11-442 / 11-642 / 11-742:
Search Engines

Learning to Rank: Neural Models

Jamie Callan
Carnegie Mellon University
callan@cs.cmu.edu

1

Outline

- Introduction
- Deep Structured Semantic Models (DSSM)
- **Deep Relevance Matching Model (DRMM)**
- Kernel-based Neural Ranking Model (K-NRM)
- Convolutional Kernel-based Neural Ranking Model (Conv-KNRM)
- BERT reranking
- DeepCT
- doc2query

2

2

Deep Relevance Matching Model (DRMM): Motivating Ideas

Much recent deep learning research uses word embeddings

- Represent a term by a weight vector (continuous representation)

Continuous representations are an old idea in IR

- LSI, LSA, PLSA, PIRCS, MatchPlus, …
  - Not terrible, but not as good as BM25, vector space, …
- Query term 'cat' matches document term 'kitten' ☺
- Query term 'cat' matches document term 'dog' ☹

Query & document terms that match exactly are a strong signal

- Prior work with continuous representations lost this signal

(Guo, et al., 2016)

Word2Vec

Word2vec is a popular method for creating continuous representations of terms

- Input: A lot of text
- Output: A vector-based term dictionary
  - Words that appear in similar contexts will have similar term vectors

Examples of similar terms (English GoogleNews)

- apple: apples, pear, fruit, berry, pears, strawberry
- pie: pies, cake, slice, cheesecake, biscuit
- man: woman, boy, teenager, girl, robber, men
- cat: cats, dog, kitten, feline, beagle, puppy

cat	kitten
0.14	0.13
0.01	0.02
0.00	0.01
0.38	0.35
0.01	0.00
0.00	0.01
0.27	0.29
: :	: :
0.67	0.60
←300→	←300→

Deep Relevance Matching Model (DRMM)

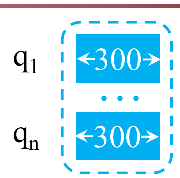
Key ideas

- Continuous representations of terms (word2vec)
- Measure the interaction between each pair of terms (q_i, d_j)
- For each query term q_i, bin interactions of different strengths
- Use a feed-forward network to combine signals for q_i
- Aggregate scores for q_i
- Modulate the influence of q_i ("gating")
- Linear combination to produce a score for (q_i, d_j)

It's simpler than it sounds…

(Guo, et al., 2016)

Deep Relevance Matching Model (DRMM): Query Representation



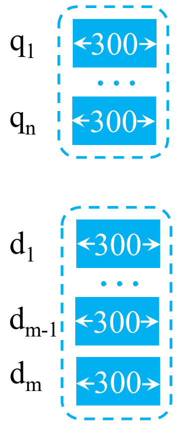
Embedding Layer

Use a continuous representation of query terms

- A 300-dimension vector for each term
- Standard word2vec

(Guo, et al., 2016)

Deep Relevance Matching Model (DRMM): Document Representation



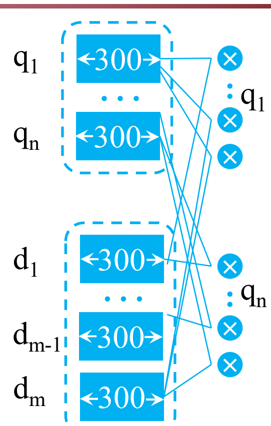
Embedding Layer

Use a continuous representation of document terms

- A 300-dimension vector for each term
- Standard word2vec

(Guo, et al., 2016)

Deep Relevance Matching Model (DRMM): Local Interactions



Embedding Layer

Compare each query term to each document term

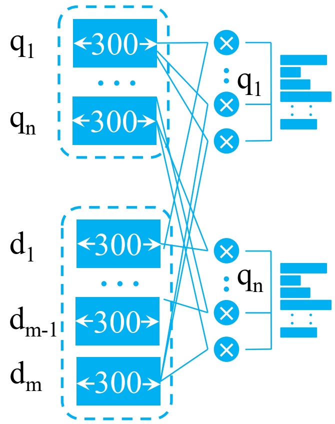
- Cosine similarity of 300-dimension embedding vectors for (q_i, d_j)
- Values are in range [-1, 1]

Note: This is an interaction model

- It considers many local interactions between q and d

(Guo, et al., 2016)

Deep Relevance Matching Model (DRMM): Pyramid (Histogram) Pooling



Embedding Layer

Pyramid Pooling

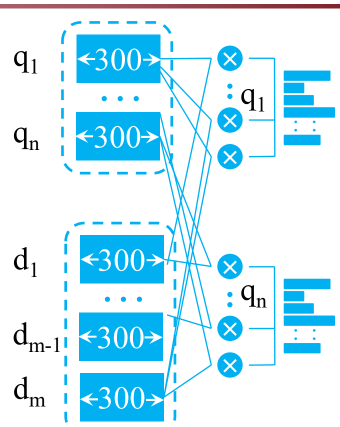
Bin values for (q_i, d_j) matches of different quality

- 1 bin for [1,1]
  - q_i and d_j match exactly
- b bins for [-1, 1)
  - q_i and d_j match softly
  - E.g., [-1, -0.8) … [0.8, 1.0)

How should values be binned?

