Outline

Morning program

- Preliminaries
- Modeling user behavior
- Semantic matching
- Learning to rank

Afternoon program

- Entities
- Generating responses
- Recommender systems
- Industry insights
- Q & A

Semantic matching

Definition

”... conduct query/document analysis to represent the meanings of query/document with richer representations and then perform matching with the representations.” - Li et al. [2014]

A promising area within neural IR, due to the success of semantic representations in NLP and computer vision.

Outline

- Morning program
  - Preliminaries
  - Modeling user behavior
  - Semantic matching
    - Using pre-trained unsupervised representations for semantic matching
    - Learning unsupervised representations for semantic matching
    - Learning to match models
    - Learning to match using pseudo relevance
    - Toolkits
  - Learning to rank
- Afternoon program
  - Entities
  - Generating responses
  - Recommender systems
  - Industry insights
  - Q & A

Unsupervised semantic matching with pre-trained representations

Word embeddings have recently gained popularity for their ability to encode semantic and syntactic relations amongst words.

How can we use word embeddings for information retrieval tasks?

Word embedding

Distributional Semantic Model (DSM): A model for associating words with vectors that can capture their meaning. DSM relies on the distributional hypothesis.

Distributional Hypothesis: Words that occur in the same contexts tend to have similar meanings [Harris, 1954].

Statistics on observed contexts of words in a corpus is quantified to derive word vectors.

- The most common choice of context: The set of words that co-occur in a context window.
- Context-counting VS. Context-predicting [Baroni et al., 2014]

From word embeddings to query/document embeddings

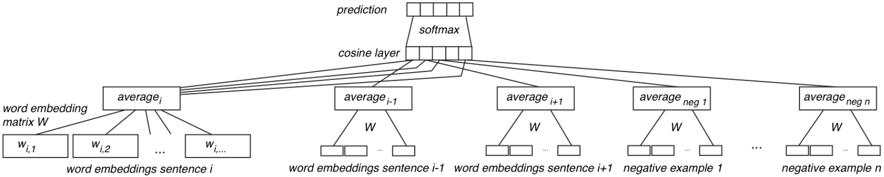
Creating representations for compound units of text (e.g., documents) from representation of lexical units (e.g., words).

From word embeddings to query/document embeddings

Obtaining representations of compound units of text (in comparison to the atomic words).

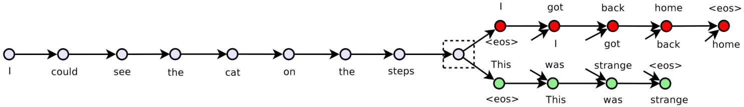
Bag of embedded words: sum or average of word vectors.

- Averaging the word representations of query terms has been extensively explored in different settings. [Vulić and Moens, 2015, Zamani and Croft, 2016b]
  - Effective but for small units of text, e.g. query [Mitra, 2015].
- Training word embeddings directly for the purpose of being averaged [Kenter et al., 2016].



From word embeddings to query/document embeddings

- Skip-Thought Vectors
  - Conceptually similar to distributional semantics: a units representation is a function of its neighbouring units, except units are sentences instead of words.

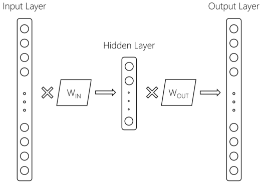


  - Similar to auto-encoding objective: encode sentence, but decode neighboring sentences.
  - Pair of LSTM-based seq2seq models with shared encoder.
- Doc2vec (Paragraph2vec) [Le and Mikolov, 2014].
- You'll hear more later about it on "Learning unsupervised representations from scratch". (Also you might want to take a look at Deep Learning for Semantic Composition)

Using similarity amongst documents, queries and terms.

Given low-dimensional representations, integrate their similarity signal within IR.

Dual Embedding Space Model (DESM) [Nalisnick et al., 2016]



Word2vec optimizes IN-OUT dot product which captures the co-occurrence statistics of words from the training corpus:

- We can gain by using these two embeddings differently

	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

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

Pre-trained word embeddings for document retrieval and ranking

DESM [Nalisnick et al., 2016]: Using IN-OUT similarity to model document aboutness.

