Information Retrieval

Zhiyuan Liu

liuzy@tsinghua.edu.cn

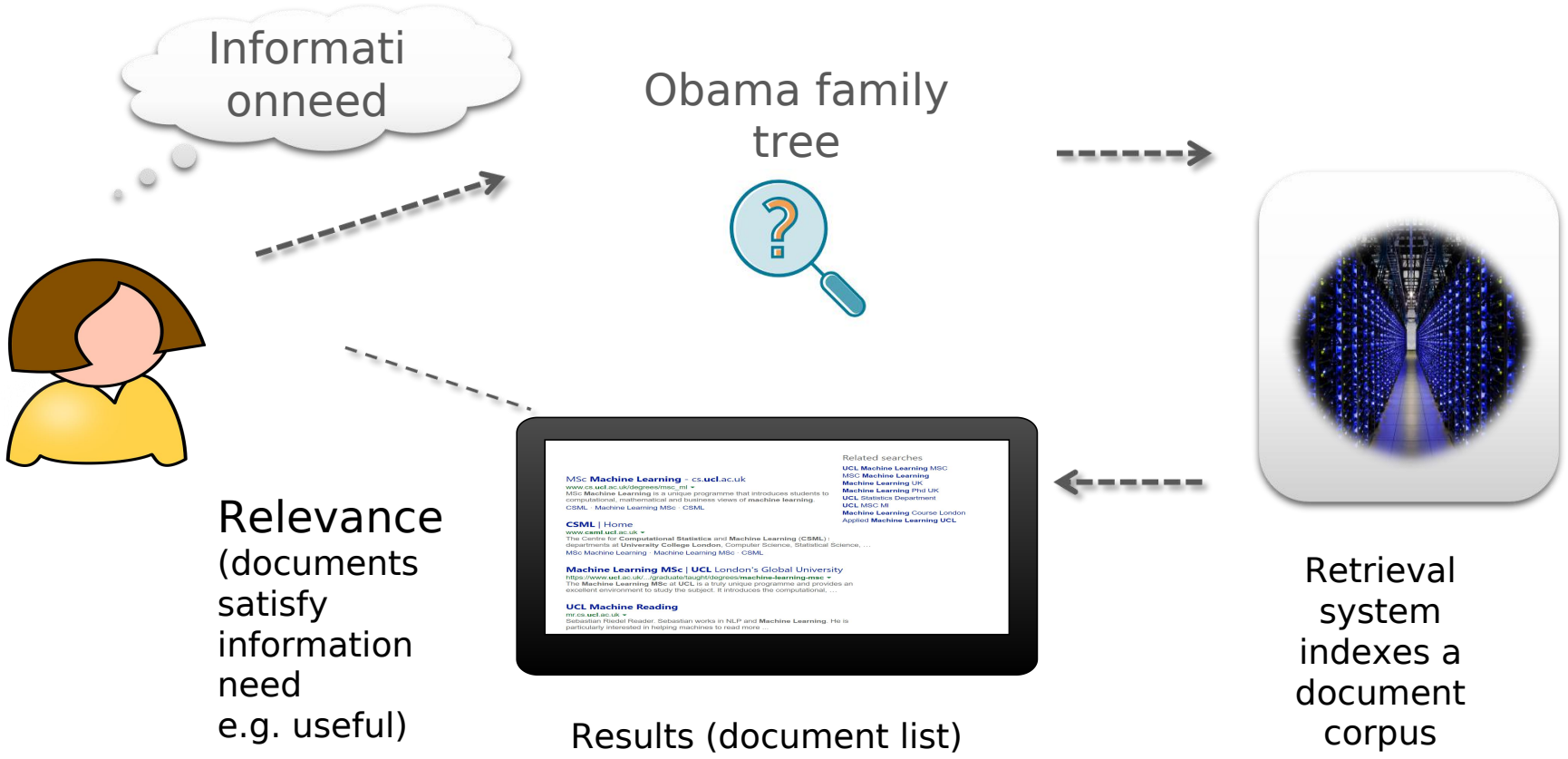
THUNLP

Outline

- Introduction to IR
  - What is Information Retrieval (IR)
  - Evaluation of IR Models
- Traditional IR Models
- Word Embedding for IR
- Neural IR Models

Information Retrieval

- What is Information Retrieval (IR)



Relevance between text queries and documents

Information Retrieval

- Applications of Information Retrieval
  - Document Ranking
    - Query: Obama family tree
    - Document:
      - **Family of Barack Obama** - Wikipedia
      - Barack **Obama Family Tree** along with family connections to other famous kin. Genealogy charts for Barack Obama may include up to 30 generations of ...
  - Question Answering
    - Query: Who is Barack Obama's sister?
    - Answer:

Maya Soetoro-Ng

Auma Obama

Information Retrieval

- Applications of Information Retrieval
  - The applications of IR can be divided into two categories:
    - Document Ranking and Question Answering

	Document Ranking	Question Answering
Query	Keywords	Natural language question
Document	Web page, news article	A fact and supporting passage
Research solution	Traditional IR Neural IR	Open Domain QA Generative QA Reading Comprehension Fact Verification
In products	Document rankers at: Google, Bing, Baidu…	Microsoft Xiaoice Watson@Jeopardy

Outline

- Introduction to IR
  - What is Information Retrieval (IR)
  - Evaluation of IR Models
- Traditional IR Models
- Word Embedding for IR
- Neural IR Models

Evaluation of IR Models

- Mean Average Precision (MAP)
  - MAP for a set of queries is the mean of the average precision score for each query

$$MAP = \frac{\sum_{q=1}^{Q} AveP(q)}{Q}$$

where Q is the number of queries

  - Suppose you have two queries:
    - **Query1:** **Four related docs**. Four related docs were retrieved and their ranks are **1, 2, 4 and 7**

$$AveP(Q1) = (1/1 + 2/2 + 3/4 + 4/7)/4 = 0.83$$

    - **Query2:** **Five related docs**. Three related docs were retrieved and their ranks are **1, 3 and 5**

$$AveP(Q2) = (1/1 + 2/3 + 3/5 + 0 + 0)/5 = 0.45$$

$$MAP = (0.83 + 0.45) / 2 = 0.64$$

Evaluation of IR Models

- Discounted Cumulative Gain (DCG)
  - DCG is often used to measure the effectiveness of web search engine and recommender systems

$$DCG_p = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{log_2(i+1)}$$

Where p is the number of docs, i is the rank position of document and rel_i is the grade of the doc

  - Why we need DCG?
    - In MAP, docs are either related (1) or unrelated (0) to the query
    - In DCG, the degree of correlation is divided into r+1 grades from 0 to r (r can be set)

Evaluation of IR Models

- Discounted Cumulative Gain (DCG)
  - For example, when you search for a query:
    - You get five results and classify them into three grades: Good (3), Fair (2) and Bad (1)
    - The grade values of these five results are $\boldsymbol{rel_1} = 3$, $\boldsymbol{rel_2} = 1$, $\boldsymbol{rel_3} = 2$, $\boldsymbol{rel_4} = 3$, and $\boldsymbol{rel_5} = 2$
    - DCG =**7** * **1** + **1** * **0.63** + **3** * **0.50** + **7** * **0.43** + **3** * **0.39** = 13.

Rank	rel_i	$2^{rel_i} - 1$	$\frac{1}{log_2(i+1)}$
1	3	**7**	**1**
2	1	**1**	**0.63**
3	2	**3**	**0.50**
4	3	**7**	**0.43**
5	2	**3**	**0.39**

$$DCG_p = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{log_2(i+1)}$$

Evaluation of IR Models

- Normalized DCG (NDCG)
  - Why we need NDCG?
    - The number of different search results may not be equal, DCG values for different searches cannot be directly compared
  - Ideal DCG (IDCG): The DCG value of gold-standard relevant documents (ordered by their ideal rank)

$$iDCG_p = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{log_2(i+1)}$$

  - NDCG

$$nDCG_p = \frac{DCGp}{iDCGp}$$

Evaluation of IR Models

- Normalized DCG (NDCG)
  - For example, when you search for a query:
    - You get five results and classify them into three grades: Good(3), Fair(2) and Bad(1)
    - The grade values of these five results are $rel_1 = 3$, $rel_2 = 1$, $rel_3 = 2$, $rel_4 = 3$, and $rel_5 = 2$
    - In this case, the best rank relevance is **3, 3, 2, 2, 1**
      - IDCG = **7** * **1** + **7** * **0.63** + **3** * **0.50** + **3** * **0.43** + **1** * **0.39** = 14.59

$$DCG_p = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{log_2(i+1)}$$

$$NDCG_p = \frac{DCGp}{iDCGp} = 13.31/14.59 = 0.9$$

Rank	rel_i	$2^{rel_i} - 1$	$\frac{1}{log_2(i+1}$
1	3	**7**	**1**
2	3	**7**	**0.63**
3	2	**3**	**0.50**
4	2	**3**	**0.43**
5	1	**1**	**0.39**

Evaluation of IR Models

- Mean Reciprocal Rank (MRR)
  - MRR evaluates lists of possible responses to a query set, ordered by the probability of correctness
  - MRR is the average of the reciprocal ranks of the first relevant results for a query set Q:

$$MRR = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i}$$

  - For example
    - MR

Query	Search Results	$\frac{1}{rank_i}$
cat	catten, cati, **cats**	1/3
torus	torii, **tori**, toruses	1/2
virus	**viruses**, virii, viri	1

Evaluation of IR Models

- The differences of MAP, NDCG and MRR
  - **MAP** only divides docs into related and unrelated categories
  - **DCG** and **NDCG** score docs into different levels according to the relevance with the query
  - **MRR** only considers the first rank of the relevant doc

Outline

- Introduction to IR
- Traditional IR Models
- Word Embedding for IR
- Neural IR Models

Traditional IR Methods

- Language modeling approach of IR
  - Given a query q and document d:

$$p(d|q) \approx p(q|d)p(d)$$

    - $\mathrm{p}(d)$ can be assumed uniform across docs
    - $\mathrm{p}(q|d) = \prod_{w \in q} p(w|d)$ depends on how to model the relationship of query word and doc
  - The language modeling approach is quite extensible
    - TF-IDF; BM25 ...

Traditional IR Methods

- TF-IDF
  - Term Frequency (TF)
    - The weight of a term that occurs in a document is simply proportional to the term frequency
    - The number of times that term t occurs in document d:

$$tf(t, D) = \frac{n_t}{n_d}$$

    - Where n_t is the number of times the term t appears in d, and n_d is the word number of the document d

Traditional IR Methods

- TF-IDF
  - Inverse Document Frequency (IDF)
    - The specificity of a term can be quantified as an inverse function of the number of documents in which term t appears
    - IDF is a measure to evaluate if term t is common or rare across the document collection D

$$\mathrm{IDF}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

    - Where N is the total number of documents in the corpus, and $|\{d \in D : t \in d\}|$ denotes the number of documents where the term t appears

Traditional IR Methods

- TF-IDF
  - A high TF-IDF value of term t requires:
    - High term frequency (TF) in the given document
    - Low document frequency (IDF) of the term in the whole collection of documents

$$\mathrm{TF\_IDF}(t, D) = \mathrm{TF}(t, D) \cdot \mathrm{IDF}(\mathrm{t}, \mathrm{D})$$

Traditional IR Methods

- BM25
  - BM25 is a bag-of-word retrieval model
    - Given a query Q, which contains n words q_1, ...q_n, the BM25 score of a document D is:

$$score(D,Q) = \sum_{i=1}^{n} IDF(q_i) \cdot \frac{f(q_i, D) \cdot (k+1)}{f(q_i, D) + k \cdot \left(1 - b + b \cdot \frac{|D|}{avgdl}\right)}$$

      - Where $f(q_i, D)$ is the term frequency of q_i in the document D, $|D|$ is the length of D, and $avgdl$ is the average document length in the document collection
    - BM25 aims to normalize term frequency according to document length

Traditional IR Methods

- BM25
  - BM25 is a bag-of-words retrieval

$$score(D,Q) = \sum_{i=1}^{n} IDF(q_i) \cdot \frac{f(q_i, D) \cdot (k+1)}{f(q_i, D) + k \cdot \left(1 - b + b \cdot \frac{|D|}{avgdl}\right)}$$

    - k and b are free parameters:
      - If k is large enough, $\frac{f(q_i,D)\cdot(k+1)}{f(q_i,D)+k\cdot\left(1-b+b\cdot\frac{|D|}{avgdl}\right)} \approx f(q_i, D)$
      - If $k = 0$, $\frac{f(q_i,D)\cdot(k+1)}{f(q_i,D)+k\cdot\left(1-b+b\cdot\frac{|D|}{avgdl}\right)} = 1$
      - $0 \leq b \leq 1$:
        - If $b = 0$, we do not consider document length
        - If $b = 1$, we normalize term frequency totally according to document length

Traditional IR Methods

- Neural Translation Language Model (NTLM)
  - Extends query likelihood:

$$p(d|q) \sim p(q|d)p(d)$$

$$p(q|d) = \prod_{t_q \in q} p(t_q|d)$$

$$p(t_q|d) = \sum_{t_d \in d} p(t_q|t_d)p(t_d|d)$$

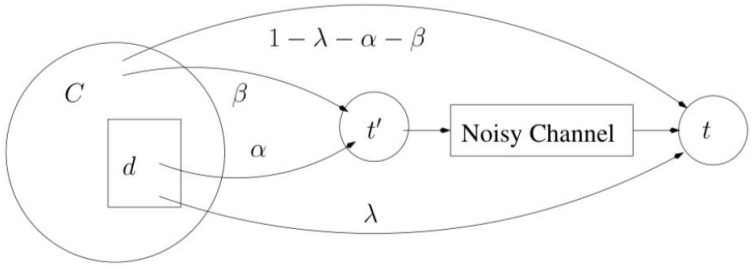
  - Uses the similarity between term embeddings as a measure for term-term translation probability $p(t_q|t_d)$

$$p(t_q|t_d) = \frac{\cos(\vec{v}_{t_q}, \vec{v}_{t_d})}{\sum_{t \in V} \cos(\vec{v}_t, \vec{v}_{t_d})}$$

Berger and Lafferty, Information retrieval as statistical translation. SIGIR 1999

Traditional IR Methods

- Generalized Language Model (GLM):
  - Term t in a query is generated by sampling independently from either the document or the document collection



$$p(t|d) = \lambda p(t|d) + \alpha \sum_{t' \in d} p(t\ |t')p(t'|d) + \beta \sum_{t' \in N_t} p(t|t')p(t'|C) + (1-\lambda-\alpha-\beta)\ p(t\ |C)$$

  - The noisy channel may transform (mutate) a term t' into a term t. Term t'' is sampled from its nearest neighbors

$$p(t|t') = \frac{sim(\vec{v}_{t'}, \vec{v}_t)}{\sum sim(\vec{v}_{t'}, \vec{v}_{t''})}$$

Ganguly, et al. Word embedding based generalized language model for information retrieval. SIGIR 2015

Traditional IR Methods

- Sequential Dependence Model (SDM):
  - Models term dependence for IR
  - Provides a good balance between retrieval effectiveness and efficiency
  - The SDM score is calculated with:
    - Unigram term frequency f_T
    - Bigram term frequency f_O (with order) and f_U (unorder)

$$p(q|d) = \lambda_T \sum_{t_q^i \in q} f_T(t_q^i|d) + \lambda_O \sum_{t_q^i, t_q^{i+1} \in q} f_O(t_q^i, t_q^{i+1}|d) + \lambda_U \sum_{t_q^i, t_q^{i+1} \in q} f_U(t_q^i, t_q^{i+1}|d)$$

    - Where $\lambda_T + \lambda_O + \lambda_U = 1$

D. Metzler and W. B. Croft, A Markov Random Field model for term dependencies. SIGIR 2015

Traditional IR Methods

- Pros
  - Have ability to deal with large scale data
  - Do not need annotated labels
- Cons
  - Have vocabulary mismatch problem
  - Perform shallow understanding for queries and documents

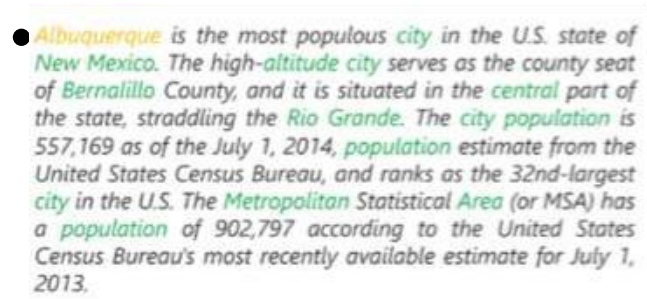
Traditional IR Methods

- Vocabulary mismatch
  - Q: How many people live in Sydney?

