Semantic Matching using Neural Networks

Information Retrieval

CSE, IIT Kharagpur

March 20th, 2020

Semantic Matching

tf-idf LM etc.

Definition

"... conduct query/document analysis to represent the meanings of query/document with richer representations and then perform matching with the representations."

i.e., go beyong keyword (lexical) matching.

We will discuss both unsupervised and supervised methods of semantic matching.

(Pseudo) Relevance Feedback

Beyond keyword matching?

→ Query Expansion

Semantic Matching: What have we seen till now?

- Query expansion
- Relevance Feedback
- Translation Model (How to model word similarity?)

Semantic Matching: What have we seen till now?

- Query expansion
- Relevance Feedback
- Translation Model (How to model word similarity?)

Disrtributional Hypothesis

Words that occur in similar contexts tend to have similar meanings.

word similarity

Distributional Semantic Models

		driving	learns	syllabus
	automobile	✓	✓	✗
similar	car	✓	✓	✗
	board	✗	✓	✓
similar	classroom	✗	✓	✓

Pointwise Mutual information

Semantic Matching: What have we seen till now?

- Query expansion
- Relevance Feedback
- Translation Model (How to model word similarity?)

Disrtributional Hypothesis

Words that occur in similar contexts tend to have similar meanings.

Word embeddings have proved to be very important for modeling semantic similarity

Word2Vec – A distributed representation

Distributional representation – word embedding?

Any word w_i in the corpus is given a distributional representation by an embedding

$$w_i \in R^d$$

i.e., a $d-$dimensional vector, which is mostly learnt! without any annotations

Word2Vec – A distributed representation

Distributional representation – word embedding?

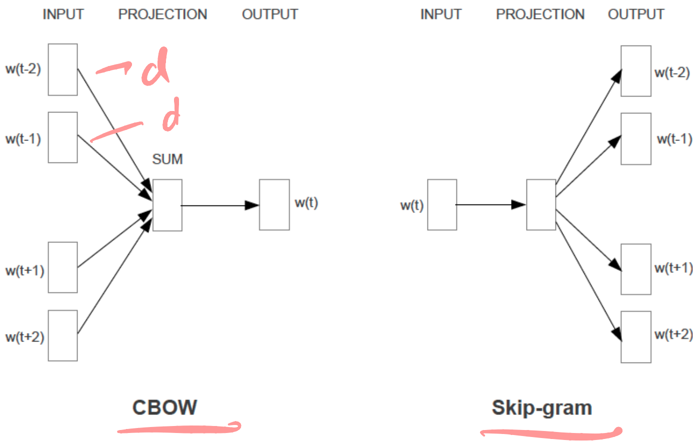
Any word w_i in the corpus is given a distributional representation by an embedding

$$w_i \in R^d$$

i.e., a $d-$dimensional vector, which is mostly learnt!

$$\textit{linguistics} = \begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \end{pmatrix} \rightarrow \text{8-dim}$$

Two Variations: CBOW and Skip-grams



What do we finally have?

- For each word w_i in vocabulary (size V), we have two vectors: v_i^{IN} and v_i^{OUT}, each of $d-$dimensions.
- Generally, you can just add these vectors and use $v_i = v_i^{IN} + v_i^{OUT}$
- Ideally, similar words will have similar vectors

How do we go about using these for the retrieval task

Cosine similarity

Query Expansion? →

Pre-trained word embeddings for query expansion

Basic Idea

Identify expansion terms using word2Vec cosine similaity

- 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.

q — tf idf / VSM — top 5 inter-section

Neural Translation Language Model

Language Model: Using Query Likelihood

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

What happens in translation language model

type mismatch

q - songs   d: hits

Neural Translation Language Model

Language Model: Using Query Likelihood

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

What happens in translation language model

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

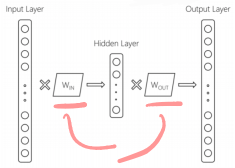
You can use similarity between term embeddings for term-term translation probability, thus

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

$p(t_q | t_d)$

Dual Embedding Space Model (DESM)

Dual Embedding



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

Nalisnick et al., 2016. Improving Document Ranking with Dual Word Embeddings. (WWW '16 Companion).

$v_{in} \cdot v_{out}$

given x $\quad$ z_1, z_2, z_3 -

BTP ✓ IR system q_{in} d_{out} / d_n

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 universities / Names
- 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

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

- IN-OUT captures more topical notion of similarity than IN-IN and OUT-OUT.