- A document is represented by the centroid of its word OUT_vectors:

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

- Query-document similarity is average of cosine similarity over query words:

$$\text{DESM}_{\text{IN-OUT}}(q,d) = \frac{1}{q} \sum_{t_q \in q} \frac{\vec{v}_{t_{q,\text{IN}}}^{\top} \vec{v}_{t_{d,\text{OUT}}}}{\|\vec{v}_{t_{q,\text{IN}}}\| \; \|\vec{v}_{t_{d,\text{OUT}}}\|}$$

- IN-OUT captures more topical notion of similarity than IN-IN and OUT-OUT.
- DESM is effective at, but only at, ranking at least somewhat relevant documents.

Pre-trained word embeddings for document retrieval and ranking

- NTLM [Zuccon et al., 2015]: Neural Translation Language Model
  - Translation Language Model: extending query likelihood:

$$
\begin{aligned}
p(d|q) &\sim p(q|d)p(d)\\
p(q|d) &= \prod_{t_q \in q} p(t_q|d)\\
p(t_q|d) &= \sum_{t_d \in d} p(t_q|t_d)p(t_d|d)
\end{aligned}
$$

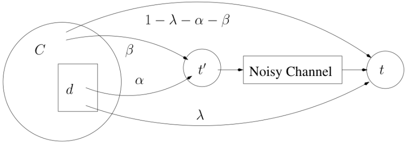
  - Uses the similarity between term embeddings as a measure for term-term translation probability $p(t_q|t_d)$.

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

Pre-trained word embeddings for document retrieval and ranking

GLM [Ganguly et al., 2015]: Generalized Language Model

- Terms in a query are generated by sampling them independently from either the document or the collection.
- The noisy channel may transform (mutate) a term t into a term t'.



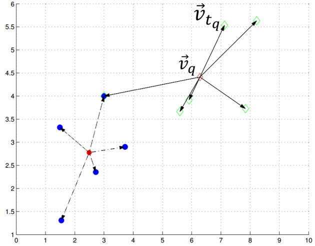
$$p(t_q|d) = \lambda p(t_q|d) + \alpha \sum_{t_d \in d} p(t_q, t_d|d) p(t_d) + \beta \sum_{t' \in N_t} p(t_q, t'|C) p(t') + 1 - \lambda - \alpha - \beta) p(t_q|C)$$

N_t is the set of nearest-neighbours of term t.

$$p(t', t|d) = \frac{\mathsf{sim}(\vec{v}_{t'}, \vec{v}_t).\mathsf{tf}(t', d)}{\sum_{t_1 \in d} \sum_{t_2 \in d} \mathsf{sim}(\vec{v}_{t_1}, \vec{v}_{t_2}).|d|}$$

Pre-trained word embeddings for query term weighting

Term re-weighting using word embeddings [Zheng and Callan, 2015].
- Learning to map query terms to query term weights.



- Constructing the feature vector $\vec{x}_{t_q}$ for term t_q using its embedding and embeddings of other terms in the same query q as:

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

- $\vec{x}_{t_q}$ measures the semantic difference of a term to the whole query.
- Learn a model to map the feature vectors the defined target term weights.

Pre-trained word embeddings for query expansion

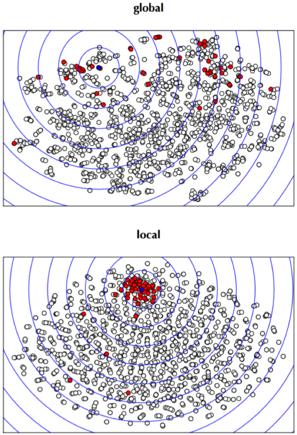
- Identify expansion terms using word2vec cosine similarity [Roy et al., 2016].
  - pre-retrieval:
    - Taking nearest neighbors of query terms as the expansion terms.
  - post-retrieval:
    - Using a set of pseudo-relevant documents to restrict the search domain for the candidate expansion terms.
  - pre-retrieval incremental:
    - Using an iterative process of reordering and pruning terms from the nearest neighbors list.
      - Reorder the terms in decreasing order of similarity with the previously selected term.
- Works better than having no query expansion, but does not beat non-neural query expansion methods.

Pre-trained word embedding for query expansion

- Embedding-based Query Expansion [Zamani and Croft, 2016a]
  Main goal: Estimating a better language model for the query using embeddings.
- Embedding-based Relevance Model:
  Main goal: Semantic similarity in addition to term matching for PRF.