(Guo, et al., 2016)

Deep Relevance Matching Model (DRMM): Pyramid (Histogram) Pooling



Embedding Layer

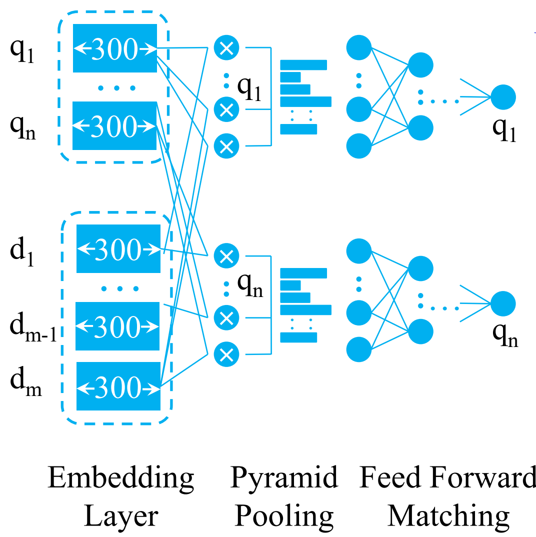
Pyramid Pooling

They tried 3 types of histograms

- Count matches in range (CH)
  - Number of matches to q_i in each range (e.g., [0.2, 0.4))
  - Essentially tf for each range
- Normalized count (NH)
  - Percentage of matches to q_i in each quality range
- Log of count (LCH)
  - log (tf) for each range (most effective method)

(Guo, et al., 2016)

Deep Relevance Matching Model (DRMM): Feed Forward Neural Network

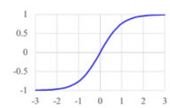


Use a feedforward network to combine the scores from the 11 bins for q_i into a match score

- 2 hidden layers

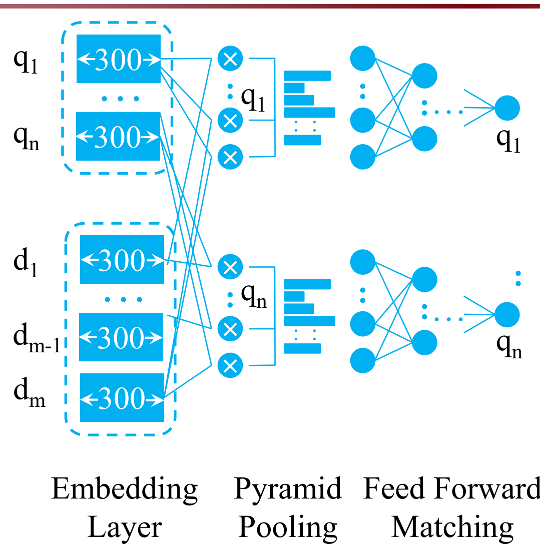
$$z_i^{(l)} = tanh(\boldsymbol{W}^{(l)} z_i^{(l-1)} + \boldsymbol{b}^{(l)})$$

i=1, …, n, l=1, …, L



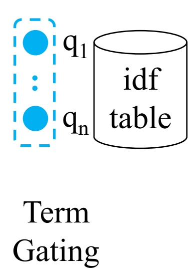
(Guo, et al., 2016)

Deep Relevance Matching Model (DRMM): Term Gating



Model term importance with gate weights

$$g_i = \frac{\exp(w\,\mathrm{idf}(q_i))}{\sum_{j=1}^{n} \exp(w\,\mathrm{idf}(q_j))}$$



(Guo, et al., 2016)

Deep Relevance Matching Model (DRMM): Term Gating

$$idf(q_i) = log\frac{N}{df_{q_i}}$$

w transforms the idf weight

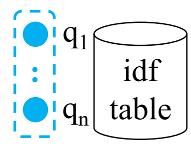
- w=0.0: g_i is $\frac{1}{n}$ for all terms (no idf)
- w=1.0: g_i is more skewed than idf
  - Increases the impact of rare terms

Guo, et al., don't report the value of w

- Removing the gate has a small effect
- Probably the learned w is close to 0

Model term importance with gate weights

$$g_i = \frac{\exp(w\,\mathrm{idf}(q_i))}{\sum_{j=1}^{n}\exp(w\,\mathrm{idf}(q_j))}$$



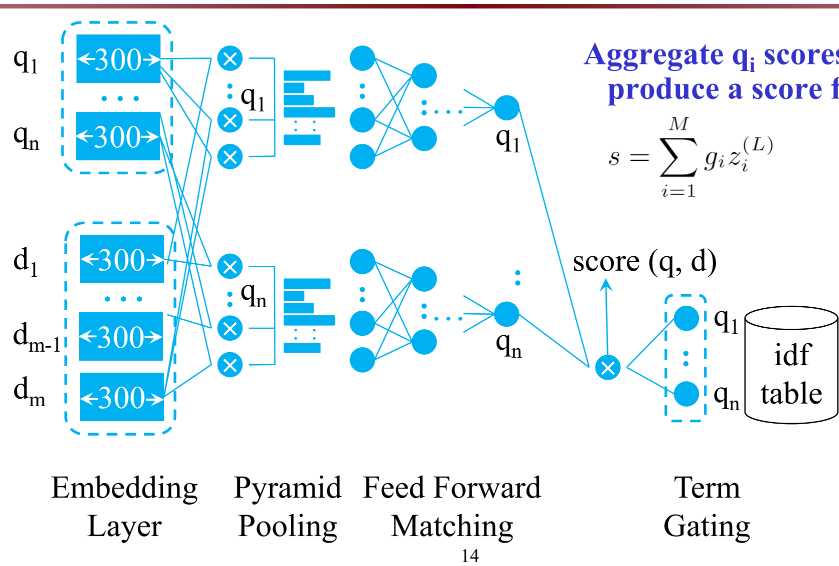
Term Gating

(Guo, et al., 2016)