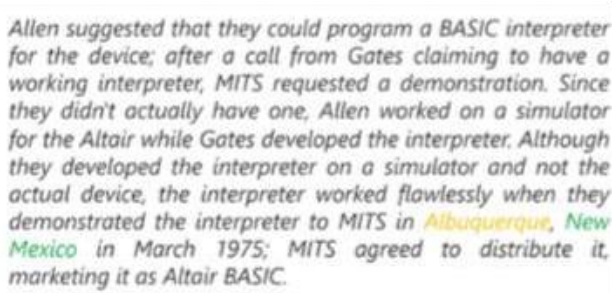
    Sydney's population is 4.9 million
    [relevant, but missing 'people' and 'live']

    Hundreds of people queueing for live music in Sydney
    [irrelevant, and matching 'people' and 'live']

- Perform shallow understanding for queries and documents



Passage about Albuquerque



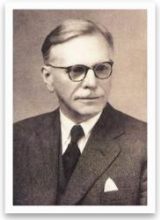
Passage not about Albuquerque

Outline

- Introduction to IR
- Traditional IR Models
- Word Embedding for IR
  - Word Representation in IR
  - Term Weight with Embedding
  - Query Expansion with Embedding
- Neural IR Models

Word Representation in IR

- Distributional semantics
  - Linguistic items with similar distributions (e.g. context words) have similar meanings



"You shall know a word by the company it keeps"

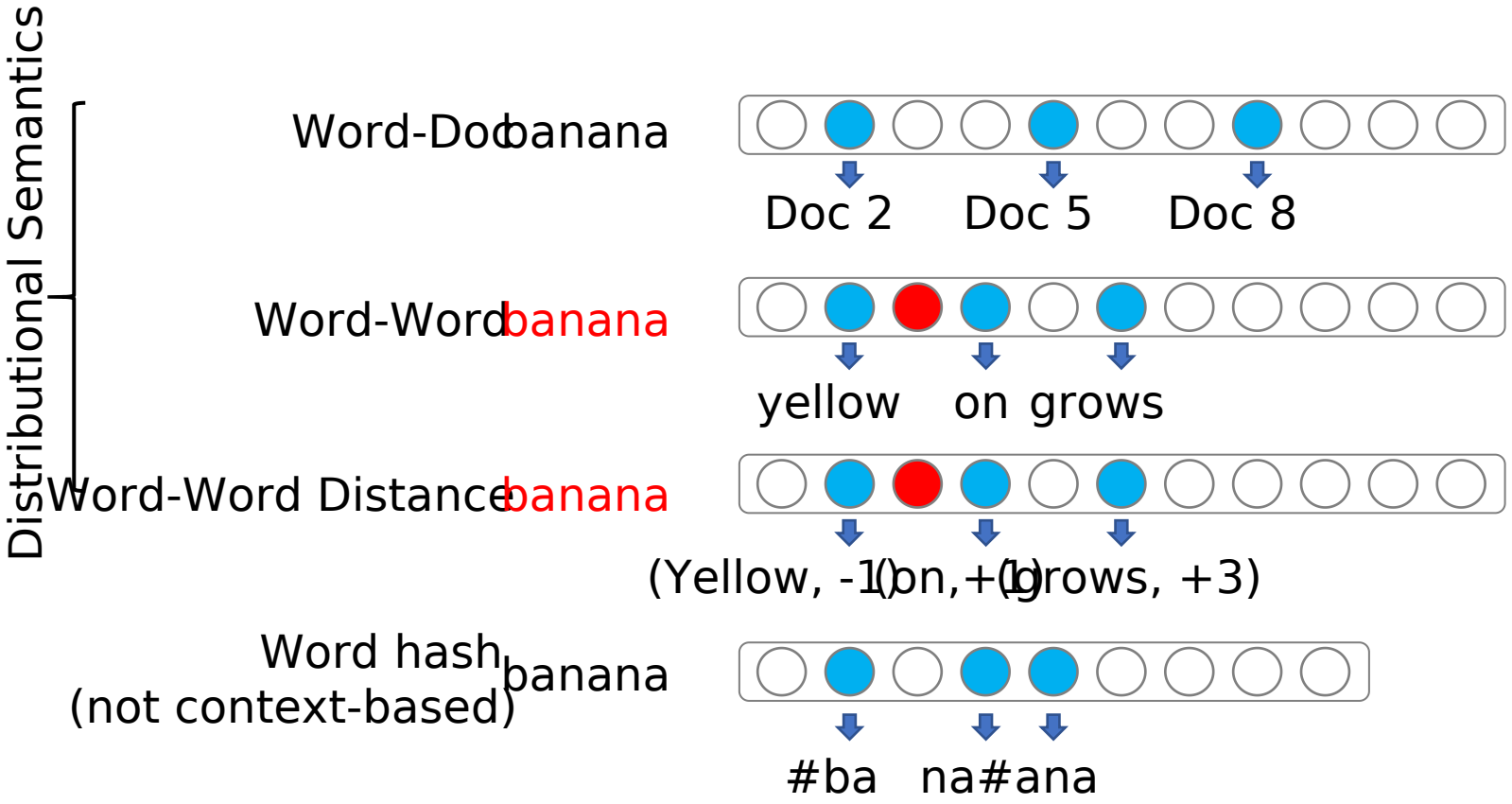
John Rupert Firth

banana — *Bananas* are one of the world's most appealing fruits.

apple — An *apple* is a sweet, edible fruit produced by an **apple** tree (Malus domestica)

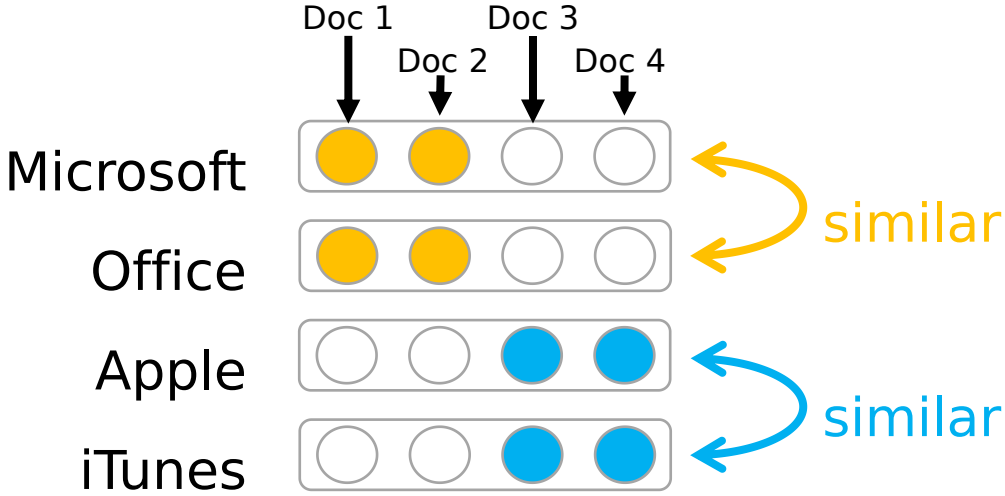
Word Representation in IR

- Different kinds of word representations



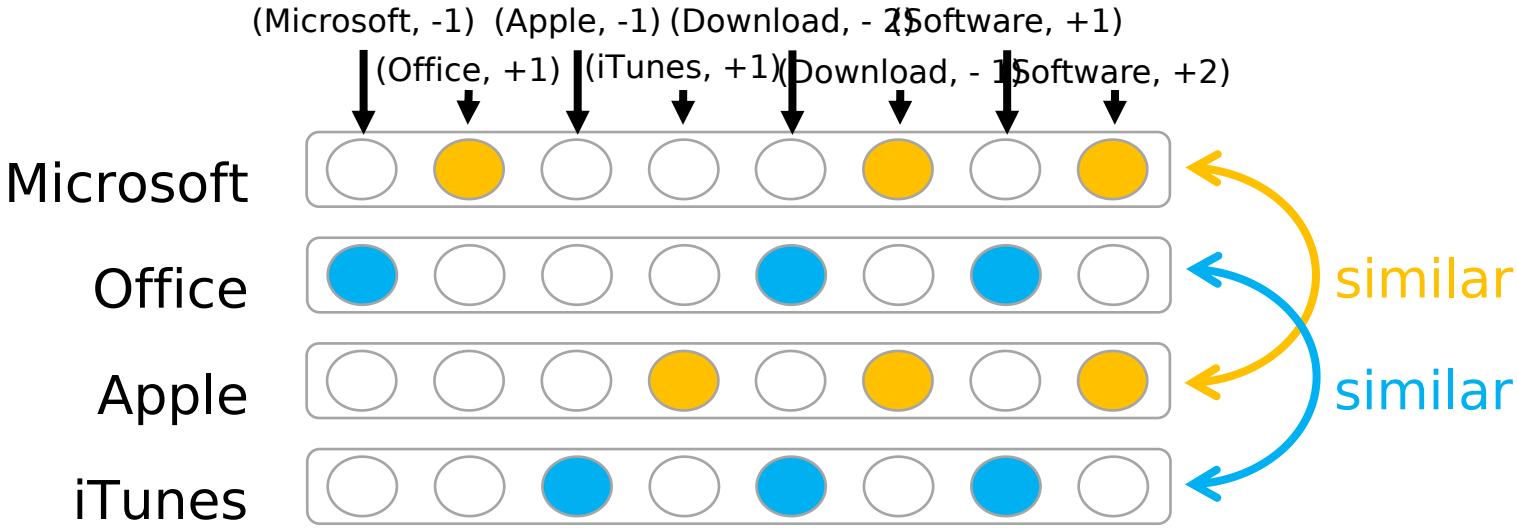
Word Representation in IR

- Word-Doc Context
  - Words co-occur in the same query or document (topical)
    - Doc 1 : “Microsoft Office Software”
    - Doc 2 : “Download Microsoft Office ”
    - Doc 3 : “Apple iTunes Software”
    - Doc 4 : “Download Apple iTunes”



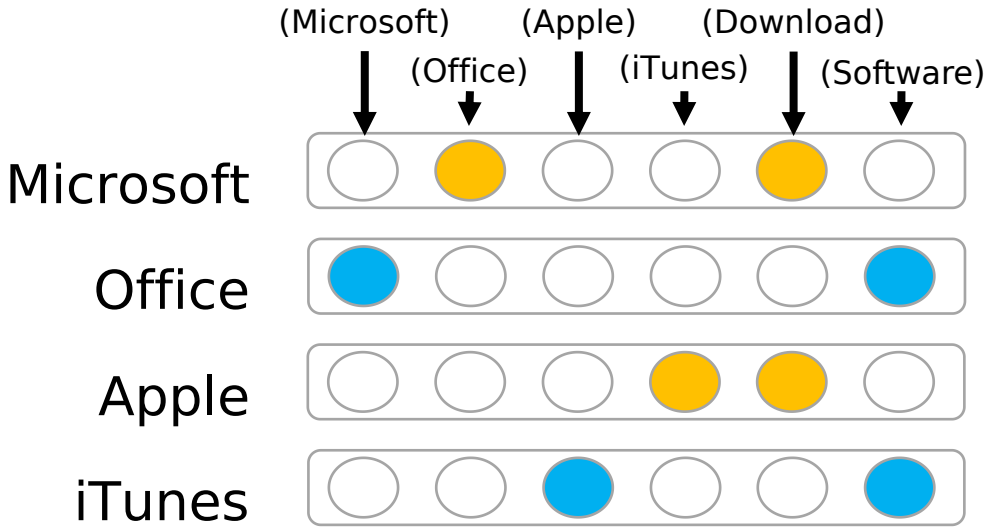
Word Representation in IR

- Word-Word Distance Context
  - Words share same function or type (typical)
    - Doc 1 : “Microsoft Office Software”
    - Doc 2 : “Download Microsoft Office ”
    - Doc 3 : “Apple iTunes Software”
    - Doc 4 : “Download Apple iTunes”



Word Representation in IR

- Word-Word Context
  - A mix of topical and typical similarity
    - Doc 1 : “Microsoft Office Software”
    - Doc 2 : “Download Microsoft Office ”
    - Doc 3 : “Apple iTunes Software”
    - Doc 4 : “Download Apple iTunes”



Word-Word Context is less sparse with more documents

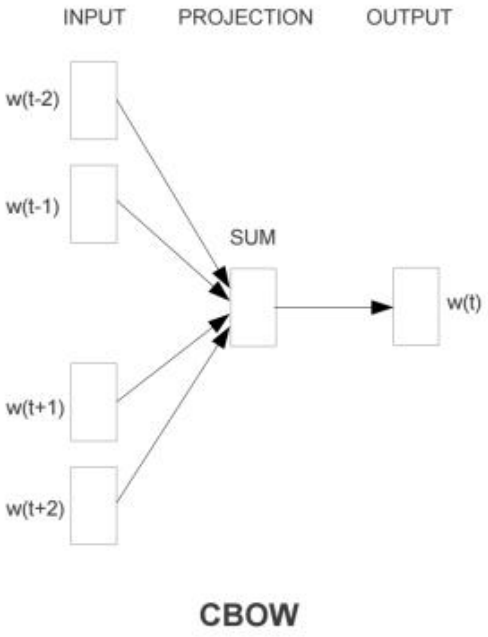
Word Representation in IR

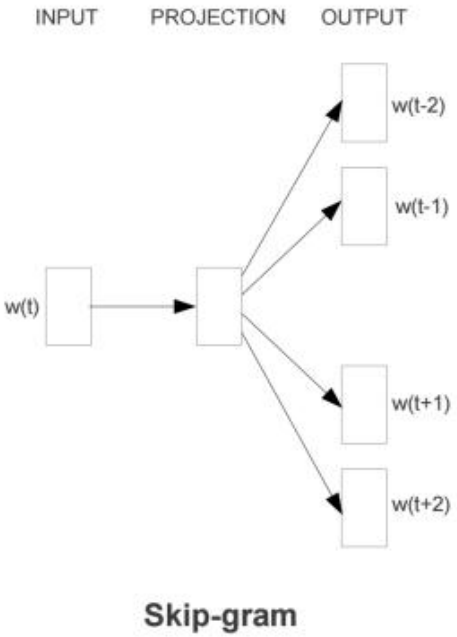
- Different kinds of word representations

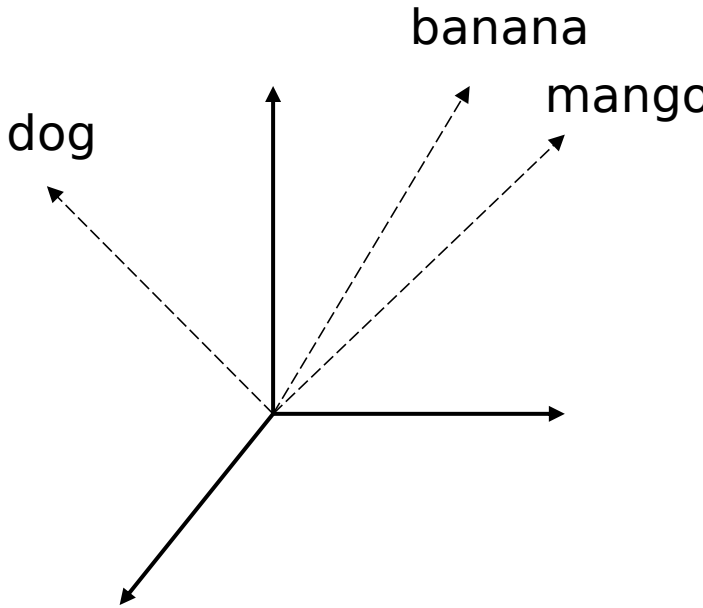
Context Data	Learning from counts matrix	Learning from individual instances
Word-Doc	LSA	Paragraph2Vec PV-DBOW
Word-Word	GloVe	Word2vec
Word-WordDistance		

