos(q, d))}$$

DSSM: Brief Summary

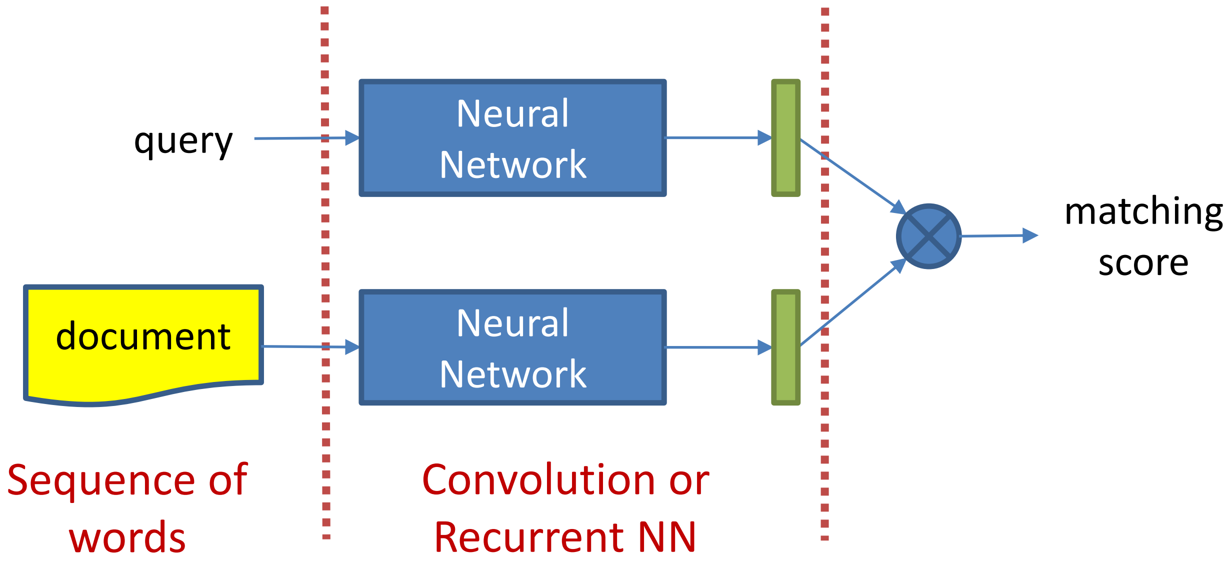
- **Inputs**: Bag of letter-trigrams as input for improving the scalability and generalizability
- **Representations**: mapping sentences to vectors with DNN: semantically similar sentences are close to each other
- **Matching**: cosine similarity as the matching function

- **Problem**: *the order information of words is missing* (bag of letter-trigrams cannot keep the word order information)



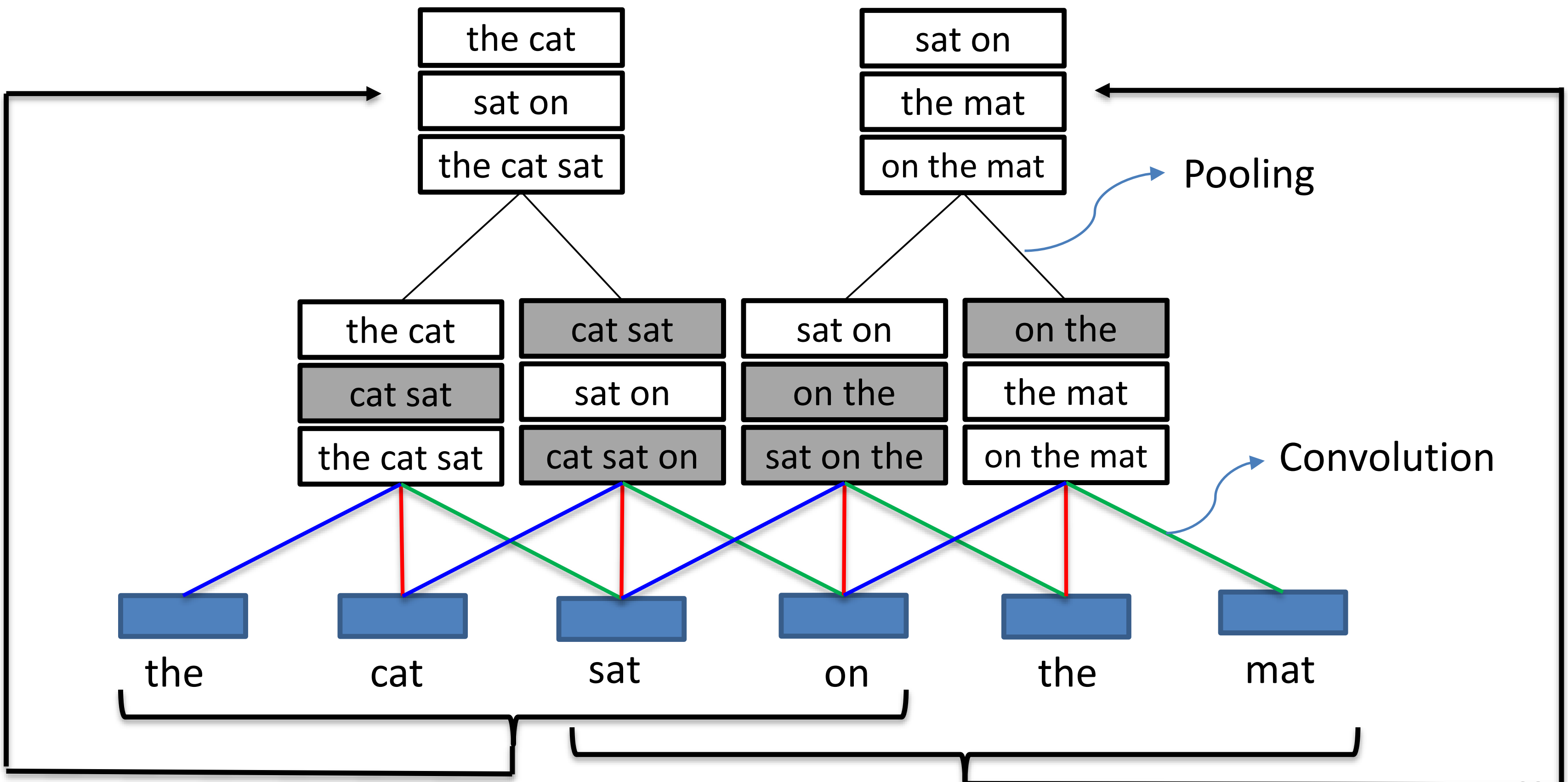
How to Capture Order Information?

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



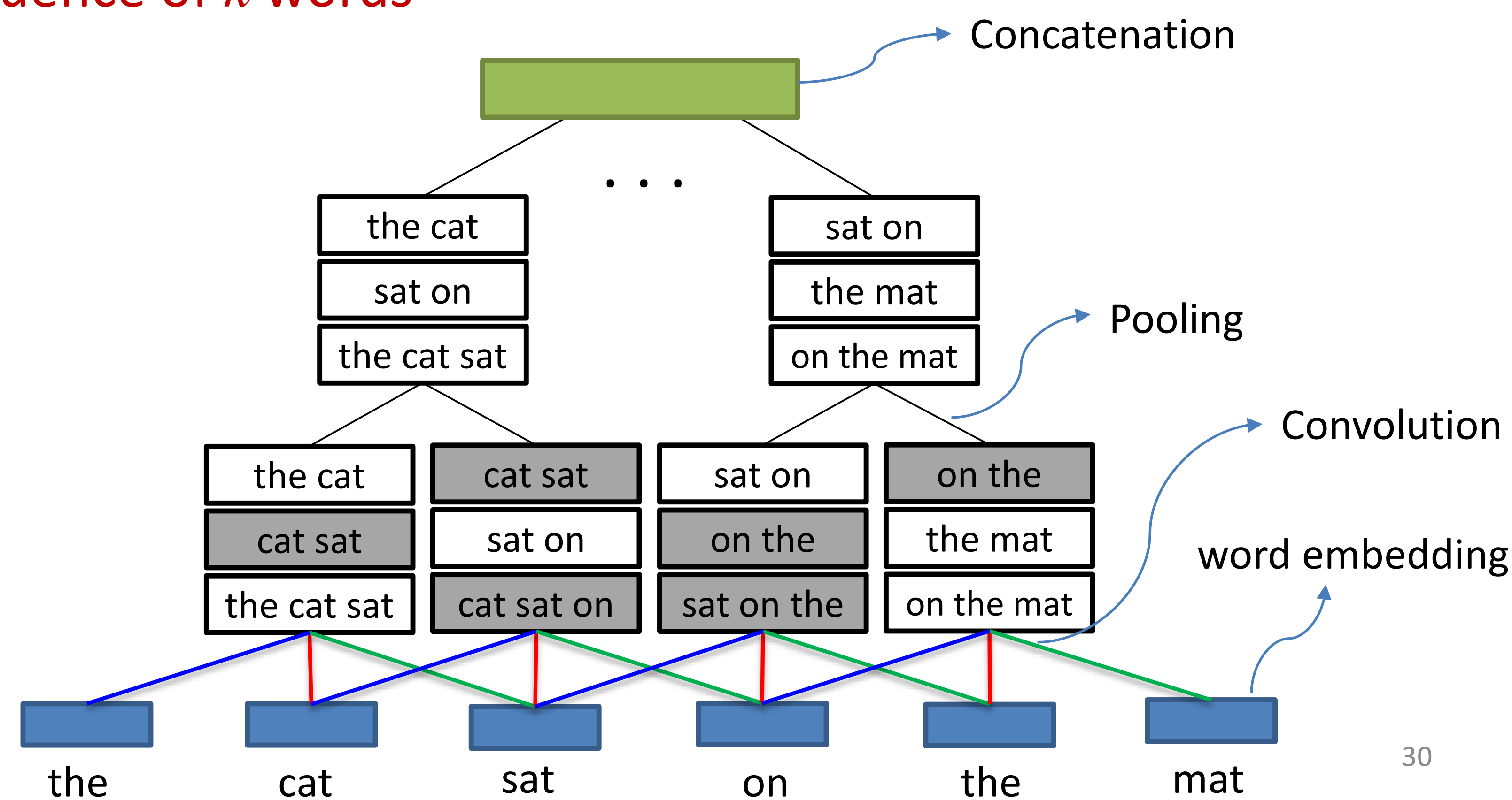
CNN can Keep the Order Information

1-D convolution and pooling operations can keep the word order information



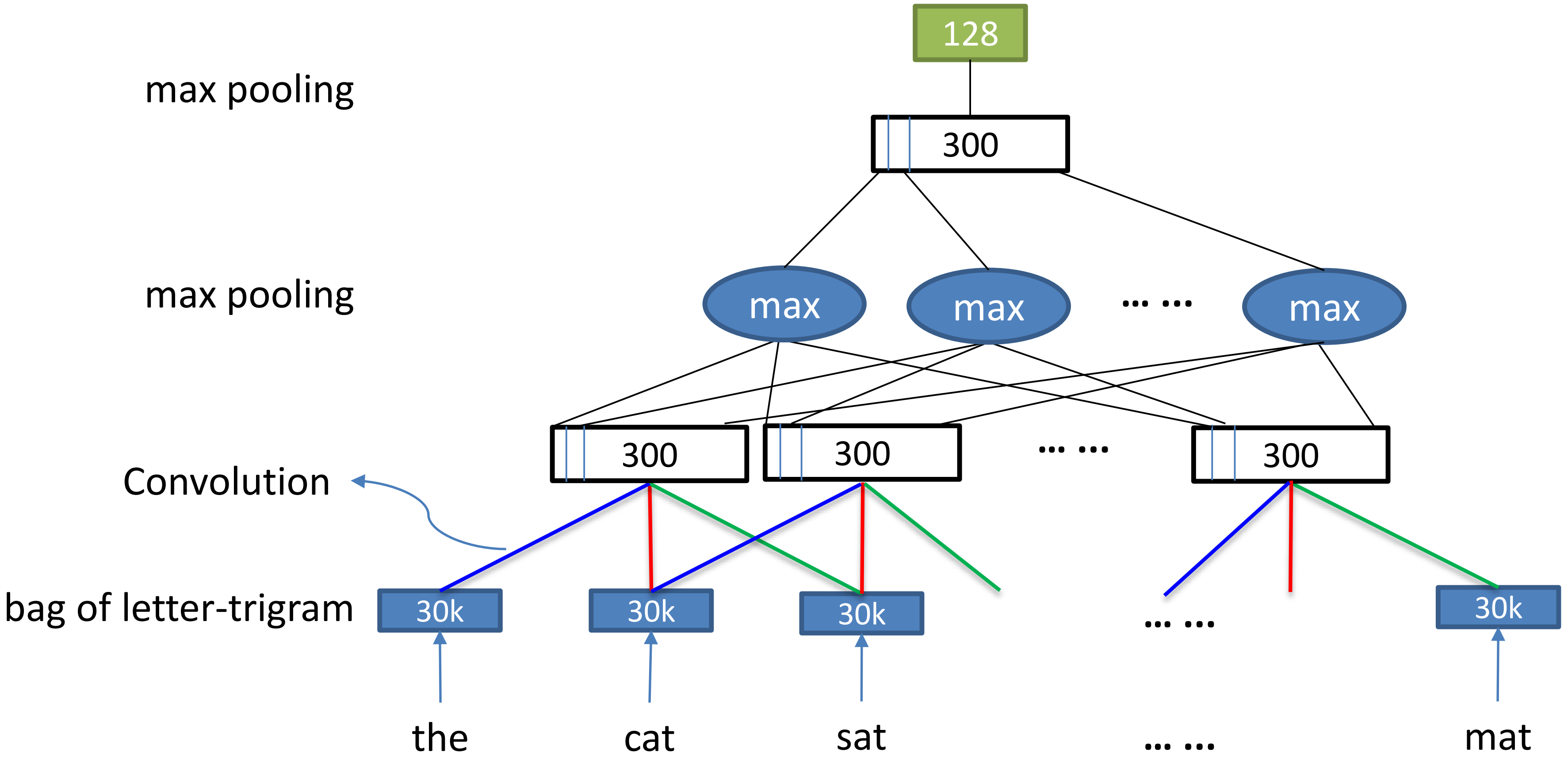
Using CNN: ARC-I (Hu et al., 2014) and CNTN (Qiu et al., 2015)

- Input: sequence of word embeddings trained on a large dataset
- Model: the convolutional operation in CNN compacts each sequence of k words

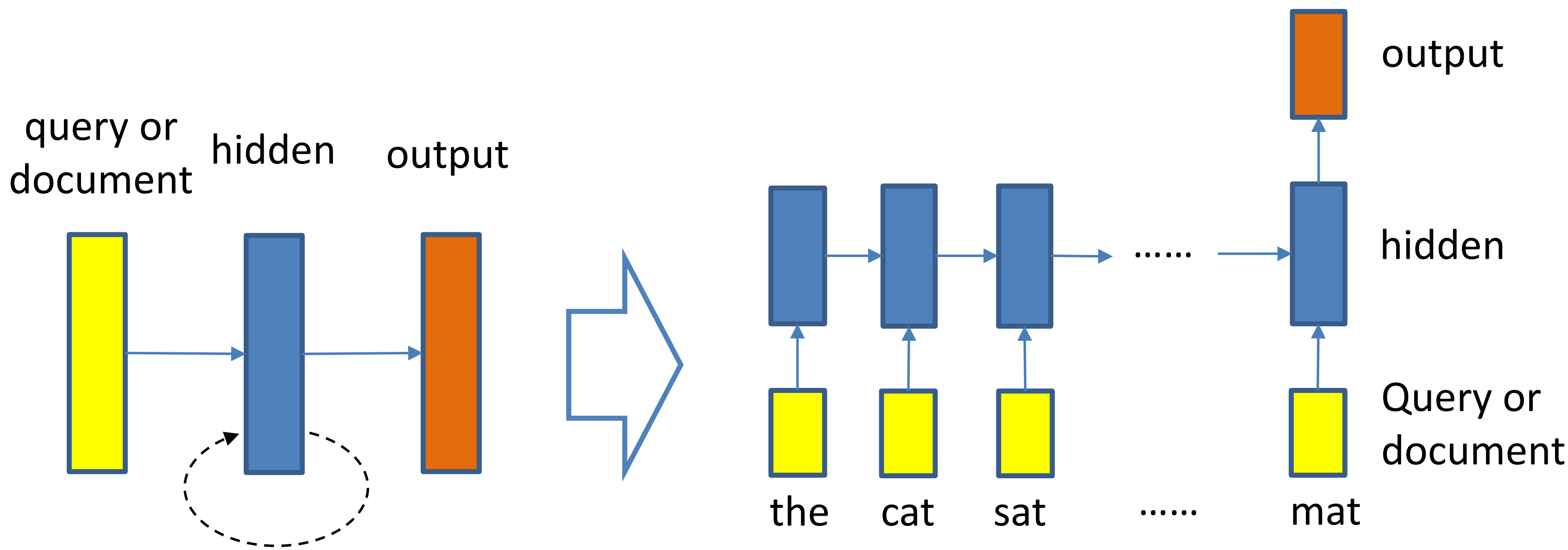


Using CNN: CDSSM (Shen et al., ’14)

The convolutional operation in CNN compacts each sequence of k words



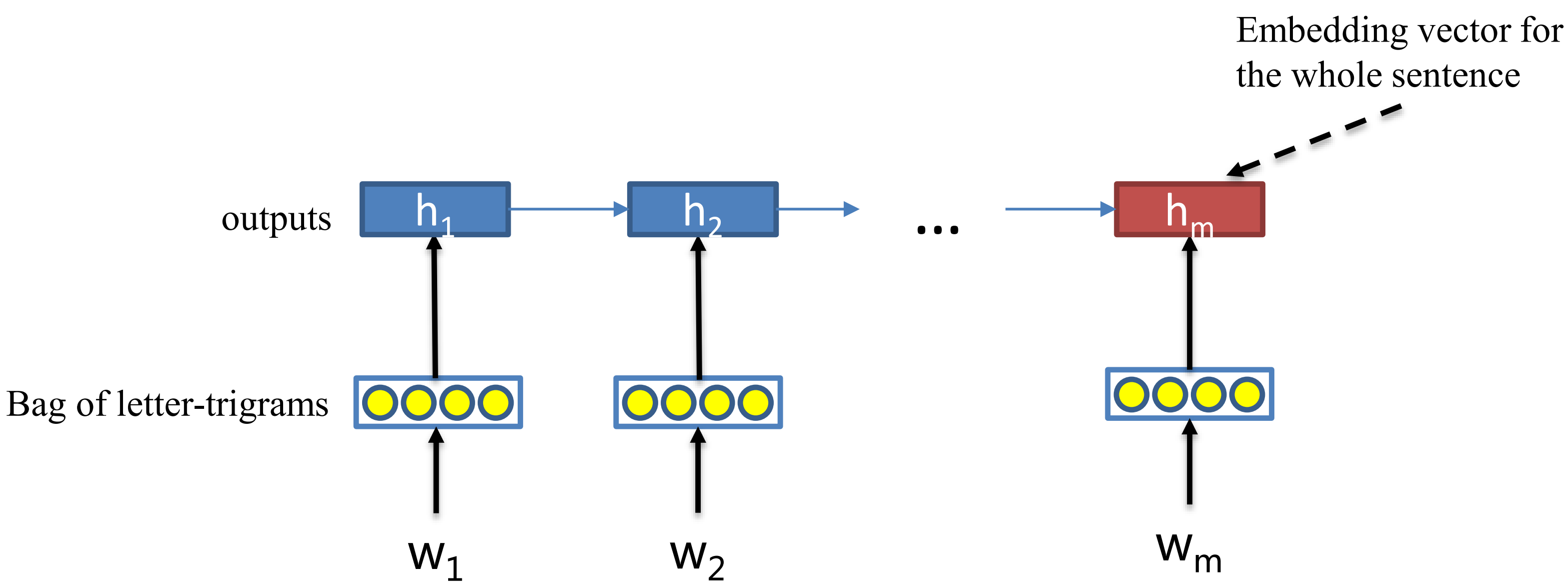
RNN can Keep the Order Information



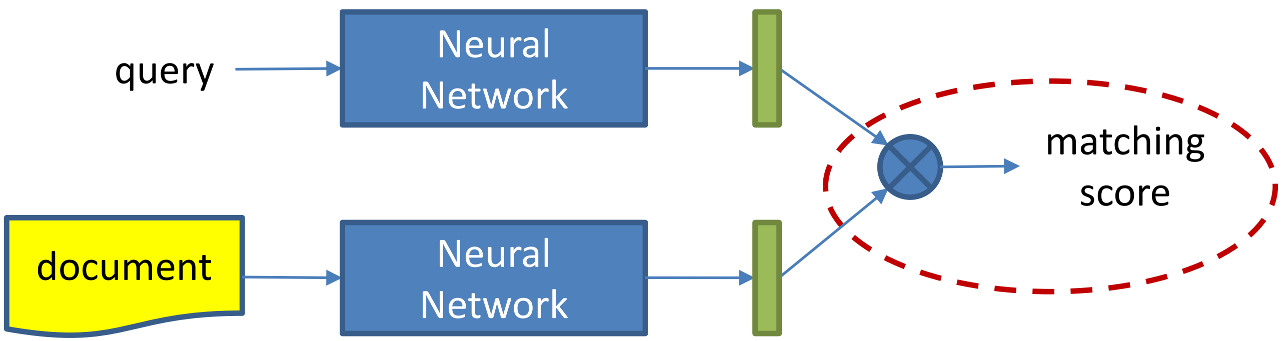
- Two popular variations: long-short term memory (LSTM) and gated recurrent unit (GRU)

Using RNN: LSTM-RNN (Palangi et al., '16)

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



Matching Function



- **Heuristic**: Cosine, Dot product
- **Learning**: MLP, Neural tensor networks

Matching Functions (cont')

- Given representations of query and document : q and d
- Similarity between these two representations:
  - Cosine Similarity (DSSM, CDSSM, RNN-LSTM)

$$s = \frac{q^T \cdot d}{|q| \cdot |d|}$$

  - Dot Product

$$s = q^T \cdot d$$

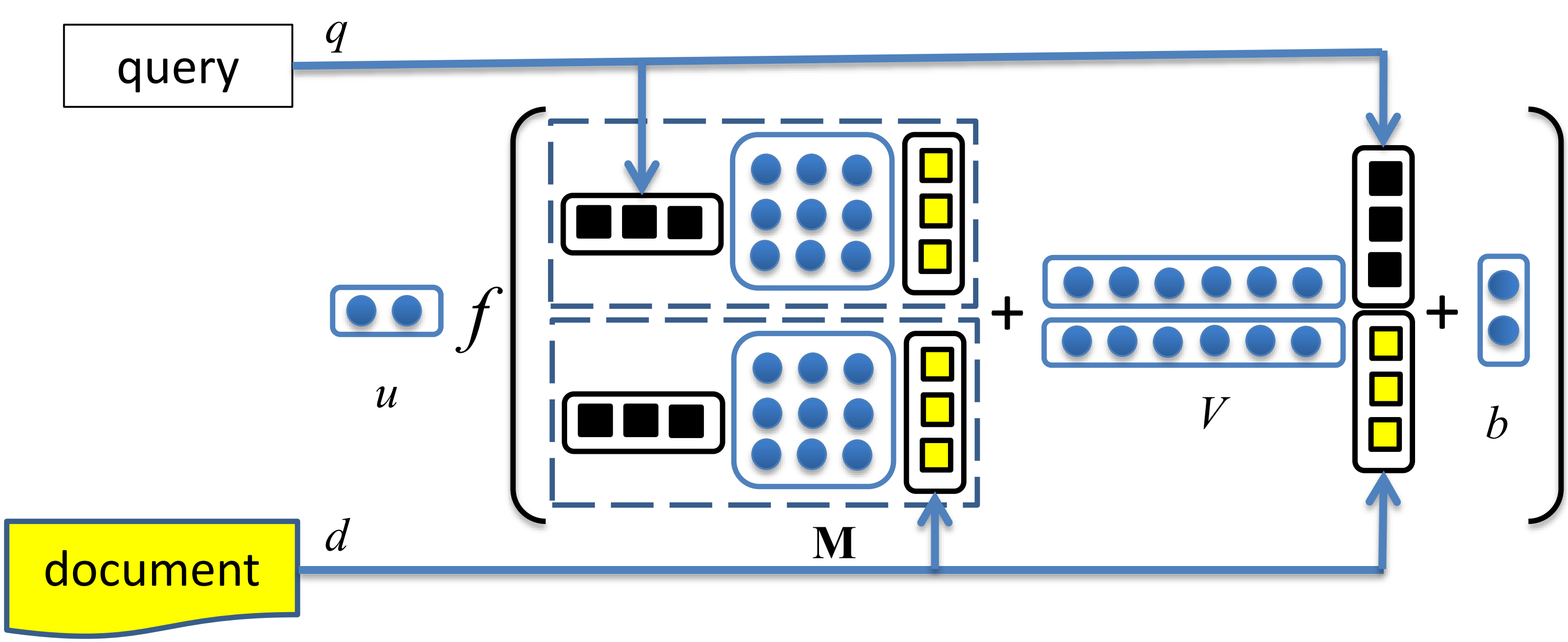
  - Multi-Layer Perception (ARC-I)

$$s = W_2 \cdot \sigma\left(W_1 \cdot \left[\begin{array}{c} q \\ d \end{array}\right] + b_1\right) + b_2$$

Matching Functions (cont')

- Neural Tensor Networks (CNTN) (Qiu et al., IJCAI '15)

$$s = u^T f\left(q^T \mathbf{M}^{[1:r]} d + V \begin{bmatrix} q \\ d \end{bmatrix} + b\right)$$

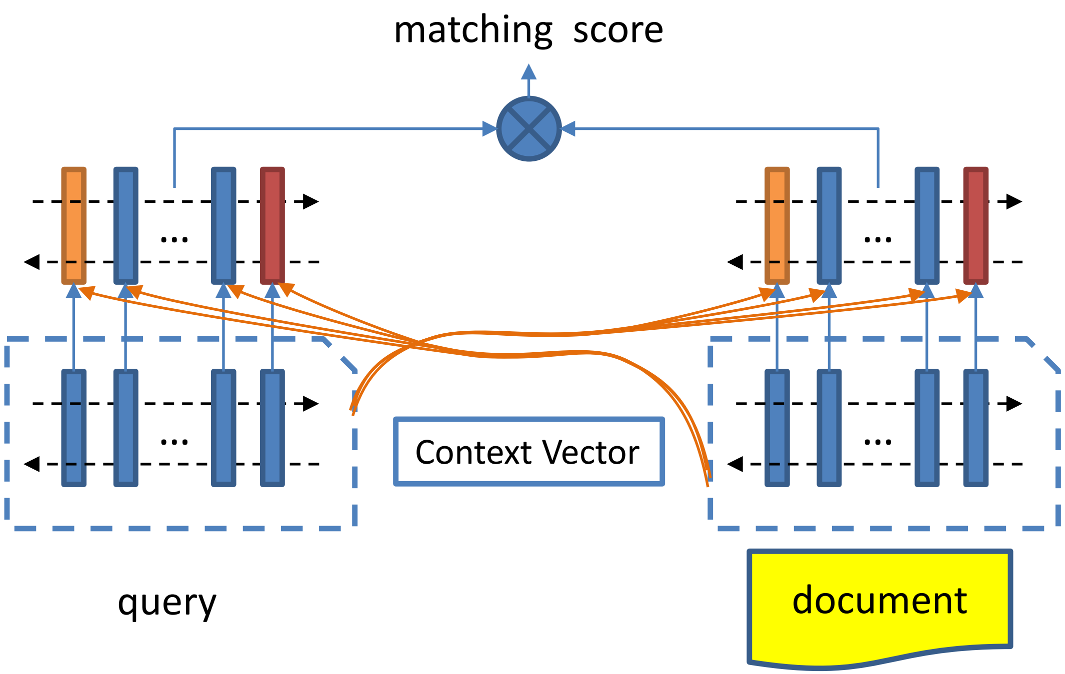


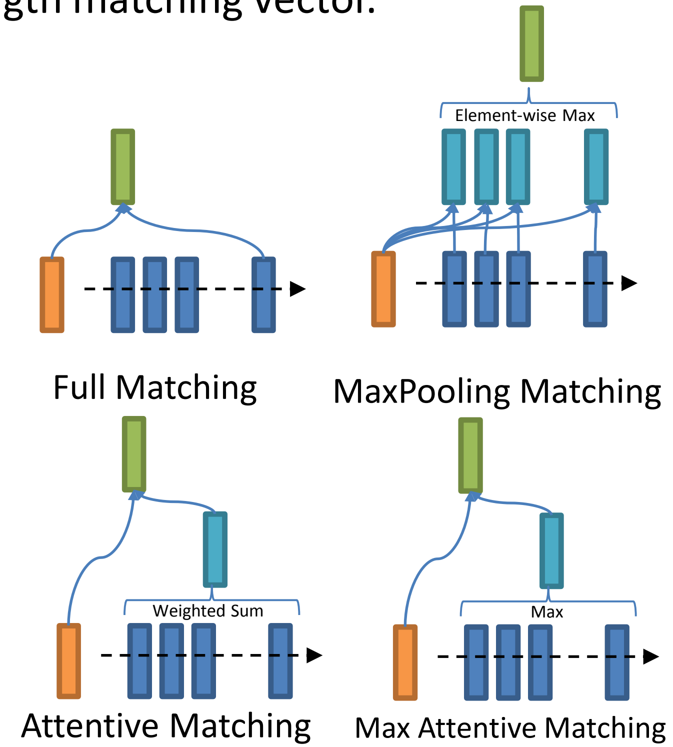
Extensions to Representation Learning Methods

- Problem: context information from the other sentence is not used during the representation generation
- Solution: rep. of the document based on the rep. of query,

BiMPM (Wang et al., IJCAI '17), CA-RNN (Chen et al., AAAI '18)

- Step 1: multiple perspectives context vector of one text is matched against all timesteps of the other.
- Step 2: aggregate the matching results into a fixed-length matching vector.