OUT-OUT IN-IN

- Train CBOW from either
  - 600 million Bing queries
  - 342 million web document sentences
- Test on 7,741 randomly sampled Bing queries
  - 5 level eval (Perfect, Excellent, Good, Fair, Bad)
- Two approaches
  1. Use DESM model to rerank top results from BM25
  2. Use DESM alone or a mixture model of it and BM25

$$MM(Q,D) = \alpha DESM(Q,D) + (1-\alpha)BM25(Q,D)$$

$$\alpha \in \mathbb{R}, 0 \leq \alpha \leq 1$$

7.5K Q's

: Top 15 doc. returned by system

tf-idf

tf

Val set

Results: Reranking k-best list

top 10

	Explicitly Judged Test Set		
	NDCG@1	NDCG@3	NDCG@10
BM25	23.69	29.14	44.77
LSA	22.41*	28.25*	44.24*
DESM (IN-IN, trained on body text)	23.59	29.59	45.51*
DESM (IN-IN, trained on queries)	23.75	29.72	46.36*
DESM (IN-OUT, trained on body text)	24.06	30.32*	46.57*
DESM (IN-OUT, trained on queries)	**25.02***	**31.14***	**47.89***

Pretty decent gains – e.g., 2% for NDCG@3

Gains are bigger for model trained on queries than docs

Results: whole ranking system

	Explicitly Judged Test Set		
	NDCG@1	NDCG@3	NDCG@10
BM25	21.44	26.09	37.53
LSA	04.61*	04.63*	04.83*
DESM (IN-IN, trained on body text)	06.69*	06.80*	07.39*
DESM (IN-IN, trained on queries)	05.56*	05.59*	06.03*
DESM (IN-OUT, trained on body text)	01.01*	01.16*	01.58*
DESM (IN-OUT, trained on queries)	00.62*	00.58*	00.81*
BM25 + DESM (IN-IN, trained on body text)	21.53	26.16	37.48
BM25 + DESM (IN-IN, trained on queries)	**21.58**	26.20	37.62
BM25 + DESM (IN-OUT, trained on body text)	21.47	26.18	37.55
BM25 + DESM (IN-OUT, trained on queries)	21.54	**26.42***	**37.86***

Unsupervised

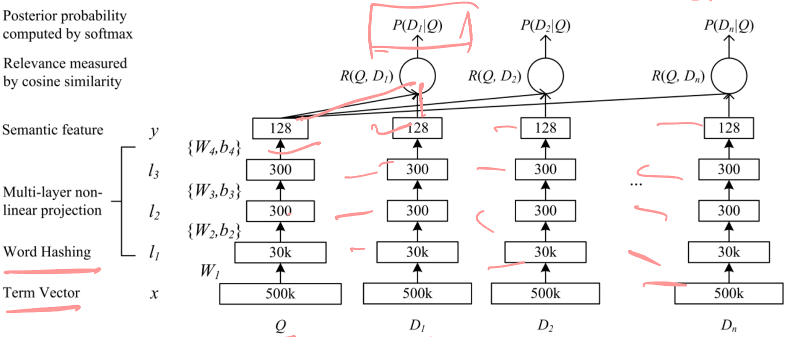
Semantic Matching – with Supervision

learn how to represent

← q

d+ rel

d- non-rel

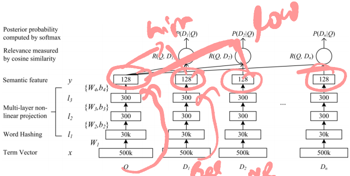


Deep Structured Semantic Model (DSSM) [Huang et al., 2013]

DSSM

1. Represent query and document as vectors $\mathbf{q}$ and $\mathbf{d}$ in a latent vector space

2. Estimate the matching degree between $\mathbf{q}$ and $\mathbf{d}$ using cosine similarity



Deep Structured Semantic Model (DSSM) [Huang et al., 2013]

Why supervised?

We **learn** to represent queries and documents in the latent vector space by forcing the vector representations

- for relevant query-document pairs (q, d^+) to be close in the latent space; and
- for irrelevant query-document pairs (q, d^-) to be far in the latent vector space

Understanding DSSM - How to represent text

500k

How to represent text (e.g., Shinjuku Gyoen)?