Word Representation in IR

- Word Embedding

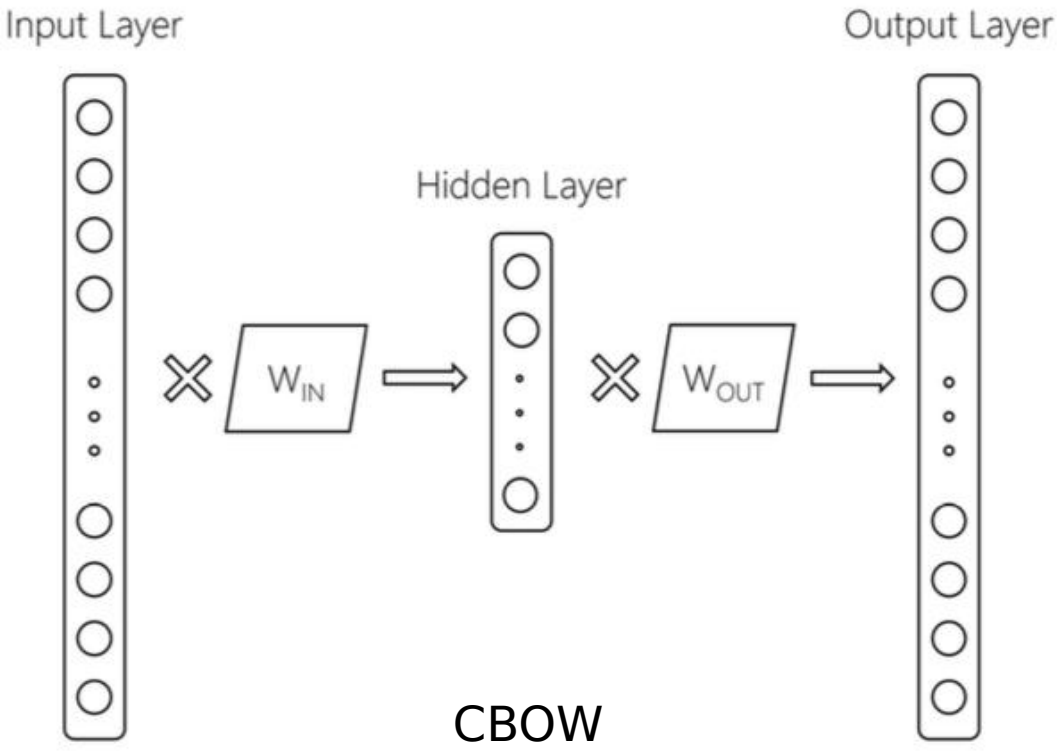






Word Representation in IR

- Dual Embedding Space Model (DESM)
  - Word embedding for IR style



Mitra, et al. A dual embedding space model for document ranking. 2016.

Word Representation in IR

- Dual Embedding Space Model (DESM)
  - DESM: Use IN-OUT similarity to model the aboutness between query and doc
    - A document is represented by the centroid of its word OUT vectors:

$$\vec{v}_{d,OUT} = \frac{1}{|d|} \sum_{t_d \in d} \frac{\vec{v}_{t_d,OUT}}{\|\vec{v}_{t_d,OUT}\|}$$

    - The query-document similarity is the average of cosine similarity over representations of query word and doc:

$$DESM_{IN-OUT}(q,d) = \frac{1}{|q|} \sum_{t_q \in q} \frac{\vec{v}^T_{t_q,IN} \vec{v}_{t_d,OUT}}{\|\vec{v}^T_{t_q,IN}\| \, \|\vec{v}_{t_d,OUT}\|}$$

Mitra, et al. A dual embedding space model for document ranking. 2016.

Word Representation in IR

- Dual Embedding Space Model (DESM)
  - IN-IN and OUT-OUT cosine similarities are high for words that are similar by function or type (typical)
  - IN-OUT cosine similarities are high between words that often co-occur in the same query or document (topical)

yale			seahawks			eminem		
IN-IN	OUT-OUT	IN-OUT	IN-IN	OUT-OUT	IN-OUT	IN-IN	OUT-OUT	IN-OUT
yale	yale	yale	seahawks	seahawks	seahawks	eminem	eminem	eminem
harvard	uconn	faculty	49ers	broncos	highlights	rihanna	rihanna	rap
nyu	harvard	alumni	broncos	49ers	jerseys	ludacris	dre	featuring
cornell	tulane	orientation	packers	nfl	tshirts	kanye	kanye	tracklist
tulane	nyu	haven	nfl	packers	seattle	beyonce	beyonce	diss
tufts	tufts	graduate	steelers	steelers	hats	2pac	tupac	performs

Mitra, et al. A dual embedding space model for document ranking. 2016.

Word Representation in IR

- Dual Embedding Space Model (DESM)
  - For the query Cambridge, what happens when a document with word giraffe replaced by Cambridge?
    - The document is scored low by DESM for the query Cambridge
    - It finds low ds in the document

Passage about giraffes

the giraffe (giraffa camelopardalis) is an african ungulate mammal, the tallest living terrestrial animal and the largest ruminant. its species name refers to its shape and its colouring. its chief distinguishing characteristics are its extremely long neck and legs, its , and its distinctive coat patterns. it is classified under the family , along with its closest extant relative, the okapi. the nine subspecies are distinguished by their coat patterns. the scattered range of giraffes extends from chad in the north to south africa in the south, and from niger in the west to somalia in the east. giraffes usually inhabit savannas, grasslands, and open woodlands.

Passage about giraffes, but 'giraffe' is replayed by 'Cambridge'

the cambridge (giraffa camelopardalis) is an african ungulate mammal, the tallest living terrestrial animal and the largest ruminant. its species name refers to its shape and its colouring. its chief distinguishing characteristics are its extremely long neck and legs, its , and its distinctive coat patterns. it is classified under the family , along with its closest extant relative, the okapi. the nine subspecies are distinguished by their coat patterns. the scattered range of giraffes extends from chad in the north to south africa in the south, and from niger in the west to somalia in the east. giraffes usually inhabit savannas, grasslands, and open woodlands.

Mitra, et al. A dual embedding space model for document ranking. 2016.

Word Representation in IR

- Dual Embedding Space Model (DESM)
  - For the query Cambridge, DESM is also confused by those docs about Oxford
    - It detects a high number of similar words in the document that frequently co-occur with the word Cambridge

Mitra, et al. A dual embedding space model for document ranking. 2016.

Outline

- Introduction to IR
- Traditional IR Models
- Word Embedding for IR
  - Word Representation in IR
  - Term Weight with Embedding
  - Query Expansion with Embedding
- Neural IR Models

Term Weight with Embedding

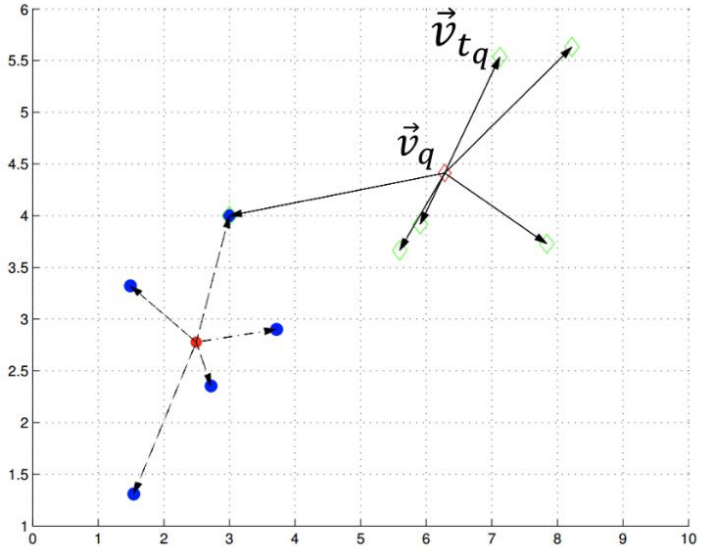
- Pre-trained word embedding for query term weighting
  - For the query Chinese river, word embedding gives several neighbors
    - The phrase Chinese river does not belong to the vocabulary of the model, we can average the embeddings of Chinese and river
    - The neighbor phrases are semantically related to the input

Word	Cosine similarity
Yangtze_River	0.667376
Yangtze	0.644091
Qiantang_River	0.632979
Yangtze_tributary	0.623527
Xiangjiang_River	0.615482
Huangpu_River	0.604726
Hanjiang_River	0.598110
Yangtze_river	0.597621
Hongze_Lake	0.594108
Yangtse	0.593442

Zheng and Callan, Learning to reweight terms with distributed representations. SIGIR 2015

Term Weight with Embedding

- Pre-trained word embedding for query term weighting



We calculate $|\vec{x}_{t_q}|$ to measure the semantic distance of a term to the whole query:

$$\vec{x}_{t_q} = \vec{v}_{t_q} - \frac{1}{|q|} \sum_{t'_q \in q} \vec{v}_{t'_q}$$

Where $\vec{v}_{t_q}$ is the embedding of term t_q and t'_q is the word from query other than t_q

Zheng and Callan, Learning to reweight terms with distributed representations. SIGIR 2015

Outline

- Introduction to IR
- Traditional IR Models
- Word Embedding for IR
  - Word Representation in IR
  - Term Weight with Embedding
  - Query Expansion with Embedding
- Neural IR Models

Query Expansion with Embedding

- Given a query: Albuquerque

Both passages have the same number of gold matches (exact match to the query word)

Those green matches cannot be found through the exact match

We need methods to consider these non-query terms

Automatic query expansion

Albuquerque is the most populous city in the U.S. state of New Mexico. The high-altitude city serves as the county seat of Bernalillo County, and it is situated in the central part of the state, straddling the Rio Grande. The city population is 557,169 as of the July 1, 2014, population estimate from the United States Census Bureau, and ranks as the 32nd-largest city in the U.S. The Metropolitan Statistical Area (or MSA) has a population of 902,797 according to the United States Census Bureau's most recently available estimate for July 1, 2013.

(a)

Allen suggested that they could program a BASIC interpreter for the device; after a call from Gates claiming to have a working interpreter, MITS requested a demonstration. Since they didn't actually have one, Allen worked on a simulator for the Altair while Gates developed the interpreter. Although they developed the interpreter on a simulator and not the actual device, the interpreter worked flawlessly when they demonstrated the interpreter to MITS in Albuquerque, New Mexico in March 1975; MITS agreed to distribute it, marketing it as Altair BASIC.

(b)

Query Expansion with Embedding

- Identify expansion terms with the cosine similarity of embeddings
- Three different strategies:
  - Pre-retrieval: Take nearest neighbors of query terms as the expansion terms
  - Post-retrieval: Use a set of pseudo-relevant docs that are retrieved at top ranks in response to the initial query to restrict the search domain for candidate expansion terms
  - Pre-retrieval incremental: Use an iterative process of reordering and pruning terms from the nearest neighbor list

Roy, et al. Using Word Embeddings for Automatic Query Expansion. Neu-IR Workshop 2016

Query Expansion with Embedding

- Identify expansion terms with the cosine similarity of embeddings

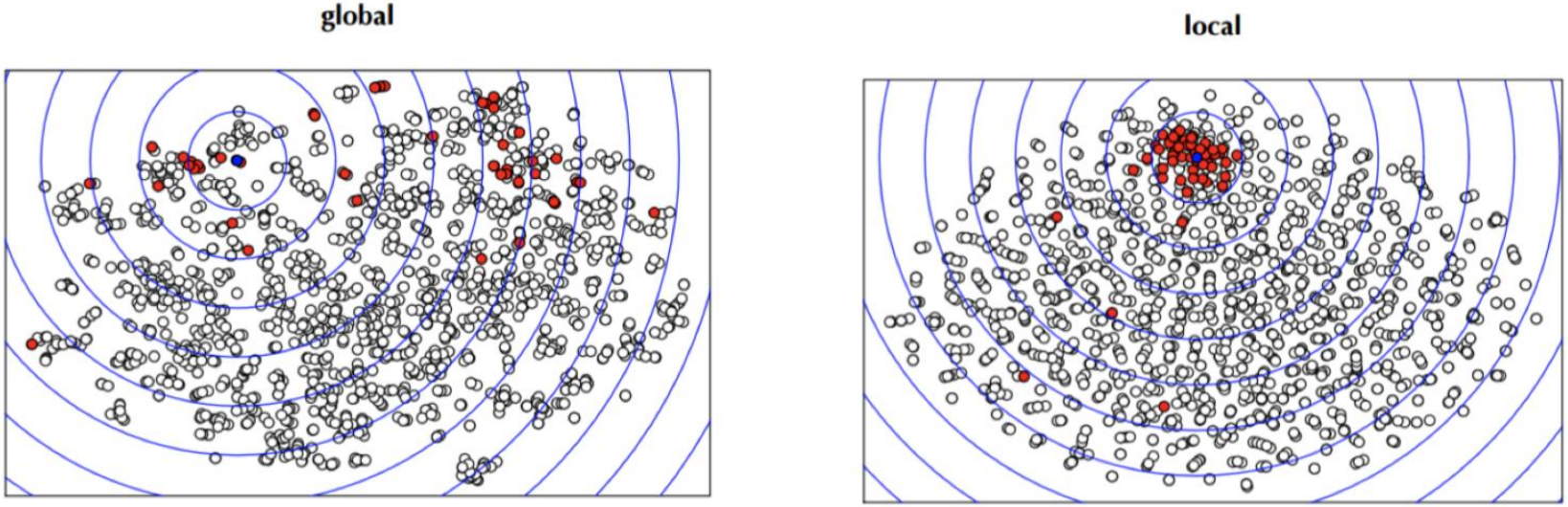
$$P(w|Q_{exp}) = \alpha P(w|Q) + (1-\alpha)\frac{sim(w,Q)}{\sum_{w_i \in Q_{exp}} sim(w_i,Q)}$$

  - Beats methods with no expansion methods, but does not beat methods with non-neural expansion

Roy, et al. Using Word Embeddings for Automatic Query Expansion. Neu-IR Workshop 2016

Query Expansion with Embedding

- Query expansion with locally-trained word embeddings
- Learned on the topically-constrained corpus
  - Train word2vec on docs from the retrieval process
  - Fine-grained word sense disambiguation



Diaz et al., Query Expansion with Locally-Trained Word Embeddings. ACL 2016.

Query Expansion with Embedding

- Query expansion with locally-trained word embeddings
- Terms that are similar to cut
  - Word2vec model trained on a general news corpus (left)
  - Train r

global	local
cutting	tax
squeeze	deficit
reduce	vote
slash	budget
reduction	reduction
spend	house
lower	bill
halve	plan
soften	spend
freeze	billion

line tax (right)

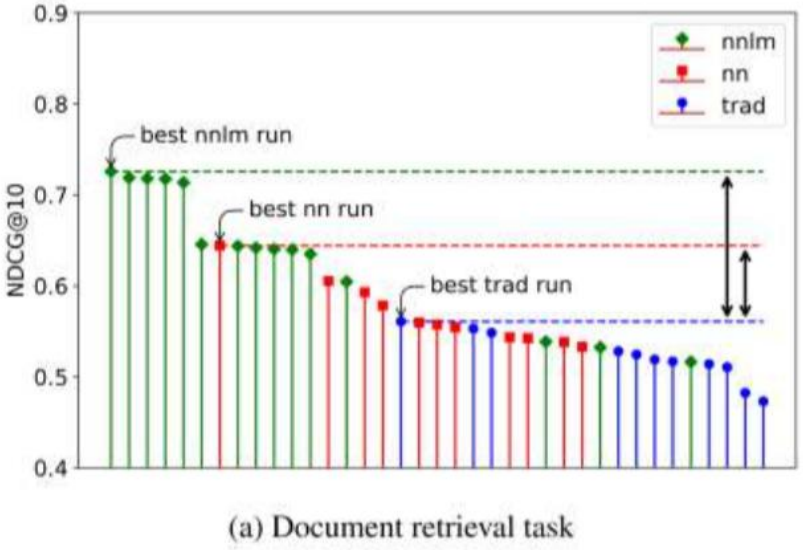
Diaz et al., Query Expansion with Locally-Trained Word Embeddings. ACL 2016.