Deep Relevance Matching Model (DRMM): Aggregation

Aggregate q_i scores to produce a score for q

$$s = \sum_{i=1}^{M} g_i z_i^{(L)}$$



(Guo, et al., 2016)

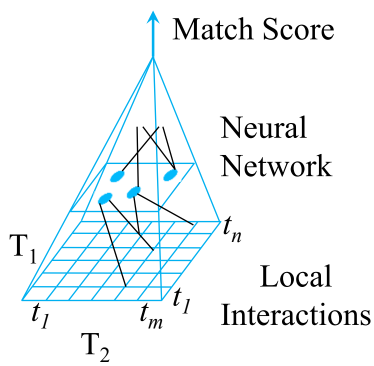
Deep Relevance Matching Model (DRMM): Type of Neural IR Model

DRMM is a type of <u>interaction-based</u> neural IR model

- Identify <u>local matches</u> between two pieces of text
  - E.g., cosine similarity of term vectors
- Learn interaction patterns for matching
  - Often hierarchical patterns
  - E.g., convolutional neural network

There are <u>many</u> interaction-based models

- DRMM, DeepMatch, ARC-II
- MatchPyramid, K-NRM



Deep Relevance Matching Model (DRMM): Computational Complexity

<u>Every</u> query matches <u>every</u> document

- There are <u>always</u> soft-matches
- The computational cost is too high to be practical for initial retrieval

DRMM is used in a re-ranking pipeline

- Use an efficient algorithm (e.g., Indri) to create a ranking
- Use DRMM to re-rank the top *n* documents

Deep Relevance Matching Model (DRMM): Training

Pairwise training with hinge loss

$$\mathcal{L}(q, d^+, d^-; \Theta) = \max(0, 1 - s(q, d^+) + s(q, d^-))$$

d^+: Relevant documents

d^-: Non-relevant documents

Training data

- Robust04: 600K documents, 50 queries
- ClueWeb09-B: 34M documents, 150 queries

(Guo, et al., 2016)

17

17

Deep Relevance Matching Model (DRMM): Effectiveness

DRMM is more effective than Indri and BM25

- Supervised vs unsupervised … not surprising

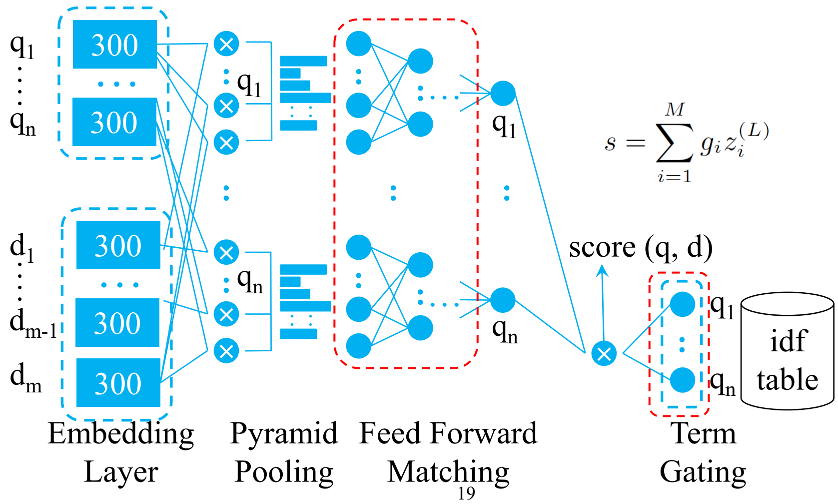
Guo, et al. didn't compare to learning-to-rank systems (!) … but we did

- DRMM is a little better than Rank-SVM
  - Should it be compared to a system that does query expansion?
- DRMM is about the same as Coordinate Ascent
  - A good list-wise LeToR algorithm

18

18

Deep Relevance Matching Model (DRMM): Where Does the Learning Occur?



(Guo, et al., 2016)

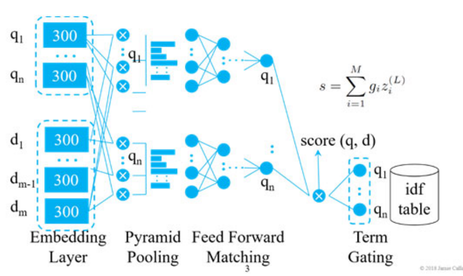
Deep Relevance Matching Model (DRMM): Where Does the Learning Occur?