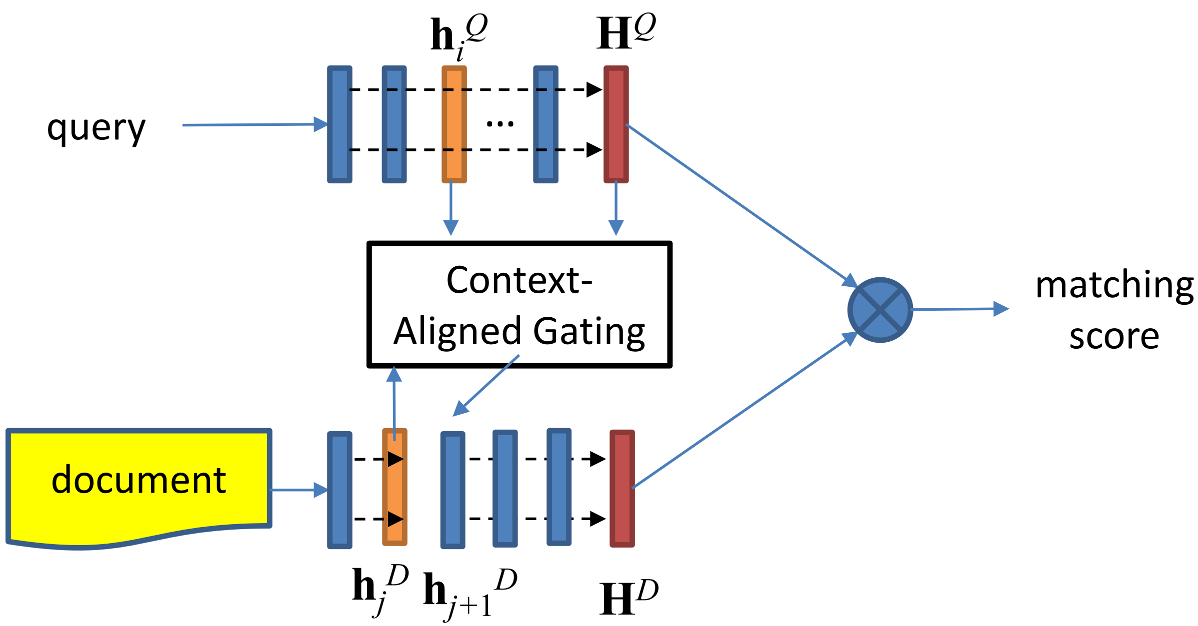


Extensions to Representation Learning Methods

- Problem: context information from the other sentence is not used during the representation generation
- Solution: rep. of the document based on the rep. of query,

BiMPM (Wang et al., IJCAI '17), **CA-RNN** (Chen et al., AAAI '18)

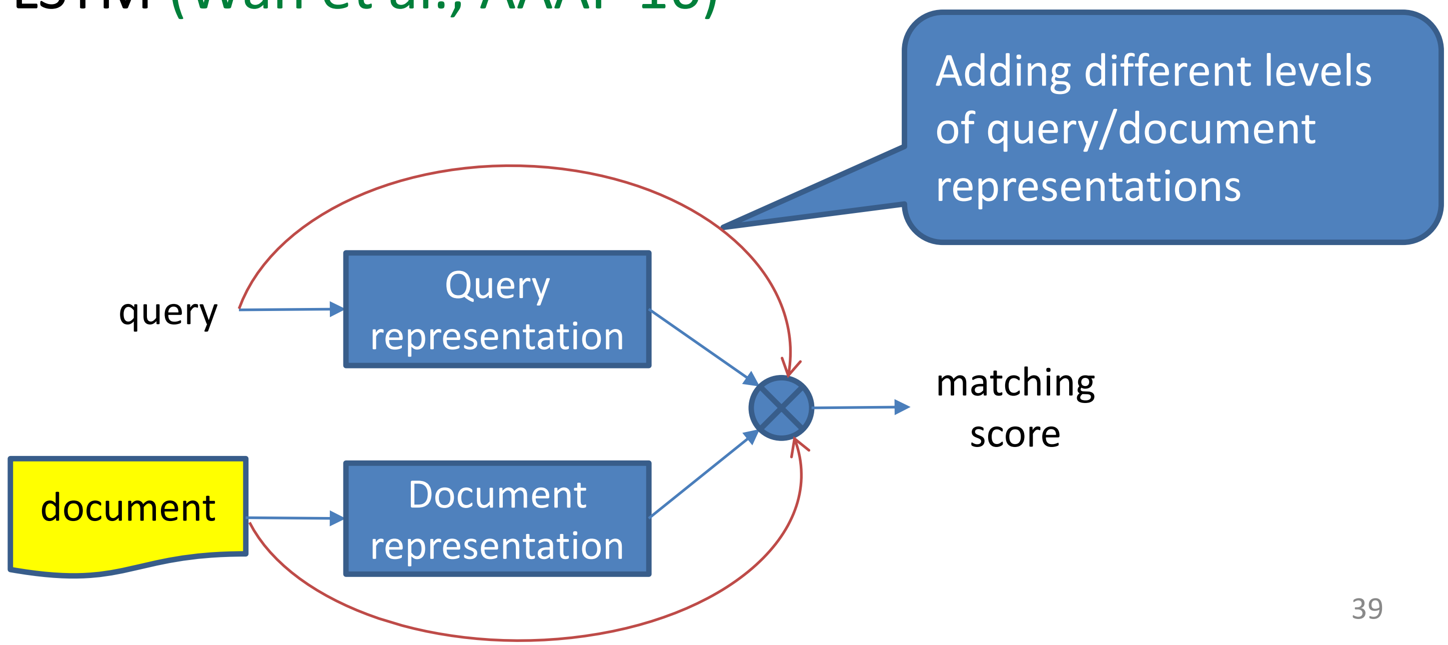
  - Step 1: Word alignment to identify the aligned words in two sentences
  - Step 2: Context alignment gating to absorb the context



Extensions to Representation Learning Methods (cont')

- Problem: representations are too coarse to conduct text match
  - Experience in IR: combining topic-level and word-level matching signals usually achieve better performances
- Solution: add fine-grained signals,

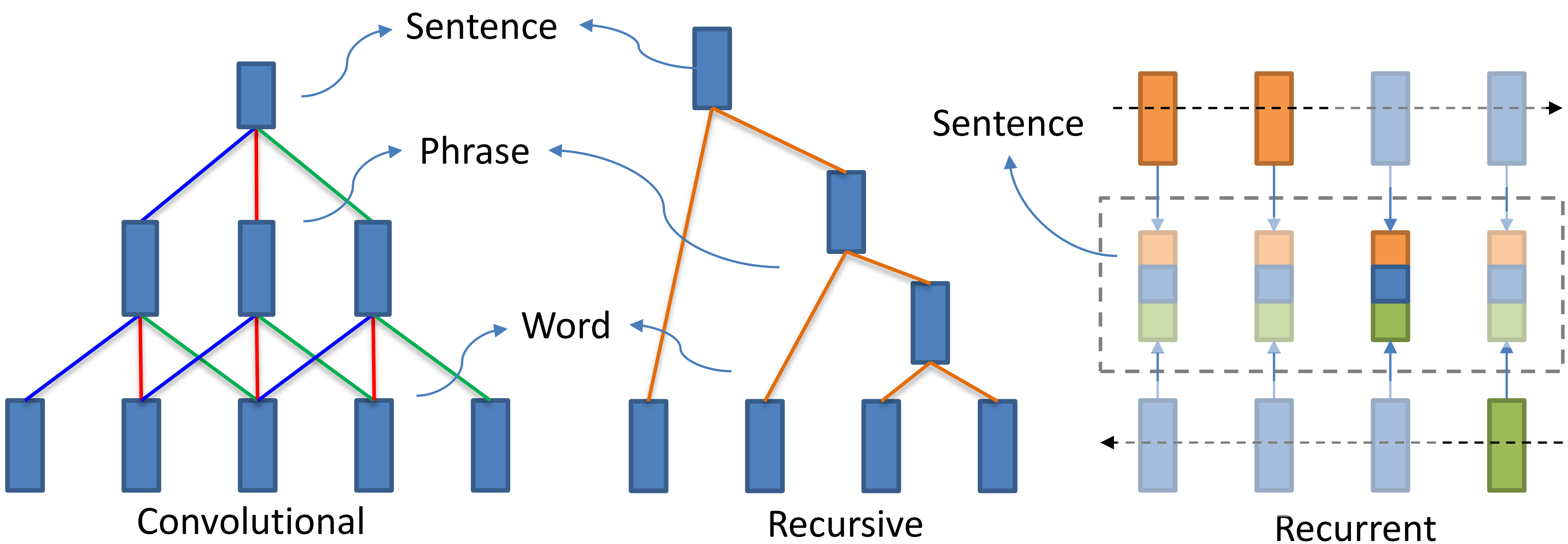
  include MultGranCNN(Yin et al., ACL '15), U-RAE (Socher et al., NIPS '11), MV-LSTM (Wan et al., AAAI '16)



Extensions to Representation Learning Methods (cont’)

- Problem: representations are too coarse to conduct text match
  - Experience in IR: combining topic-level and word-level matching signals usually achieve better performances
- Solution: add fine-grained signals,

  include MultGranCNN(Yin et al., ACL ‘15), U-RAE (Socher et al., NIPS ‘11), MV-LSTM (Wan et al., AAAI ‘16)



Experimental Results

	Model	P@1	MRR
Traditional methods	BM25	0.579	0.726
Representation learning for matching	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

Based on Yahoo! Answers dataset (60,564 question-answer pairs)

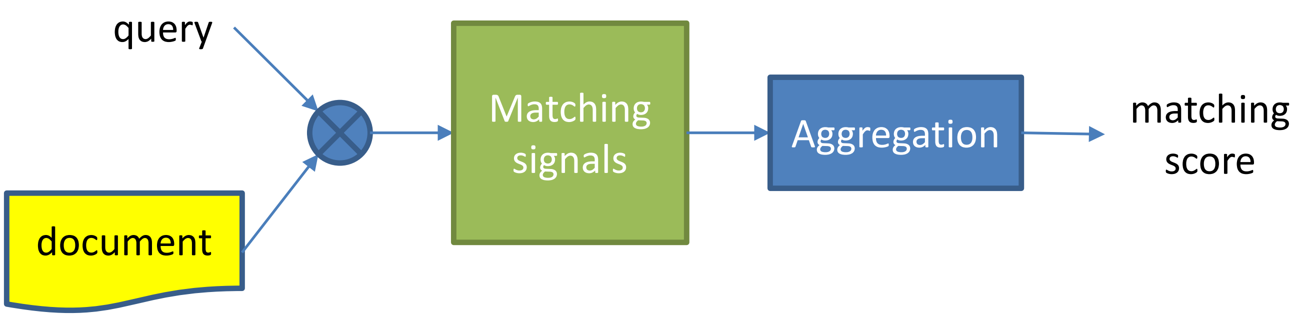
- Representation learning methods outperformed baselines
  - Semantic representation is important
- LSTM-RNN performed better than ARC-I and CNTN
  - Modeling the order information does help
- MultiGranCNN and MV-LSTM are the best performing methods
  - Fine-grained matching signals are useful

Short Summary

- Two steps
  - 1. Calculate representations for query and document
  - 2. Conduct matching
- Representations for query and document
  - Using DNN
  - Using CNN and RNN to capture order information
  - Representing one sentence using the other as context
- Matching function
  - Dot product (cosine similarity)
  - Multi-layer Perceptron
  - Neural tensor networks

Outline

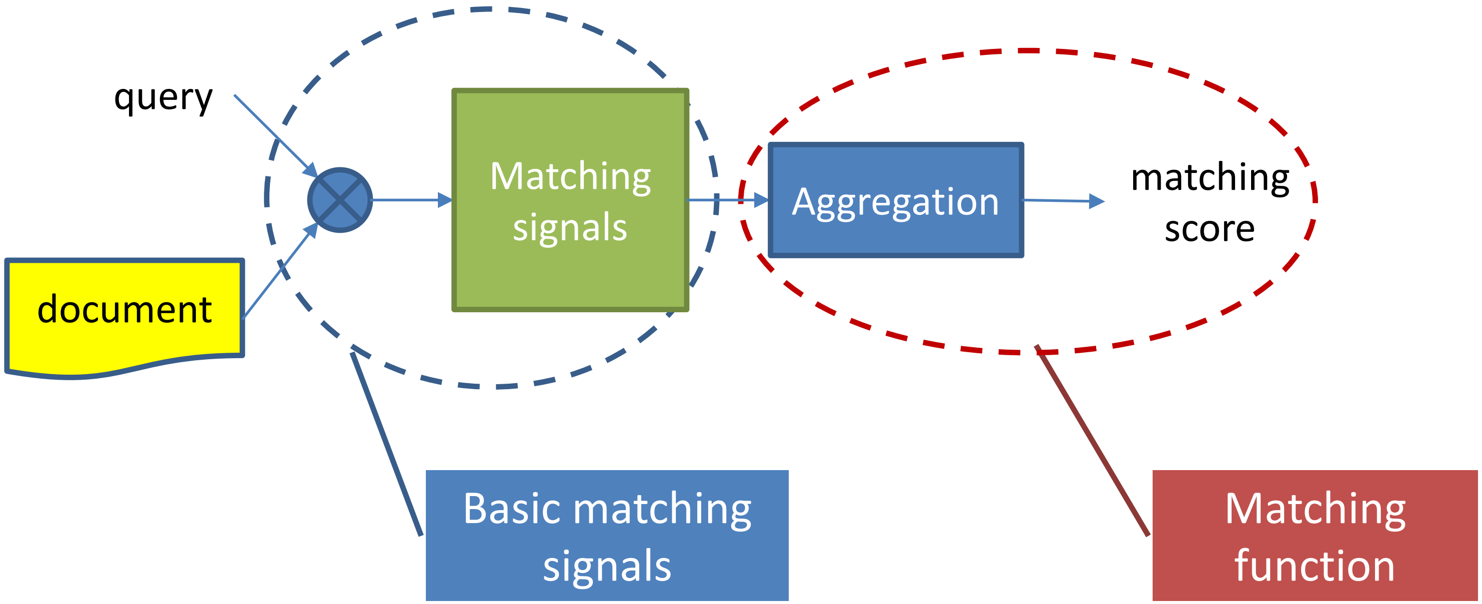
- Introduction
- Deep Semantic Matching
  - Methods of Representation Learning
  - Methods of Matching Function Learning
- Reinforcement Learning to Rank
  - Formulation IR Ranking with RL
  - Approaches
- Summary



METHODS OF MATCHING FUNCTION LEARNING

Matching Function Learning

- Step 1: construct basic low-level matching signals
- Step 2: aggregate matching patterns



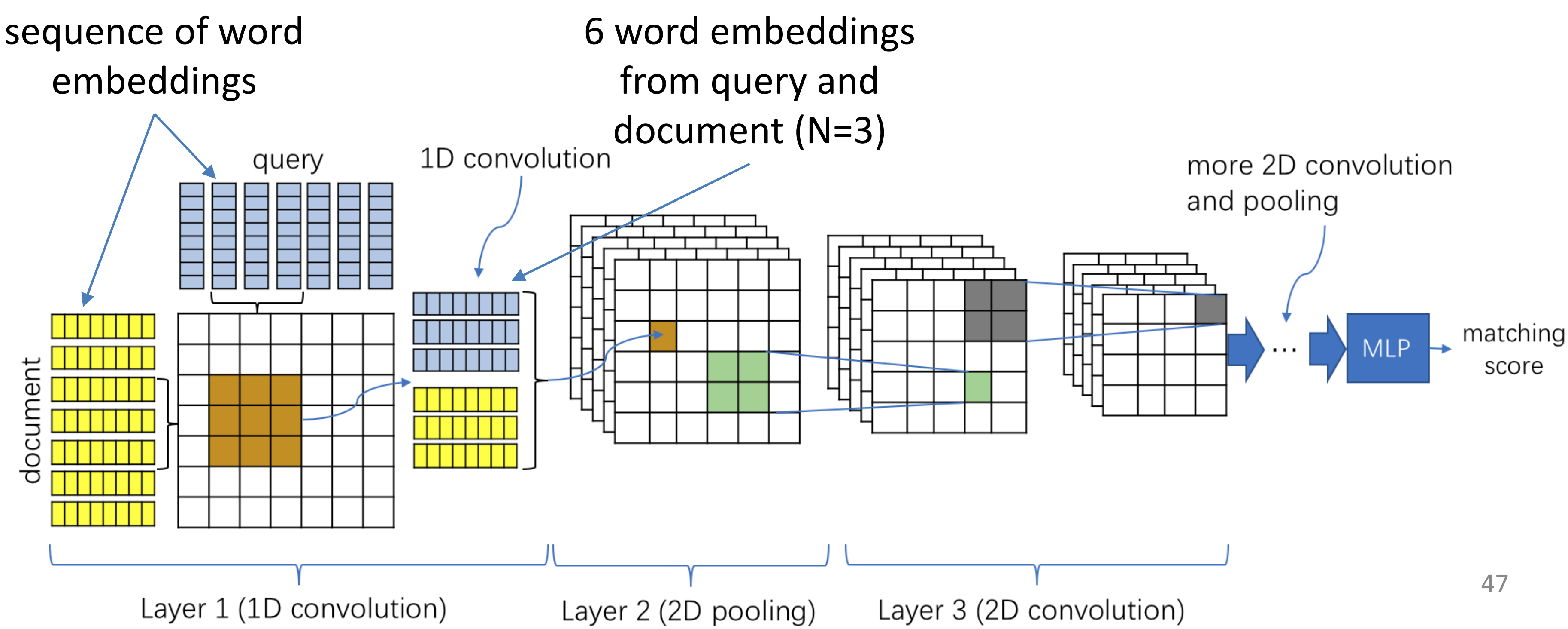
Typical Matching Function Learning Methods

- For short text (e.g., sentence) similarity matching
  - ARC II (Hu et al., NIPS ’14)
  - MatchPyramid (Pang et al., AAAI ’16)
  - Match-SRNN (Wan et al., IJCAI ’16)

- For query-document relevance matching
  - DRMM (Guo et al., CIKM ’16) and aNMM (Yang et al., CIKM ‘16)
  - K-NRM (Xiong et al., SIGIR ‘17) and Conv-KNRM (Dai et al., WSDM ‘18)
  - DeepRank (Pang et al., CIKM ‘17) and PACRR (Hui et al., EMNLP ’17)
  - DUET (Mitra et al., WWW ‘17)

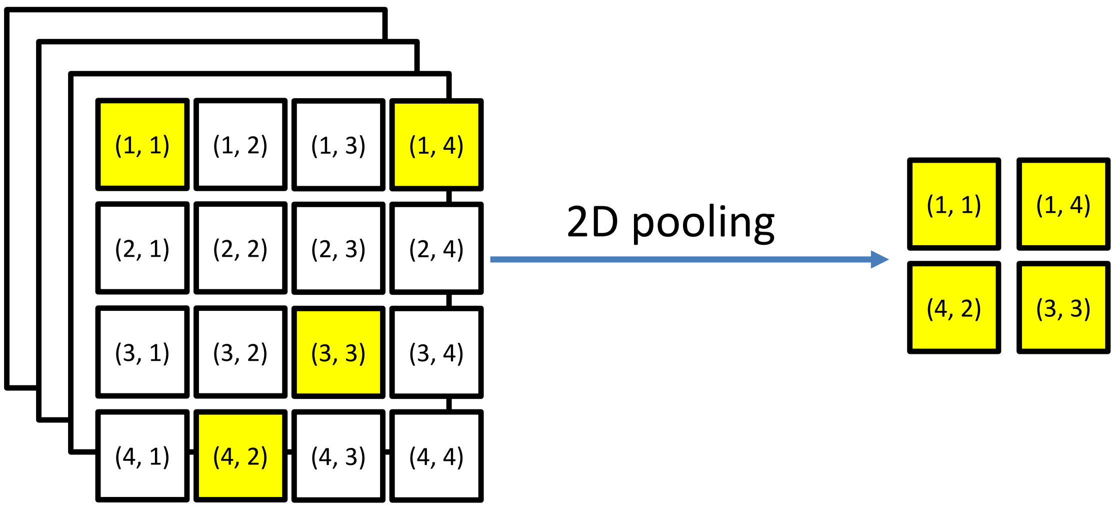
ARC-II (Hu et al., NIPS '14)

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



ARC-II (cont')

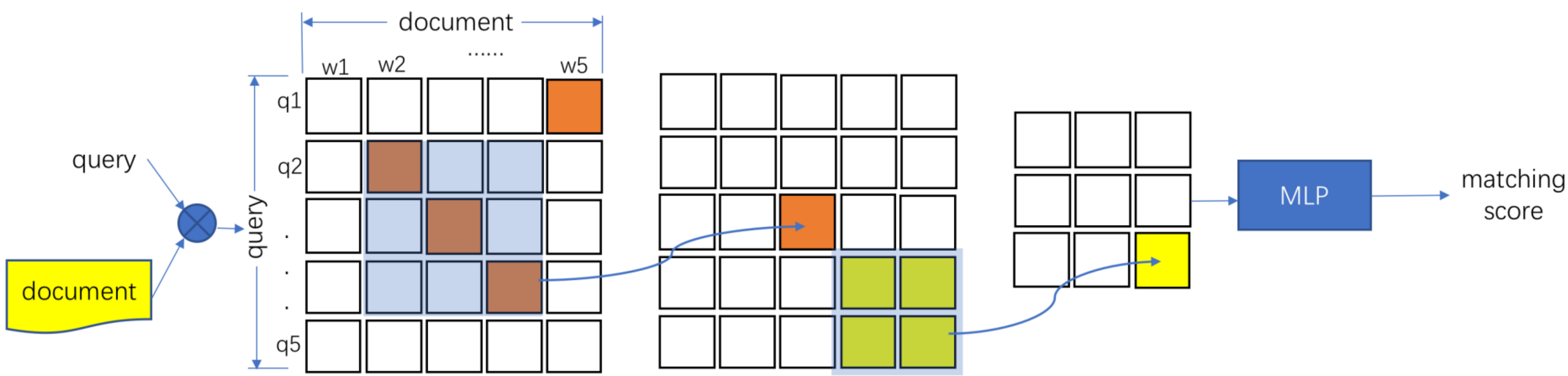
- Keeping word order information
  - Both the convolution and pooling are order preserving



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

MatchPyramid (Pang et al., AAAI '16)

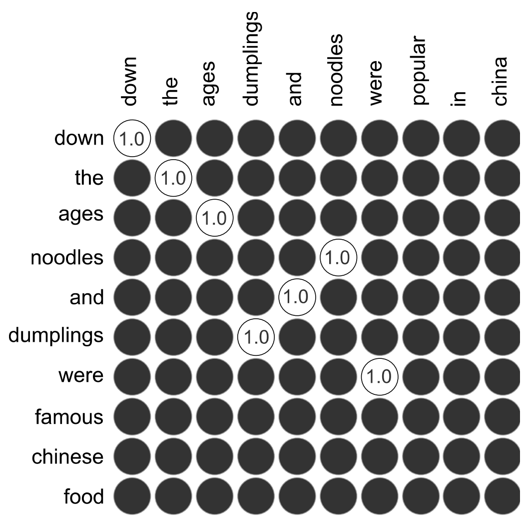
- Inspired by image recognition
- Basic matching signals: word-level matching matrix
- Matching function: 2D convolution + MDP



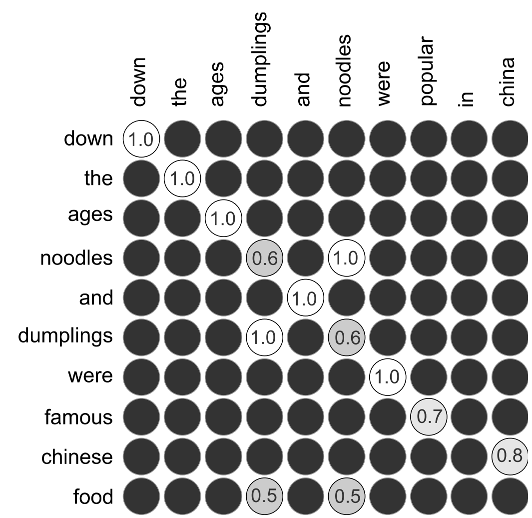
Word matching matrix 2D convolution pooling

Matching Matrix: Basic Matching Signals

- Each entry calculated based on
  - Word-level exact matching (0 or 1)
  - Semantic similarity based on embeddings of words