Outline

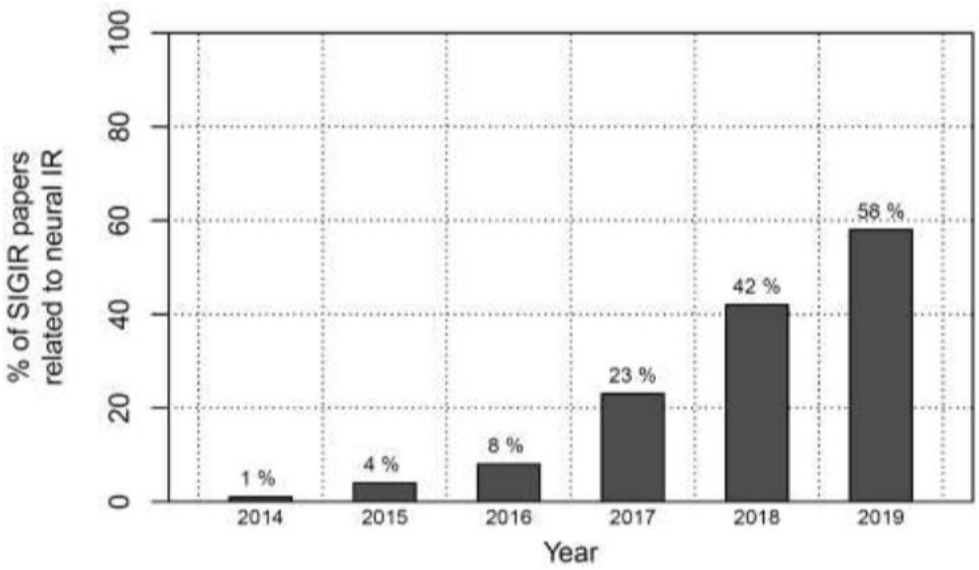
- Introduction to IR
- Traditional IR Models
- Word Embedding for IR
- Neural IR Models
  - Neural Models for IR
  - Representation-based IR Models
  - Interaction-based IR Models
  - Further Combination
  - Data Challenge in Neural IR

Neural Models for IR

- Why choose neural models
- Neural models outperform traditional IR models significantly
- Being neural has become a tendency for IR

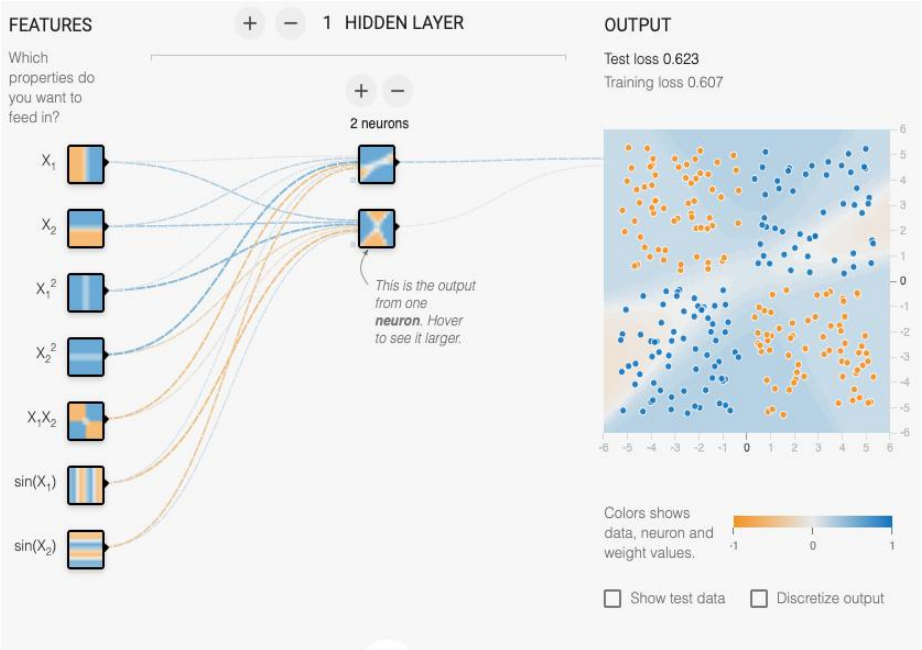


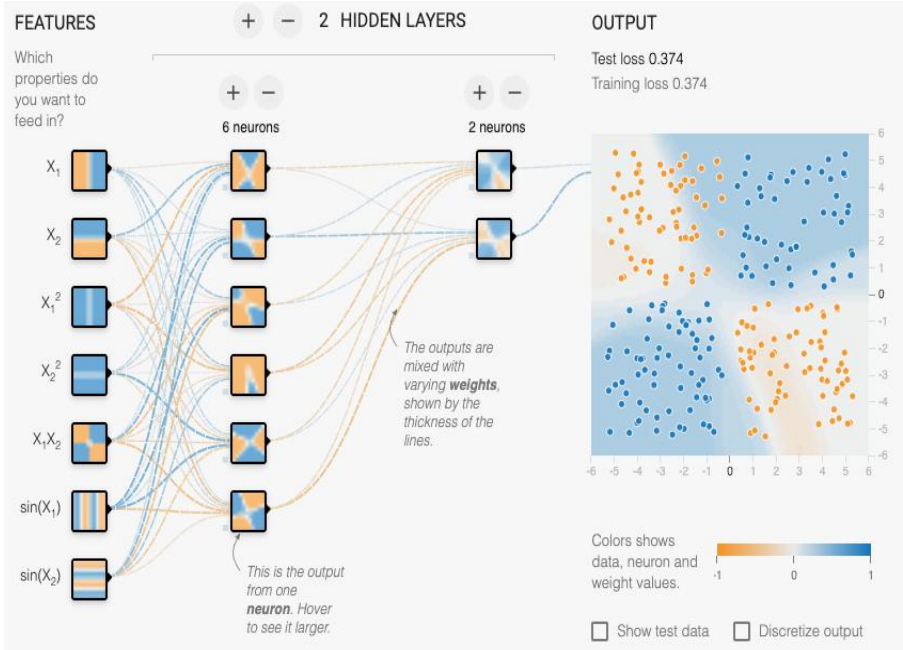
(a) Document retrieval task



Neural Models for IR

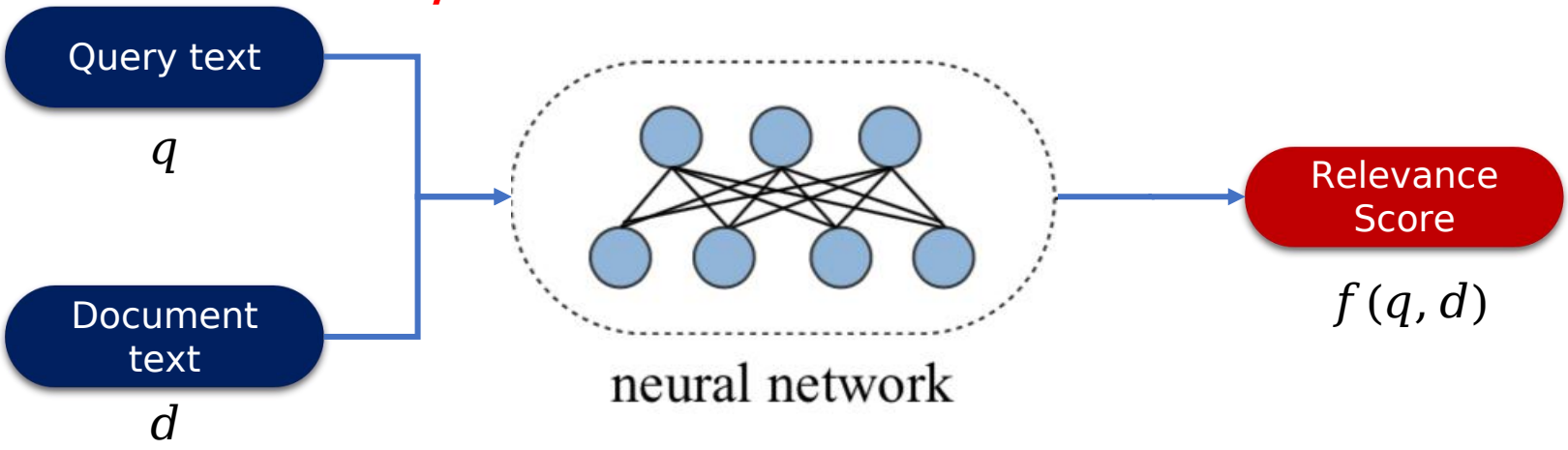
- Why choose neural models
- Deeper model has stronger ability to fit data





Neural Models for IR

- Given a query q and a document d, use a neural network to get relevance score $f(q, d)$
- Train neural model according to a rank based metric
- Rank based metrics such as DCG or MRR are non-smooth / non-differentiable

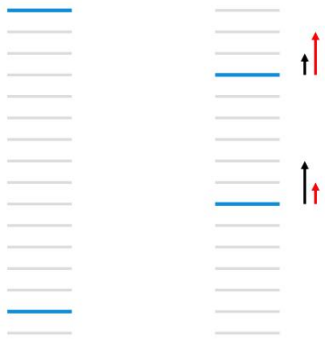


Neural Models for IR

- Learning-to-rank methods to optimize neural models
- Pointwise approach: Predict the q, d relevance y (derived from binary or graded human judgments) with a regression or classification model
  - $L = \|y - f(q, d)\|^2$
- Pairwise approach: Predict pairwise preference between documents for a query
  - $L = \phi(f(q, d_+) - f(q, d_-))$, and the ϕ can be
    - Hinge function $\phi(z) = max(0, 1 - z)$
    - Exponential function $\phi(z) = e^{-z}$
    - Logistic function $\phi(z) = \log(1 + e^{-z})$

Neural Models for IR

- Learning-to-rank methods to optimize neural models
- Listwise approach
  - NDCG is higher for left but pairwise error is less for right
  - Errors at higher ranks are more problematic than at lower ranks
  - LambdaRank loss: Multiply actual gradients with the change in NDCG by swapping the rank positions of [illegible]ocuments
    - $L = L_{Rank}$ [illegible]$CG|$

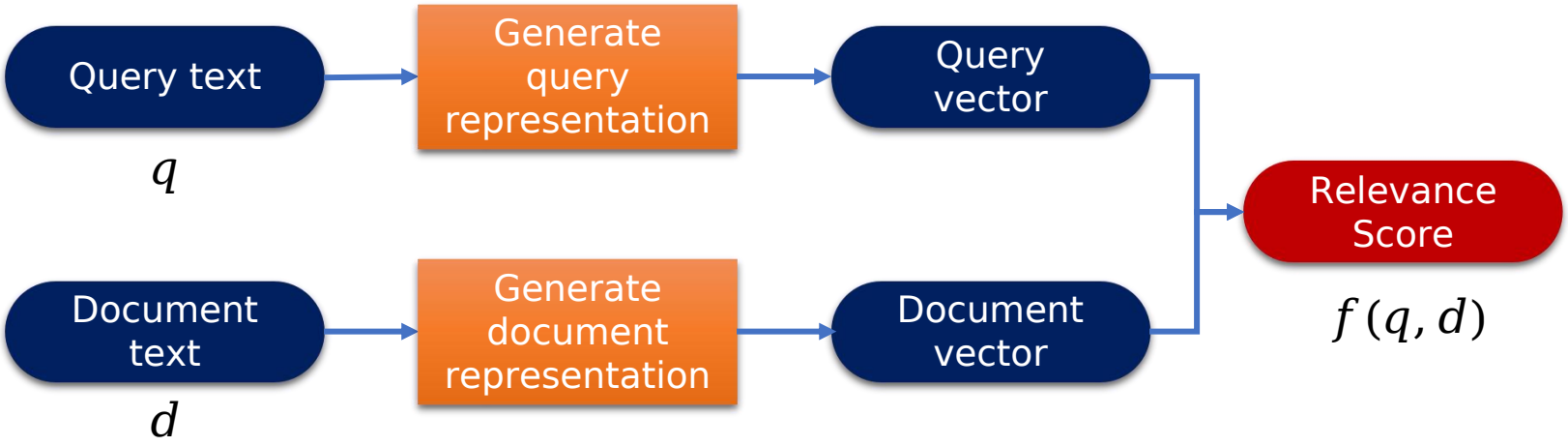


Outline

- Introduction to IR
- Traditional IR Models
- Word Embedding for IR
- Neural IR Models
  - Neural Models for IR
  - Representation-based IR Models
  - Interaction-based IR Models
  - Further Combination
  - Data Challenge in Neural IR

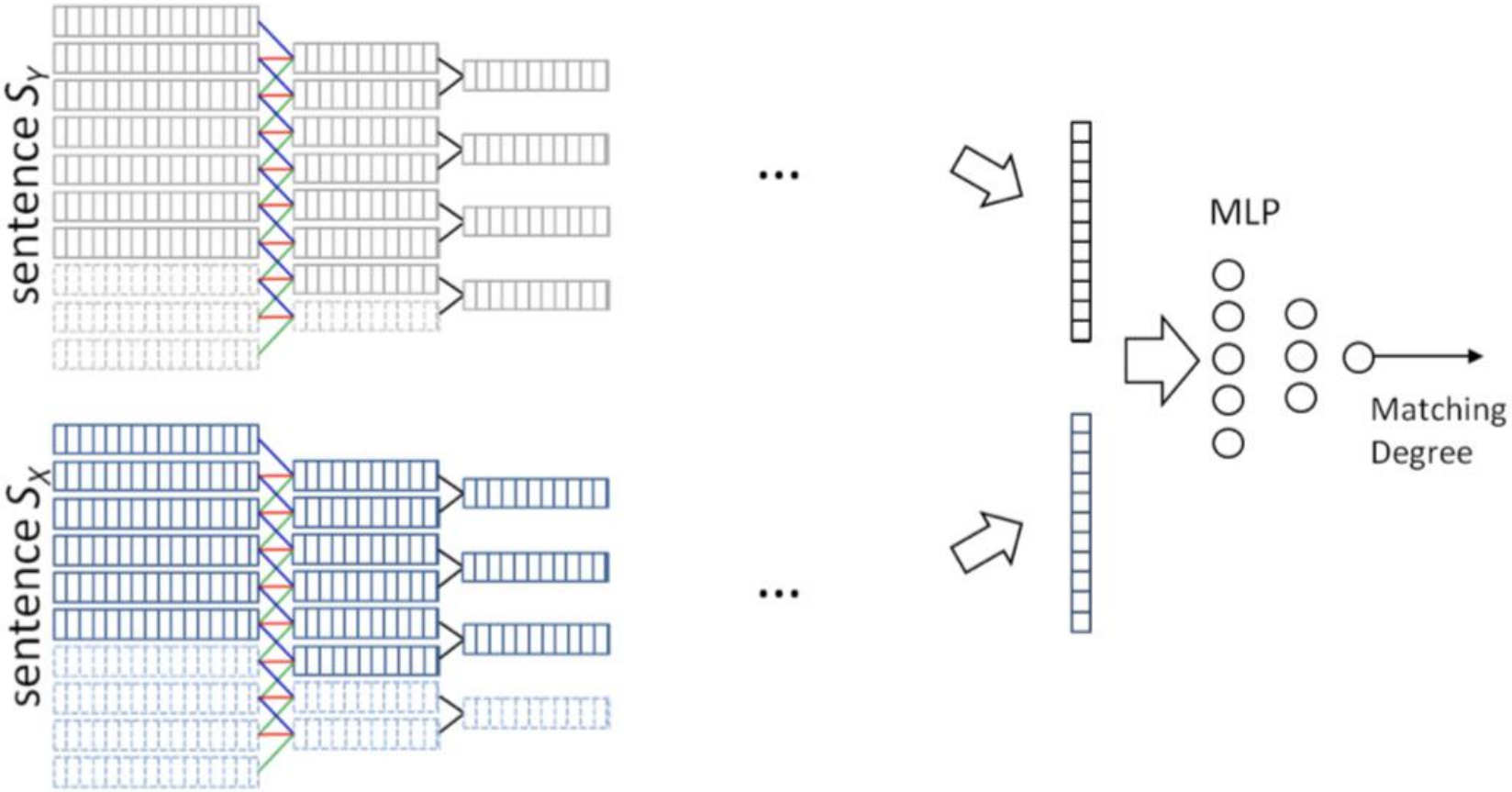
Representation-based IR Models

- Representation-based IR models
  - Use neural networks to generate query and document representations
  - Then estimate the relevance of the query and document