DRMM learns how to combine evidence

- How to combine ‘exact match’, ‘strong match’ and ‘weak match’ signals

The word embeddings are static

- Learning cannot propagate weights through the histogram layer



(Guo, et al., 2016)

Deep Relevance Matching Model (DRMM): Similarities and Differences

Similarity to older models

- A bag-of-words model
- Exact-match of query terms to document terms
- log (tf)
- idf
- Summation of scores for each query term

Differences with older models

- Exact- and soft-match of query terms and document terms
  - Continuous representations
- Binning for matches of different quality
  - A bin for exact matches
  - Bins for 'close' and 'far' matches
- Transformed idf
  - Perhaps little effect
- Non-linear combination of match values of different quality

Outline

Introduction
Deep Structured Semantic Models (DSSM)
Deep Relevance Matching Model (DRMM)
Kernel-based Neural Ranking Model (K-NRM)
Convolutional Kernel-based Neural Ranking Model (Conv-KNRM)
BERT reranking
DeepCT
doc2query

Kernel-Based Neural Ranking Model (K-NRM): Motivating Ideas

Borrow as many good ideas as possible from DRMM

Use end-to-end training to train all parts of the model

- E.g., train word embeddings for <u>search tasks</u>
- This requires a new type of pooling layer
  – Pyramid pooling (binning) is not differentiable

Train from search log data

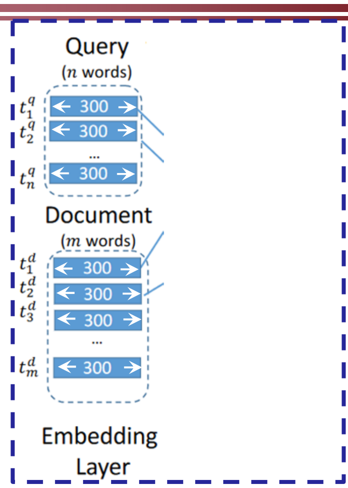
- Much <u>more</u> data than TREC relevance assessments
- <u>Noisier</u> data than TREC relevance assessments

(Xiong, et al., 2017)

23

23

Kernel-Based Neural Ranking Model (K-NRM): Text Representations



Represent query and document terms with embeddings

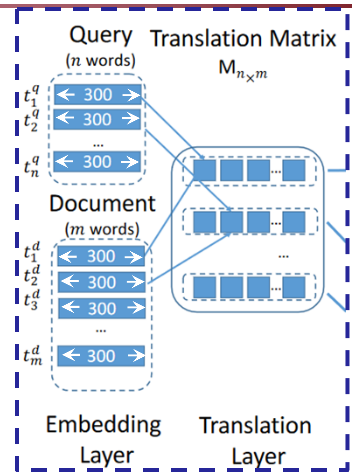
- 300 dimensions, as in DRMM
- Initialize with word2vec

(Xiong, et al., 2017)

24

24

Kernel-Based Neural Ranking Model (K-NRM): Local Interactions



(q_i, d_j) translation model

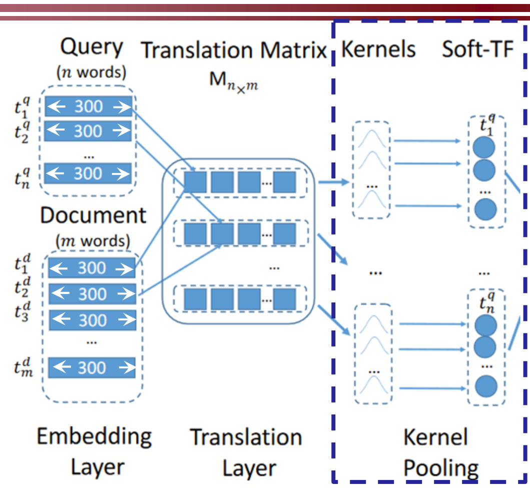
- Cosine similarities as translation scores
  - As in DRMM
- A *virtual* translation matrix
  - K-NRM just instantiates the entries needed for (q, d)

(Xiong, et al., 2017)

25

25

Kernel-Based Neural Ranking Model (K-NRM): Pooling



Kernel pooling bins matches of different strengths

- Similar to DRMM bins
  - But, differentiable
- How many similarity scores are softly in (1.0, 0.8] ?
- RBF Kernel $\mu = 0.9, \sigma = 0.1$
- $K_k(M_i) = \sum_{j=1}^{m} \exp(-\frac{M_{ij}-\mu_k}{2\sigma_k{}^2})$
- Soft-TF: softly count soft-match term frequencies

(Xiong, et al., 2017)

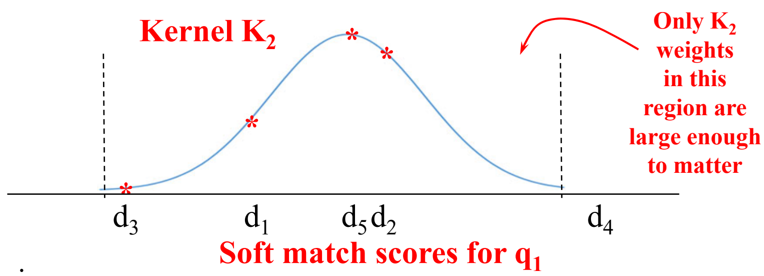
26

26

Kernel-Based Neural Ranking Model (K-NRM): Pooling

Example

- Query term: q_1
- Doc terms: $d_1 \ldots d_5$



4 soft-matches are in range for bin 2 (K_2)

- Apply the kernel to each match in range
- Sum K_2 ‘soft match’ scores to get a ‘soft tf’ score for q_i

Soft match and soft counting

Repeat for each kernel

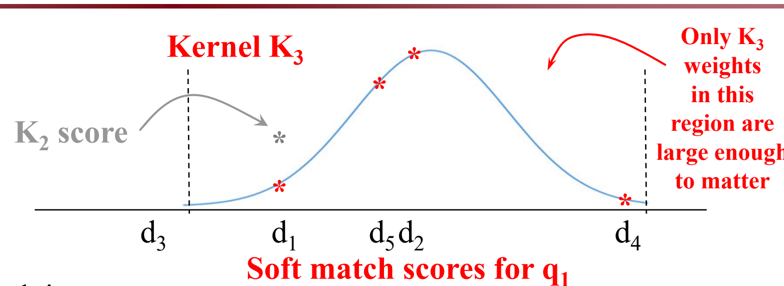
- 11 kernels, centered on different quality ranges