Pre-trained word embedding for query expansion

Query expansion with locally-trained word embeddings [Diaz et al., 2016].



- Main idea: Embeddings be learned on topically-constrained corpora, instead of large topically-unconstrained corpora.
- Training word2vec on documents from first round of retrieval.
- Fine-grained word sense disambiguation.
- A large number of embedding spaces can be cached in practice.

Outline

Morning program

- Preliminaries
- Modeling user behavior
- Semantic matching
    - Using pre-trained unsupervised representations for semantic matching
    - Learning unsupervised representations for semantic matching
    - Learning to match models
    - Learning to match using pseudo relevance
    - Toolkits
- Learning to rank

Afternoon program

- Entities
- Generating responses
- Recommender systems
- Industry insights
- Q & A

Learning unsupervised representations for semantic matching

Pre-trained word embeddings can be used to obtain

- a query/document representation through compositionality, or
- a similarity signal to integrate within IR frameworks.

Can we learn unsupervised query/document representations directly for IR tasks?

LSI, pLSI and LDA

History of latent document representations

Latent representations of documents that are learned from scratch have been around since the early 1990s.

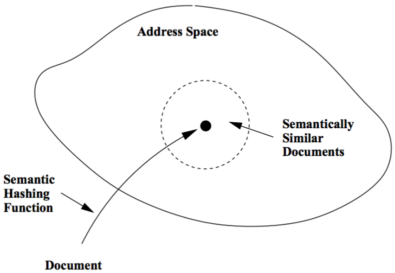
- Latent Semantic Indexing [Deerwester et al., 1990],
- Probabilistic Latent Semantic Indexing [Hofmann, 1999], and
- Latent Dirichlet Allocation [Blei et al., 2003].

These representations provide a semantic matching signal that is complementary to a lexical matching signal.

Semantic Hashing

Salakhutdinov and Hinton [2009] propose Semantic Hashing for document similarity.

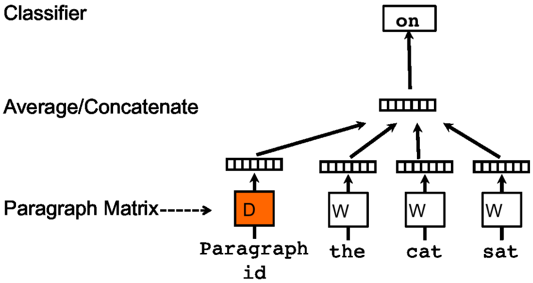
- Auto-encoder trained on frequency vectors.
- Documents are mapped to memory addresses in such a way that semantically similar documents are located at nearby bit addresses.
- Documents similar to a query document can then be found by accessing addresses that differ by only a few bits from the query document address.



Schematic representation of Semantic Hashing. Taken from Salakhutdinov and Hinton [2009].

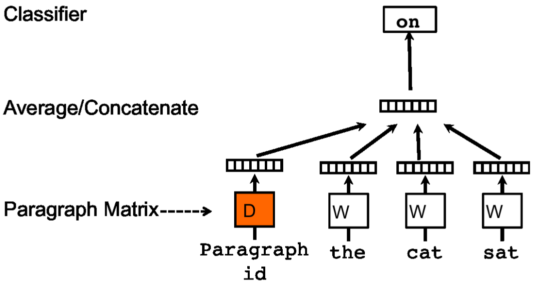
Distributed Representations of Documents [Le and Mikolov, 2014]

- Learn document representations based on the words contained within each document.
- Reported to work well on a document similarity task.
- Attempts to integrate learned representations into standard retrieval models [Ai et al., 2016a,b].

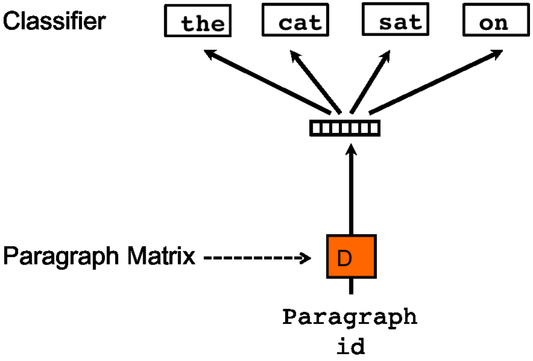


Overview of the Distributed Memory document vector model. Taken from Le and Mikolov [2014].