Representation-based IR Models

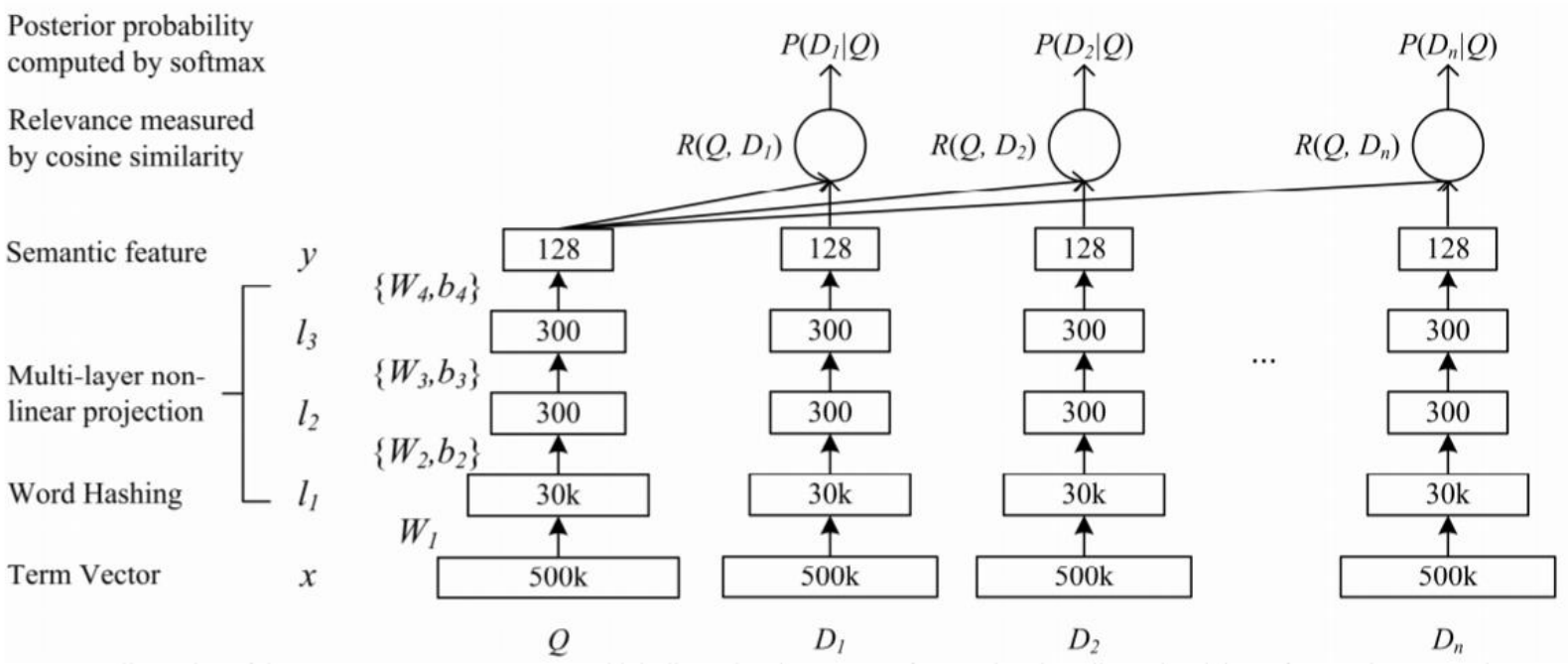
- ARC-I: Stacked layers of convolution and max-pooling



Hu et al. Convolutional Neural Network Architectures for Matching Natural Language

Representation-based IR Models

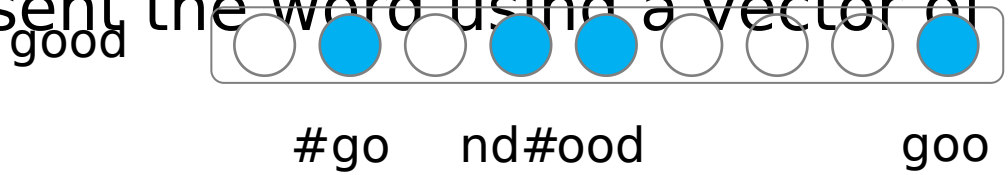
- Deep Semantic Similarity Model (DSSM)
  - Input: Character trigram counts after word hashing
  - Query and document relevance is estimated by the cosine similarity of their representations



Huang et al. Learning Deep Structured Semantic Models for Web Search using Clickthrough Data. CIKM 2013

Representation-based IR Models

- Deep Semantic Similarity Model (DSSM)
- Word hashing: The word hashing method aims to reduce the dimension of the word representation
  - Given a word
    - **good**
  - Add a mark (#) to the start and end of the word
    - **#good#**
  - Break the word into letter n-grams
    - **trigrams: #go, goo, ood, od#**
  - Represent the word using a vector of letter n-grams



Huang et al. Learning Deep Structured Semantic Models for Web Search using Clickthrough Data. CIKM 2013

Representation-based IR Models

- Deep Semantic Similarity Model (DSSM)
- Word hashing analysis
  - Why word hashing: Vocabulary is often too large (out of vocabulary)
  - Collision in word hashing vectors:
    - Different words may have a same word hashing vector
      - For example, '#bananna#' and '#bannana#'
    - The collision probability is very low:

Vocabulary	Type	Unique Key	Collision
40K	Bigram	1107	18
	Trigram	10306	2
500K	Bigram	1607	1192
	Trigram	30621	22

Huang et al. Learning Deep Structured Semantic Models for Web Search using Clickthrough Data. CIKM 2013

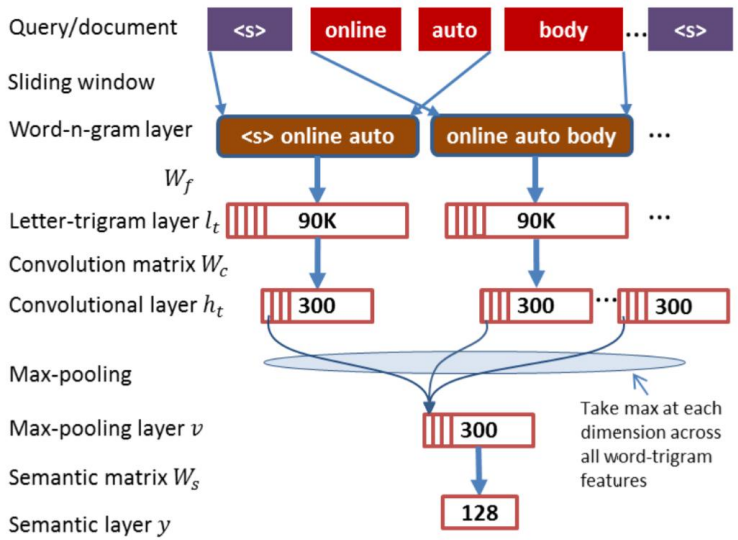
Representation-based IR Models

- Convolutional Latent Semantic Model (CLSM)
- A convolutional layer extract contextual features for each word with its neighboring words: Capture context information for queries and docs
  - Word-n-grams obtained by running a sliding window over an input sequence
  - Get the representation of each composition through word-hashing

Huang et al. Learning Deep Structured Semantic Models for Web Search using Clickthrough Data. CIKM 2013

Representation-based IR Models

- Convolutional Latent Semantic Model (CLSM)
- Max-pooling layer discovers and combines word-n-gram features into a fixed-length vector
- Semantic layer extracts the high-level feature for the inpu



Shen et al., A Latent Semantic Model with Convolutional-Pooling Structure for Information Retrieval, CIKM 2014

Representation-based IR Models

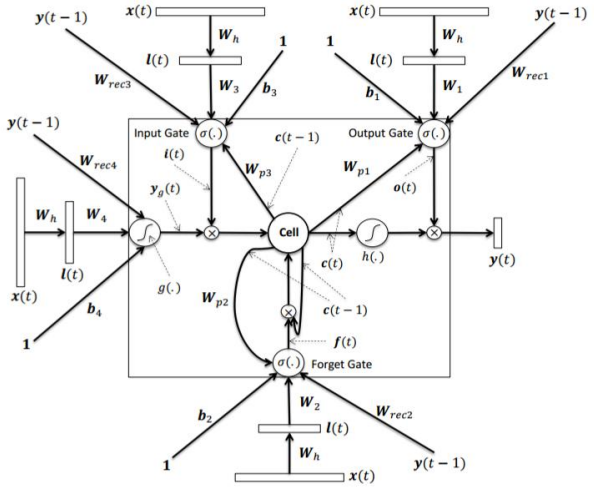
- Convolutional Latent Semantic Model (CLSM)
- The cosine similarities between learned word-n-gram feature vectors of ***office*** and ***body*** in different contexts

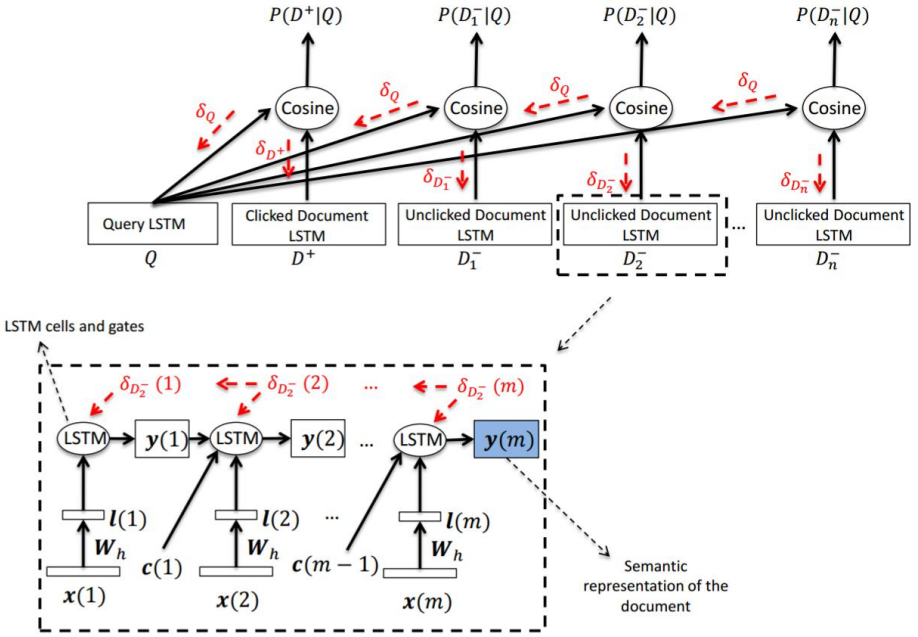
microsoft *office* software		car *body* shop	
Free *office* 2000	0.550	car *body* kits	0.698
download *office* excel	0.541	auto *body* repair	0.578
word *office* online	0.502	auto *body* parts	0.555
apartment *office* hours	0.331	wave *body* language	0.301
massachusetts *office* location	0.293	calculate *body* fat	0.220
international *office* berkeley	0.274	forcefield *body* armour	0.165

Shen et al., A Latent Semantic Model with Convolutional-Pooling Structure for Information Retrieval, CIKM 2014

Representation-based IR Models

- LSTM-DSSM: Replaces convolution layers with LSTM
- User-click signal can indicate the semantic similarity between queries and docs





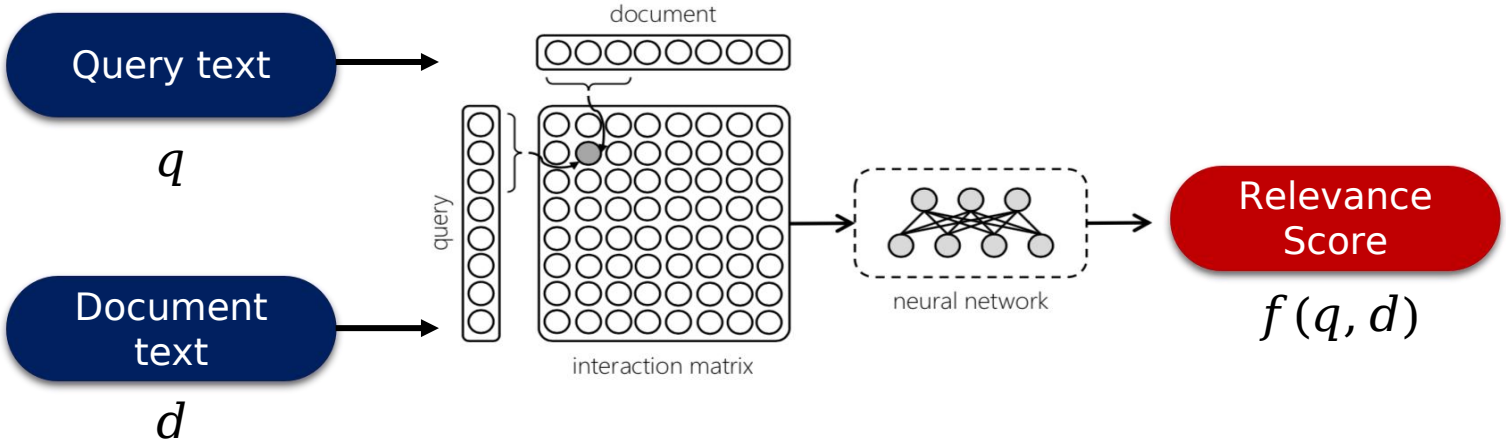
Palangi et al., Deep Sentence Embedding Using Long Short-Term Memory Networks: Analysis and Application to Information Retrieval. 2016.

Outline

- Introduction to IR
- Traditional IR Models
- Word Embedding for IR
- Neural IR Models
  - Neural Models for IR
  - Representation-based IR Models
  - Interaction-based IR Models
  - Further Combination
  - Data Challenge in Neural IR

Interaction-based IR Models

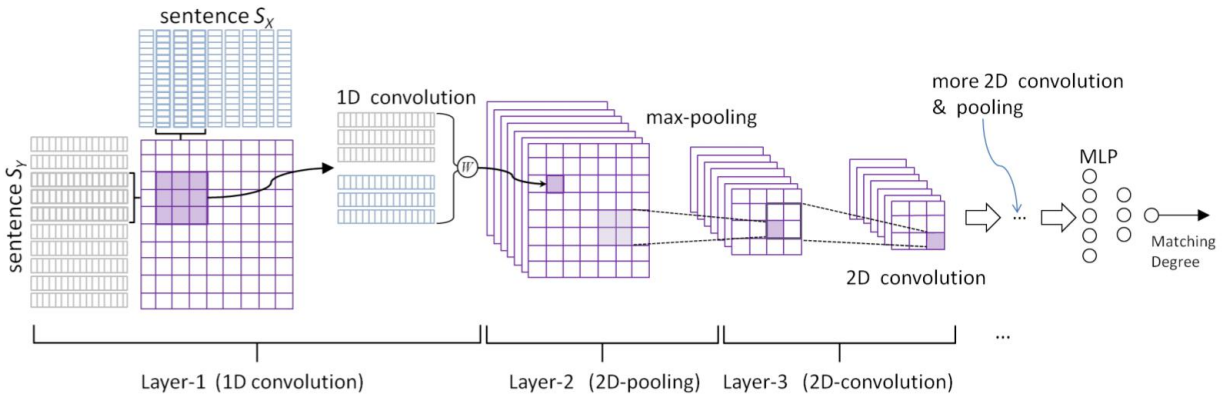
- Interaction-based IR models: Establish an interaction matrix M
  - M_{ij} is obtained by comparing the i^{th} word in query and the j^{th} word in doc
    - For example, $M_{ij} = \cos(\vec{v}_{t_i}, \vec{v}_{t_j})$
- Employ neural networks to extract features and get the ranking score



Kenter et al., Neural Networks for Information Retrieval. WSDM 2018.