(Xiong, et al., 2017)

Kernel-Based Neural Ranking Model (K-NRM): Pooling

Example

- Query term: q_1
- Doc terms: $d_1 \ldots d_5$



4 soft-matches are in range for bin 3 (K_3)

- Apply the kernel to each match in range
- Sum K_3 ‘soft match’ scores to get a ‘soft tf’ score for q_i

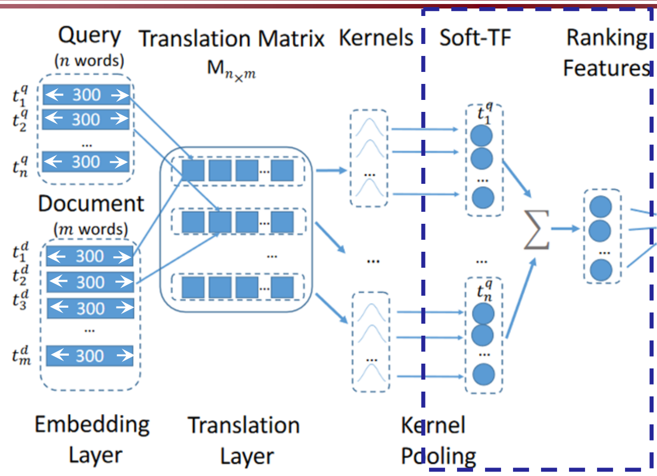
Soft match and soft counting

Repeat for each kernel

- 11 kernels, centered on different quality ranges

(Xiong, et al., 2017)

Kernel-Based Neural Ranking Model (K-NRM): Aggregation



Log-sum the soft-TF features for all query terms

- Log-sum penalizes query terms with very few matches

Output: K ranking features for a (q, d) pair ($\phi(M)$)

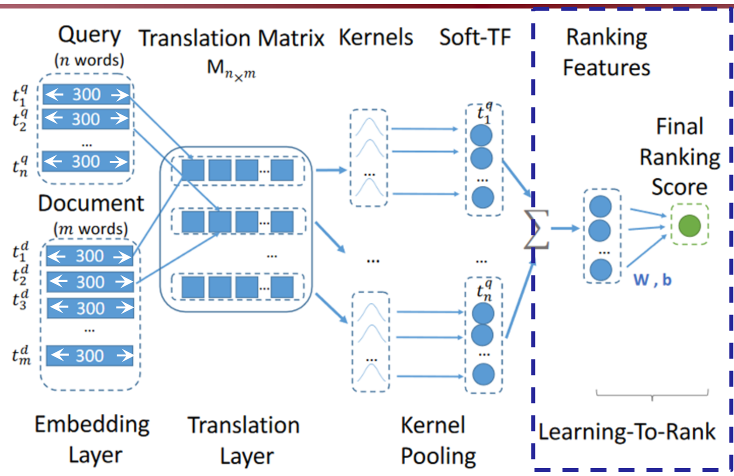
- One feature for each kernel (match level)

(Xiong, et al., 2017)

29

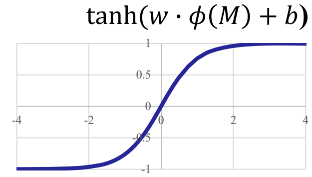
29

Kernel-Based Neural Ranking Model (K-NRM): Aggregation



Combine features into the ranking score

- A one layer feedforward NN
- $f(q, d) =$ $\tanh(w \cdot \phi(M) + b)$



Pairwise training

(Xiong, et al., 2017)

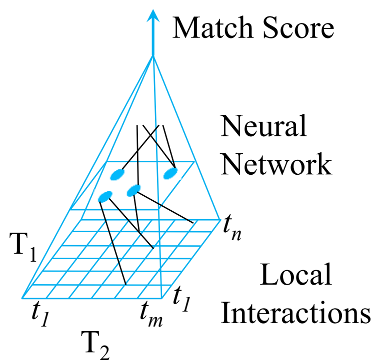
30

30

Kernel-Based Neural Ranking Model (K-NRM): Type of Neural IR Model

K-NRM is a type of interaction-based neural IR model
- Identify local matches between two pieces of text
  - E.g., cosine similarity of term vectors
- Learn interaction patterns for matching
  - Often hierarchical patterns
  - E.g., convolutional neural network

There are many interaction-based models
- DRMM, DeepMatch, ARC-II
- MatchPyramid, K-NRM



Kernel-Based Neural Ranking Model (K-NRM): Computational Complexity

Every query matches every document
- There are always soft-matches
- The computational cost is too high to be practical for initial retrieval

K-NRM is used in a re-ranking pipeline
- Use an efficient algorithm (e.g., Indri) to create a ranking
- Use K-NRM to re-rank the top n documents

Kernel-Based Neural Ranking Model (K-NRM): Training

Pairwise training with hinge loss

- p (d_6 | q) > p (d_1 | q)
- $\mathcal{L}(q, d^+, d^-; \Theta) = \max(0, 1 - s(q, d^+) + s(q, d^-))$

d^+ score (relevant) d^- score (not relevant) Training pair

What does it mean? Consider *$s(q, d^+) - s(q, d^-)$*:

- > 1: The loss is 0. The model did a nice job on this pair.
- [0 .. 1]: A small loss guides the model to push the scores of d^+ and d^- farther apart
- < 0: The loss is > 1, and indicates the seriousness of the misclassification

Used for maximum margin classification

Kernel-Based Neural Ranking Model (K-NRM): Training

Search log training data

- 95K queries, 31M search sessions
- Clicked titles > not clicked titles

Testing data

- 1K head queries, 4M search sessions
- Clicked documents assumed to be relevant

(Xiong, et al., 2017)

Kernel-Based Neural Ranking Model (K-NRM): Effectiveness

K-NRM is (much) more effective than DRMM, SVM-Rank, Coordinate Ascent and several other baselines

- Moves 1st click from rank 4 to rank 3

Method	MRR		W/T/L
Lm	0.2193	−9.19%	416/09/511
BM25	0.2280	−5.57%	456/07/473
RankSVM	0.2241	−7.20%	450/78/473
Coor-Ascent	$0.2415^{\ddagger}$	–	–/–/–/
Trans	0.2181	−9.67%	406/08/522
DRMM	$0.2335^{\ddagger}$	−3.29%	419/12/505
CDSSM	$0.2321^{\ddagger}$	−3.90%	405/11/520
K-NRM	$\mathbf{0.3379}^{\dagger\ddagger\S\P}$	+39.92%	507/05/424

(Xiong, et al., 2017)

Kernel-Based Neural Ranking Model (K-NRM): Sources of Effectiveness

Embeddings trained end-to-end for adhoc search

- In standard word2vec, hotel ≈ motel and boston ≈ dallas
- In search, 'boston hotel' ≈ 'boston motel', but not to 'dallas hotel'
- Many word2vec word pairs are decoupled
  – E.g., (wife, husband); (China-Unicom, China-mobile)
- New soft matches are discovered
  – E.g., (MH370, search); (pdf, reader); (192.168.0.1, router)
- Matching strengths are changed
  – (cloud, share) ↑, (oppor9, oppor) ↓

58% of word pairs change bins

(Xiong, et al., 2017)

Kernel-Based Neural Ranking Model (K-NRM): Sources of Effectiveness

Kernel-pooling for combining matches in a score range

- More effective than other standard pooling methods

Kernel-Based Neural Ranking Model (K-NRM): How Does it Compare?

Many similarities to DRMM, but…

- Embeddings tuned for adhoc search
  - Thus it needs much more training data
- More consistent (end-to-end) training
- A simpler model
  - Exact-match features
  - Stronger soft-match features
  - Standard learning-to-rank to combine features
  - No idf
    - » Not needed to rerank the top 10-15 from a strong engine?
    - » Perhaps binning down-weights common terms?

Outline

Conv-KNRM: Motivating Ideas

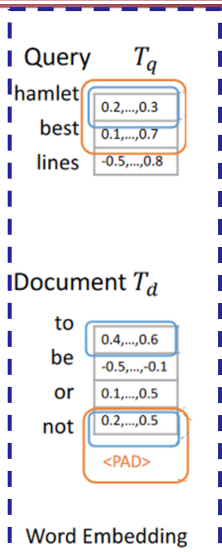
Borrow as many good ideas as possible from K-NRM

Extend the model to support n-grams

- Compose unigram embeddings to create n-gram embeddings
  – 'deep' + 'learning' vs. deep_learning
- Allow matching between n-grams of different lengths
  – E.g., 'deep learning' and 'convolutional neural network')

(Dai, et al., 2018)

Conv-KNRM: Text Representation

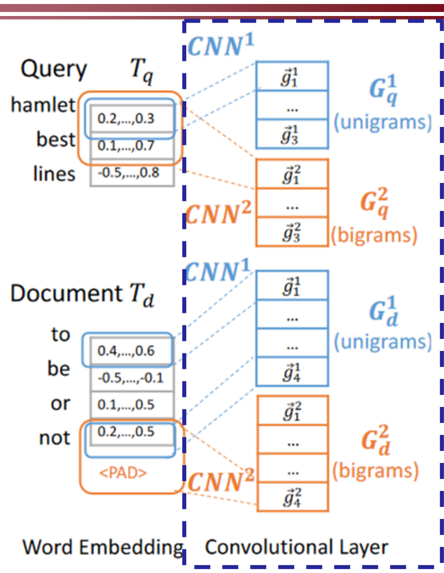


Represent query and document terms with embeddings

- 300 dimensions, as in DRMM and K-NRM
- Initialize with word2vec

(Dai, et al., 2018)

Conv-KNRM: Text Representation

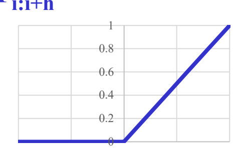


To form one n-gram of length *h* use a convolutional filter to combine *h* term embeddings

- E.g., for terms T_i, T_{i+1}, …, T_{i+h}
- $v = w_v \cdot T_{i:i+h} \qquad v \in \mathbb{R}$
- *F* filters produce a vector *V*
  - *F* different combinations