Two Doc2Vec Architectures [Le and Mikolov, 2014]



Overview of the Distributed Memory document vector model. Taken from Le and Mikolov [2014].



Overview of the Distributed Bag of Words document vector model. Taken from Le and Mikolov [2014].

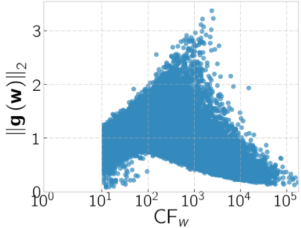
Neural Vector Spaces for Unsupervised IR [Van Gysel et al., 2017a]

- Learns query (term) and document representations directly from the document collection.

- Outperforms existing latent vector space models and provides semantic matching signal complementary to lexical retrieval models.

- Learns a notion of term specificity.

- Luhn significance: mid-frequency words are more important for retrieval than infrequent and frequent words.



Relation between query term representation L2-norm within NVSM and its collection frequency. Taken from [Van Gysel et al., 2017a].

Outline

Morning program

- Preliminaries
- Modeling user behavior
- Semantic matching
  - Using pre-trained unsupervised representations for semantic matching
  - Learning unsupervised representations for semantic matching
  - Learning to match models
  - Learning to match using pseudo relevance
  - Toolkits
- Learning to rank

Afternoon program

- Entities
- Generating responses
- Recommender systems
- Industry insights
- Q & A

Text matching as a supervised objective

Text matching is often formulated as a supervised objective where pairs of relevant or paraphrased texts are given.

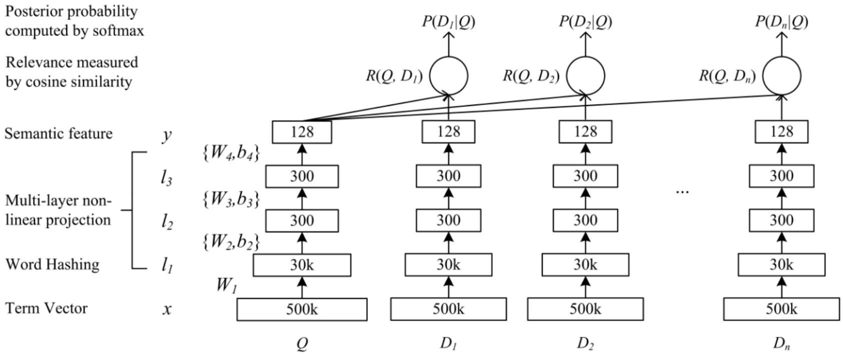
In the next few slides, we'll go over different architectures introduced for supervised text matching. Note that this is a mix of models originally introduced for (i) relevance ranking, (ii) paraphrase identification, and (iii) question answering among others.

Representation-based models

Representation-based models construct a fixed-dimensional vector representation for each text separately and then perform matching within the latent space.

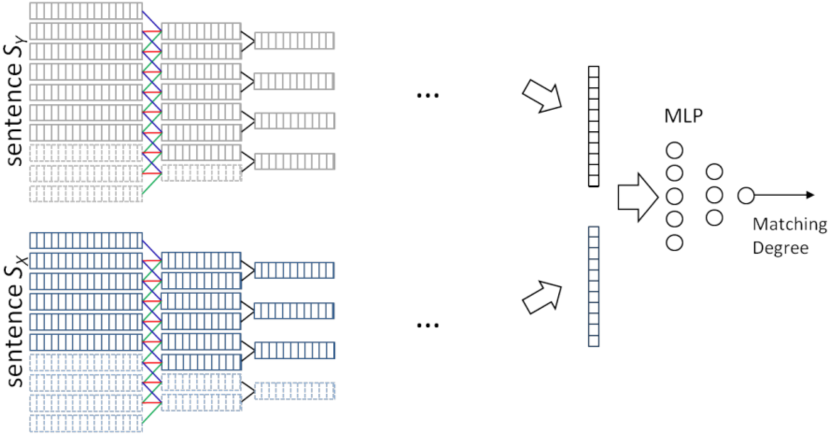
(C)DSSM [Huang et al., 2013, Shen et al., 2014]

- Siamese network between query and document, performed on character trigrams.
- Originally introduced for learning from implicit feedback.



ARC-I [Hu et al., 2014]

- Similar to DSSM, perform 1D convolution on text representations separately.
- Originally introduced for paraphrasing task.