Exact match

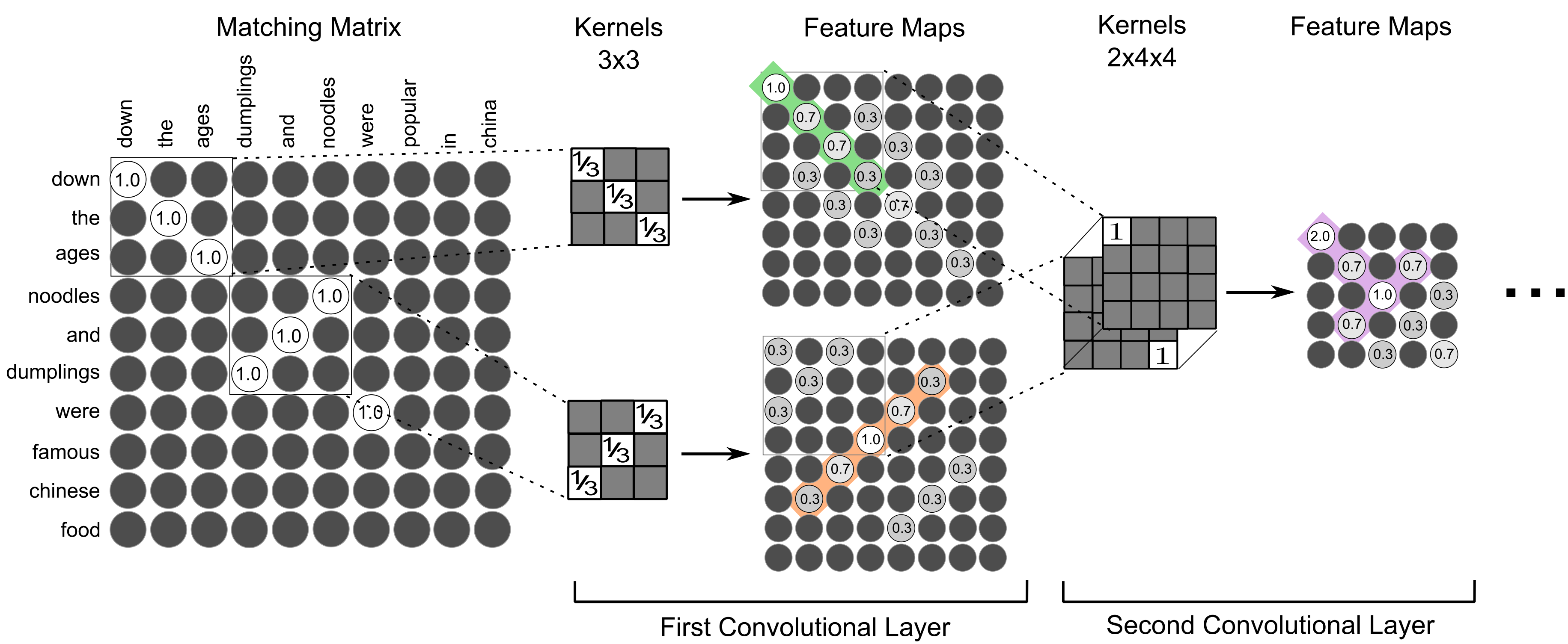


Cosine similarity

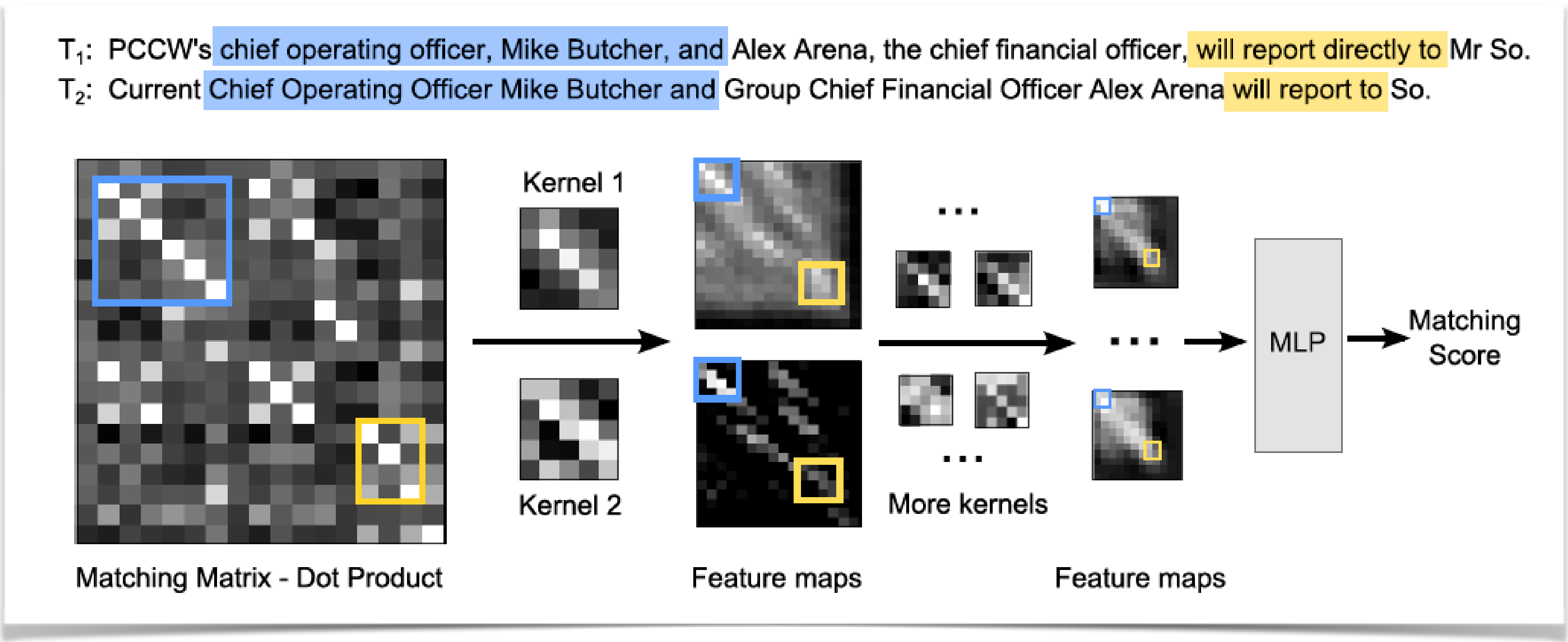
- Positions information of words is kept

Matching Function: 2D Convolution

- Discovering the matching patterns with CNN, stored in the kernels

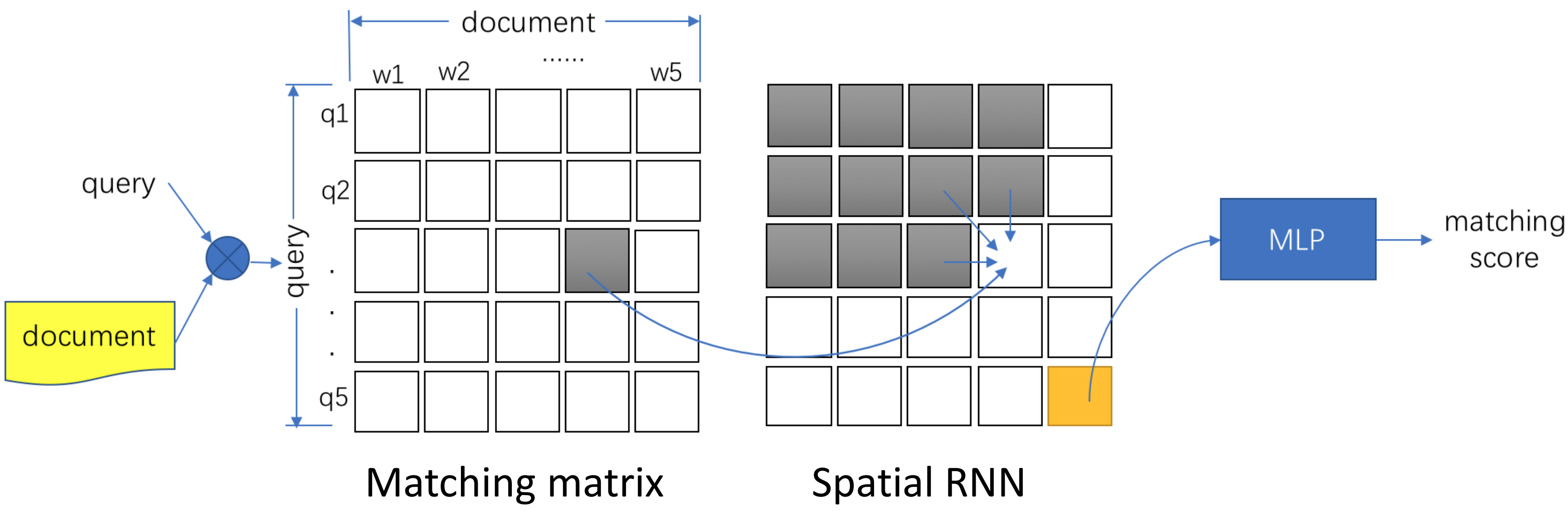


Discovered Matching Patterns

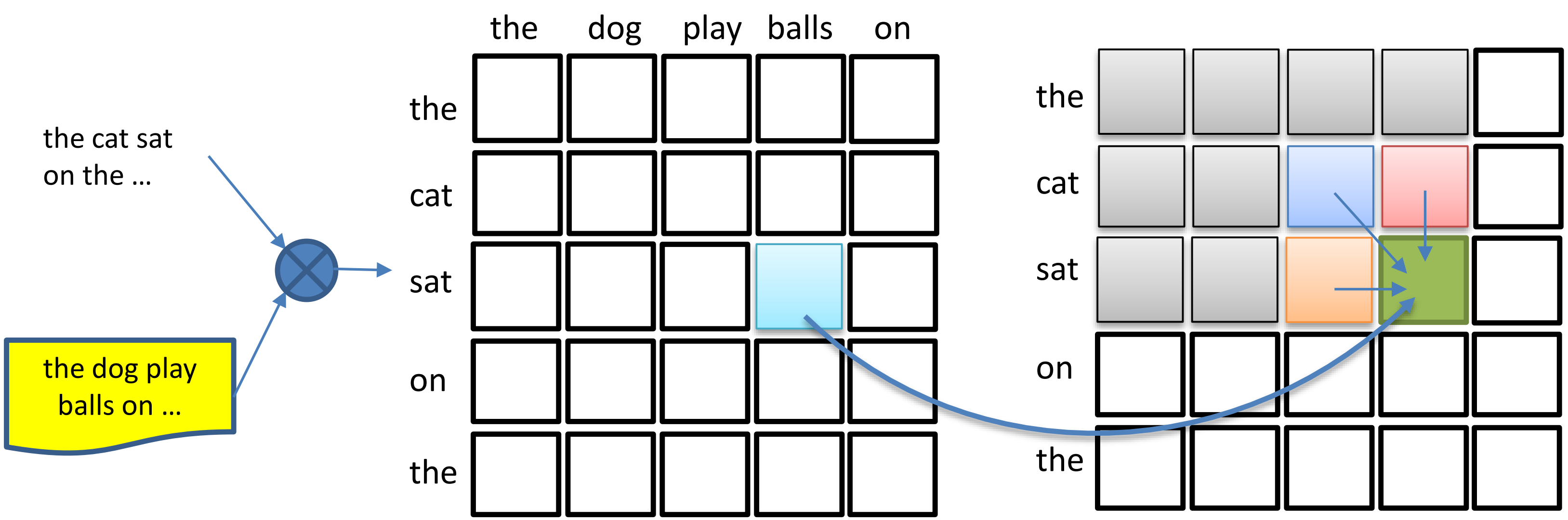


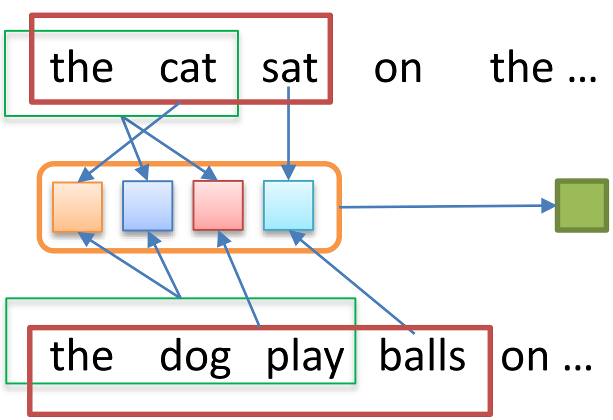
Match-SRNN (Wan et al., IJCAI '16)

- Based on spatial recurrent neural network (SRNN)
- Basic matching signals: word-level matching matrix
- Matching function: Spatial RNN + MLP



Match-SRNN: Recursive Matching Structure

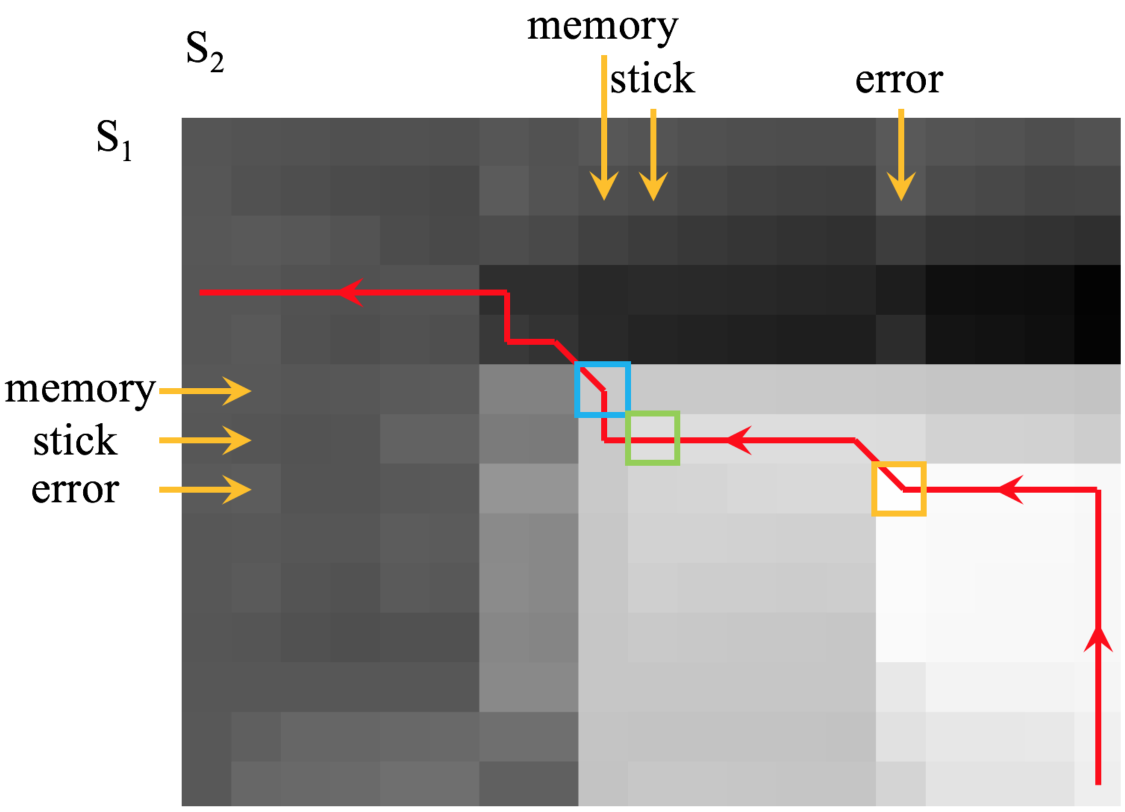




- Calculated recursively (from top left to bottom right)
- All matching signals between the prefixes been utilized
  - **Current position**: sat <—> balls
  - **Substrings**:
    - the cat <—> the dog play
    - the cat <—> the dog play balls
    - the cat sat <—> the dog play

Visualized Matching Signal Aggregation

- 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."*

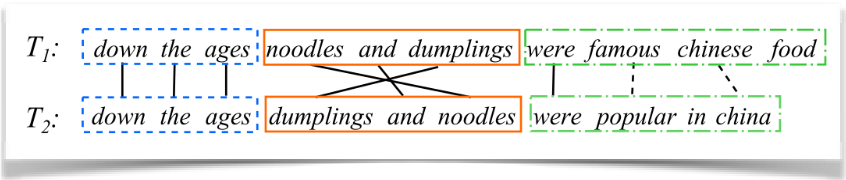


Short Summary

- Two steps
  - 1. Construct basic matching signals
  - 2. Aggregate matching patterns
- Basic matching signals
  - Matching matrix (based on exact match, dot product, or/and cosine similarity)
- Aggregate matching patterns
  - CNN/Spatial RNN + MLP
  - Kernel pooling + nonlinear combination
  - Feed forward networks

Similarity ≠ Relevance

(Pang et al., Neu-IR workshop '16)



Similarity matching

- Whether two sentences are semantically similar
- Homogeneous texts with comparable lengths
- Matches at all positions of both sentences
- Symmetric matching function
- Representative task: Paraphrase Identification

Relevance matching

- Whether a document is relevant to a query
- Heterogeneous texts (keywords query, document) and very different in lengths
- Matches in different parts of documents
- Asymmetric matching function
- Representative task: ad-hoc retrieval

Relevance Matching ?

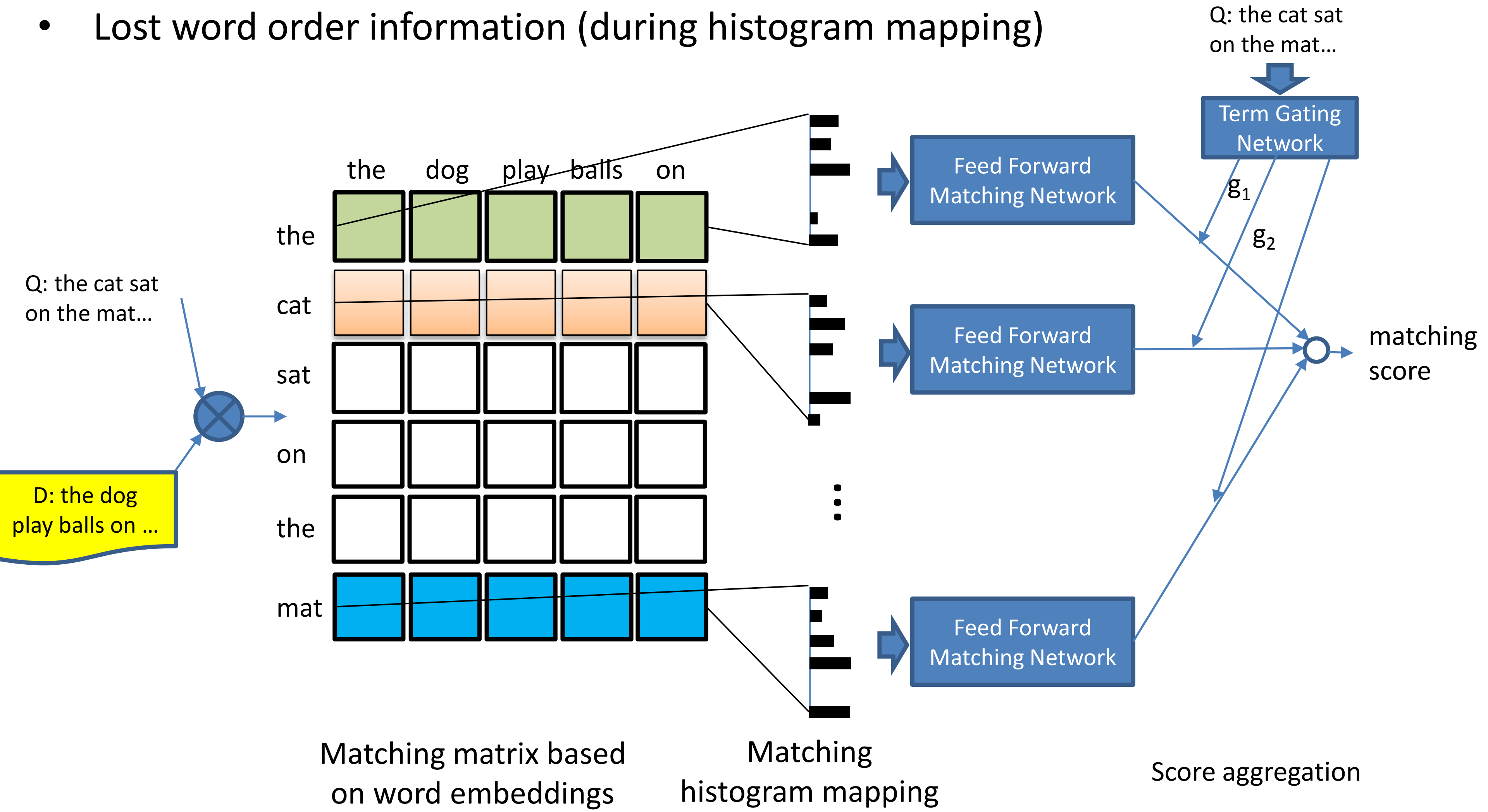
- Global Distribution of Matching Signals
  - DRMM (Guo et al., CIKM ’16) and aNMM (Yang et al., CIKM ‘16)
  - K-NRM (Xiong et al., SIGIR ‘17) and Conv-KNRM (Dai et al., WSDM ‘18)
- Local Context of Matching Positions
  - DeepRank (Pang et al., CIKM ‘17) and PACRR (Hui et al., EMNLP ’17)
- Others
  - DUET (Mitra et al., WWW ‘17)

Relevance Matching based on Global Distribution of Matching Signals

- Step 1: calculate matching signals for each query term
- Step 2: statistic each query term's matching signal distributions
- Step 3: aggregate the distributions
- Pros
  - Matching between short query text and long document text
  - Robust: matching signals from irrelevant document words
- Cons: lost term order information

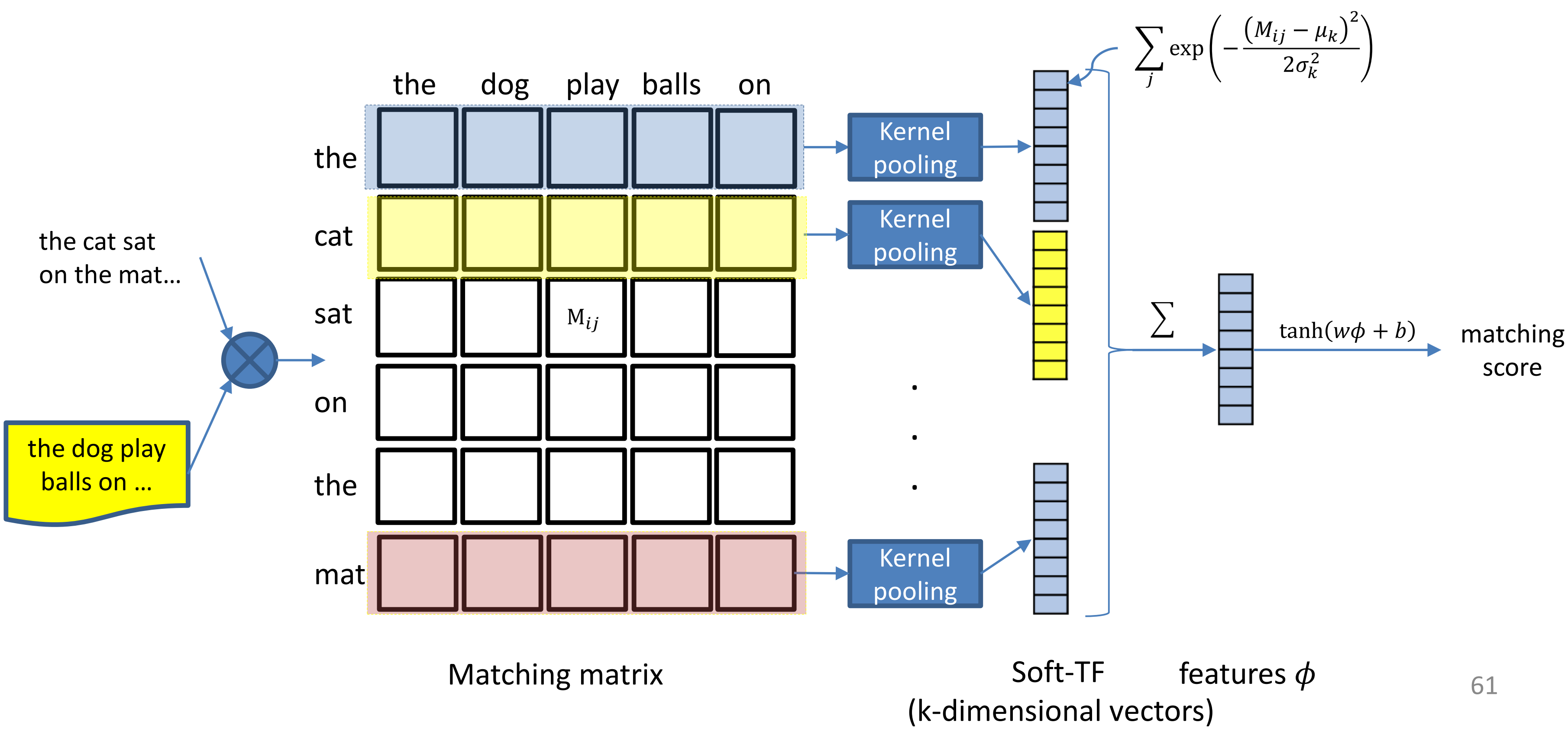
Deep Relevance Matching Model (DRMM) (Guo et al., CIKM '16)

- Matching histogram mapping for summarizing each query matching signals
- Term gating network for weighting the query matching signals
- Lost word order information (during histogram mapping)



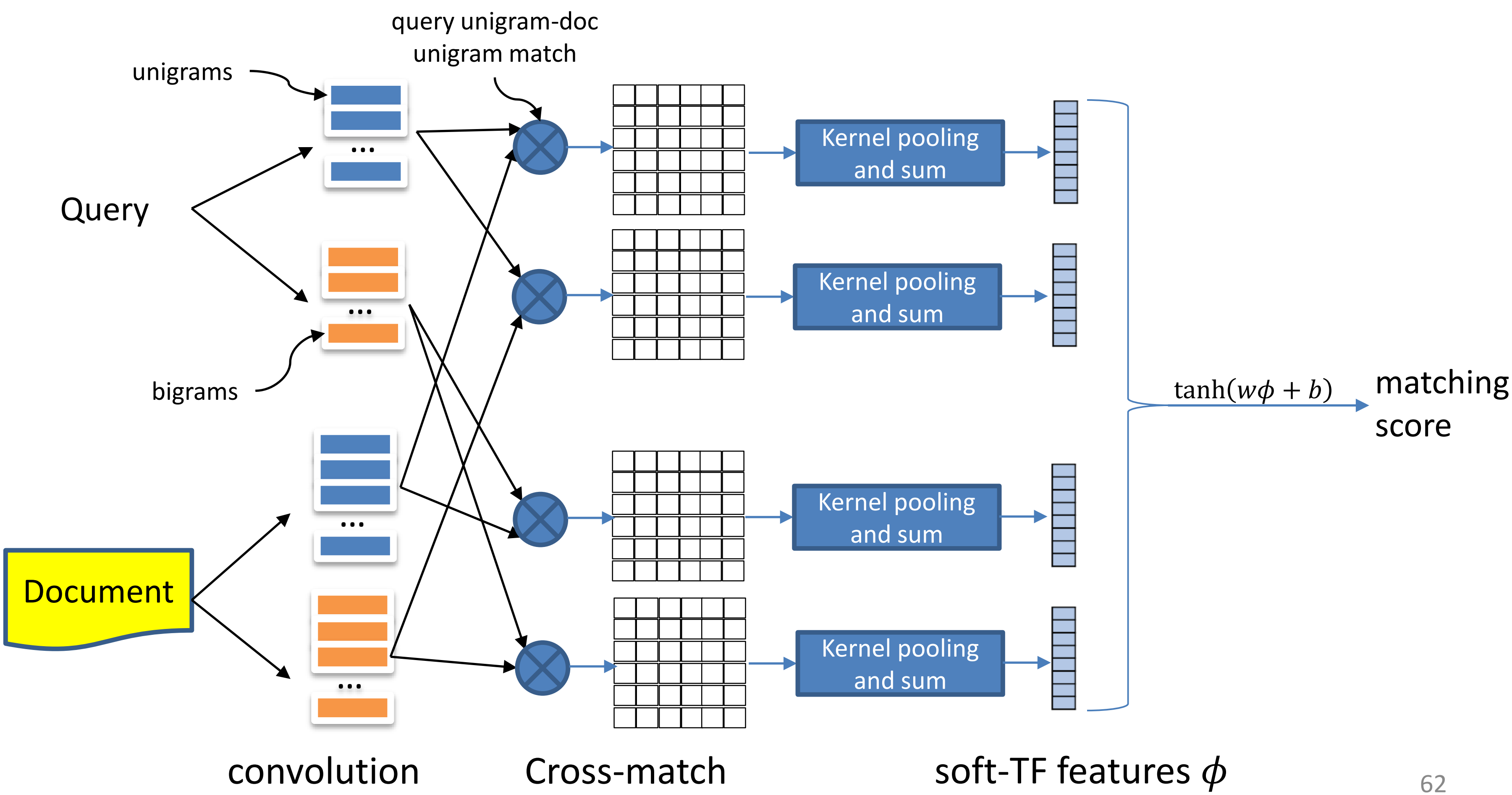
K-NRM: Kernel Pooling as Matching Function (Xiong et al., SIGIR '17)