Interaction-based IR Models

- ARC-II: Takes the sliding window on the sentence, and model all word-n-grams through the one-dimensional convolution
- Obtains an interaction matrix between two sentences (Concatenation word-n-gram representations)
- Obtains a high level representation through the tw



Hu et al., Convolutional Neural Network Architectures for Matching Natural Language Sentences, 2014

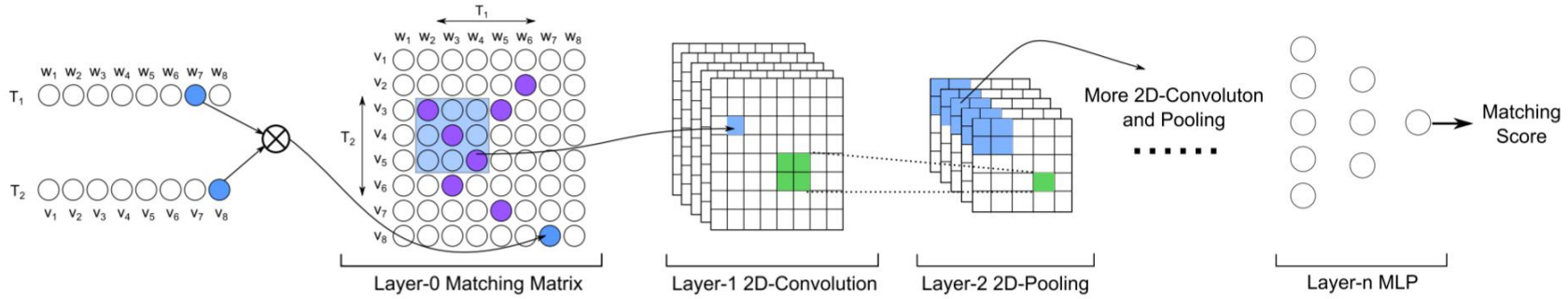
Interaction-based IR Models

- ARC-I and ARC-II
- ARC-I (representation-based) is a special case of ARC-II (interaction-based)
- ARC-II offers the capability for internal abstraction on the sentence
- ARC-II can naturally incorporate two processes
  - Modeling and aggregation compositions of each sentence
  - Extraction and fuse matching patterns between sentences

Hu et al., Convolutional Neural Network Architectures for Matching Natural Language Sentences, 2014

Interaction-based IR Models

- MatchPyramid
  - Interaction matrix
  - Hierarchical convolution (N convolutional layers)
  - Matching score aggregation (MLP)



Pang et al., A Study of MatchPyramid Models on Ad-hoc Retrieval. SIGIR 2016.

Interaction-based IR Models

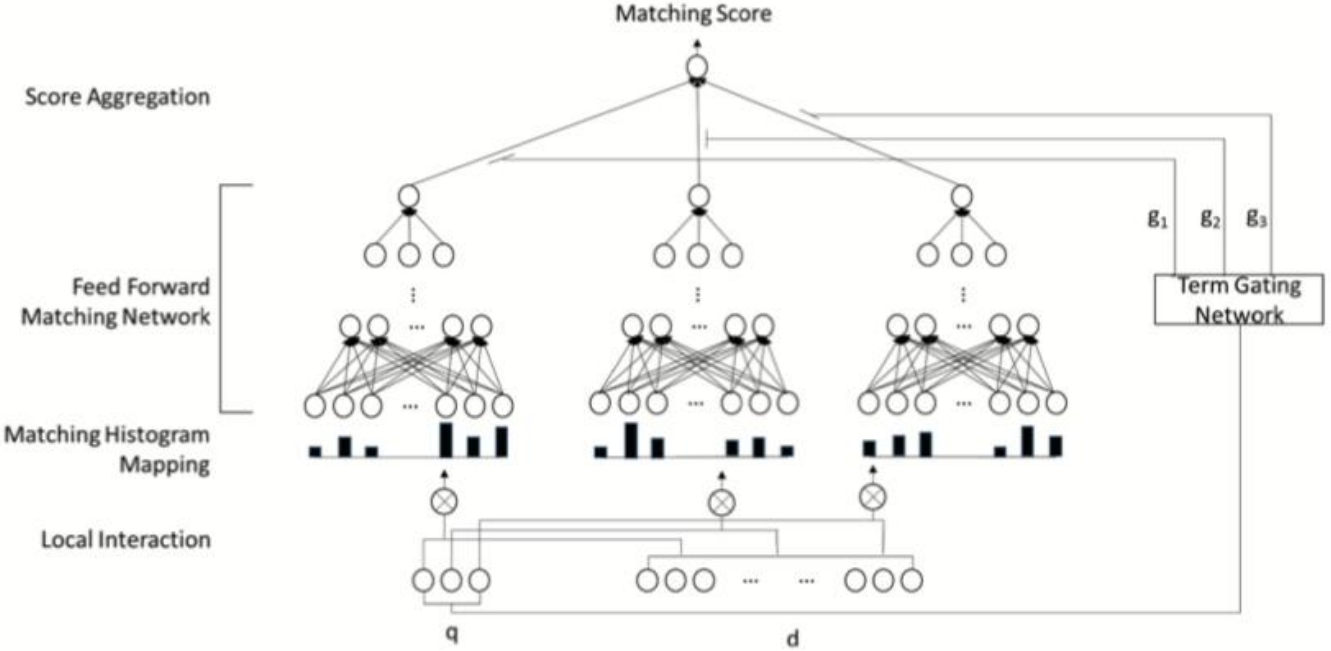
- Similarity Functions in MatchPyramid:
  - Indicator Function produces either 1 or 0 to indicate whether two words are identical
  - Cosine views the angle between two word vectors as the similarity
  - Dot Product further considers the norm of word vectors, as compared to the cosine
  - Gaussian Kernel is a well-known similarity function

Model	MAP	nDCG@20
MP-Ind	0.225	0.387
MP-Dot	0.095	0.149
MP-Cos	0.189	0.340
MP-Gau	**0.226**	**0.403**

Pang et al., A Study of MatchPyramid Models on Ad-hoc Retrieval. SIGIR 2016.

Interaction-based IR Models

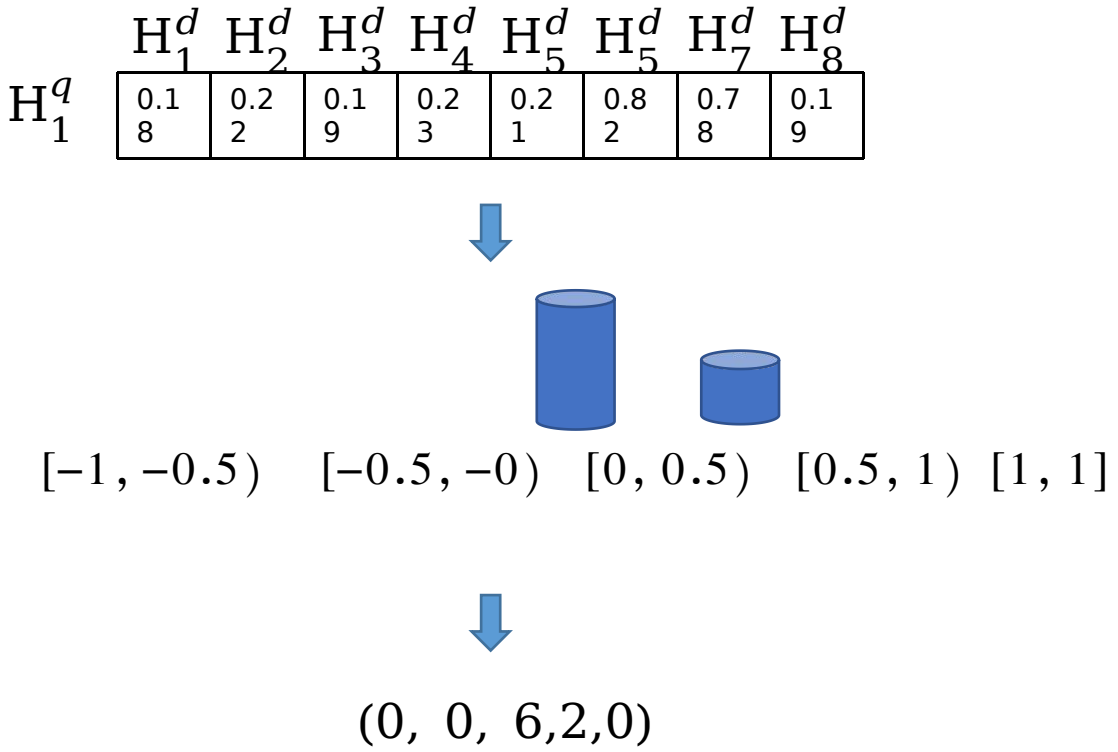
- Deep Relevance Matching Model (DRMM)
  - distribute similarity scores of word pairs to different bins to get word frequency distribution through counting
  - use neural network to extract matching patterns



Guo et al., A Deep Relevance Matching Model for Ad-hoc Retrieval. CIKM 2016.

Interaction-based IR Models

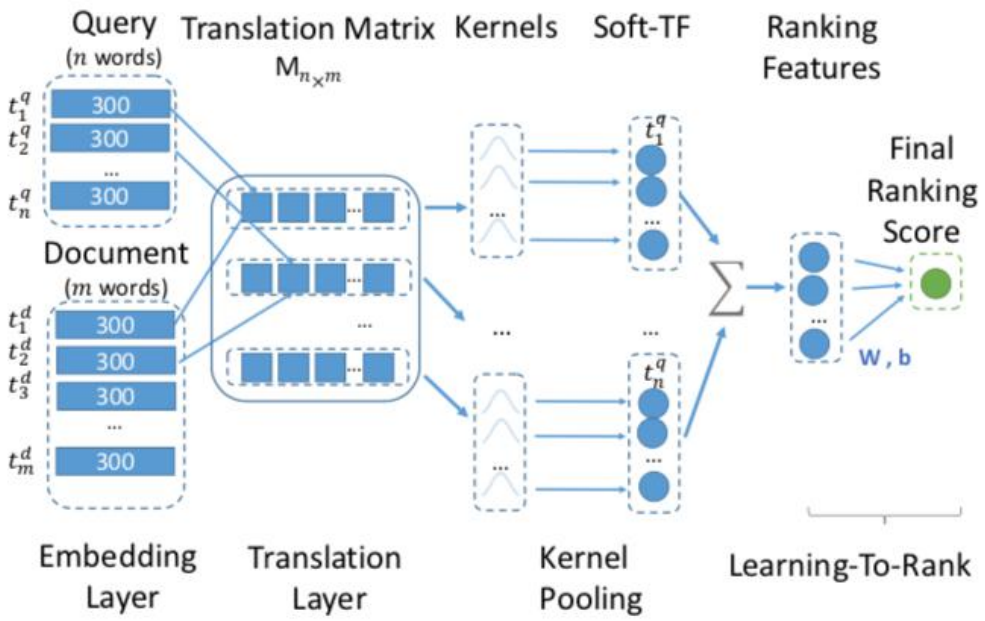
- Deep Relevance Matching Model (DRMM)
  - Matching histogram mapping



Guo et al., A Deep Relevance Matching Model for Ad-hoc Retrieval. CIKM 2016.

Interaction-based IR Models

- Kernel-based Neural Ranking Model (K-NRM)
  - Learning embedding tailored for relevance ranking
  - End-to-end training from user feedback (User click signal)
  - Soft-matching at word level



Xiong et al., End-to-End Neural Ad-hoc Ranking with Kernel Pooling. SIGIR 2017.

Interaction-based IR Models

- Kernel-based Neural Ranking Model (K-NRM)
  - Embedding layer maps each word to an L-dimension vector
  - Then K-NRM constructs an interaction matrix M
  - Kernel-Pooling converts word-word interactions to the query-document ranking feature
  - Learning-to-Rank (LeToR) combines the ranking feature to produce the final ranking score

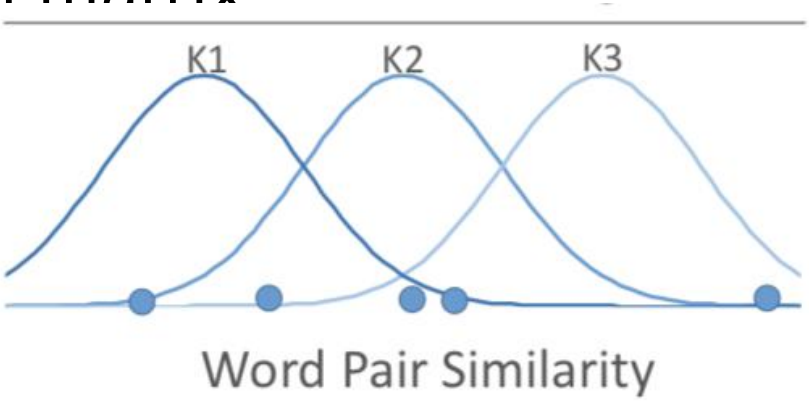
Xiong et al., End-to-End Neural Ad-hoc Ranking with Kernel Pooling. SIGIR 2017.

Interaction-based IR Models

- Kernel-based Neural Ranking Model (K-NRM)
  - Radial Basis Function (RBF) Kernel:

$$K_k(M_i) = \sum_j \exp(-\frac{(M_{ij} - \mu_k)^2}{2\sigma_k^2})$$

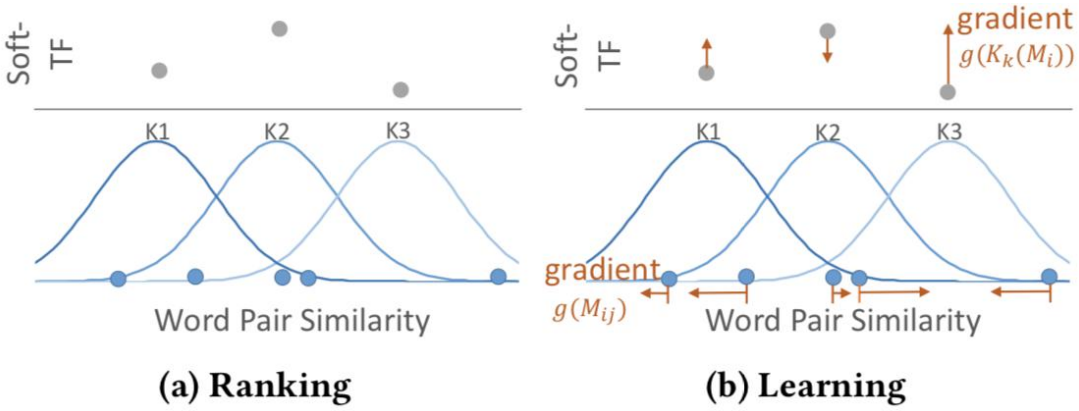
  - Where K_k is the k-th kernel, μ_k is the mean of kernel k, σ defines the kernel width, and M is the interaction matrix



Xiong et al., End-to-End Neural Ad-hoc Ranking with Kernel Pooling. SIGIR 2017.

Interaction-based IR Models

- Kernel-Pooling in K-NRM
  - Soft-TF
    - Uses kernels to softly count the frequencies of word pairs at different similarity levels
    - Counts soft-match pairs at multiple similarity levels using Kernels



$$\phi(M) = \sum_{i=1}^{n} log\overrightarrow{K}(M_i)$$

$$\overrightarrow{K}(M_i) = \{K_1(M_i), ..., K_k(M$$

Xiong et al., End-to-End Neural Ad-hoc Ranking with Kernel Pooling. SIGIR 2017.

Interaction-based IR Models

- Kernel-based Neural Ranking Model (K-NRM)
- Examples of word pairs: During training, K-NRM adjusts word embeddings to produce soft matches that can better separate relevant and irrelevant docs

From	To	Word Pairs
$\mu = 0.9$ (0.20, −)	$\mu = 0.1$ (0.23, −)	(wife, husband), (son, daughter), (China-Unicom, China-Mobile)
$\mu = 0.5$ (0.26, −)	$\mu = 0.1$ (0.23, −)	(Maserati, car),(first, time) (website, homepage)
$\mu = 0.1$ (0.23, −)	$\mu = -0.3$ (0.30, +)	(MH370, search), (pdf, reader) (192.168.0.1, router)
$\mu = 0.1$ (0.23, −)	$\mu = 0.3$ (0.26, −)	(BMW, contact-us), (Win7, Ghost-XP)
$\mu = 0.5$ (0.26, −)	$\mu = -0.3$ (0.30, +)	(MH370, truth), (cloud, share) (HongKong, horse-racing)
$\mu = -0.3$ (0.30, +)	$\mu = 0.5$ (0.26, −)	(oppor9, OPPOR), (6080, 6080YY), (10086, www.10086.com)

Values in parenthesis are MRR of the individual kernel, indicating the importance of the kernel.

'+' means word pair appearances in the corresponding kernel are positively correlated with relevance; '−' means negatively correlated.

Xiong et al., End-to-End Neural Ad-hoc Ranking with Kernel Pooling. SIGIR 2017.