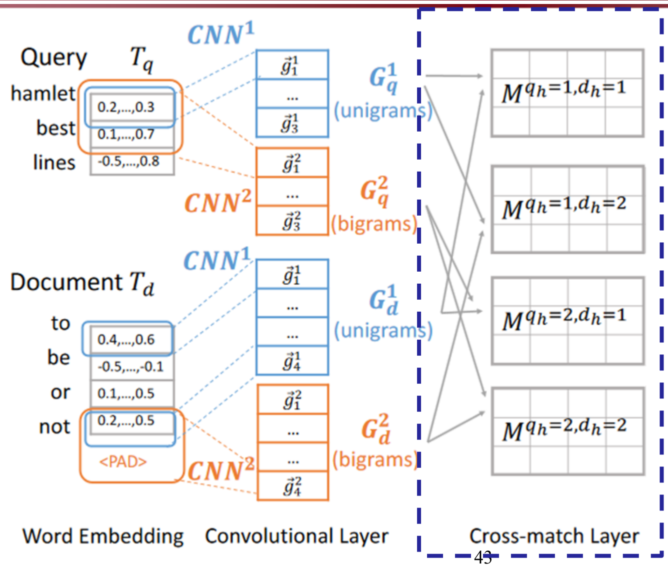
The n-gram embedding of $T_{i:i+h}$

- $g_i^h = relu\left(w^h \cdot V^h + b^h\right)$
- *relu* $(\mathbf{x}) = \max(\mathbf{0}, \mathbf{x})$
  - "rectifier linear unit"



(Dai, et al., 2018)

Conv-KNRM: Local Interactions



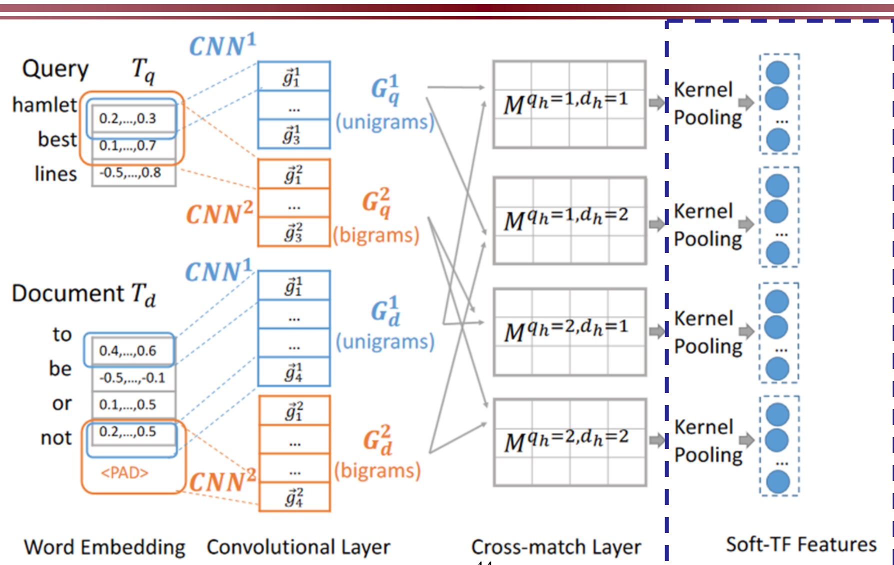
A cross-match layer allows n-grams of different lengths to match

- Similar to K-NRM's translation matrix
- 'cat movies' can match 'silly kitten videos'

(Dai, et al., 2018)

43

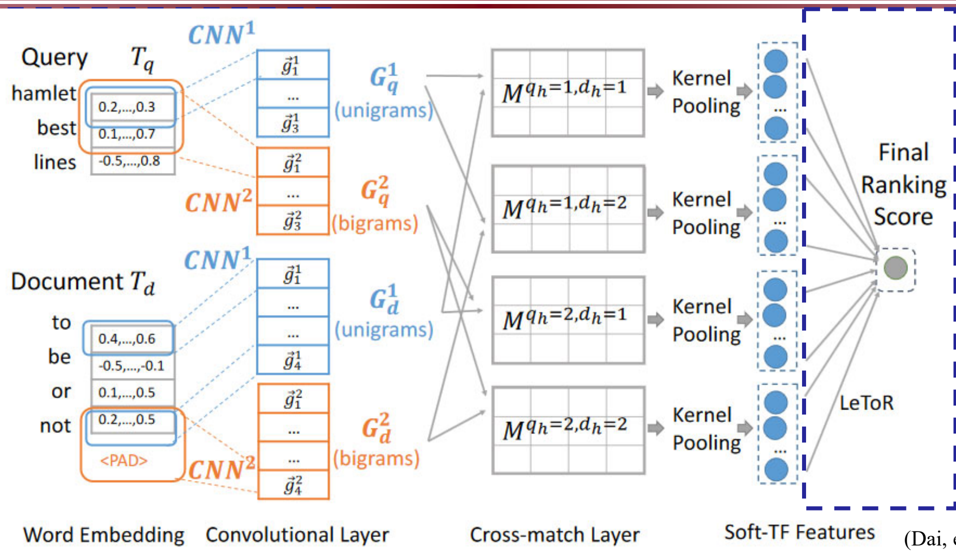
Conv-KNRM: Pooling



(Dai, et al., 2018)

44

Conv-KNRM: Aggregation



(Dai, et al., 2018)

Conv-KNRM

Conv-KNRM is a type of interaction-based neural IR model

- Identify local matches between two pieces of text
- Learn interaction patterns for matching

Conv-KNRM is used in a re-ranking pipeline

- Use an efficient algorithm (e.g., Indri) to create a ranking
- Use Conv-KNRM to re-rank the top *n* documents