- Basic matching signals: cosine similarity of word embeddings
- Ranking function: kernel pooling + nonlinear feature combination
- Semantic gap: embedding and soft-TF bridge the semantic gap
- Word order: kernel pooling and sum operations lost order information



Conv-KNRM (Dai et al., WSDM '18)

- Based on KNRM
- N-gram cross-matching to capture the word order information

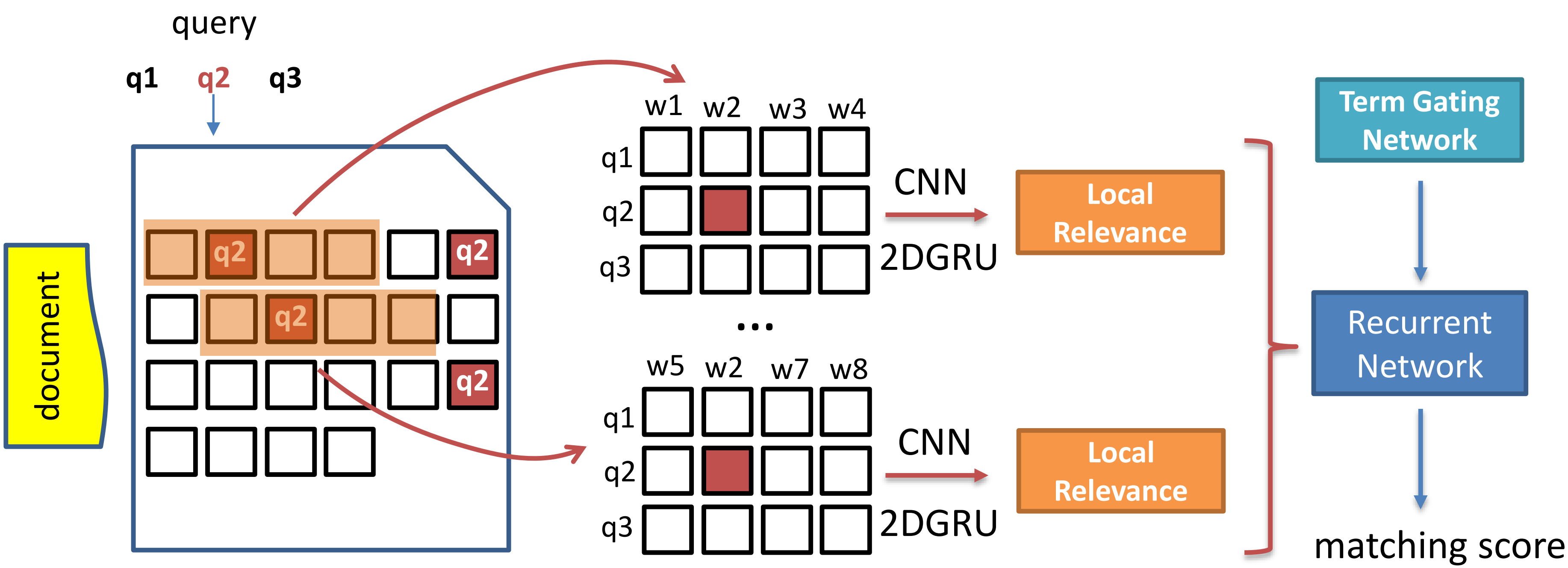


Relevance Matching based on Local Context of Matching Positions

- Step 1: find matching positions for each query term
- Step 2: calculate matching signals within the local context
- Step 3: aggregate the local signals
- Advantages:
  - Matching between short query text and long document text
  - Robust: filtered out irrelevant context
  - Keep order information within the context

DeepRank (Pang et al., CIKM '17)

- Calculate relevance by mimicking the human relevance judgement process



1. Detecting Relevance locations:
focusing on locations of query terms when scanning the whole document

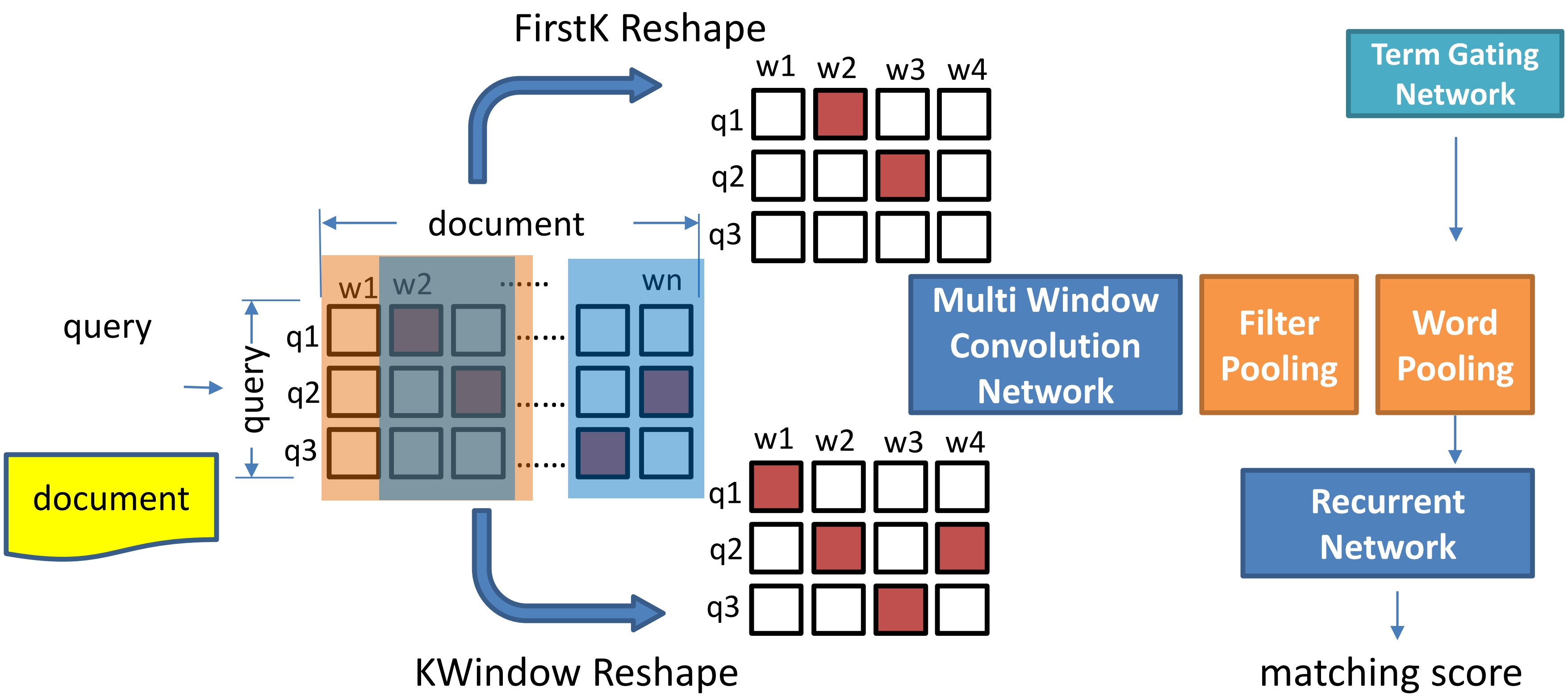
2. Determining local relevance:
relevance between query and each location context, using MatchPyramid/MatchSRNN etc.

3. Matching signals aggregation:

$$F(\mathbf{q},\mathbf{d}) = \sum_{w\in\mathbf{q}} (E_w \mathbb{I})^T \cdot \mathcal{T}(w)$$

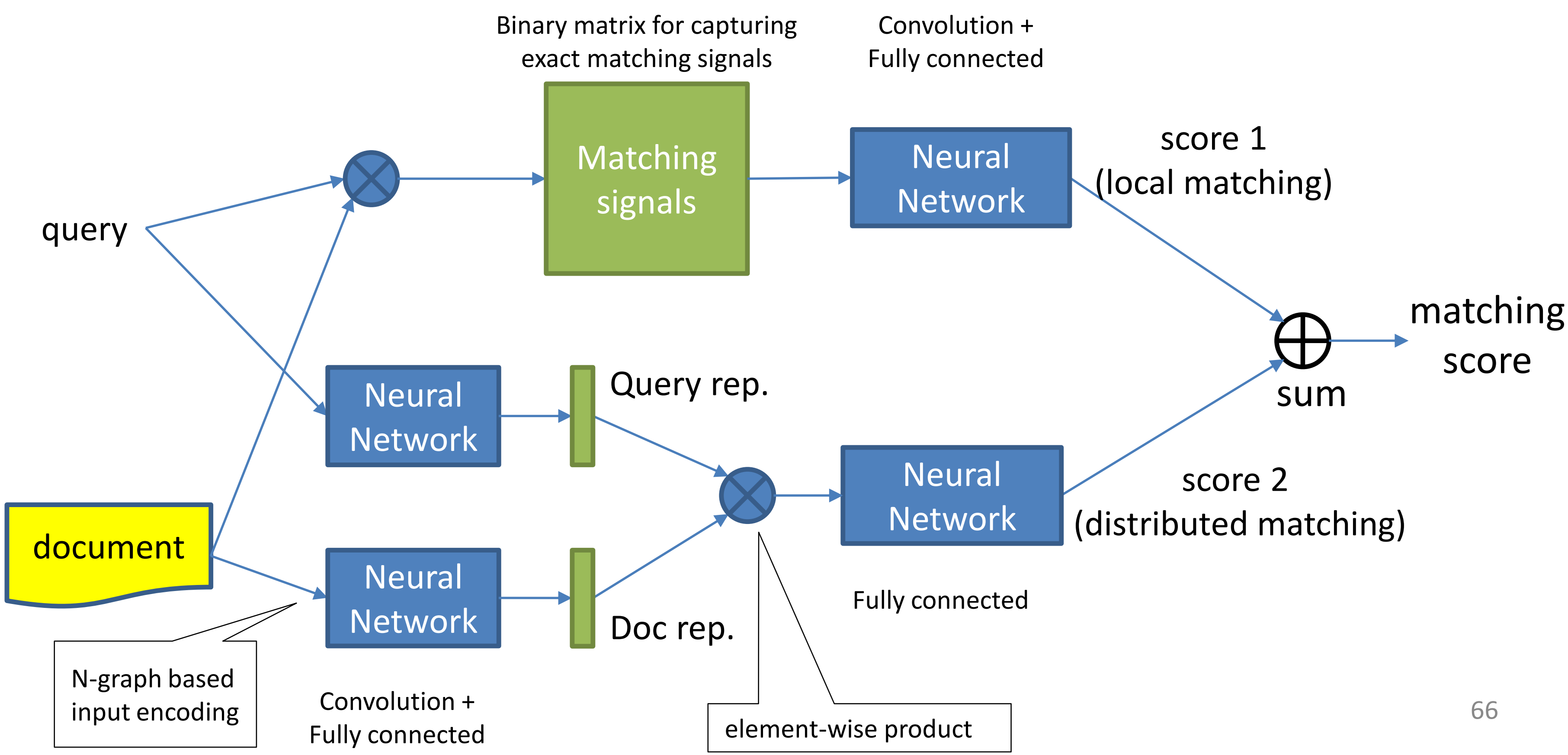
Position-Aware Neural IR Model (PACRR, Hui et al., EMNLP '17)

- Hypothesis: relevance matching is determined by some positions in documents
  - The first k words in document.
  - The most similar context positions in document.



Representation Learning + Matching Function Learning (Duet, Mitra et al., WWW '17)

- Hypothesis: matching with distributed representations complements matching with local representations
  - Local matching: matching function learning
  - Distributed matching: representation learning



Experimental Evaluation

	Method	P@1	MRR
Traditional IR	BM25	0.579	0.457
Representation Learning methods	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
Matching Function Learning	ARC-II	0.591	0.765
	MatchPyramid	0.764	0.867
	Match-SRNN	0.790	0.882

Based on Yahoo! Answers dataset (60,564 question-answer pairs)

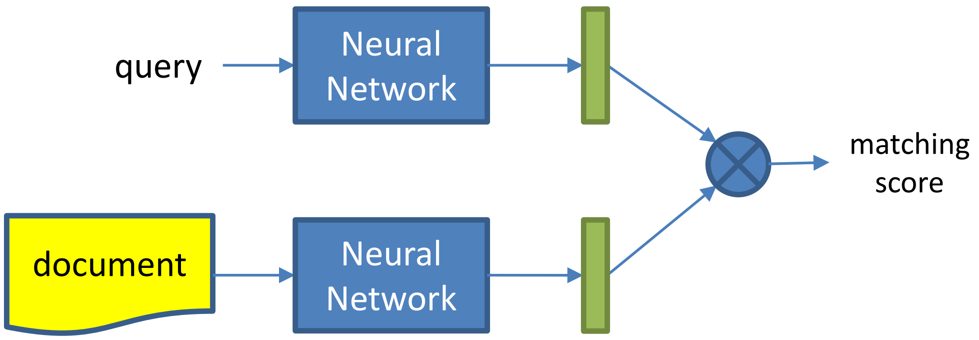
- Matching function learning based methods outperformed the representation learning ones

Short Summary

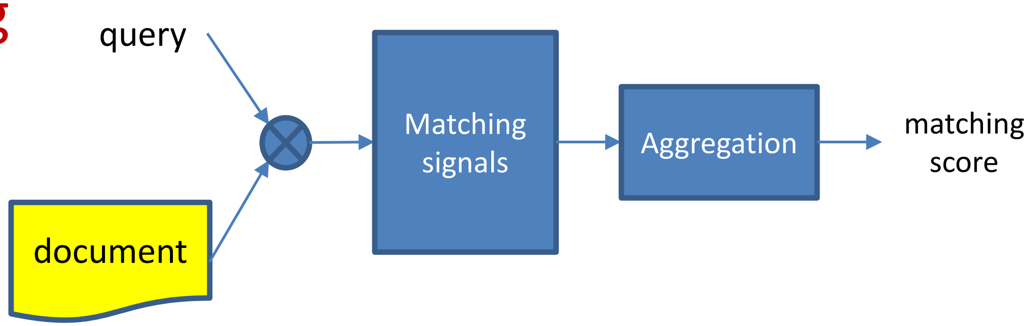
- Methods based on global distributions of matching strengths
  - 1. calculating term matching strength distributions
  - 2. aggregating the distributions to a matching score
- Methods based on local context of matched terms
  - 1. Identifying the relevance locations / contexts
  - 2. Matching the whole query with the local contexts
  - 3. Aggregating the local matching signals

Summary of Deep Matching Models in Search

- Representation learning: representing queries and document in semantic space



- Matching function learning: discovering and aggregating the query-document matching patterns



References

- Clark J. Google turning its lucrative web search over to ai machines[J]. Bloomberg Technology. Publicado em, 2015, 26.
- Metz C. AI is transforming Google search[J]. The rest of the web is next. WIRED Magazine, 2016.
- Huang P S, He X, Gao J, et al. Learning deep structured semantic models for web search using clickthrough data[C]//Proceedings of the 22nd ACM international conference on Conference on information & knowledge management. ACM, 2013: 2333-2338.
- Hu B, LuShen Y, He X, Gao J, et al. A latent semantic model with convolutional-pooling structure for information retrieval[C]//Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management. ACM, 2014: 101-110.
- Z, Li H, et al. Convolutional neural network architectures for matching natural language sentences[C]//Advances in neural information processing systems. 2014: 2042-2050.
- Qiu X, Huang X. Convolutional Neural Tensor Network Architecture for Community-Based Question Answering[C]//IJCAI. 2015: 1305-1311.
- Palangi H, Deng L, Shen Y, et al. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval[J]. IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP), 2016, 24(4): 694-707.
- Yin W, Schütze H. Multigrancnn: An architecture for general matching of text chunks on multiple levels of granularity[C]//Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). 2015, 1: 63-73.
- Socher R, Huang E H, Pennin J, et al. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection[C]//Advances in neural information processing systems. 2011: 801-809.
- Wan S, Lan Y, Guo J, et al. A Deep Architecture for Semantic Matching with Multiple Positional Sentence Representations[C]//AAAI. 2016, 16: 2835-2841.

References

- Pang L, Lan Y, Guo J, et al. Text Matching as Image Recognition[C]//AAAI. 2016: 2793-2799.
- Shengxian Wan, Yanyan Lan, Jun Xu, Jiafeng Guo, Liang Pang, and Xueqi Cheng. 2016. Match-SRNN: modeling the recursive matching structure with spatial RNN. In Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI'16), 2922-2928.
- Ankur P. Parikh, Oscar Tackstrom, Dipanjan Das, and Jakob Uszkoreit. A Decomposable Attention Model for Natural Language Inference. In Proceedings of EMNLP, 2016.
- Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. Convolutional Neural Networks for Soft-Matching N-Grams in Ad-hoc Search. In Proceedings of WSDM 2018.
- Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, Russell Power. End-to-End Neural Ad-hoc Ranking with Kernel Pooling. In Proceedings of SIGIR 2017.
- Bhaskar Mitra, Fernando Diaz, and Nick Craswell. Learning to match using local and distributed representations of text for web search. In Proceedings of WWW 2017.
- Jiafeng Guo, Yixing Fan, Qiqing Yao, W. Bruce Croft, A Deep Relevance Matching Model for Ad-hoc Retrieval. In Proceedings of CIKM 2016.
- Liu Yang, Qingyao Ai, Jiafeng Guo, W. Bruce Croft, aNMM: Ranking Short Answer Texts with Attention-Based Neural Matching Model. In Proceedings of CIKM 2016.
- Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu and Xueqi Cheng. DeepRank: a New Deep Architecture for Relevance Ranking in Information Retrieval. In Proceedings of CIKM 2017.
- Qin Chen, Qinmin Hu, Jimmy Xiangji Huang, Liang He. CA-RNN: Using Context-Aligned Recurrent Neural Networks for Modeling Sentence Similarity. . In Proceedings of AAAI 2018.
- Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Xueqi Cheng. A Study of MatchPyramid Models on Ad-hoc Retrieval. In Proceedings of SIGIR 2016 Neu-IR Workshop.

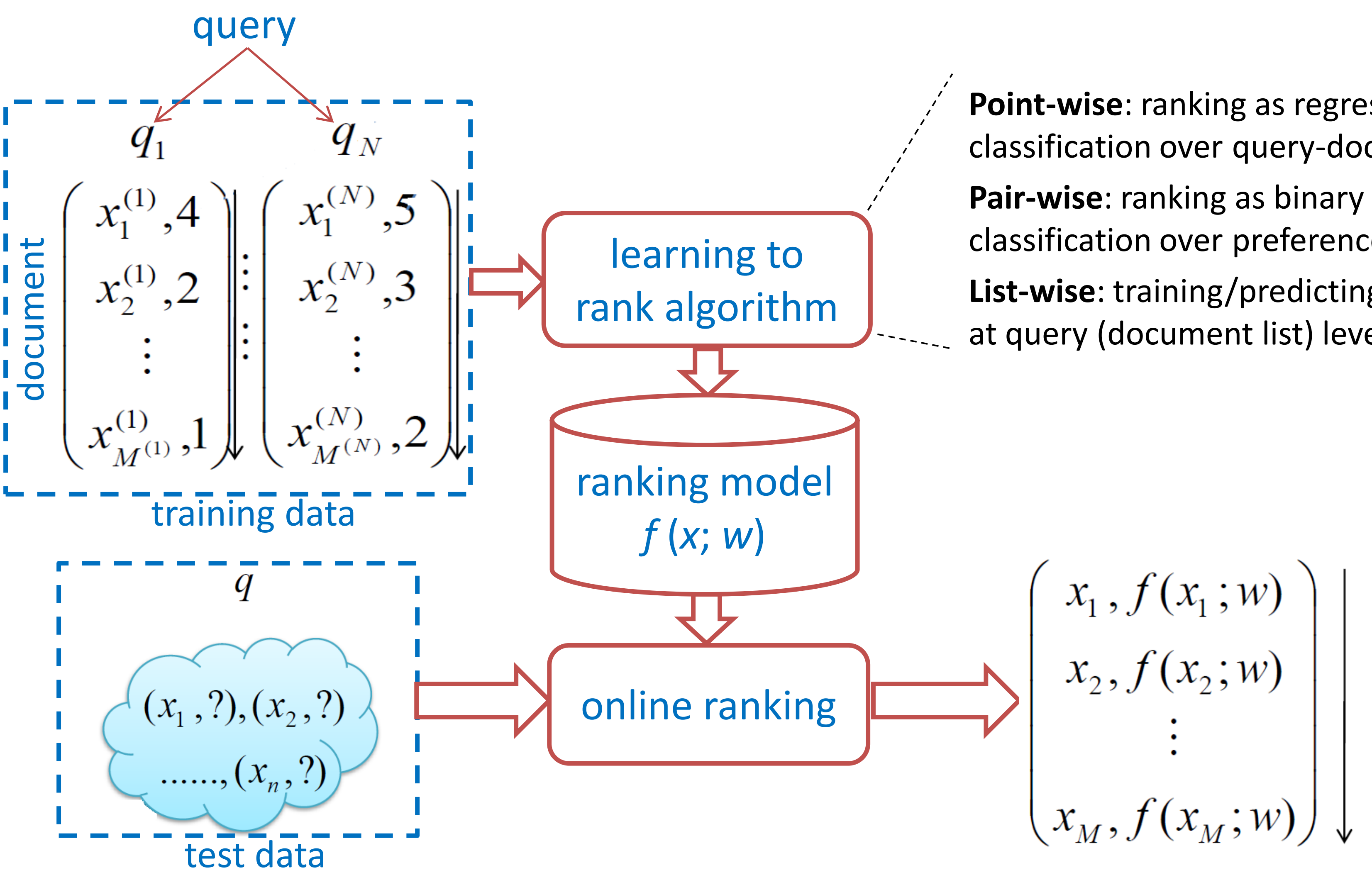
10 MINUTES BREAK !

Outline

- Introduction
- Deep Semantic Matching
  - Methods of Representation Learning
  - Methods of Matching Function Learning
- Reinforcement Learning to Rank
  - Formulation IR Ranking with RL
  - Approaches
- Summary

Traditional Learning to Rank for Web Search

- Machine learning algorithms for relevance ranking

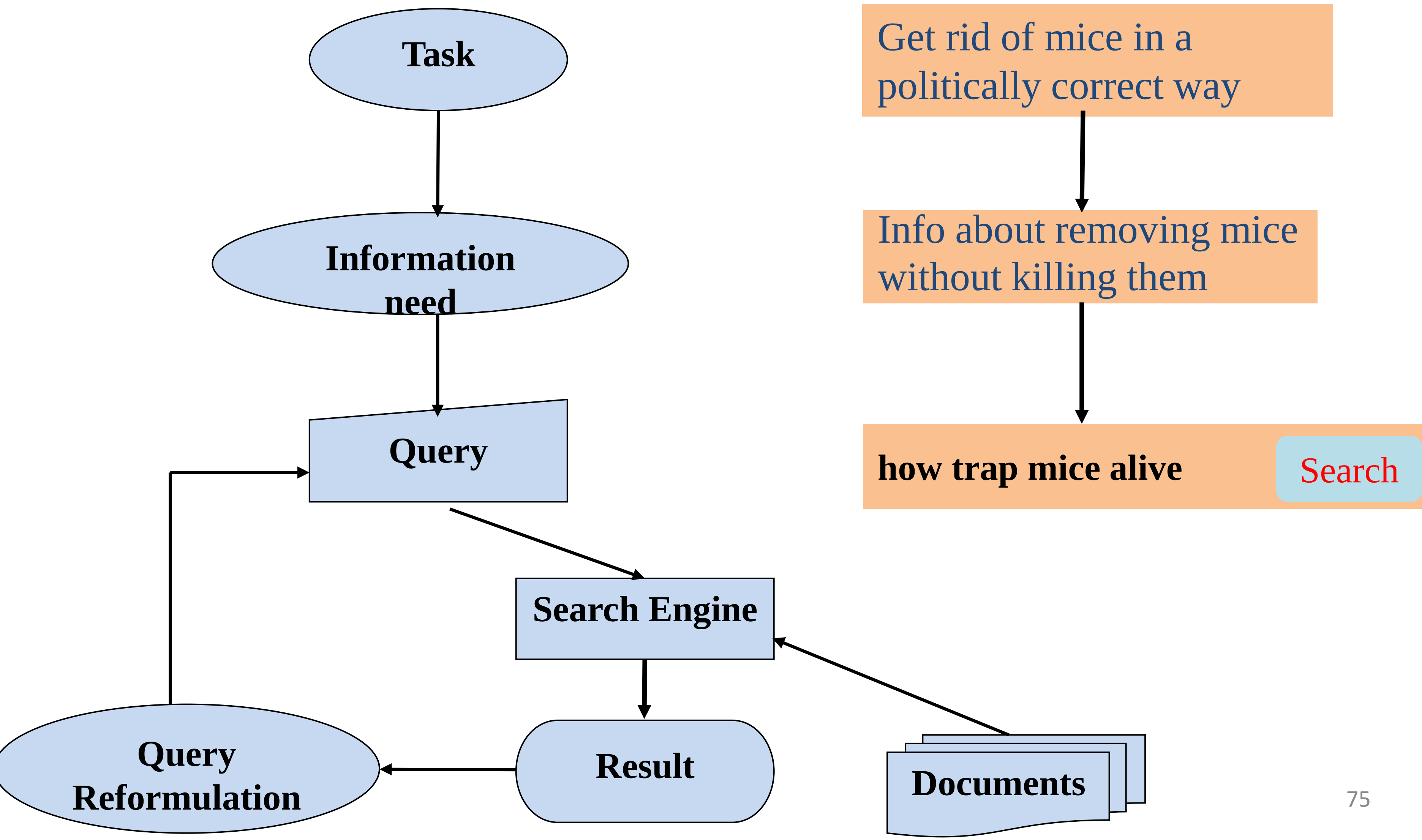


Point-wise: ranking as regression or classification over query-documents

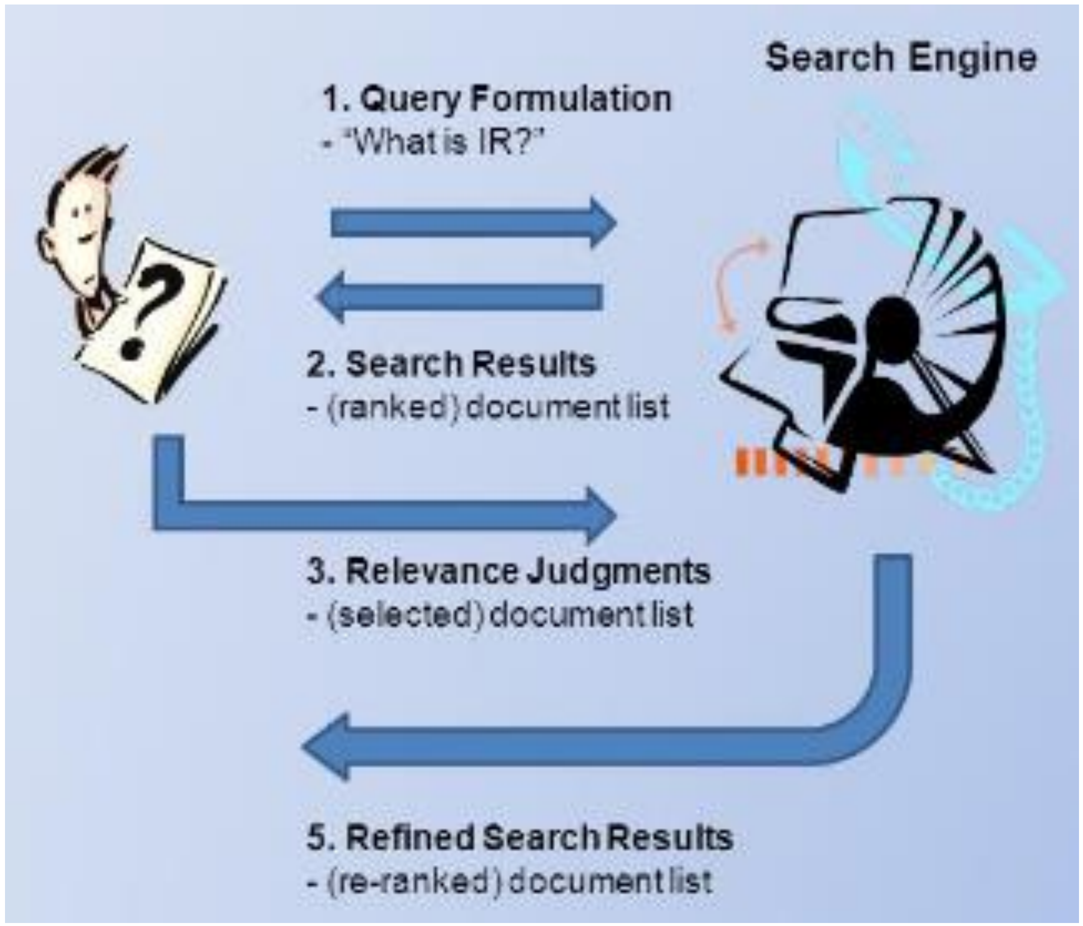
Pair-wise: ranking as binary classification over preference pairs