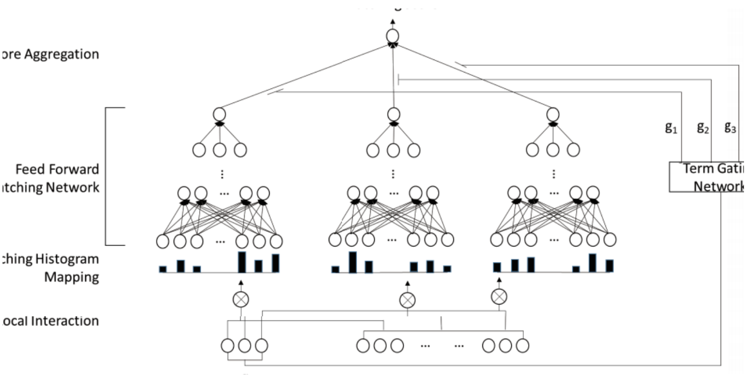
Interaction-based models

Interaction-based models compute the interaction between each individual term of both texts. An interaction can be identity or syntactic/semantic similarity.

The interaction matrix is subsequently summarized into a matching score.

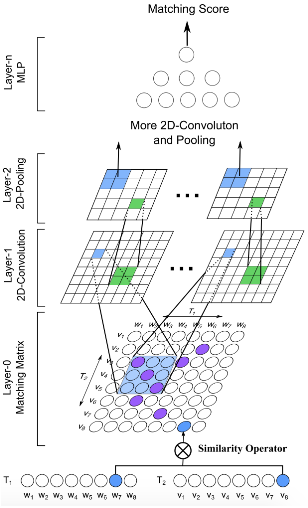
DRMM [Guo et al., 2016]

- Compute term/document interactions and matching histograms using different strategies (count, relative count, log-count).
- Pass histograms through feed-forward network for every query term.
- Gating network that produces an attention weight for every query term; per-term scores are then aggregated into a relevance score using attention weights.



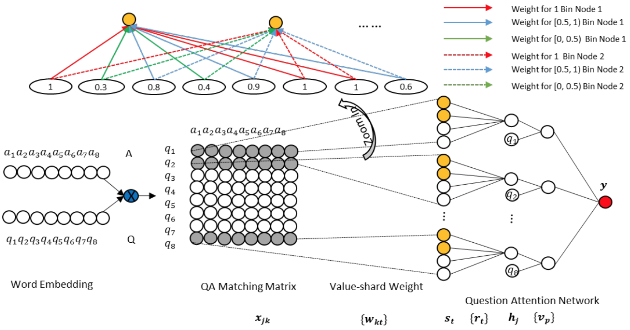
MatchPyramid [Pang et al., 2016]

- Interaction matrix between query/document terms, followed by convolutional layers.
- After convolutions, feed-forward layers determine matching score.



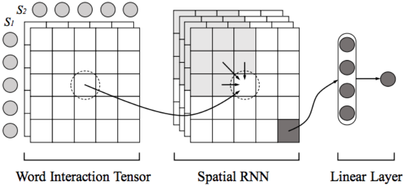
aNMM [Yang et al., 2016]

- Compute word interaction matrix.
- Aggregate similarities by running multiple kernels.
- Every kernel assigns a different weight to a particular similarity range.
- Similarities are aggregated to the kernel output by weighting them according to which bin they fall in.


Weight for 1 Bin Node 1
Weight for [0.5, 1) Bin Node 1
Weight for [0, 0.5) Bin Node 1
Weight for 1 Bin Node 2
Weight for [0.5, 1) Bin Node 2
Weight for [0, 0.5) Bin Node 2
……
1
0.3
0.8
0.4
0.9
1
1
0.6
zoom
$a_1 a_2 a_3 a_4 a_5 a_6 a_7 a_8$
$a_1 a_2 a_3 a_4 a_5 a_6 a_7 a_8$
A
$q_1 q_2 q_3 q_4 q_5 q_6 q_7 q_8$
Q
q_1
q_2
q_3
q_4
q_5
q_6
q_7
q_8
q_1
q_2
q_8
y
Word Embedding
QA Matching Matrix
Value-shard Weight
Question Attention Network
x_{jk}
$\{w_{kt}\}$
s_t
$\{r_t\}$
h_j
$\{v_p\}$