Trained in the same way as K-NRM

- Pairwise training with hinge loss

(Dai, et al., 2018)

Conv-KNRM: Effectiveness

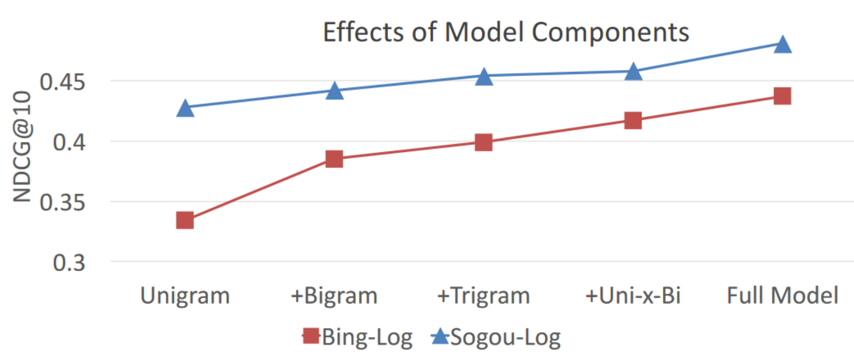
Conv-KNRM is more effective than K-NRM and several other baselines

Method	Sogou-Log MRR		Bing-Log MRR	
BM25	0.228	−33%	0.102	−61%
RankSVM	0.224	−34%	0.207	−22%
Coor-Ascent	0.242	−29%	0.208	−22%
DRMM	0.234	−31%	0.200	−25%
CDSSM	0.232	−32%	0.212	−20%
MP	0.240	−29%	$0.244^{\dagger\ddagger\S}$	−8%
K-NRM	$0.338^{\dagger\ddagger\S\P}$	––	$0.265^{\dagger\ddagger\S\P}$	––
Conv-KNRM	$0.358^{\dagger\ddagger\S\P*}$	+5%	$0.354^{\dagger\ddagger\S\P*}$	+34%

(Dai, et al., 2018)

Conv-KNRM: Effectiveness

How do the different model components contribute?



(Dai, et al., 2018)

Conv-KNRM:
Sources of Effectiveness

N-grams

- A convolutional approach to forming n-grams
- Fewer parameters to train than discrete n-grams
  - E.g., ‘white’ + ‘house’ vs. white_house

Cross-matching n-grams of different lengths

- E.g., ‘deep learning’ matches ‘convolutional neural networks’
- Another use of soft matching
- This is the most important part of Conv-KNRM

(Dai, et al., 2018)

Outline

Introduction

Deep Structured Semantic Models (DSSM)

Deep Relevance Matching Model (DRMM)

Kernel-based Neural Ranking Model (K-NRM)

Convolutional Kernel-based Neural Ranking Model (Conv-KNRM)

Summary

Summary

Continuous representations are popular again

- Lexical (DSSM), conceptual (DRMM, K-NRM, Conv-KNRM)

Two main types of architectures

- Representation-based vs. interaction-based

Integration of exact-match and soft-match signals

- Older systems were discrete or continuous, not both
- The combination seems effective and reliable (robust)

Some architectures require much training data, some don't

- E.g., trained (much data) vs static (little data) embeddings

51

51

Summary

No feature engineering … but much network engineering

- Ignore the hype … not necessarily less work

Poor understanding of how well and why the system works

- Early neural rankers compared to weak baselines (i.e., not LeToR)
- What is the contribution of different parts of the network?
- Did the system learn (good), or did it memorize (less good)?
  - Neural ranking systems are good at memorizing data

Some research embeds familiar ideas in complicated networks

- log (tf), idf, proximity, multiple bags-of-words (title, body, …)

52

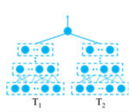
52

Summary

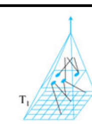
Right now the importance of neural ranking is unclear

- Two dozen or so architectures have been published recently
  - – No clear winners yet
- Interesting possibilities
  - – We are beginning to see convincing wins over strong LeToR systems
- Strong opinions on both sides … much debate

An interesting area to watch



Outline



Introduction
Deep Structured Semantic Models (DSSM)
Deep Relevance Matching Model (DRMM)
Kernel-based Neural Ranking Model (K-NRM)
Convolutional Kernel-based Neural Ranking Model (Conv-KNRM)
Summary

For More Information

- W. R. Caid, S. T. Dumais, and S. I. Gallant. Learned vector-space models for document retrieval. Information Processing and Management, 31(3), pp. 419-429, 1995.
- J. Guo, Y. Fan, Q. Ai, and W. B. Croft. A deep relevance matching model for ad-hoc retrieval. CIKM 2017.
- B. Hu, Z. Lu, H. Li, and Q. Chen. Convolutional neural network architectures for matching natural language sentences. NIPS 2014.
- P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck. Learning deep structured semantic models for web search using clickthrough data. CIKM 2013.
- K. L. Kwok. A network approach to probabilistic information retrieval. ACM Transactions on Information Systems, 13(3), pp 324-353, 1995.
- L. Pang, Y. Lan, J. Guo, J. Xu, and X. Cheng. A deep investigation of deep IR models. SIGIR 2017 Neu-IR workshop.
- Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil. Learning semantic representations using convolutional neural networks for web search. WWW 2014.
- C. Xiong, Z. Dai, J. Callan, Z. Liu, and R. Power. End-to-end neural ad-hoc ranking with kernel pooling. SIGIR 2017.