List-wise: training/predicting ranking at query (document list) level

Retrieving Information is a Process

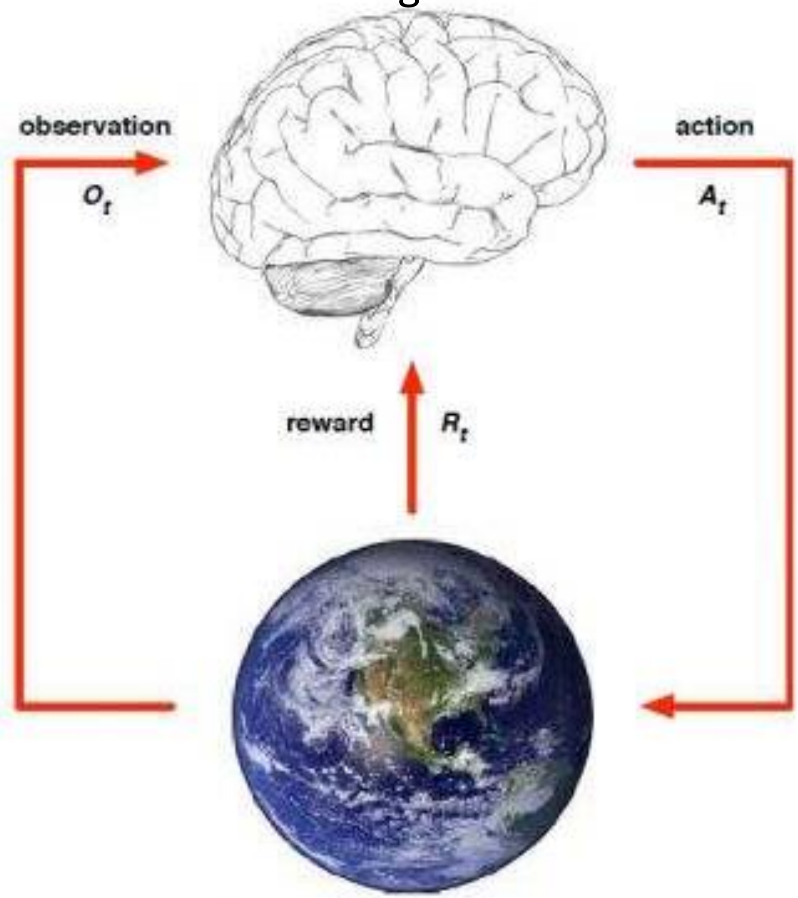


With (Multiple Rounds of) Interactions between Users and Search Engines



Reinforcement Learning: Modeling the Interactions

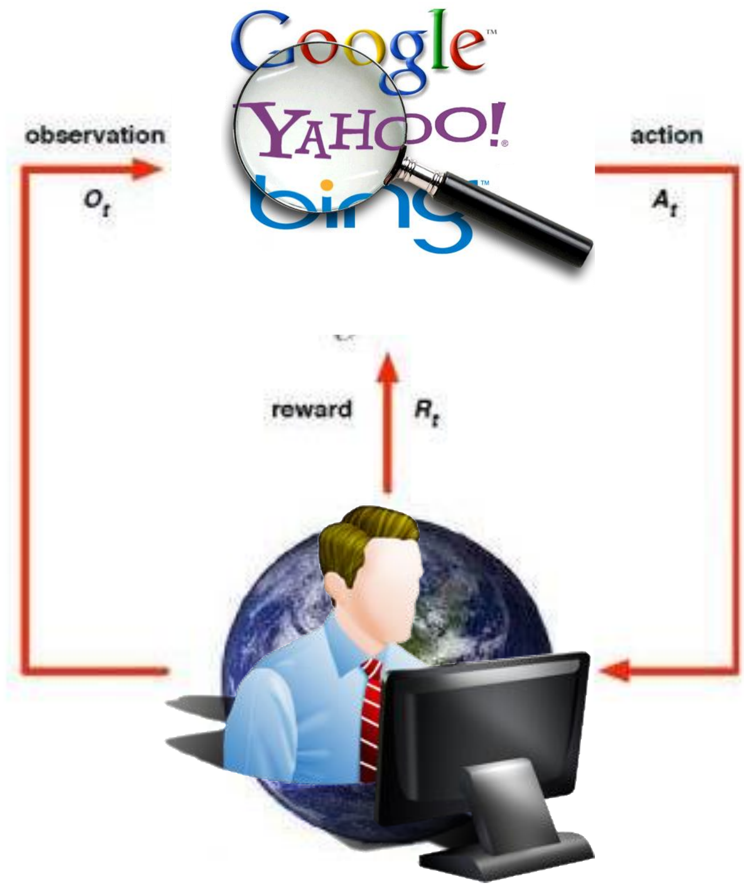
Agent



Environment

Interactions between AlphaGo and its Opponent

Interactions between Search Engine and Search Users



Different definitions of the components (**time steps**, **actions**, **rewards** etc.) leads to different IR tasks

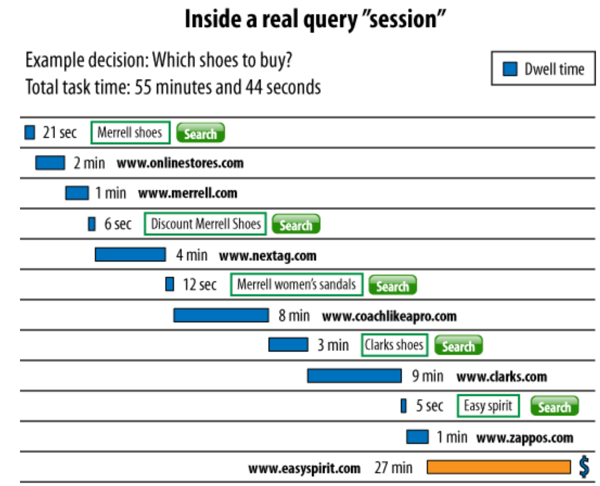
Granularity of Time Steps

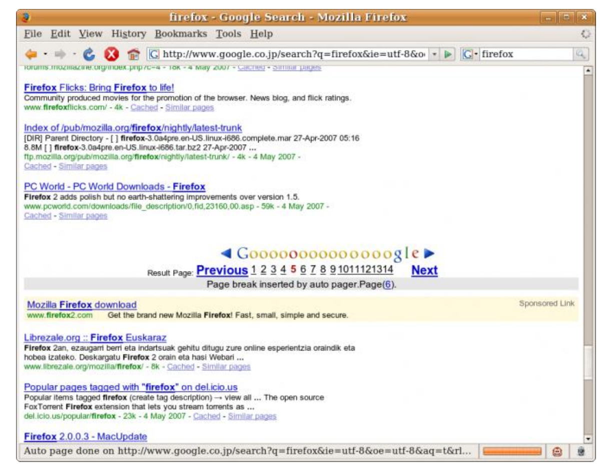
- At each time step, the user may
  - – Submit a new query

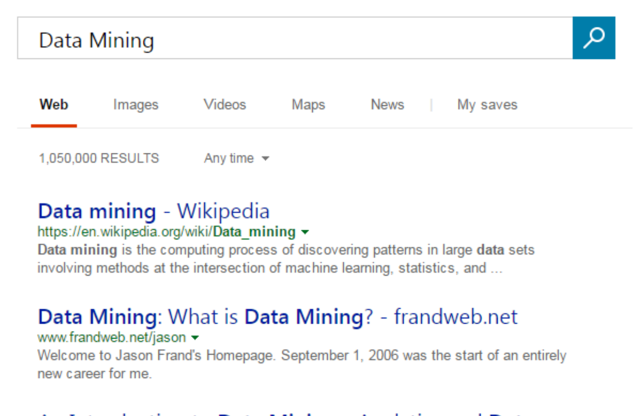
    e.g., session search
  - – Browse a result page

    e.g., multi-page search
  - – Browse an item

    e.g., relevance ranking, search result diversification

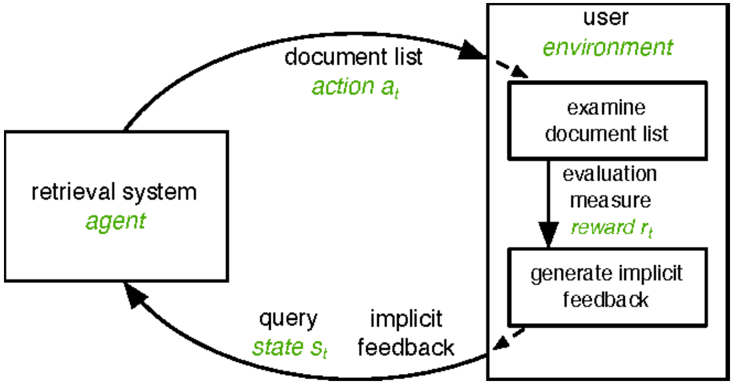




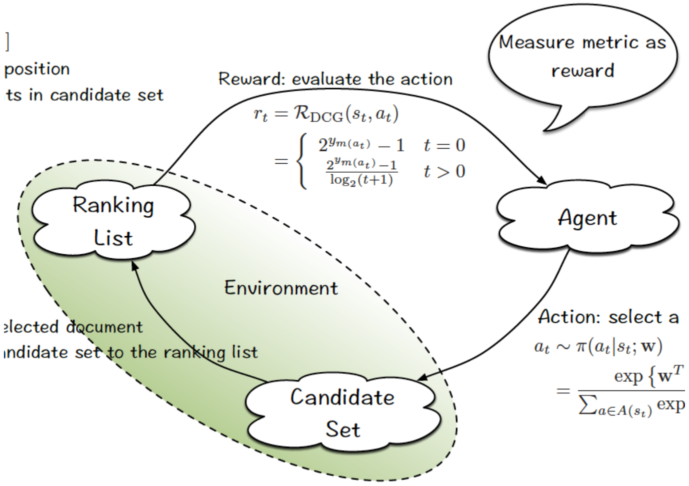


How to Get the Rewards?

- From real users
  - E.g., online learning to rank



- From simulated environment



RL Approaches to IR

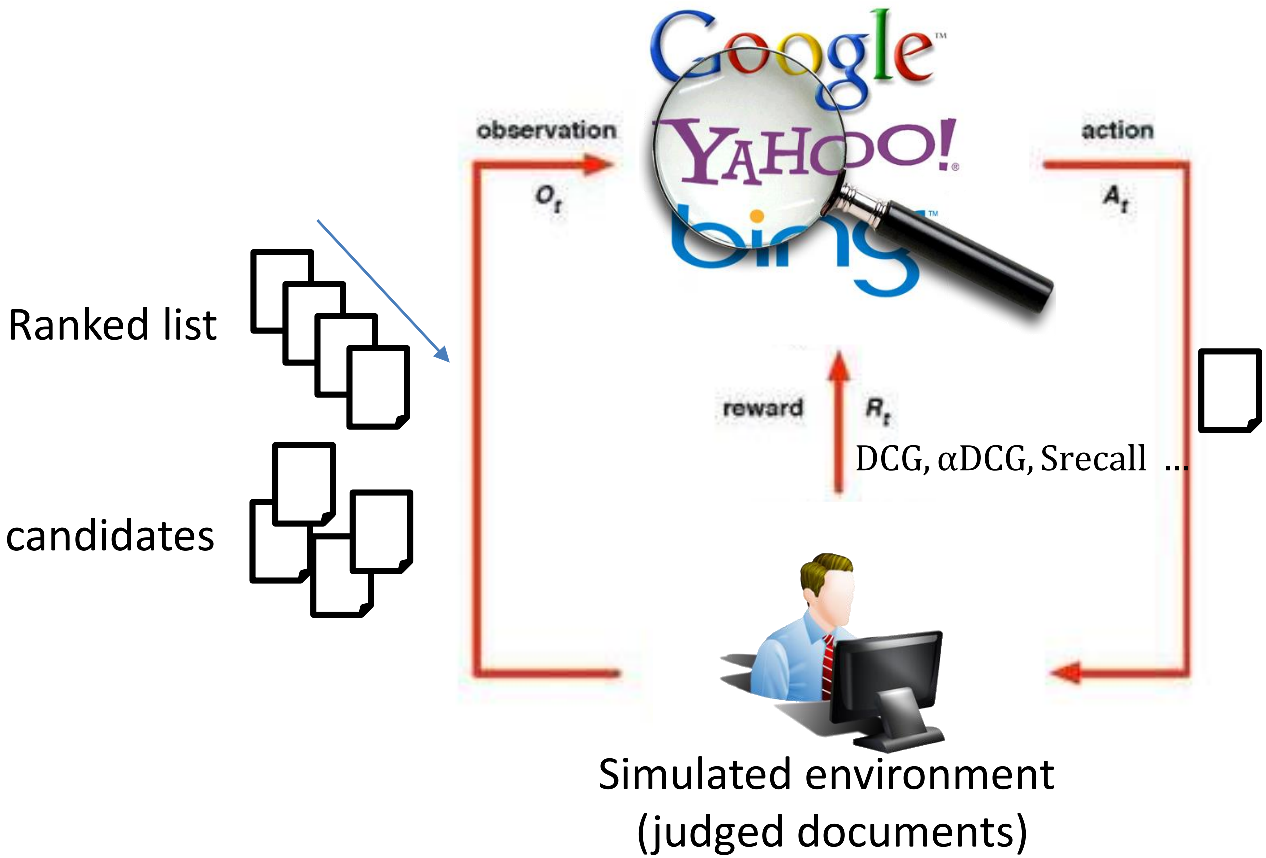
		Granularity of Time Steps		
		One item per step	One result page per step	One query per step
Source of Rewards	Simulation	**Relevance ranking** MDPRank (Zeng et al., '17) **Diverse ranking** MDP-DIV (Xia et al., '17); M2Div (Feng et al., '18)	N/A	N/A
	Real users	**Online ranking** Dueling Bandits (Yue et al., '09), (Hofmann et al., IRJ '13)	**Multi-Page search** MDP-MPS (Zeng et al., '18); DPG-FBE (Hu et al., Arxiv '18); IES (Jin et al, '13)	**Session search** QCM (Guan et al, '13); Win-Win (Luo et al,'14); DPL (Luo et al, '15)

APPROACHES

RL Approaches to IR

		Granularity of Time Steps		
		One item per step	One result page per step	One query per step
Source of Rewards	Simulation	**Relevance ranking** MDPRank (Zeng et al., ‘17) **Diverse ranking** MDP-DIV (Xia et al., ’17); M2Div (Feng et al., ‘18)	N/A	N/A
	Real users	**Online ranking** Dueling Bandits (Yue et al., ‘09), (Hofmann et al., IRJ ’13)	**Multi-Page search** MDP-MPS (Zeng et al., ’18); DPG-FBE (Hu et al., Arxiv ’18); IES (Jin et al, ’13)	**Session search** QCM (Guan et al, ’13); Win-Win (Luo et al,’14); DPL (Luo et al, ’15)

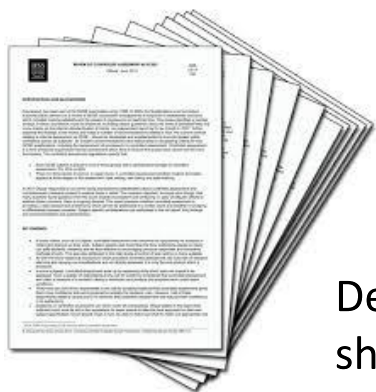
Interaction Framework of Relevance/Diverse Ranking



- **Action**: Selects a document and puts ranking list
- **Observation**: query, top *t* ranked list, candidate set
- **Reward**: designed based on rank evaluation measures

Modeling Ranking with MDP

Candidate document set



Decide which doc should be selected for the 2nd rank

query

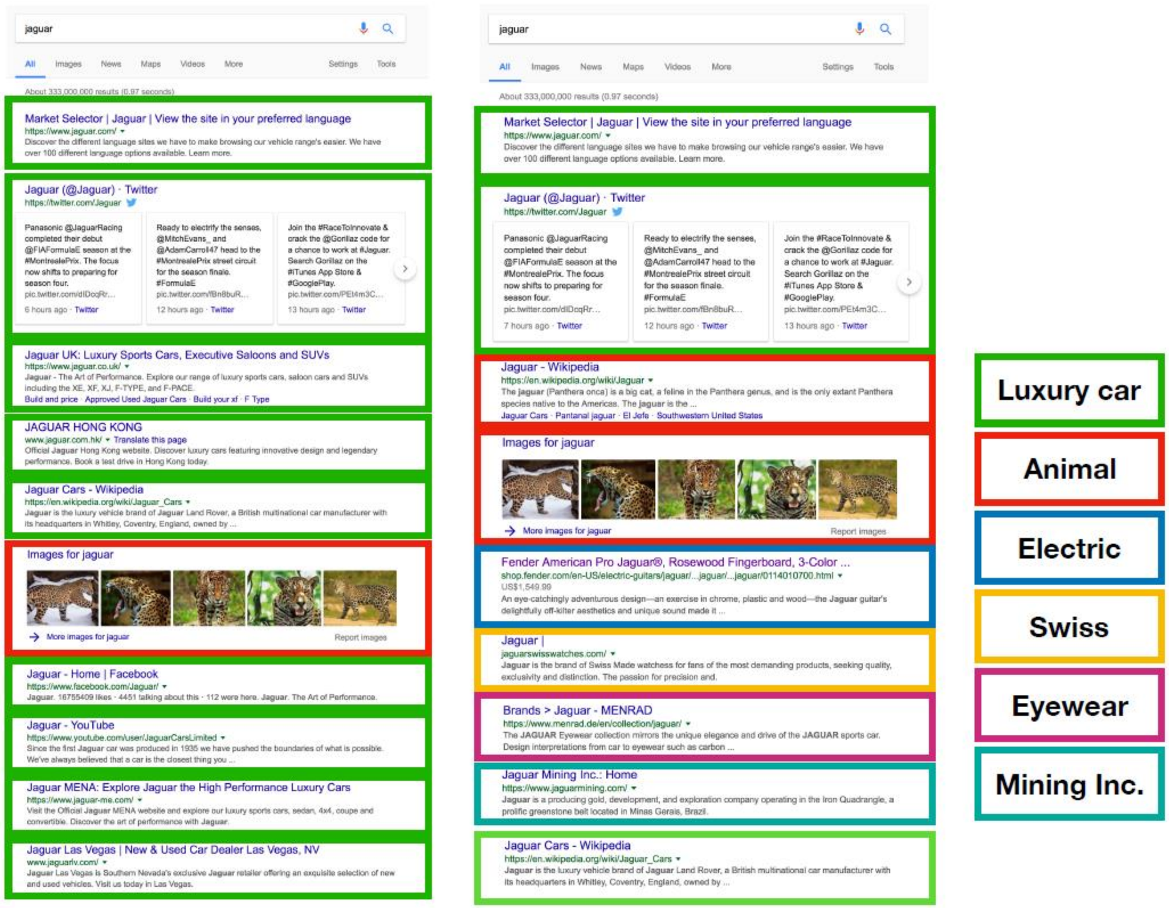
Rank 1: doc 1

Rank 2: ?

MDP factors	Corresponding ranking factors
Time steps	The ranking positions
State	Query, preceding docs, candidate docs etc.
Policy	Distribution over candidate docs
Action	Selecting a doc and placing it to current position
Reward	Defining reward based on IR evaluation measures (e.g., DCG)
State transition	Depends on the definition of the state

Search Result Diversification

Query: jaguar



- Query: information needs are ambiguous and multi-faceted
- Search results: may contain redundant information
- Goal: covering as much subtopics as possible with a few documents

Modeling Diverse Ranking with MDP (MDP-DIV) (Xia et al., SIGIR ‘17)

- Key points
  - Mimic user top-down browsing behaviors
  - Model dynamic information needs with MDP state
- States $s_t = [Z_t, X_t, \mathbf{h}_t]$
  - Z_t: sequence of t preceding documents, $Z_0 = \phi$
  - X_t: set of candidate documents, $X_0 = X$
  - $\mathbf{h}_t \in R^K$: latent vector, encodes user perceived utility from preceding documents, initialized with the information needs form the query:

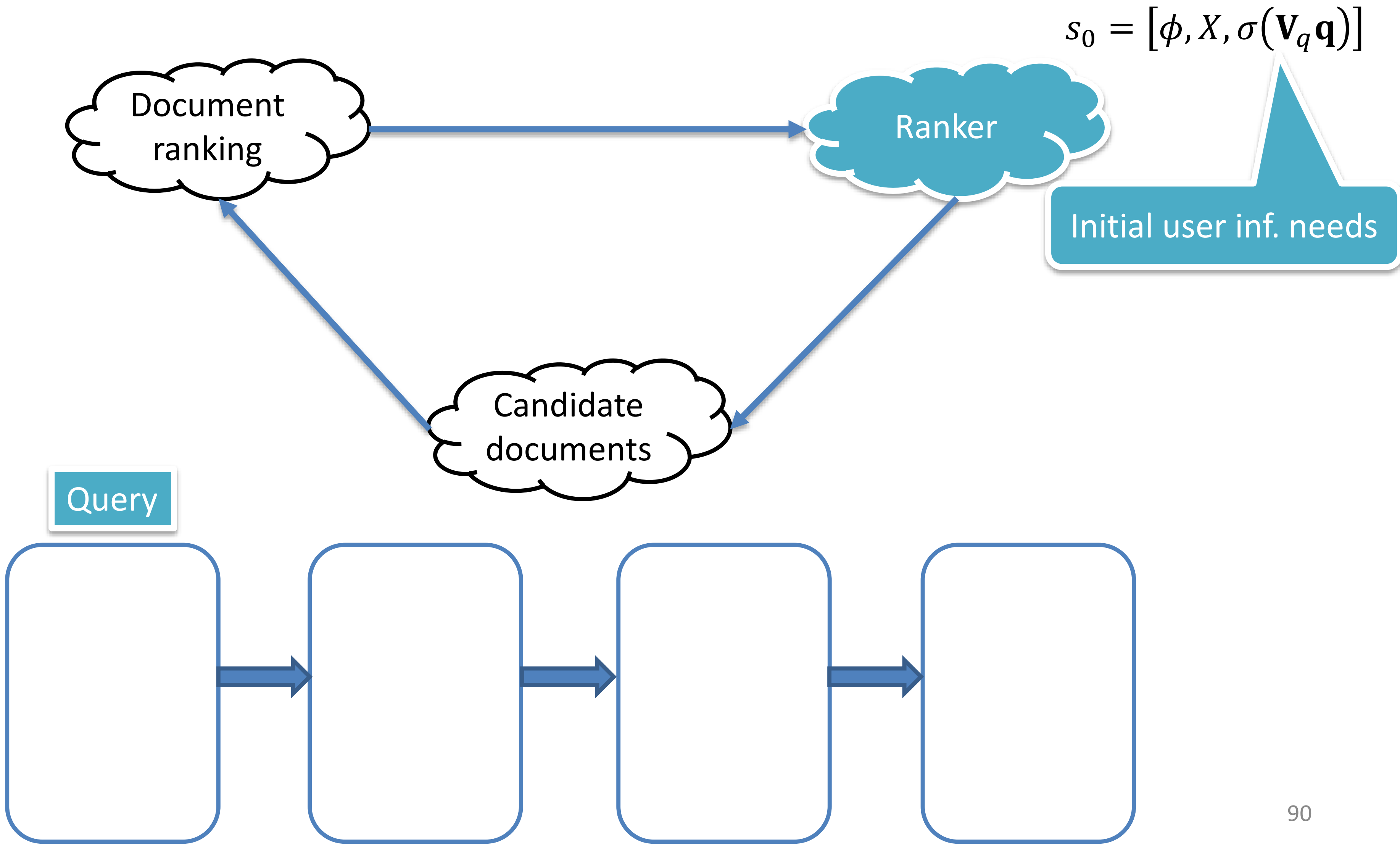
$$\mathbf{h}_0 = \sigma\left(\mathbf{V}_q \mathbf{q}\right)$$