1. Bag of Words (BoW) [large vocabulary (500000 words)]

   { 0, . . . , 0 (apple), 0, . . . , 0, 1 (gyoen), 0, . . . , 0, 1 (shinjuku), 0, . . . , 0 }

   500k dim

2. Bag of Letter Trigrams (BoLT) [small vocabulary (30621 letter 3-grams)]

   { 0, . . . , 0 (abc), 0, . . . , 1 (_gy), 0, . . . , 0, 1 (_sh), 0, . . . , 0, 1 (en_), 0, . . . , 0, 1 (gyo), 0, . . . , 0, 1 (hin), 0, . . . , 0, 1 (inj), 0, . . . , 0, 1 (juk), 0, . . . , 0, 1 (ku_), 0, . . . , 0, 1 (oen), 0, . . . , 0, 1 (shi), 0, . . . , 0, 1 (uku), 0, . . . , 0, 1 (yoe), 0 }

30k

Bat

mammal

Wood

2 vectors

Understanding DSSM - Architecture

$\{d_1, \ldots d_{15}\}$

4 3 0 1 2

$$\mathbf{x} = \text{BoW}(\text{text})$$

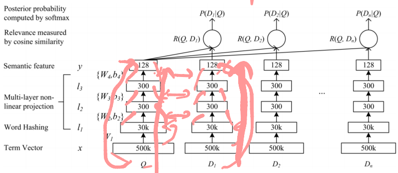
$$\mathbf{l}_1 = \text{WordHashing}(\mathbf{x})$$ 30K

$$\mathbf{l}_2 = \tanh(W_2\mathbf{l}_1 + b_2)$$

$$\mathbf{l}_3 = \tanh(W_3\mathbf{l}_2 + b_3)$$

$$\mathbf{l}_4 = \tanh(W_4\mathbf{l}_3 + b_4)$$

[128]



Web queries Q

$\{q_1 \ldots q_i \ldots q_n\}$

Siamese

26 x 26 x 26

Rel NR NR

$\vec{q}$ (128)

$\vec{d_1}$ $\vec{d_2}$ $\vec{d_n}$

$$\max \underline{P(D_i^+ | \theta)}$$

softmax

$$score(D_i | \theta) \approx \underline{\lambda} \cdot \cos(D_i, \theta)$$

hyper-parameters

$$e^{\lambda \cos(D_i', \theta)} / NR$$

$$\max \underline{P(D_i^+ | \theta)} \quad \frac{e^{\lambda \cos(D_i, \theta)}}{\sum_{D_i'} e^{\lambda \underline{\cos(D_i', \theta)}}} \approx$$

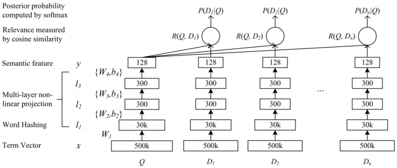
$$\underline{-\log P(D_i^+ | \theta)}$$

Error

DSSM - Training Objective

Likelihood

$$\prod_{(q,d^+)\in \text{DATA}} P(d^+ \mid q) \to \max$$



$$P(d^+ \mid q) = \frac{e^{\gamma \cos(\mathbf{q},\mathbf{d}^+)}}{\sum_{d\in D} e^{\gamma \cos(\mathbf{q},\mathbf{d})}} \approx \frac{e^{\gamma \cos(\mathbf{q},\mathbf{d}^+)}}{\sum_{d\in D^+\cup D^-} e^{\gamma \cos(\mathbf{q},\mathbf{d})}}$$

NR

Evaluation Details

- 16,510 English queries sampled from one year query log files of Bing
- Each query is associated with 15 web document titles
- Relevance judgement on a scale of 0 to 4

DSSM - Results

	NDCG		
Model	**@1**	**@3**	**@10**
TF-IDF	0.319	0.382	0.462
BM25	0.308	0.373	0.455
WTM	0.332	0.400	0.478
LSA	0.298	0.372	0.455
PLSA	0.295	0.371	0.456
DAE	0.310	0.377	0.459
BLTM	0.337	0.403	0.480
DPM	0.329	0.401	0.479
DSSM	0.362	0.425	0.498

web queries

doc titles

1 DPESM

Unsupervised

IN- OUT sim

+ BM25

Rerank

2 Supervised

latent space

q

D

9

seq. of words

Naive

average word vector

10 words