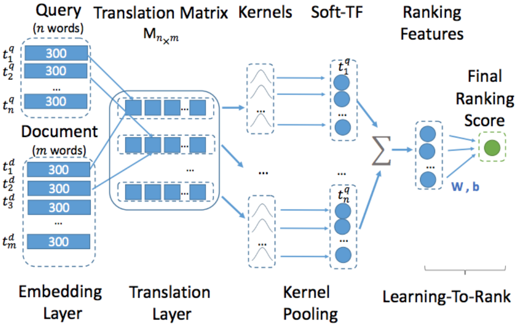
Match-SRNN [Wan et al., 2016b]

- Word interaction layer, followed by a spatial recurrent NN.
- The RNN hidden state is updated using the current interaction coefficient, and the hidden state of the prefix.



K-NRM [Xiong et al., 2017b]

- Compute word-interaction matrix, apply k kernels to every query term row in interaction matrix.
- This results in k-dimensional vector.
- Aggregate the query term vectors into a fixed-dimensional query representation.



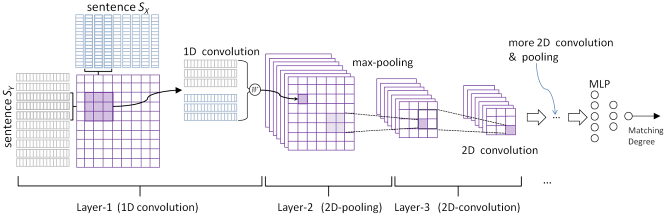
Hybrid models

Hybrid models consist of (i) a representation component that combines a sequence of words (e.g., a whole text, a window of words) into a fixed-dimensional representation and (ii) an interaction component.

These two components can occur (1) in serial or (2) in parallel.

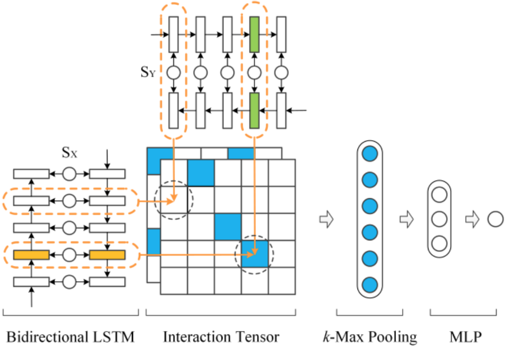
ARC-II [Hu et al., 2014]

- Cascade approach where word representation are generated from context.
- Interaction matrix between sliding windows, where the interaction activation is computed using a non-linear mapping.
- Originally introduced for paraphrasing task.



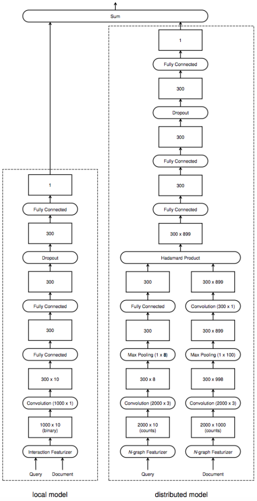
MV-LSTM [Wan et al., 2016a]

- Cascade approach where input representations for the interaction matrix are generated using a bi-directional LSTM.
- Differs from pure interaction-based approaches as the LSTM builds a representation of the context, rather than using the representation of a word.
- Obtains fixed-dimensional representation by max-pooling over query/document; followed by feed-forward network.



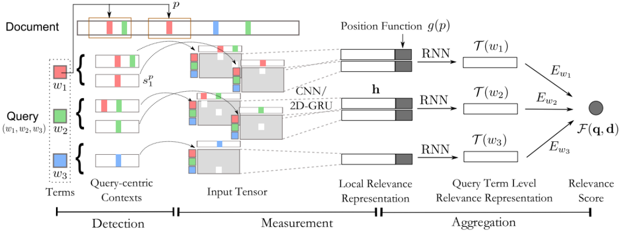
Duet [Mitra et al., 2017]

- Model has an interaction-based and a representation-based component.
  - Interaction-based component consist of a indicator matrix showing where query terms occur in document; followed by convolution layers.
  - Representation-based component is similar to DSSM/ARC-I, but uses a feed-forward network to compute the similarity signal rather than cosine similarity.
- Both are combined at the end using a linear combination of the scores.