Modeling Diverse Ranking with MDP

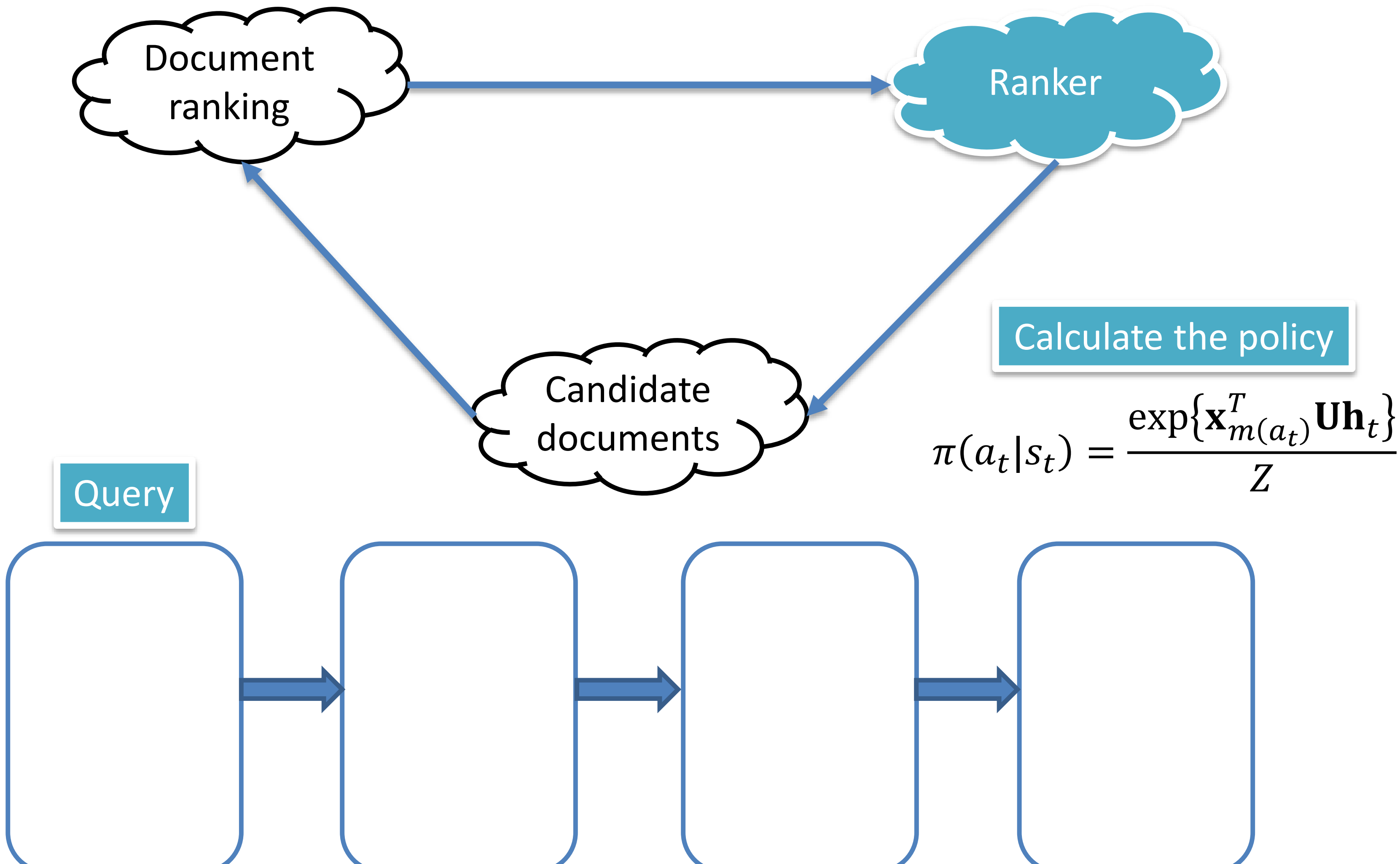
$\mathbf{x}_{m(a_t)}$: document embedding

MDP factors	Corresponding diverse ranking factors
Time steps	The ranking positions
State	$s_t = [Z_t, X_t, \mathbf{h}_t]$
Policy	$\pi(a_t \mid s_t = [Z_t, X_t, \mathbf{h}_t]) = \frac{\exp\{\mathbf{x}_{m(a_t)}^T \mathbf{U}\mathbf{h}_t\}}{Z}$
Action	Selecting a doc and placing it to rank $t+1$
Reward	Based on evaluation measure αDCG, SRecall etc. For example: $R = \alpha\mathrm{DCG}[t+1] - \alpha\mathrm{DCG}[t]$; $R = \mathrm{SRecall}[t+1] - \mathrm{SRecall}[t]$
State Transition	$s_{t+1} = T(s_t = [Z_t, X_t, \mathbf{h}_t], a_t) = [Z_t \oplus \{\mathbf{x}_{m(a_t)}\}, X_t \backslash \{\mathbf{x}_{m(a_t)}\}, \sigma(\mathbf{V}\mathbf{x}_{m(a_t)} + \mathbf{W}\mathbf{h}_t)]$

Ranking Process: Initialize State

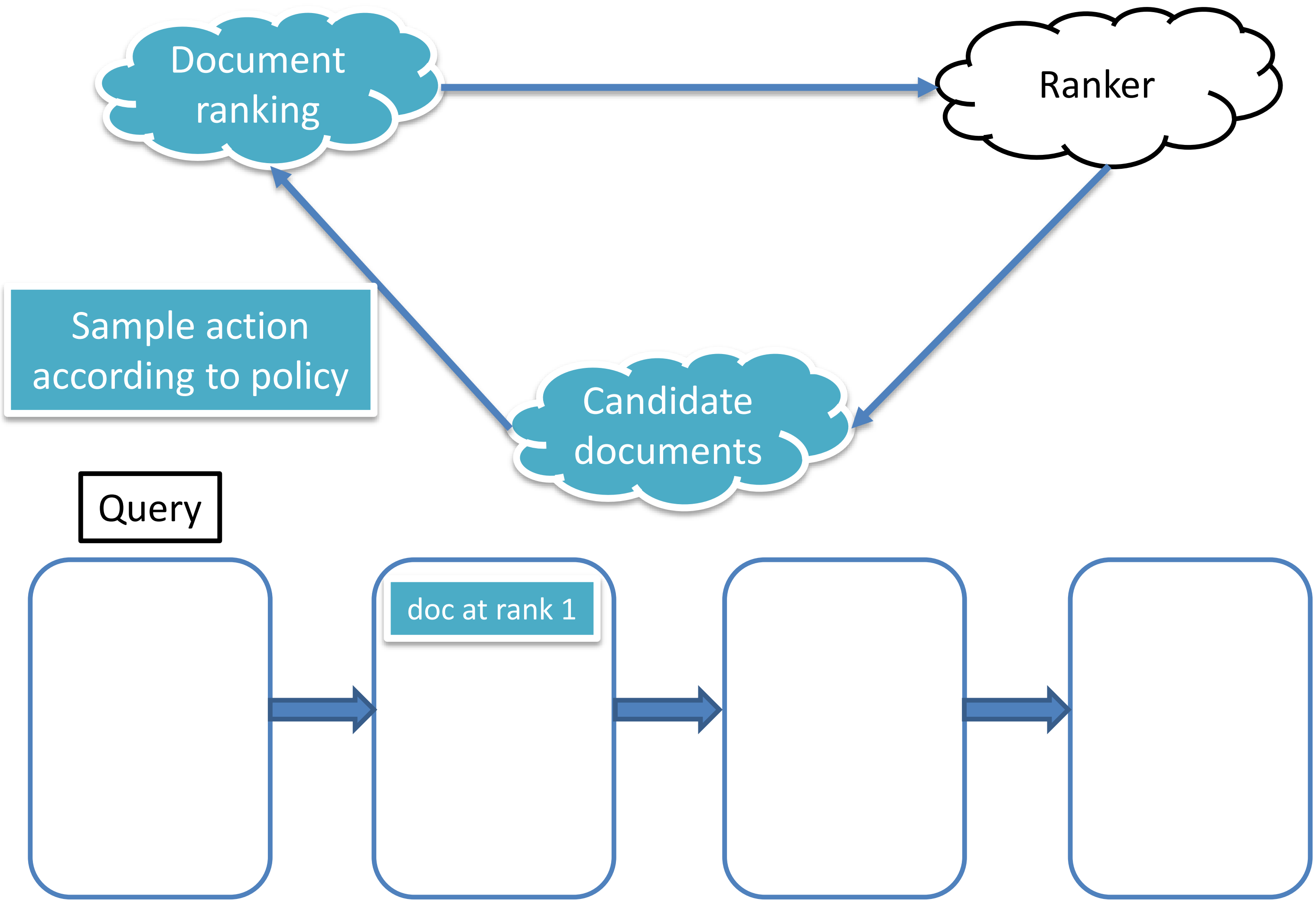


Ranking Process: Policy

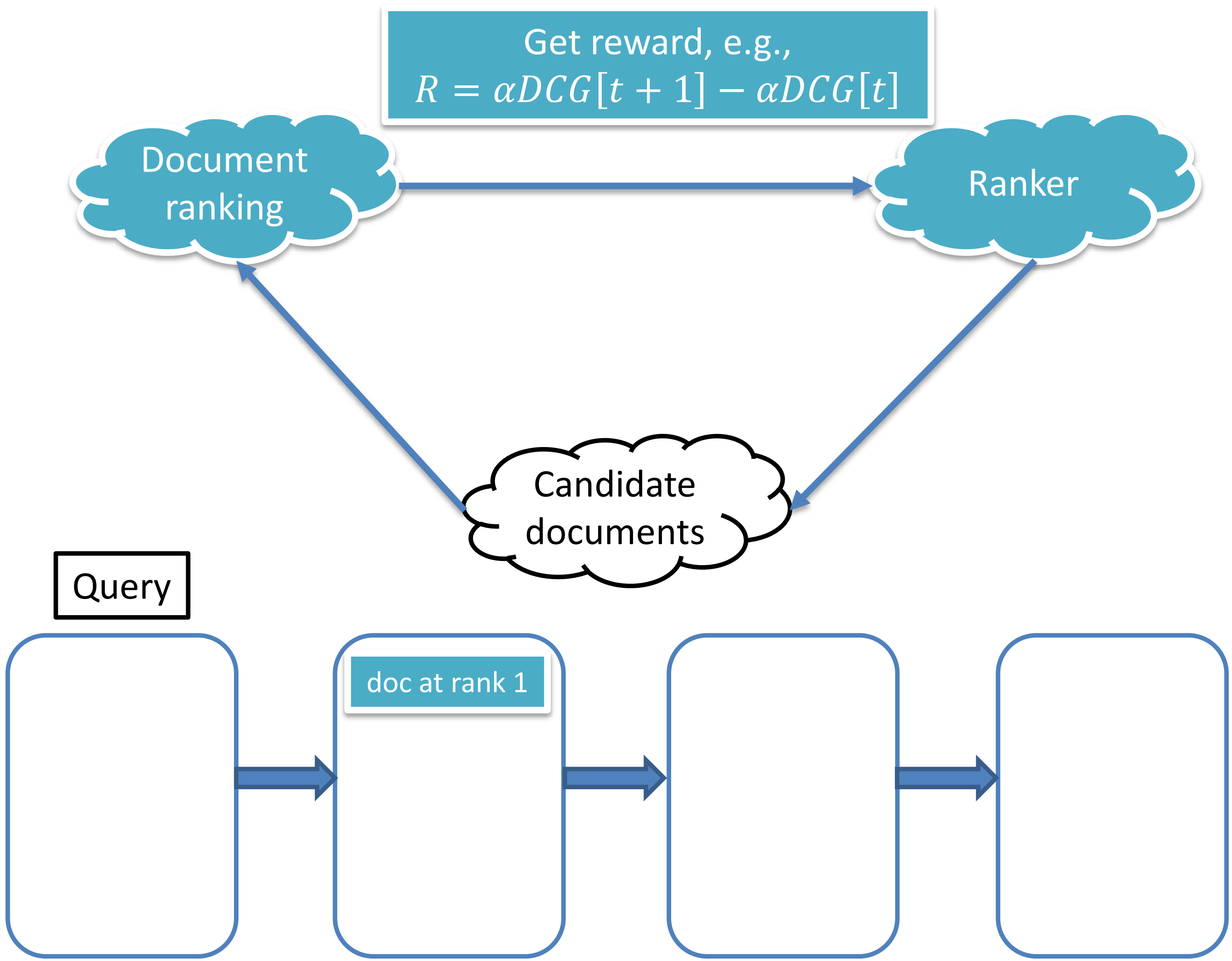


$$\pi(a_t|s_t) = \frac{\exp\{\mathbf{x}_{m(a_t)}^T \mathbf{U}\mathbf{h}_t\}}{Z}$$

Ranking Process: Action



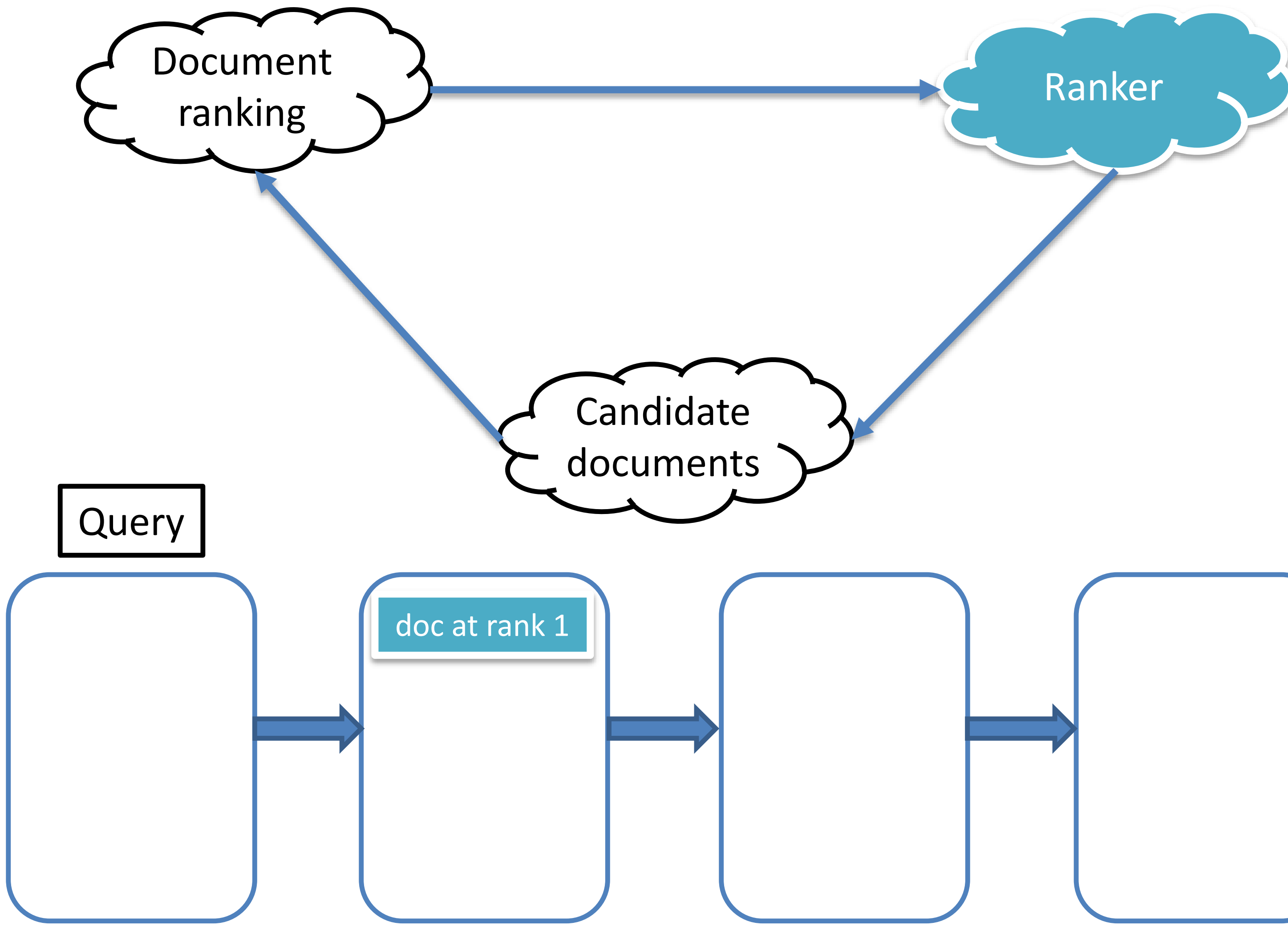
Ranking Process: Reward



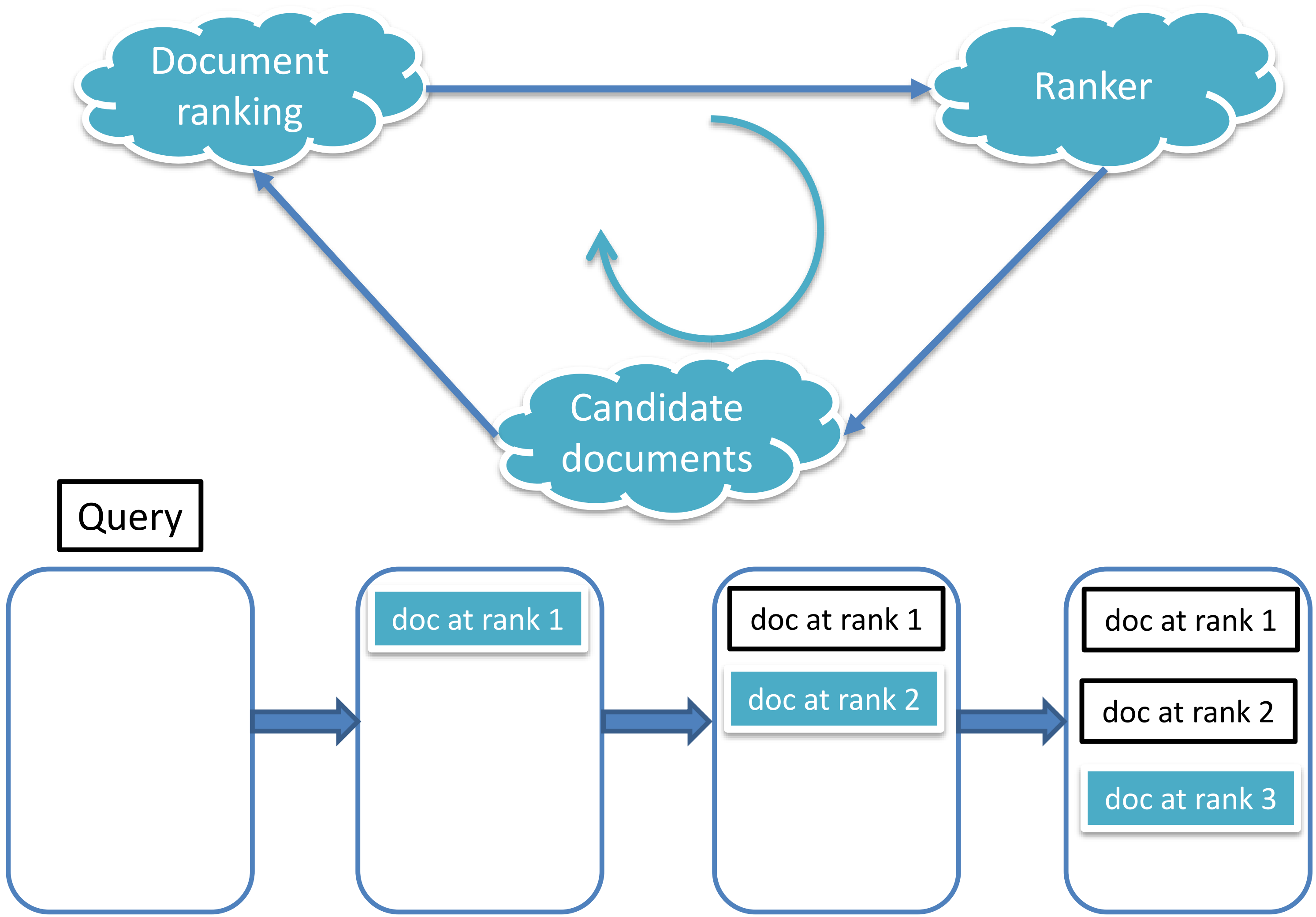
Ranking Process: State Transition

Update ranked list, candidate set, and latent vector

$$s_{t+1} = \left[Z_t \oplus \{\mathbf{x}_{m(a_t)}\}, X_t \backslash \{\mathbf{x}_{m(a_t)}\}, \sigma\left(\mathbf{V}\mathbf{x}_{m(a_t)} + \mathbf{W}\mathbf{h}_t\right)\right]$$



Ranking Process: Iterate



Learning with Policy Gradient

- Model parameters $\mathbf{\Theta} = \{\mathbf{V}_q, \mathbf{U}, \mathbf{V}, \mathbf{W}\}$
- Learning objective: maximizing expected return (discounted sum of rewards) of each training query

$$\max_{\mathbf{\Theta}} v(\mathbf{q}) = E_\pi G_0 = E_\pi \left[\sum_{k=0}^{M-1} \gamma^k r_{k+1} \right]$$

  - Directly optimizes evaluation measure as $G_0 = \alpha\text{DCG}@M$

- Monte-Carlo stochastic gradient ascent is used to conduct the optimization (REINFORCE algorithm)

$$\widehat{\nabla_{\mathbf{\Theta}} v(\mathbf{q})} = \gamma^{\mathrm{t}} \mathrm{G_t} \nabla_{\mathbf{\Theta}} \log \pi(a_t | s_t; \mathbf{\Theta})$$

Greedy Decisions in MDP-DIV

Algorithm 3 MDP-DIV online ranking

Input: Parameters $\mathbf{\Theta} = \{\mathbf{V}_q, \mathbf{U}, \mathbf{V}, \mathbf{W}\}$, query $\mathbf{q}$, documents X
Output: Permutation of documents τ

Initialize $s \leftarrow [\emptyset, X, \sigma(\mathbf{V}_q\mathbf{q})]$ {Equation (1)}
$M \leftarrow |X|$
for $t = 0$ **to** $M - 1$ **do**
  $A \leftarrow A(s)$ {Possible actions according to X in state s}
  $\hat{a} \leftarrow \arg\max_{a \in A} \pi(a|s; \mathbf{\Theta})$ {Choosing most possible action}
  $\tau[t+1] \leftarrow m(\hat{a})$ {Document $\mathbf{x}_{m(a)}$ is ranked at $t+1$}
  $[\mathcal{Z}, X, \mathbf{h}] \leftarrow s$
  $s \leftarrow [\mathcal{Z} \oplus \{\mathbf{x}_{m(\hat{a})}\}, X \setminus \{\mathbf{x}_{m(\hat{a})}\}, \sigma(\mathbf{V}\mathbf{x}_{m(\hat{a})} + \mathbf{W}\mathbf{h})]$
end for
return τ

- Full exploitation as there is no supervision information can be provided

Search global optimal solution amounts to the problem of subset selection, NP-hard!

Why Greedy?

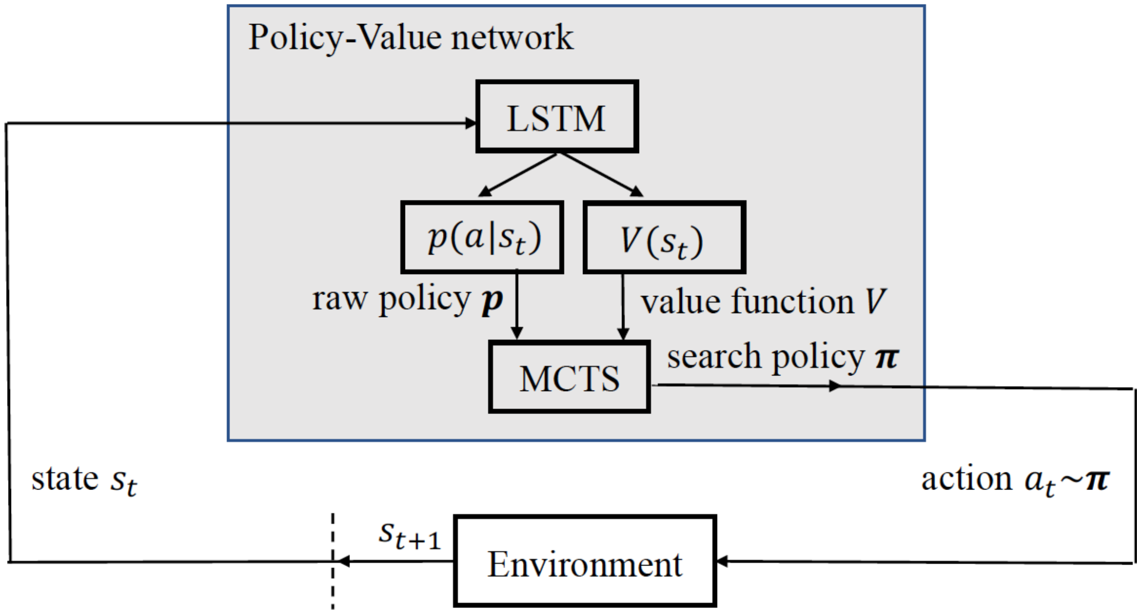
- Training: exploration and exploitation
- Online ranking: exploitation only
- From the viewpoint modeling the environment
  - Environment model simulates the rewards!
  - Training: supervision information available, can judge the quality of exploration
  - Online ranking: no supervision information, cannot make the judgement (no reward)

- The environment model cannot be generalized to unseen query!