Interaction-based IR Models

- Conv-KNRM
  - Queries and docs often match at n-gram level
    - For example:
      - Query: “Convolutional Neural Networks”
      - Doc: “Deep Learning Tutorial for beginners...”
    - Traditional IR approach: exact match n-grams
    - Interaction-based Neural IR models
      - Capture soft match using word embeddings

Dai et al., Convolutional Neural Networks for Soft-Matching N-Grams in Ad-hoc Search. WSDM 2018

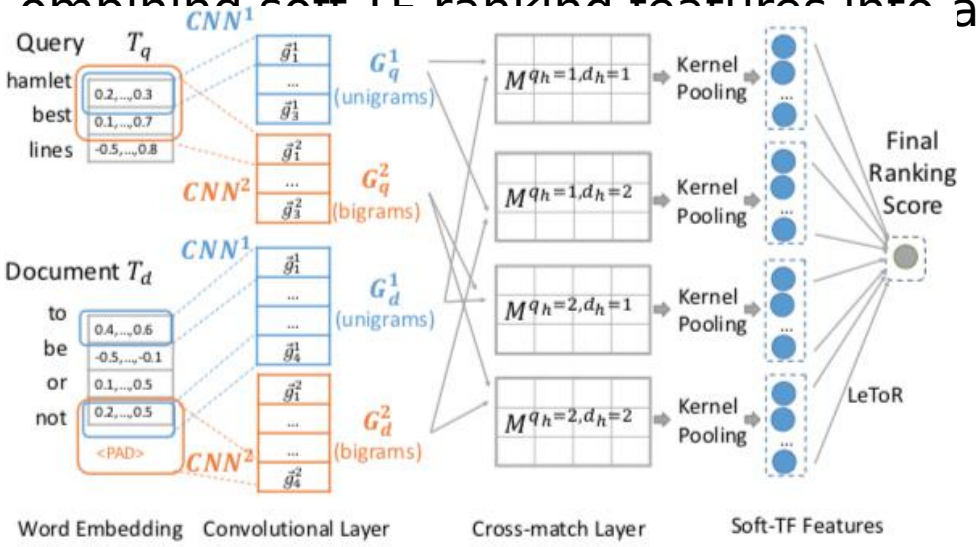
Interaction-based IR Models

- Conv-KNRM
  - Convolutional layer
    - Applys convolution layers to compose n-grams from the text
  - Cross-Match Layer
    - Builds similarity matrices between n-grams
      - Query unigrams to document unigrams
      - Query unigrams to document bigrams
      - Query bigrams to document unigrams
      - Query bigrams to document bigrams
      - ...

Dai et al., Convolutional Neural Networks for Soft-Matching N-Grams in Ad-hoc Search. WSDM 2018

Interaction-based IR Models

- Conv-KNRM
  - Ranking with N-gram Translations:
    - Kernel-Pooling
      - Using K Gaussian kernels to extract features of word n-gram pairs
    - Learning-to-Rank (LeToR):
      - Combining soft-TF ranking features into a ranking score



Dai et al., Convolutional Neural Networks for Soft-Matching N-Grams in Ad-hoc Search. WSDM 2018

Interaction-based IR Models

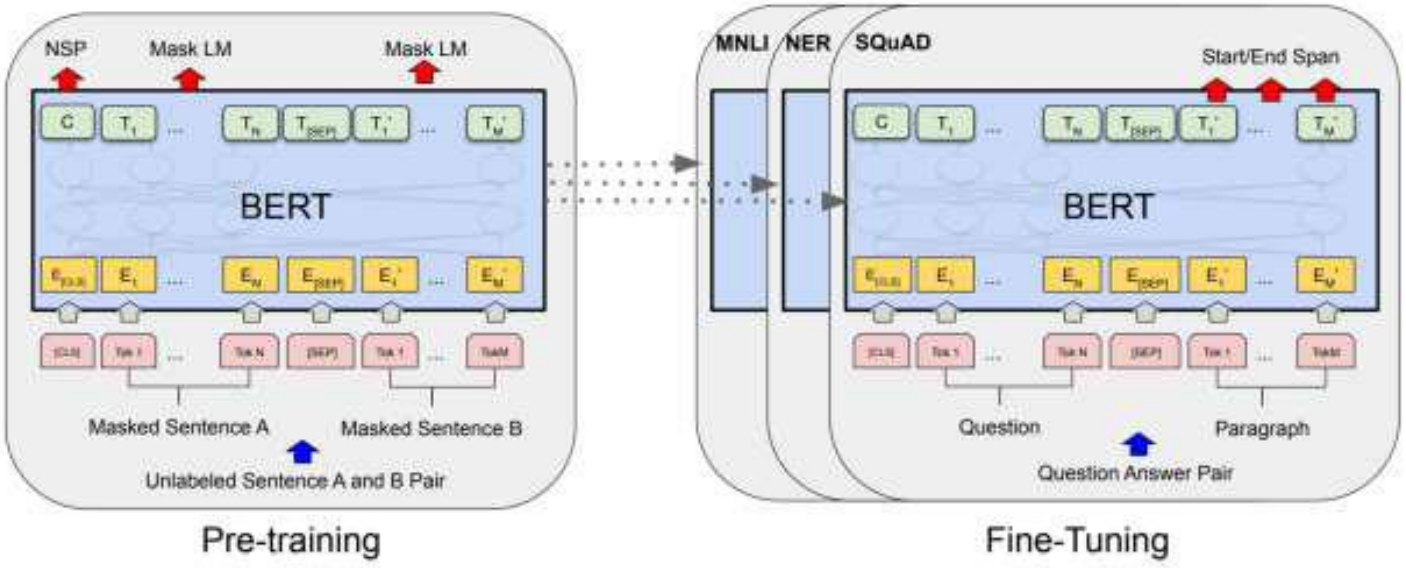
- Conv-KNRM
  - Examples of matched n-grams between query and snippets
    - Black phrases contribute more to the relevance score than gray ones

Query	Snippet
sewing instructions	...home free resources! newsletter sewing ideas...quilting 101 what is a quilt...
atypical squamous cells	...treatment decision tools cervical cancer : prevention and early detection...
moths	... grouping of moth families commonly known as the 'smaller moths ' (micro , lepidoptera)...
fickle creek farm	.. bed & breakfast inns extended stay lodging rv parks where to eat & drink nightlife ...
university phoenix	campus locations programs : bachelor degree masters degrees account degrees business degree...
wedding budget calculator	...planning tips photographs bridal board my perfect planner tools my check lists...

Dai et al., Convolutional Neural Networks for Soft-Matching N-Grams in Ad-hoc Search. WSDM 2018

Interaction-based IR Models

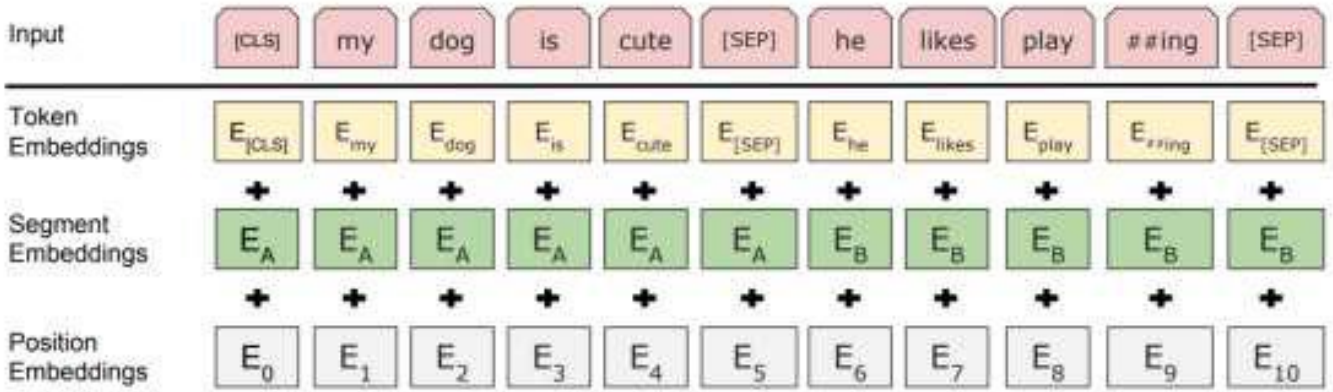
- BERT
  - Stacked transformer layers
  - BERT is pretrained on two tasks
    - Masked language modeling
    - Next sentence prediction



Devlin, et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NACCL 2019

Interaction-based IR Models

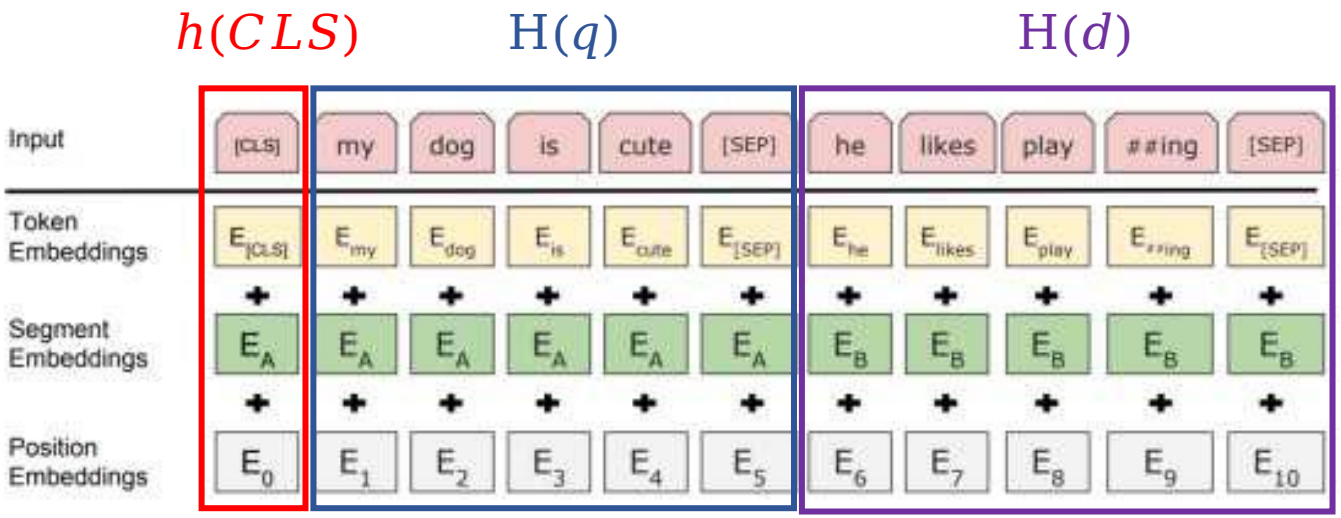
- BERT
  - Input: WordPiece embedding + position embedding + segment embedding
    - WordPiece: Convert words to subwords
      - Keeping the Secret of Genetic Testing [Keeping, the, Secret, of, Gene, ##tic, Testing]
      - He like play [He, like, play, ##ing]



Devlin, et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL 2019

Interaction-based IR Models

- BERT ranker
  - Given a query q and a document d.
    - Three kinds of representations are calculated
      - [CLS] representation $h(CLS)$
      - Query representation $\mathrm{H}(q)$
      - Document representation $\mathrm{H}(d)$

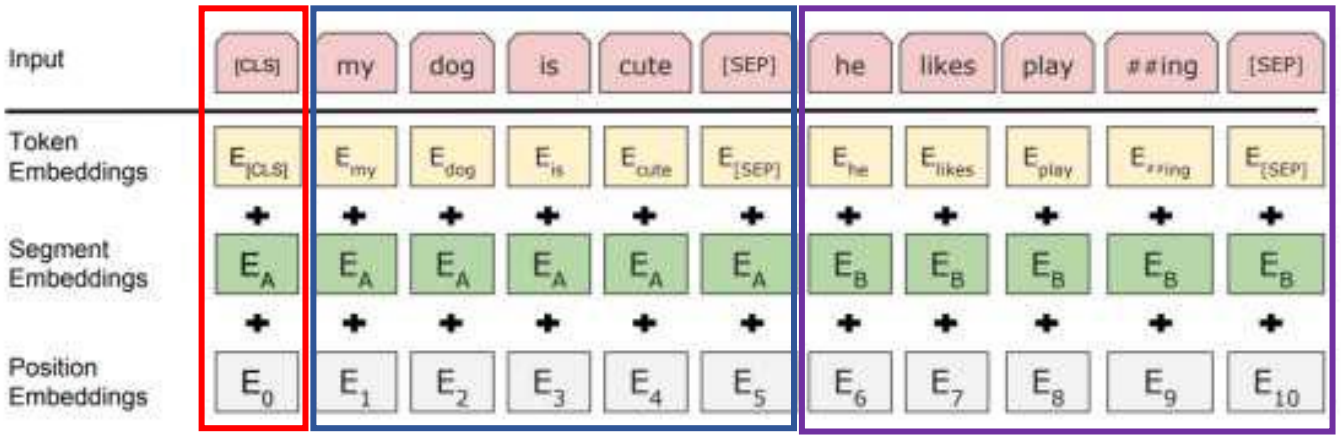


Qiao et al. Understanding the Behaviors of BERT in Ranking. 2019.

Interaction-based IR Models

- BERT ranker
  - Given a query q and a document d
    - The relevance score $f(q, d)$ can be calculated:
      - $f(q, d) = MLP(h(CLS))$ with [CLS] representation
      - Or $f(q, d) = MLP(\phi(H(q), H(d)))$ with query and document representations. ϕ can be representation-based and interaction-based architectures

$h(CLS)$ $\mathrm{H}(q)$ $\mathrm{H}(d)$



Qiao et al. Understanding the Behaviors of BERT in Ranking. 2019.

Interaction-based IR Models

- BERT ranker
  - Compared to Conv-KNRM, BERT mainly improves ranking performance on the question answering task
  - BERT performs better on natural language understanding than keyword matching

	MS MARCO Passage Ranking				ClueWeb09-B Ad hoc Ranking			
Method	**MRR@10 (Dev)**		**MRR@10 (Eval)**		**NDCG@20**		**ERR@20**	
Base	0.1762	−9.45%	0.1649	+13.44%	$0.2496^{\S}$	−6.89%	0.1387	−14.25%
LeToR	0.1946	–	0.1905	–	0.2681	–	0.1617	–
K-NRM	$0.2100^{\dagger\ddagger}$	+7.92%	0.1982	+4.04%	0.1590	−40.68%	0.1160	−28.26%
Conv-KNRM	$0.2474^{\dagger\ddagger\S}$	+27.15%	0.2472	+29.76%	$0.2118^{\S}$	−20.98%	$0.1443^{\S}$	−10.78%
Conv-KNRM (Bing)	n.a.	n.a.	n.a.	n.a.	$0.2872^{\dagger\ddagger\S\P}$	+7.12%	$0.1814^{\dagger\ddagger\S\P}$	+12.18%
BERT (Rep)	0.0432	−77.79%	0.0153	−91.97%	0.1479	−44.82%	0.1066	−34.05%
BERT (Last-Int)	$0.3367^{\dagger\ddagger\S\P}$	+73.03%	0.3590	+88.45%	$0.2407^{\S\P}$	−10.22%	$0.1649^{\dagger\S\P}$	+2.00%
BERT (Mult-Int)	$0.3060^{\dagger\ddagger\S\P}$	+57.26%	0.3287	+72.55%	$0.2407^{\S\P}$	−10.23%	$0.1676^{\dagger\S\P}$	+3.64%
BERT (Term-Trans)	$0.3310^{\dagger\S\P}$	+70.10%	0.3561	+86.93%	$0.2339^{\S\P}$	−12.76%	$0.1663^{\dagger\S\P}$	+2.81%

Qiao et al. Understanding the Behaviors of BERT in Ranking. 2019.