DeepRank [Pang et al., 2017]

- Focus only on exact term occurrences in document.
- Compute interaction between query and window surrounding term occurrence.
- RNN or CNN then combines per-window features (query representation, context representations and interaction between query/document term) into matching score.



Outline

- Morning program
  - Preliminaries
  - Modeling user behavior
  - Semantic matching
    - Using pre-trained unsupervised representations for semantic matching
    - Learning unsupervised representations for semantic matching
    - Learning to match models
    - Learning to match using pseudo relevance
    - Toolkits
  - Learning to rank
- Afternoon program
  - Entities
  - Generating responses
  - Recommender systems
  - Industry insights
  - Q & A

Beyond supervised signals: semi-supervised learning

The architectures we presented for learning to match all require labels. Typically these labels are obtained from domain experts.

However, in information retrieval, there is the concept of pseudo relevance that gives us a supervised signal that was obtained from unsupervised data collections.

Pseudo test/training collections

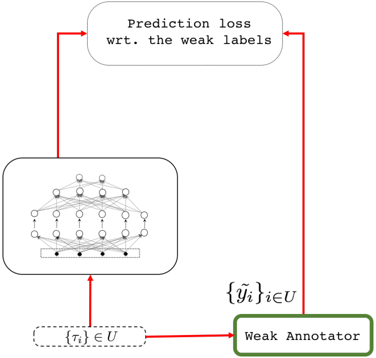
Given a source of pseudo relevance, we can build pseudo collections for training retrieval models [Asadi et al., 2011, Berendsen et al., 2013].

Sources of pseudo-relevance

Typically given by external knowledge about retrieval domain, such as hyperlinks, query logs, social tags, ...

Training neural networks using pseudo relevance

Training a neural ranker using weak supervision [Dehghani et al., 2017].



Main idea: Annotating a large amount of unlabeled data using a weak annotator (Pseudo-Labeling) and design a model which can be trained on weak supervision signal.

- Function approximation. (re-inventing BM25?)
- Beating BM25 using BM25!

Training neural networks using pseudo relevance

Generating weak supervision training data for training neural IR model [MacAvaney et al., 2017].

- Using a news corpus with article headlines acting as pseudo-queries and article content as pseudo-documents.
- Problems:
  - **Hard-Negative**
  - **Mismatched-Interaction**: (example: "When Bird Flies In", a sports article about basketball player Larry Bird)
- Solutions:
  - **Ranking filter**:
    - top pseudo-documents are considered as negative samples.
    - only pseudo-queries that are able to retrieve their pseudo-relevant documents are used as positive samples.
  - **Interaction filter**:
    - building interaction embeddings for each pair.
    - filtering out based on similarity to the template query-document pairs.

Query expansion using neural word embeddings based on pseudo relevance

Locally trained word embeddings [Diaz et al., 2016]

- Performing topic-specific training, on a set of topic specific documents that are collected based on their relevance to a query.

Relevance-based Word Embedding [Zamani and Croft, 2017].

- Relevance is not necessarily equal to semantically or syntactically similarity:
  - “united state” as expansion terms for “Indian American museum”.
- Main idea: Defining the “context”
  Using the relevance model distribution for the given query to define the context.
  So the objective is to predict the words observed in the documents relevant to a particular information need.
- The neural network will be constraint by the given weights from RM3 to learn word embeddings.

Outline

Morning program

- Preliminaries
- Modeling user behavior
- Semantic matching
  - Using pre-trained unsupervised representations for semantic matching
  - Learning unsupervised representations for semantic matching
  - Learning to match models
  - Learning to match using pseudo relevance
  - Toolkits
- Learning to rank

Afternoon program

- Entities
- Generating responses
- Recommender systems
- Industry insights
- Q & A

Document & entity representation learning toolkits

gensim : `https://github.com/RaRe-Technologies/gensim` [Řehůřek and Sojka, 2010]

SERT : `http://www.github.com/cvangysel/SERT` [Van Gysel et al., 2017b]

cuNVSM : `http://www.github.com/cvangysel/cuNVSM` [Van Gysel et al., 2017a]

HEM : `https://ciir.cs.umass.edu/downloads/HEM` [Ai et al., 2017]

MatchZoo : `https://github.com/faneshion/MatchZoo` [Fan et al., 2017]