Ways to Address the Problem

- Exhaustive search (Brute-force search)
  - Enumerating all possible candidate rankings
  - Checking their performances at each position
  - Output the best ranking
  - Global optimal solution but extremely costly
- Monte Carlo tree search (MCTS)
  - Search tree based on random sampling
  - Near-optimal solution but much faster
  - A environment model that can be generated!
  - Adopted by AlphaGo Zero

MCTS Enhanced MDP for Diverse Ranking (Feng et al., SIGIR '18)



$$
\begin{aligned}
\mathbf{f}_k &= \sigma(\mathbf{W}_f \mathbf{x}_k + \mathbf{U}_f \mathbf{h}_{k-1} + \mathbf{b}_f),\\
\mathbf{i}_k &= \sigma(\mathbf{W}_i \mathbf{x}_k + \mathbf{U}_i \mathbf{h}_{k-1} + \mathbf{b}_i),\\
\mathbf{o}_k &= \sigma(\mathbf{W}_o \mathbf{x}_k + \mathbf{U}_o \mathbf{h}_{k-1} + \mathbf{b}_o),\\
\mathbf{c}_k &= \mathbf{f}_k \circ \mathbf{c}_{k-1} + \mathbf{i}_k \circ \tanh(\mathbf{W}_c \mathbf{x}_k + \mathbf{U}_c \mathbf{h}_{k-1} + \mathbf{b}_c),\\
\mathbf{h}_k &= \mathbf{o}_k \circ \tanh(\mathbf{c}_k),
\end{aligned}
$$

$$\mathrm{LSTM}(s) = \left[{\mathbf{h}_t}^T, {\mathbf{c}_t}^T\right]^T$$

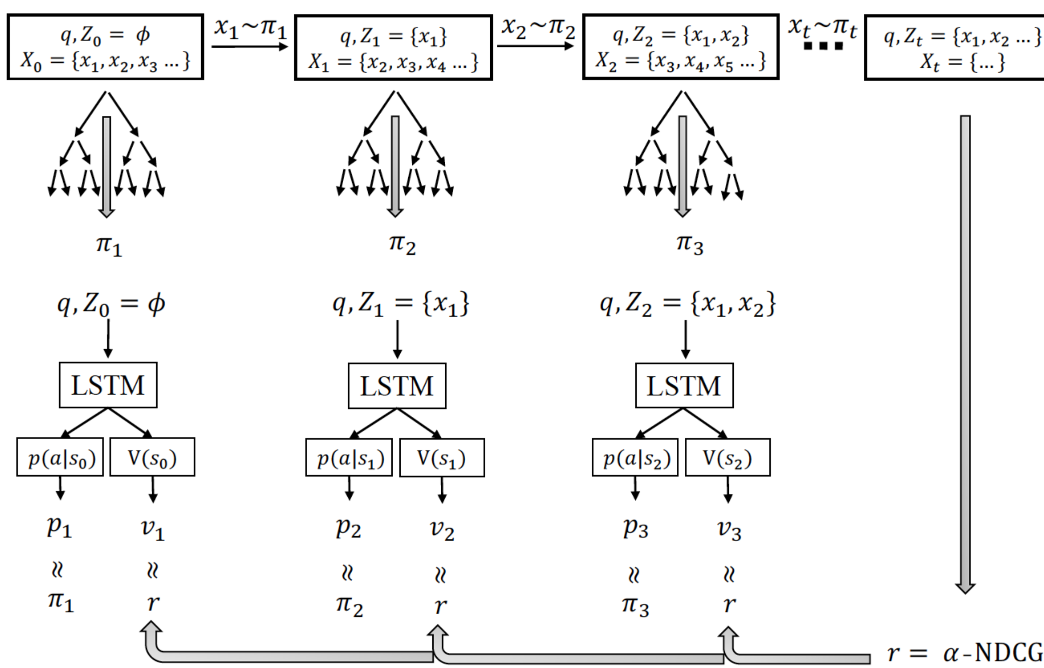
$$V(s) = \sigma\left(\langle \mathbf{w}, \mathrm{LSTM}(s)\rangle + b_v\right)$$

$$p(a|s) = \frac{\exp\left\{\mathbf{x}_{m(a)}^T \mathbf{U}_p \ \mathrm{LSTM}(s)\right\}}{\sum_{a' \in \mathcal{A}(s)} \exp\left\{\mathbf{x}_{m(a')}^T \mathbf{U}_p \ \mathrm{LSTM}(s)\right\}}$$

- Ranking as an MDP
- MCTS guided by the predicted policies and values

Learning the Parameters

$$\ell(E, r) = \sum_{t=1}^{|E|} \left((V(s_t) - r)^2 + \sum_{a \in \mathcal{A}(s_t)} \pi_t(a|s_t) \log \frac{1}{p(a|s_t)} \right)$$



- Predicted value is as close to the real α-NDCG as possible
- Raw policy is as close to the search policy as possible

Relation with AlphaGo Zero

- Task formalization
  - Playing of Go: alternating Markov game
  - Diverse ranking: sequential document selection
- Supervision information
  - AlphaGo Zero: results of self-play
  - Diverse ranking: human labels and the predefined evaluation measure
- Shared neural networks
  - AlphaGo Zero: residual network with raw board positions as inputs
  - Diverse ranking: LSTM with sequence of selected documents

Evaluation

Method	α-NDCG@5	α-NDCG@10	ERR-IA@5	ERR-IA@10
MMR	0.2753	0.2979	0.2005	0.2309
xQuAD	0.3165	0.3941	0.2314	0.2890
PM-2	0.3047	0.3730	0.2298	0.2814
SVM-DIV	0.3030	0.3699	0.2268	0.2726
R-LTR	0.3498	0.4132	0.2521	0.3011
PAMM(α-NDCG)	0.3712	0.4327	0.2619	0.3029
NTN-DIV(α-NDCG)	0.3962	0.4577	0.2773	0.3285
MDP-DIV(α-DCG)	0.4189	0.4762	0.2988	0.3494
M^2Div(without MCTS)	0.4386*	0.4835	0.3435*	0.3668*
M^2Div(with MCTS)	**0.4424***	**0.4852**	**0.3459***	**0.3686***

Why MCTS helps?

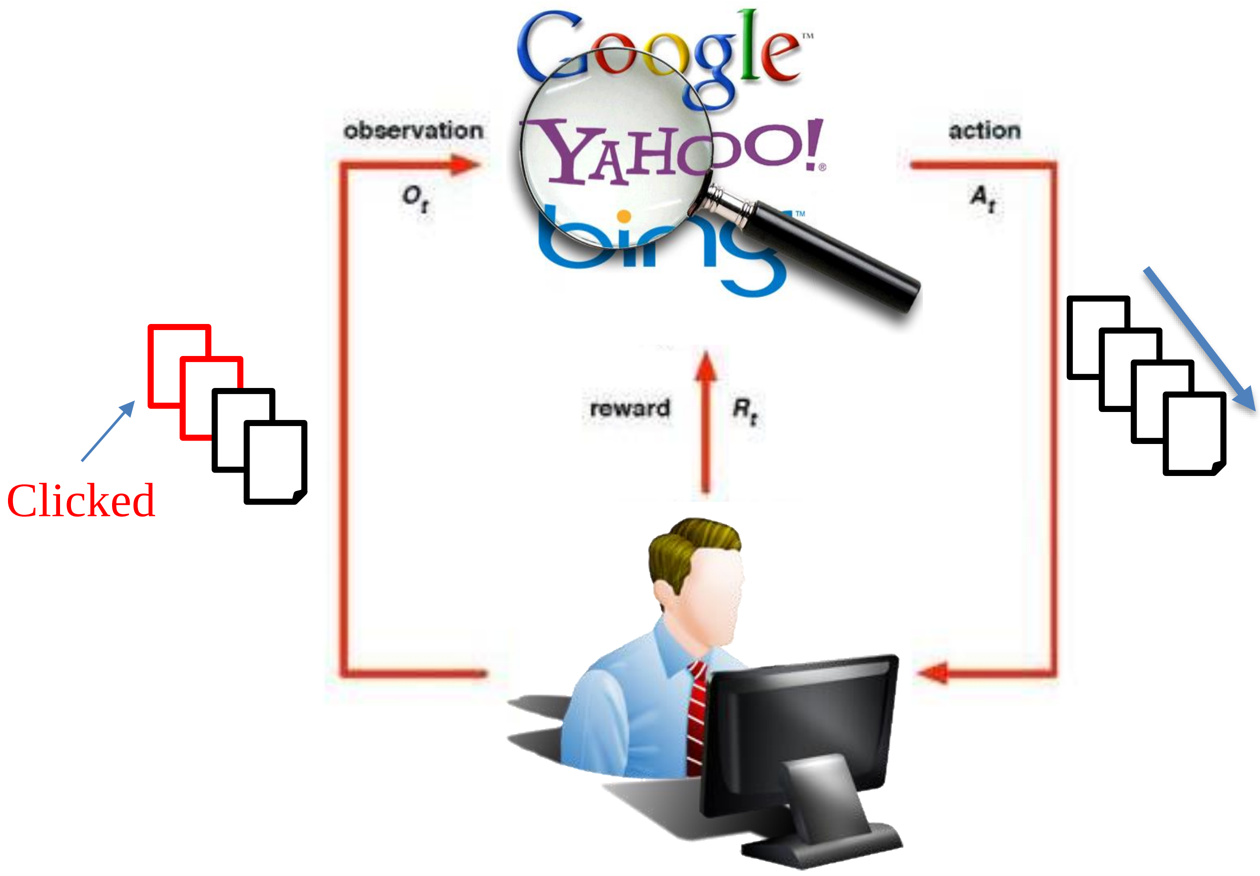
- Model-free RL: the agent does not know
  - How state will change in response to its actions
  - What immediate reward it will receive

- Model-free RL v.s. Model-based RL
  - Model-free RL don’t have to learn a model of the environment to find a good policy: policy gradient, Q-learning, Actor-critic
  - Model-based RL: agent make predictions about what the next state and reward will be (MCTS tries to do this, invoked the knowledge about ranking)

RL Approaches to IR

		Granularity of Time Steps		
		One item per step	One result page per step	One query per step
Source of Rewards	Simulation	**Relevance ranking** MDPRank (Zeng et al., '17) **Diverse ranking** MDP-DIV (Xia et al., '17); M2Div (Feng et al., '18)	N/A	N/A
	Real users	**Online ranking** Dueling Bandits (Yue et al., '09), (Hofmann et al., IRJ '13)	**Multi-Page search** MDP-MPS (Zeng et al., '18); DPG-FBE (Hu et al., Arxiv '18); IES (Jin et al, '13)	**Session search** QCM (Guan et al, '13); Win-Win (Luo et al,'14); DPL (Luo et al, '15)

Interaction Framework of Online Ranking



- **Action**: generate a document ranking list
- **Observation**: user behavior on the ranking list, e.g., browsing, click etc.
- **Reward**: calculated based on user clicks

Ranked Bandit Algorithm [Radlinski et al., ICML '08]

- For addressing diverse ranking problem
  - MAB_i for each rank i
  - Each arm corresponds to a document
- Runs an MAB instance at each rank
  - Step 1: MAB_1 is responsible for choosing document shown at rank 1
  - Step 2: MAB_2 is responsible for choosing document shown at rank 2
  - … until top K documents are selected
- Show top K to users and receive response
  - Rewards: 1 if clicked and 0 if not

Ranked Bandit Algorithm (cont')

Algorithm 2 Ranked Bandits Algorithm

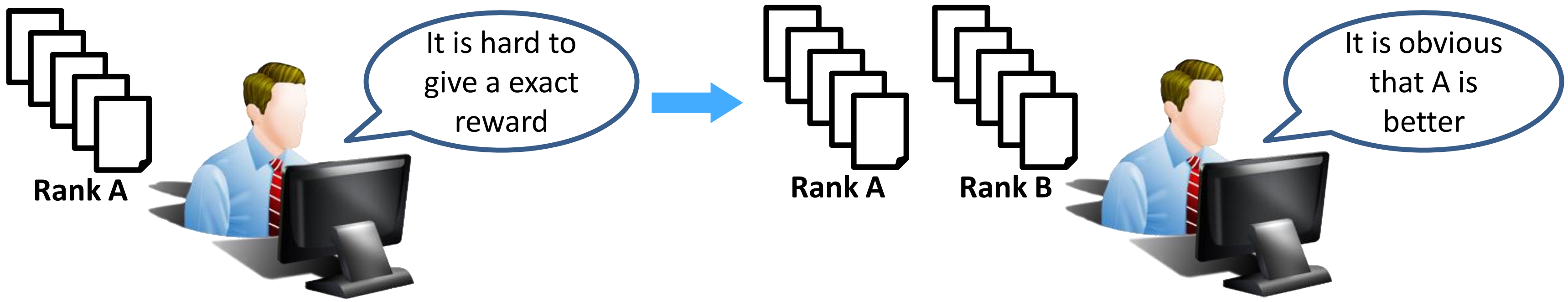
initialize $\mathsf{MAB}_1(n), \ldots, \mathsf{MAB}_k(n)$ *Initialize MABs*
for t = 1 … T **do**
  for i = 1 … k **do** *Sequentially select documents*
    $\hat{b}_i(t) \leftarrow$ select-arm (MAB_i)
    if $\hat{b}_i(t) \in \{b_1(t), .., b_{i-1}(t)\}$ **then** *Replace repeats*
      $b_i(t) \leftarrow$ arbitrary unselected document
    else
      $b_i(t) \leftarrow \hat{b}_i(t)$
    end if
  end for
  display $\{b_1(t), \ldots, b_k(t)\}$ to user; record clicks
  for i = 1 … k **do** *Determine feedback for* MAB_i
    if user clicked $b_i(t)$ **and** $\hat{b}_i(t) = b_i(t)$ **then**
      $f_{it} = 1$
    else
      $f_{it} = 0$
    end if
    update $(\mathsf{MAB}_i, arm = \hat{b}_i(t), reward = f_{it})$
  end for
end for

Document selection for k positions

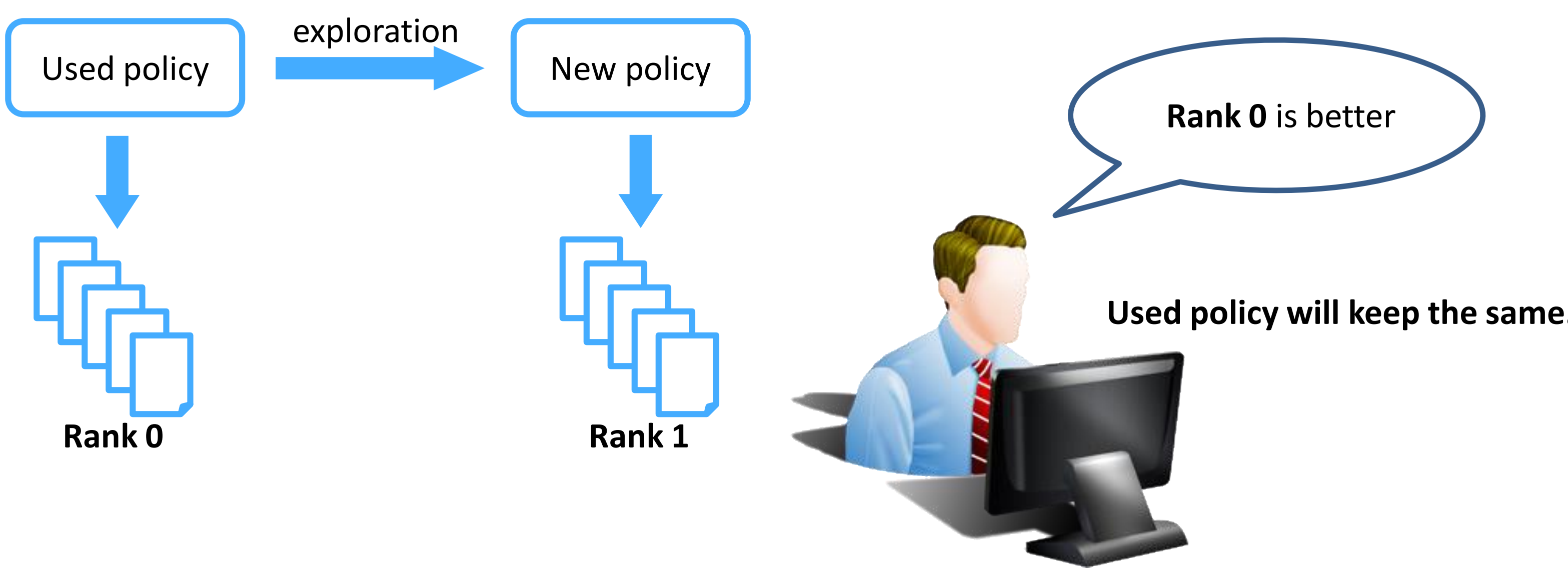
Update bandits

Dueling Bandits (Yue et al., ICML '09)

Which ranking list is better based on user responses (clicks)?

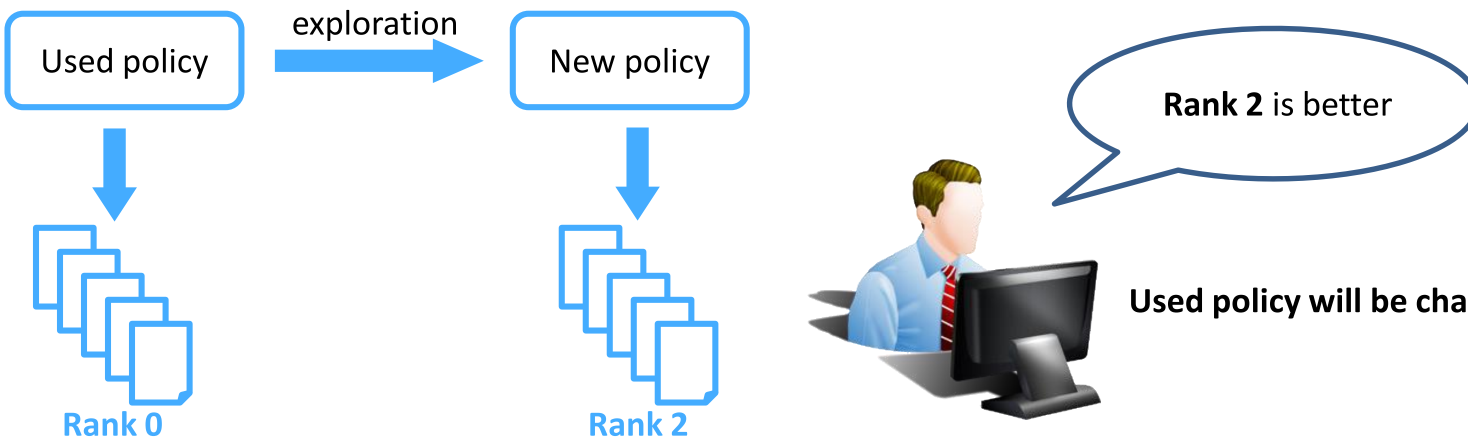


Dueling Bandit Gradient Descent(DBGD): update reject case



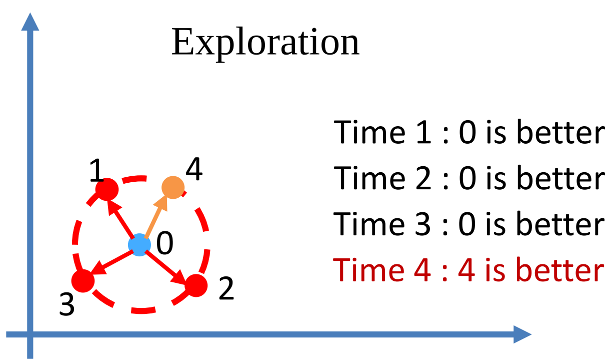
Dueling Bandits (cont’)

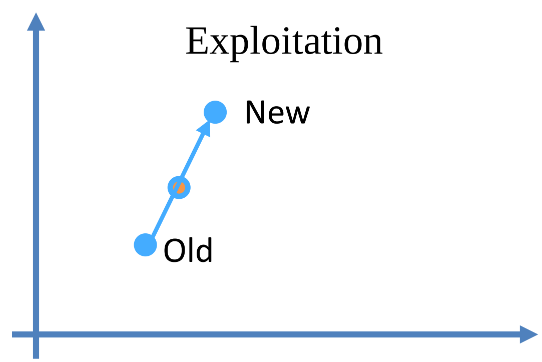
Dueling Bandit Gradient Descent(DBGD): update accept case



Used policy will be changed.

Exploration-exploitation tradeoff





Exploration parameter:

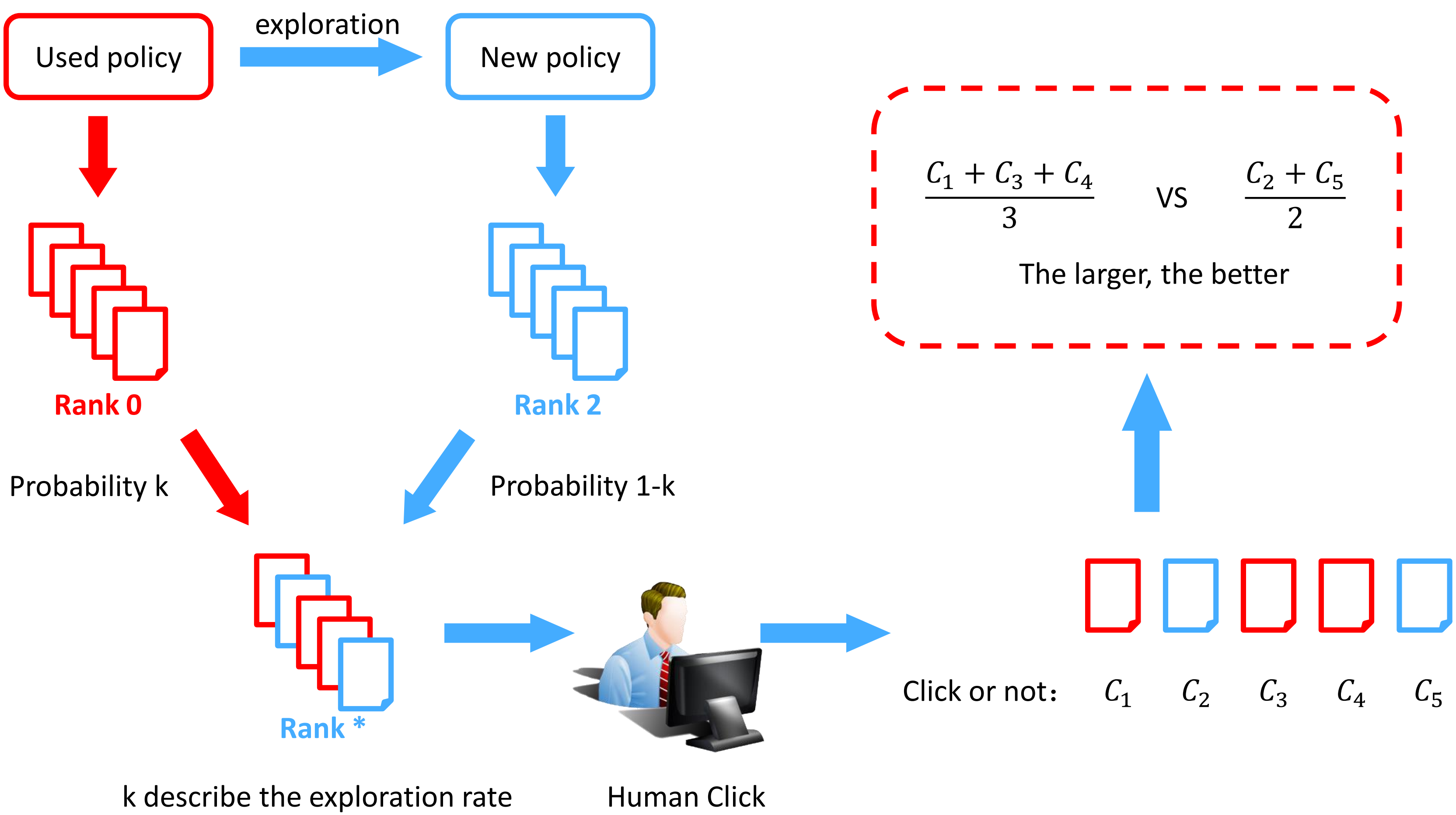
Radius of

Exploitation parameter:

Length of

Balancing Exploration and Exploitation (Hofmann et al., IRJ ‘13)

It is not natural to request users judging two ranking lists for one query!

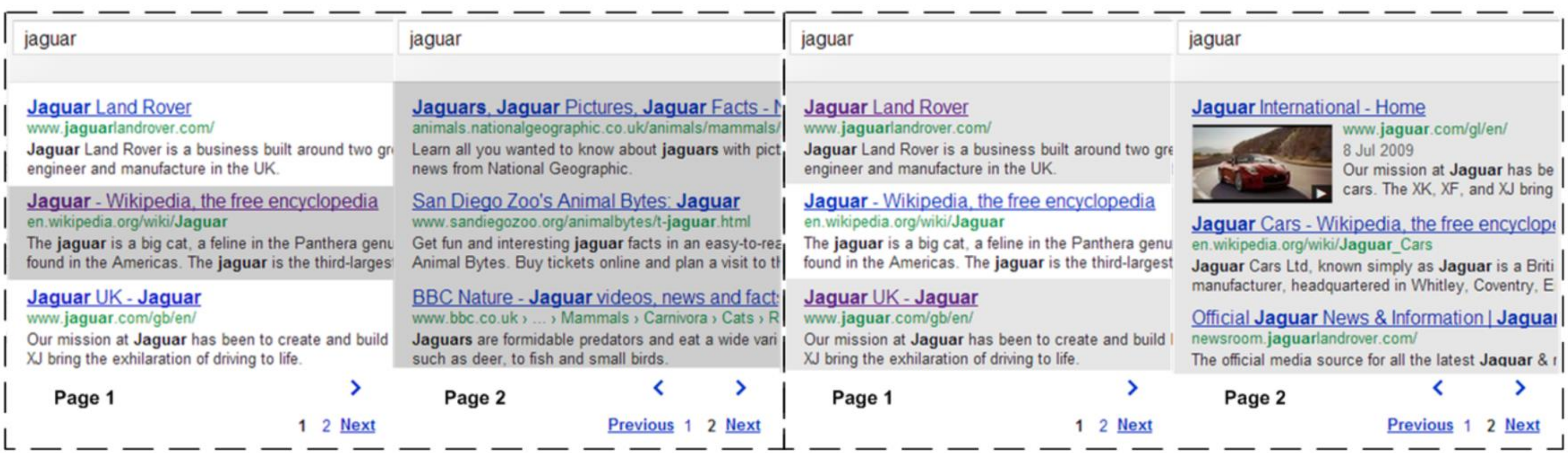


RL Approaches to IR

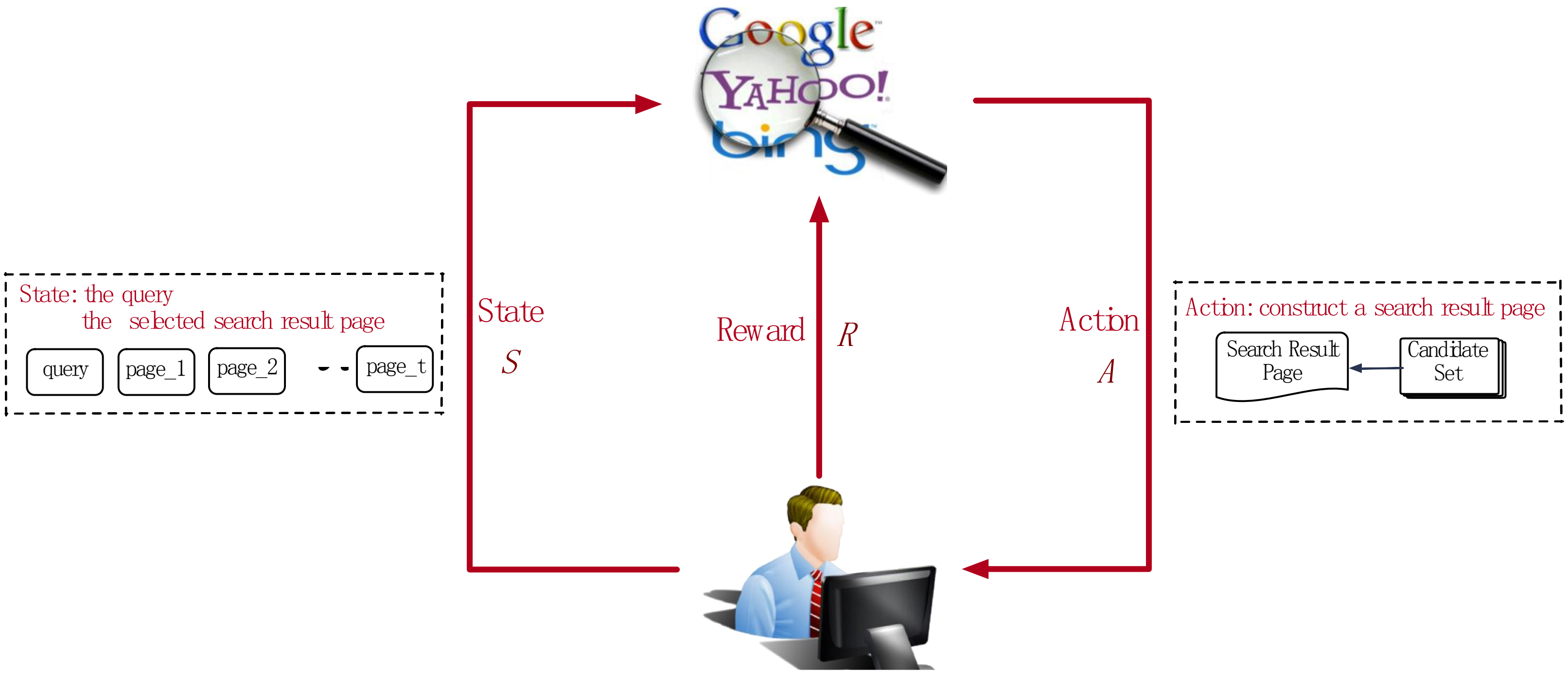
		Granularity of Time Steps		
		One item per step	One result page per step	One query per step
Source of Rewards	Simulation	**Relevance ranking** MDPRank (Zeng et al., '17) **Diverse ranking** MDP-DIV (Xia et al., '17); M2Div (Feng et al., '18)	N/A	N/A
	Real users	**Online ranking** Dueling Bandits (Yue et al., '09), (Hofmann et al., IRJ '13)	**Multi-Page search** MDP-MPS (Zeng et al., '18); DPG-FBE (Hu et al., Arxiv '18);	**Session search** QCM (Guan et al, '13); Win-Win (Luo et al,'14); DPL (Luo et al, '15)

Multi-Page Search

- Web search engines typically provide **multiple pages** of search results, each contains 10 blue links
- Recall minded or exploratory search users are likely to access more than one page
- How to rank the remaining webpages given historical user actions?