Outline

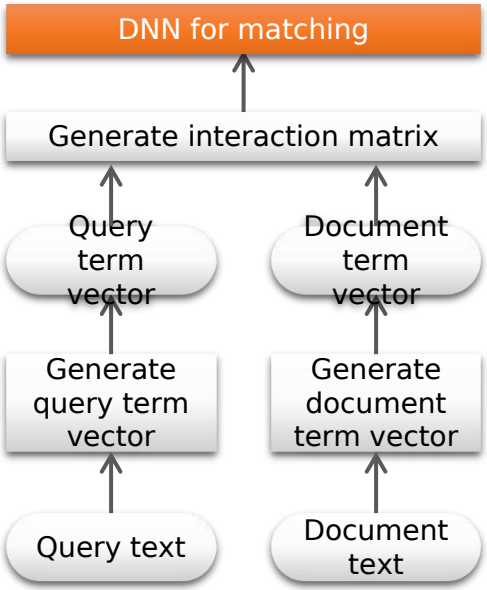
- Introduction to IR
- Traditional IR Models
- Word Embedding for IR
- Neural IR Models
  - Neural Models for IR
  - Representation-based IR Models
  - Interaction-based IR Models
  - Further Combination
  - Data Challenge in Neural IR

Further Combination

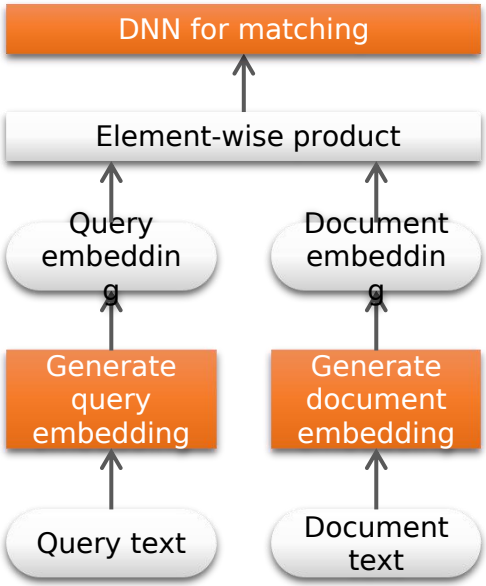
- The library vs. librarian dilemma
  - The library (Know about the world)
    - The distributed model knows more about “Barack Obama” than “Bhaskar Mitra” and will perform better on the former
  - The librarian (Know how to find information without much prior domain knowledge)
    - The local model does not understand “Barack Obama” and “Bhaskar Mitra” but might perform better for the latter query compared to the distributed model

Further Combination

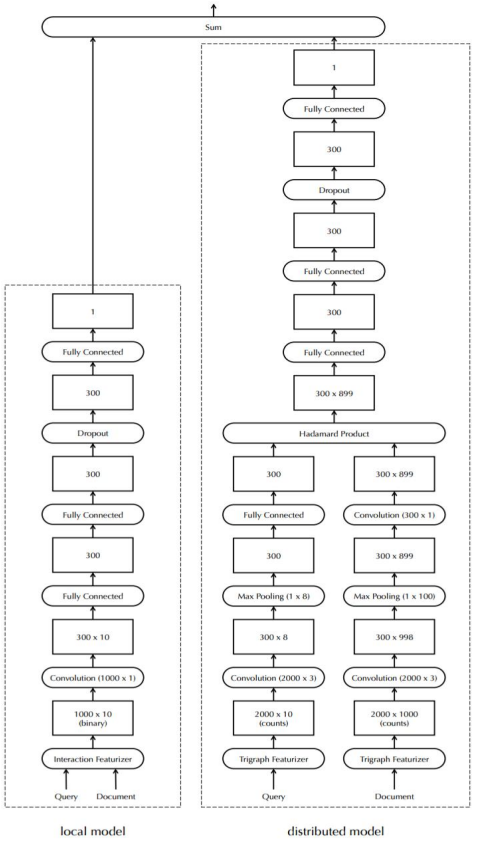
- Duet model
  - A linear combination of local and distributed models



interaction model



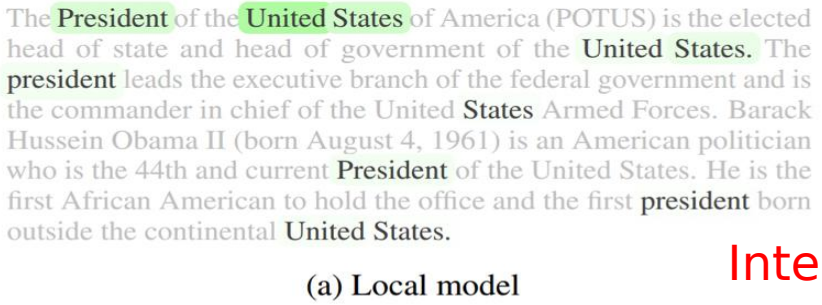
representation model



Mitra et al., Learning to Match using Local and Distributed Representations of Text for Web Search. WWW 2017

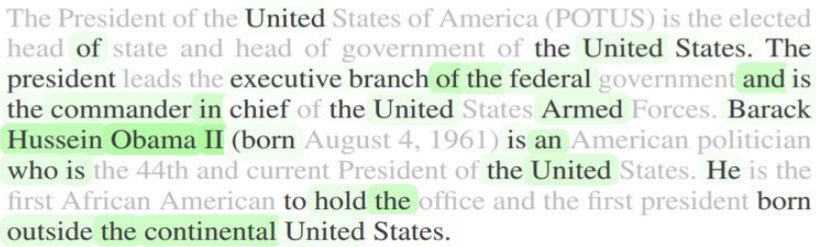
Further Combination

- An example of duet model
  - Individually removing each term for the query "united states president"
    - Darker green parts drop significantly in retrieval score



(a) Local model

Interaction-based



(b) Distributed model

Representation-based

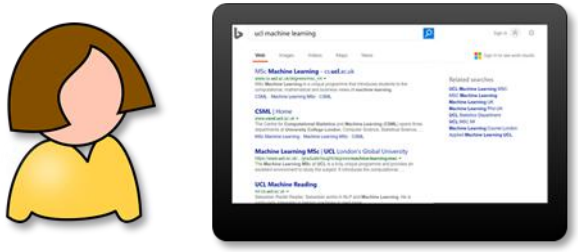
Mitra et al., Learning to Match using Local and Distributed Representations of Text for Web Search. WWW 2017

Outline

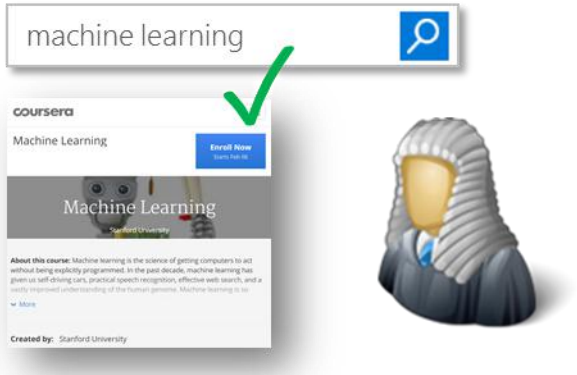
- Introduction to IR
- Traditional IR Models
- Word Embedding for IR
- Neural IR Models
  - Neural Models for IR
  - Representation-based IR Models
  - Interaction-based IR Models
  - Further Combination
  - Data Challenge in Neural IR

Data Challenge in Neural IR

- Neural IR models are fully supervised
  - Traditional IR uses human labels as ground truth for evaluation
  - So ideally we want to train our ranking models on human labels
  - User interaction data from industry is usually not available for most people and may contain different biases compared to human annotated labels



user interaction / click data



human annotated labels

Data Challenge in Neural IR

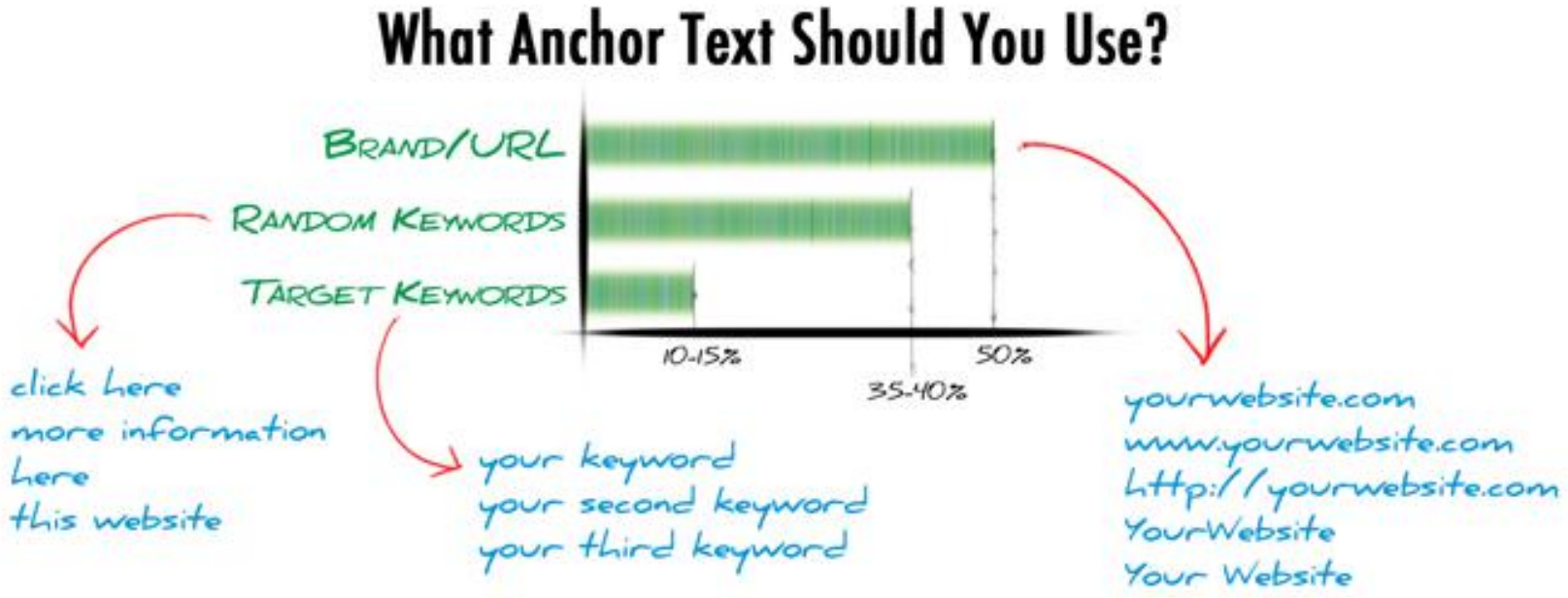
- Anchor texts are similar to query texts
- Anchor-document relations are approximate to the relevance between query and document

New York City Transit Police

The **New York City Transit Police Department** was a law enforcement agency in New York City that existed from 1953 to 1995, and is currently part of the NYPD. The roots of this organization go back to 1936 when Mayor Fiorello H. La Guardia authorized the hiring

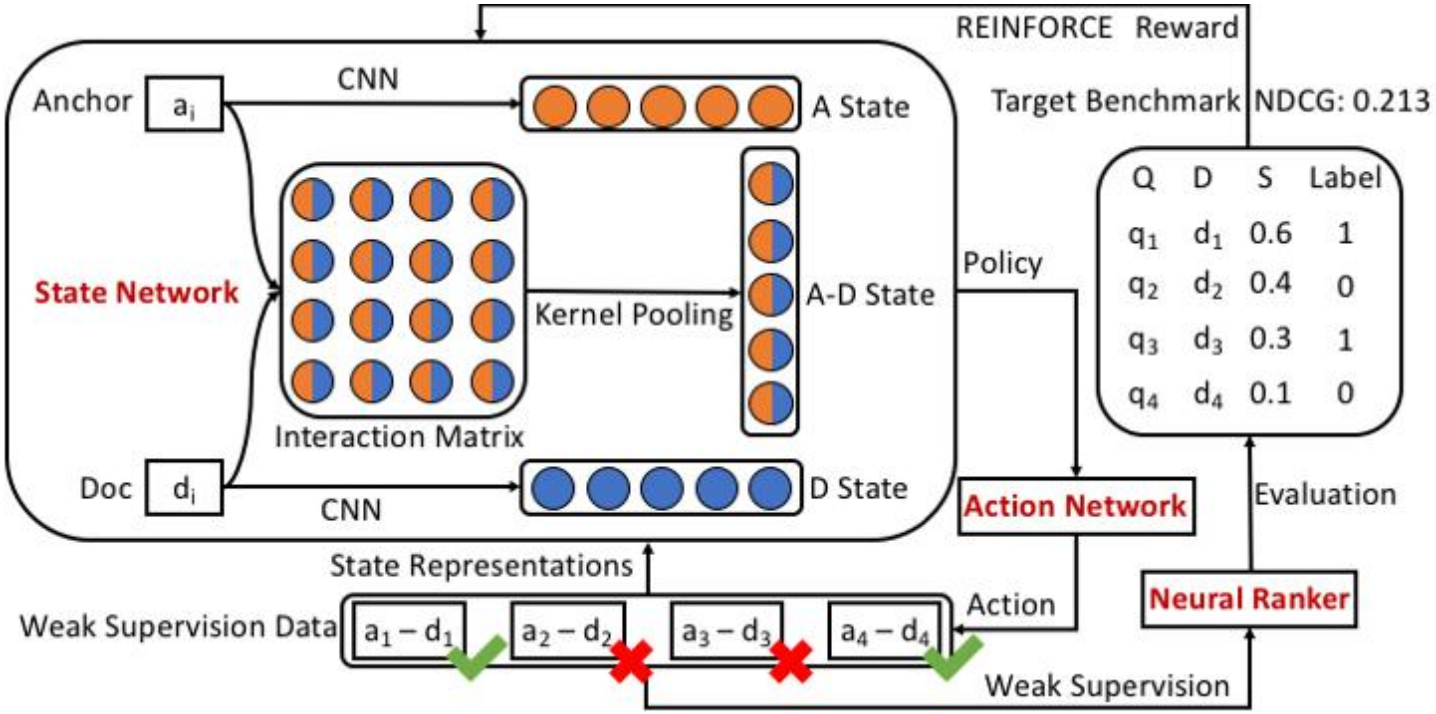
Data Challenge in Neural IR

- Anchor-document data could be very noisy, and the noise data may hurt performance of neural IR methods



Data Challenge in Neural IR

- Reinforcement data selection (ReinfoSelect)
  - Learns to select anchor-document pairs that best weakly supervise the neural ranker



Zhang et al., Selective Weak Supervision for Neural Information Retrieval. WWW 2020.

Data Challenge in Neural IR

- Reinforcement data selection (ReinfoSelect)
  - Anchor-document week supervisions significantly improve IR performance of Conv-KNRM and pre-trained language model BERT

Model	Method	ClueWeb09 NDCG@20	ClueWeb09 ERR@20	Robust04 NDCG@20	Robust04 ERR@20	ClueWeb12 NDCG@20	ClueWeb12 ERR@20
Conv-KNRM	**No Weak Supervision**	0.2873	0.1597	0.4267	0.1168	0.1123	0.0915
	ReInfoSelect	**0.3096**	**0.1611**	**0.4423**	**0.1202**	**0.1225**	***0.1044***
BERT	**No Weak Supervision**	0.2999	0.1631	0.4258	0.1163	0.1190	0.0963
	ReInfoSelect	***0.3261***	***0.1669***	***0.4500***	***0.1220***	***0.1276***	***0.0997***

Zhang et al., Selective Weak Supervision for Neural Information Retrieval. WWW 2020.

Neural IR Methods

- Two categories: representation-based and interaction-based
- Deal with vocabulary mismatch problem with word embeddings
- Help better understand natural language with sophisticated neural architectures
- Some challenges exist such as data challenge

Summary

- Information Retrieval: to select documents from a large-scale document collection to satisfy user needs
- Evaluation Metrics: MAP, NDCG, MRR
- Traditional IR Methods: deal with large scale data without annotation; vocabulary mismatch and shallow understanding
- Word Embedding for IR: word representation, term-weighting, query expanding
- Neural IR Methods: help better understand; data challenge

Reading Material

- Must-read papers

· PACRR: A Position-Aware Neural IR Model for Relevance Matching. EMNLP 2017 [link]

· Entity-Duet Neural Ranking: Understanding the Role of Knowledge Graph Semantics in Neural Information Retrieval. ACL 2018 [link]

· A Deep Look into Neural Ranking Models for Information Retrieval. 2019 [link]

· Selective Weak Supervision for Neural Information Retrieval. WWW 2020 [link]

- Further reading

· Explicit Semantic Ranking for Academic Search via Knowledge Graph Embedding. WWW 2017 [link]

· Query suggestion with feedback memory network. WWW 2018 [link]

· NPRF: A Neural Pseudo Relevance Feedback Framework for Ad-hoc Information Retrieval. EMNLP 2018 [link]

· Towards Better Text Understanding and Retrieval through Kernel Entity Salience Modeling. SIGIR 2018 [link]

For reading material recommendation of this course, please refer to our github


清华大学
TSINGHUA UNIVERSITY
-1911-


Tsinghua University
NLP
Natural Language Processing

THUNLP