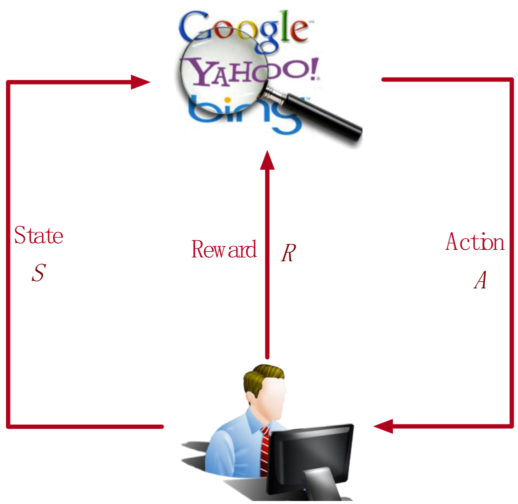


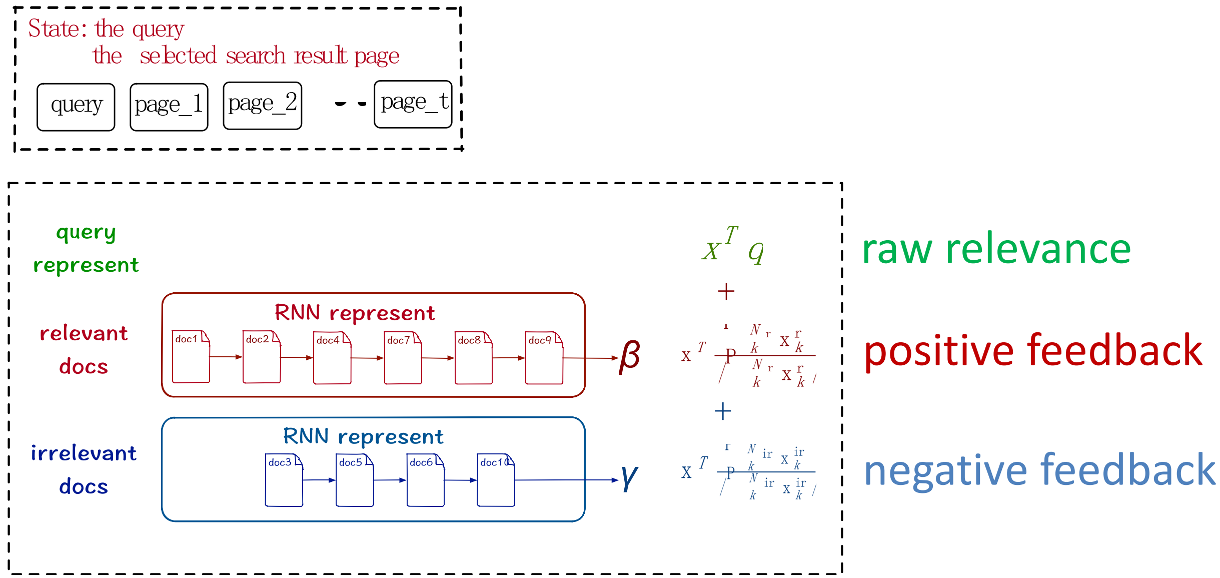
Multi-Page Search as MDP



- Agent (Search engine)
  - Construct search result page
- Environment (user)
  - Issues query, takes actions based on the search results
- Reward
  - Based on user activities, e.g., clicks, dwell time

Relevance Feedback based on MDP
MDP-MPS (Zeng et al., ICTIR '18)





- MDP as a relevance feedback model
  - State: query, user historical clicks
  - Policy: rank score = raw relevance + positive feedback + negative feedback
  - Action: construct a search result page based on policy
  - Reward: DCG improvements over the result page
- Learning: maximizing the cumulated rewards

$$L(\Theta) = \mathbb{E}_{\mathcal{E}\sim\pi}[\sum_{k=1}^{M\times T} \hat{\gamma}^{k-1} r_k].$$

E-commerce Search as MDP
DPG-FBE (Hu et al., Arxiv '18)

- Product search as multi-step ranking
  - 1. User issues a query
  - 2. Search engine ranks items related to the query and displays top K
  - 3. User makes operations (continue, convention, abandon) on the page
  - 4. User issue page request, search engine re-ranks the rest of items and display top K
  - ……

DPG-FBE (cont’)

$S = H_C \cup H_B \cup H_L$

1. all continuation events

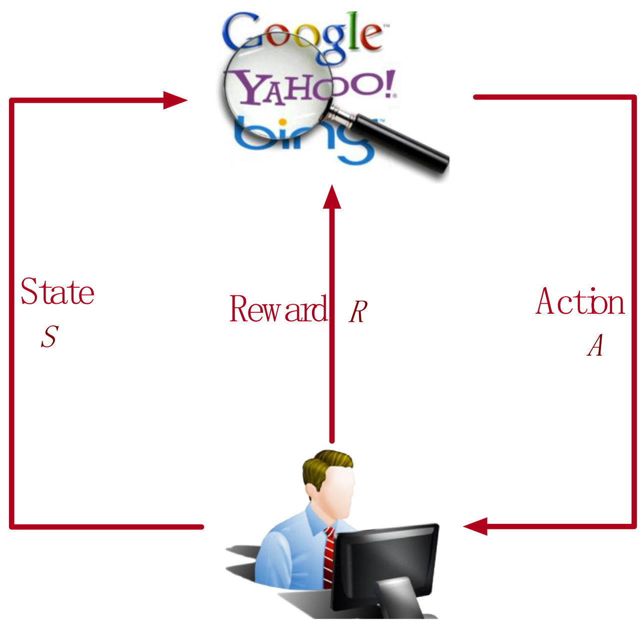
$$H_C = \{C(h_t) \mid \forall h_t \in H_t, 0 \leq t < T\}$$

2. all conversion events

$$H_B = \{B(h_t) \mid \forall h_t \in H_t, 0 < t \leq T\}$$

3. all abondon events

$$H_L = \{L(h_t) \mid \forall h_t \in H_t, 0 < t \leq T\}$$



The action space A
contains all possible ranking functions

At each time step,
the search engine chose a rank function
which could construct a item page

- The measure metric as reward:

$$\mathcal{R}(C(h_t), a, s') = \begin{cases} m(h_{t+1}) & \text{if } s' = B(h_{t+1}), \\ 0 & \text{otherwise,} \end{cases}$$

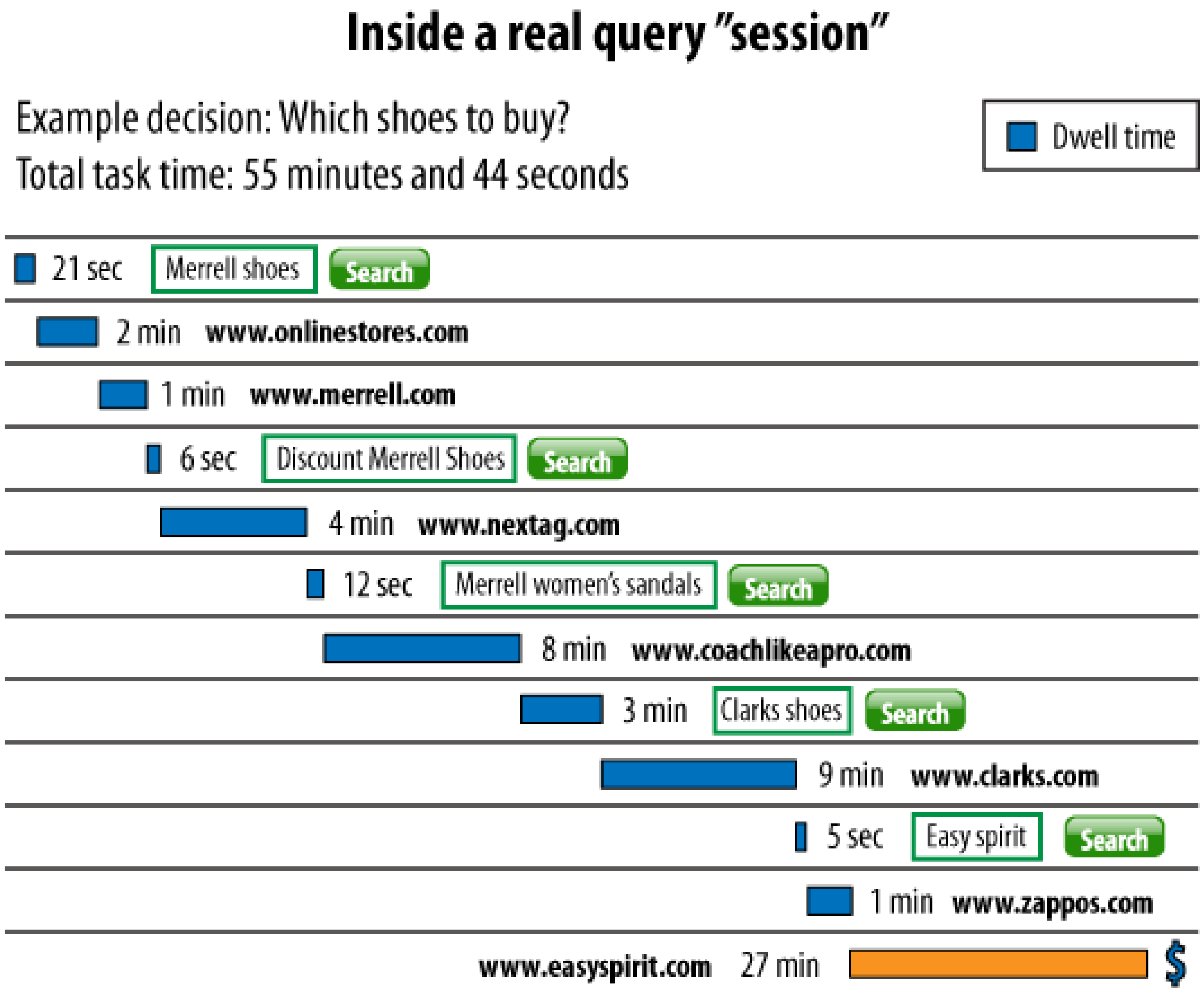
- Maximize the reward:

$$L(\Theta) = \mathbb{E}_{\mathcal{E}\sim\pi}[\sum_{k=1}^{M\times T} \hat{\gamma}^{k-1} r_k].$$

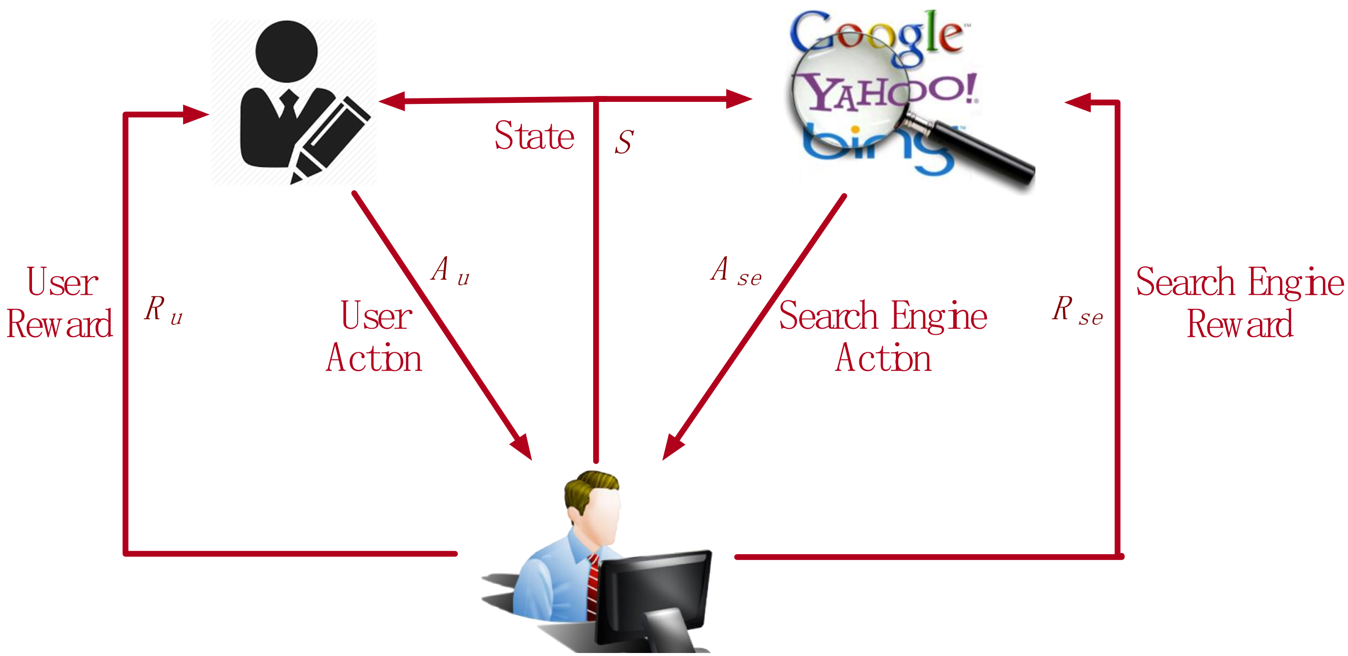
RL Approaches to IR

		Granularity of Time Steps		
		One item per step	One result page per step	One query per step
Source of Rewards	Simulation	**Relevance ranking** MDPRank (Zeng et al., '17) **Diverse ranking** MDP-DIV (Xia et al., '17); M2Div (Feng et al., '18)	N/A	N/A
	Real users	**Online ranking** Dueling Bandits (Yue et al., '09), (Hofmann et al., IRJ '13)	**Multi-Page search** MDP-MPS (Zeng et al., '18); DPG-FBE (Hu et al., Arxiv '18);	**Session search** QCM (Guan et al, '13); Win-Win (Luo et al,'14); DPL (Luo et al, '15)

Session Search



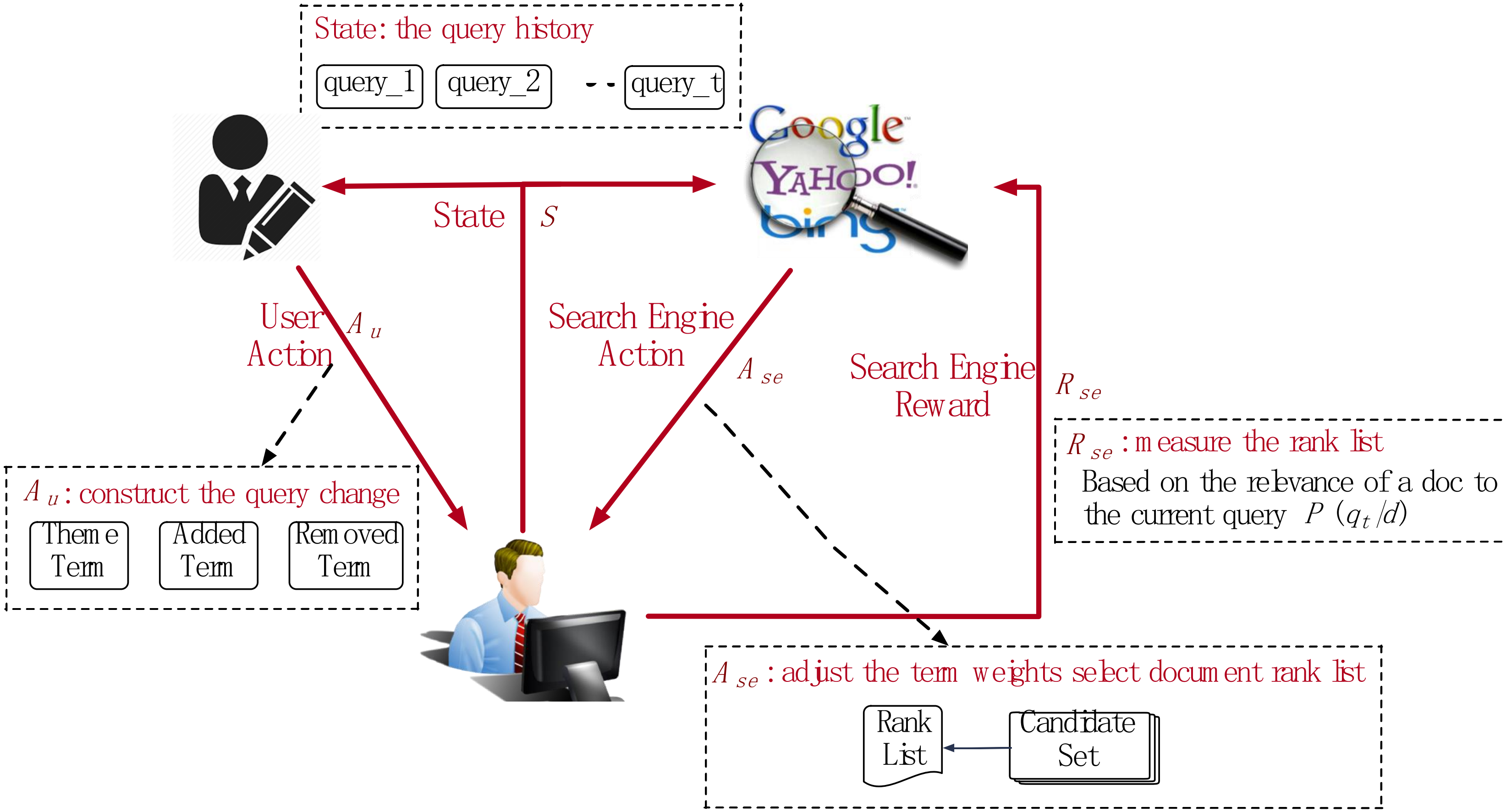
Session search as dual agent game



- User agent browns the document rank list and change query.
  - User action A_u => Query change (Theme terms, Added terms and Removed terms)

- Search engine agent Observes the query change from the user agent and construct the rank list.
  - Search engine action A_{se} => Adjustments on the term weights, (decreasing, increasing and maintaining term weights).

Query Change Model
(Guan et al., SIGIR'13)



- Model the relevant of a document d to the current query q_i as

$$score(q_i, d) = P(q_i|d) + \gamma \sum_a P(q_i|q_{i-1}, D_{i-1}, a) \max_{D_{i-1}} P(q_{i-1}|D_{i-1})$$

Experimental result

Search accuracy on TREC 2012 Session

Approach	nDCG@10	nDCG	MAP	nERR@10
Lemur	0.2474	0.2627	0.1274	0.2857
TREC’12 median	0.2608	0.2648	0.1440	0.2626
TREC’12 best	0.3221	0.2865	0.1559	0.3595
PRF	0.2074	0.2335	0.1065	0.2415
Rocchio	0.2446	0.2714	0.1281	0.2950
Rocchio-CLK	0.2916	0.2866	0.1449	0.3366
Rocchio-SAT	0.2889	0.2836	0.1467	0.3254
QCM	**0.3353**	**0.3054**	**0.1529**	**0.1534**
Win-Win	0.2941	0.2691	0.1346	0.3403

RL Approaches to IR

		Granularity of Time Steps		
		One item per step	One result page per step	One query per step
Source of Rewards	Simulation	**Relevance ranking** MDPRank (Zeng et al., ‘17) **Diverse ranking** MDP-DIV (Xia et al., ’17); M2Div (Feng et al., ‘18)	**Opportunity!**	
	Real users	**Online ranking** Dueling Bandits (Yue et al., ‘09), (Hofmann et al., IRJ ’13)	**Multi-Page search** MDP-MPS (Zeng et al., ’18); DPG-FBE (Hu et al., Arxiv ’18); IES (Jin et al, ’13)	**Session search** QCM (Guan et al, ’13); Win-Win (Luo et al,’14); DPL (Luo et al, ’15)

Discussion
Environment Simulation v.s. Real User

- Almost all methods try to simulate the user actions
  - Interaction with real users are expensive (time, implementation etc.)
  - Nonoptimal results hurt user experience
  - Seems the click models trained with log data work well in most cases
  - On-policy algorithms were well studied

- However, simulated responses $\neq$ real user responses
  - Performances heavily depend on the quality of simulation (e.g., calculation of the rewards)
  - Can the simulation model generate well to all queries and documents?

Discussion
on-policy v.s. off-policy

- On-policy: learn policy π from experience sampled from π
  - Need real-time interactions with search users,
  - or simulated environment

- Off-policy: learn policy π from experience sampled from μ
  - Training: learn ranking policy π from click-through / labeled data (data sampled from μ)
  - Online ranking: ranking document with π (usually only exploitation)
  - Available of large scale click-through data making off-policy attractive

Discussion
Modeling the Environment

- Environment accepts state and action, outputs next state and reward
- MDP-DIV and MDPRank: rewards based on human relevance labels
  - Cannot generalize to new queries and documents
  - Training: exploration + exploitation;
    Online ranking: exploitation only
- M^2Div: Monte Carlo tree search based on value estimation
  - On-policy: identical policy at training and online

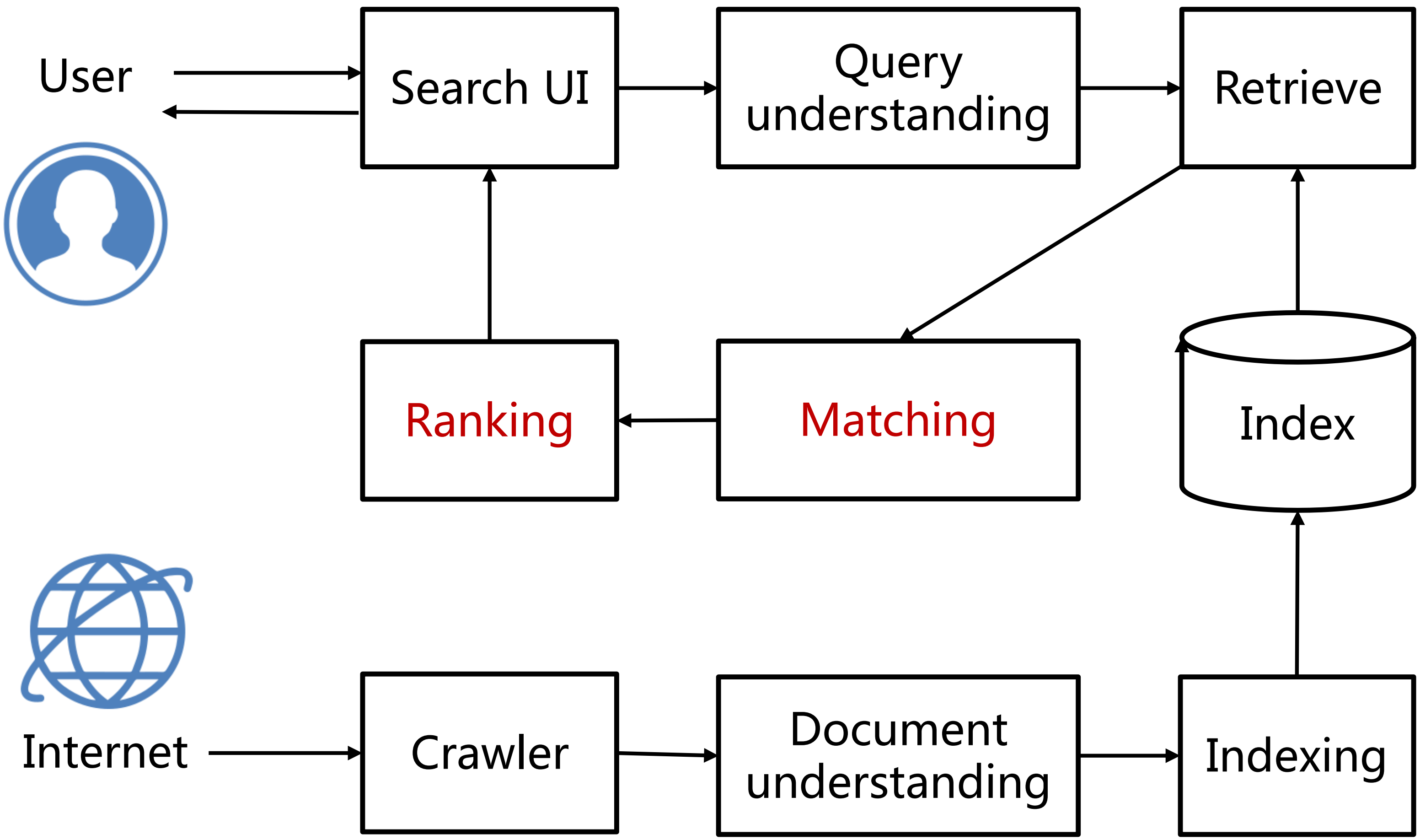
Looking Forward: Beyond Ranking

- Reinforcement information retrieval
  - Semantic matching (He et al., submitted to CCIR '18)
  - Sequence tagging (Lao et al., ArXiv '18)
  - Gradient quantization (Cui et al., ICTIR '18)
- Reinforcement information access
  - IR/Recommendation/Ads: two sides of the same coin

Outline

- Introduction
- Deep Semantic Matching
  - Methods of Representation Learning
  - Methods of Matching Function Learning
- Reinforcement Learning to Rank
  - Formulation IR Ranking with RL
  - Approaches
- Summary

Summary


User
Search UI
Query understanding
Retrieve
Ranking
Matching
Index
Internet
Crawler
Document understanding
Indexing

Deep Semantic Matching

- Methods of Representation Learning
  - Step 1: calculate representation $\phi(x)$
  - Step 2: conduct matching $F\big(\phi(x), \phi(y)\big)$
- Methods of Matching Function Learning
  - Step 1: construct basic low-level matching signals
  - Step 2: aggregate matching patterns
- Similarity Matching ≠ Relevance Matching
  - Methods based on global distributions of matching strengths
  - Methods based on local context of matched terms

Reinforcement Learning to Rank

- Ranking as agent-environment interaction
  - Agent: search engine
  - Environment: user
- Different definitions of time steps and rewards leads to different RLTR algorithms
  - Relevance ranking
  - Diverse ranking
  - Online learning to rank
  - Session search

  ……

Challenges

- Data: building better benchmarks
  - Large-scale text matching data
  - Large-scale user-item matching data with rich attributes.
- Model: data-driven + knowledge-driven
  - Most current methods are purely data-driven
  - Prior information (e.g., domain knowledge, large-scale knowledge based) is helpful and should be integrated into data-driven learning in a principled way.
- Task: multiple criteria
  - Existing work have primarily focused on similarity
  - Different application scenarios should have different matching goals
  - Other criteria such as novelty, diversity, and explainability should be taken into consideration

Thanks!

Jun Xu, Liang Pang

Institute of Computing Technology
Chinese Academy of Sciences

中国科学院计算技术研究所
INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES