Deep Semantic Similarity Model for Text Processing

Presented by Xiaodong He and Jianfeng Gao
DSSM for learning the semantic meaning of texts

Learning the semantic meaning of texts is a key problem in NLP
Semantic Embedding

Word embedding: representing the meaning of a word by a vector

From discrete symbolic representation to continuously-valued vector representation

\[ f(\text{cat}) = \text{one-hot word vector} \]

The index of “cat” in the vocabulary

\[ f(\text{cat}) = \text{word embedding vector} \]

Common neural network based word embedding approaches

(Bengio 2001; Schwenk et al., 2006; Collobert et al., 2011; Mikolov et al. 2011, 2013, etc.)
However, a decomposable, robust representation is preferable for large scale NL tasks

New words, misspellings, and word fragments frequently occur (generalizability)

Vocabulary of real-world big data tasks could be huge (scalability)

e.g., 100M+ unique words in a modern commercial search engine log
From Word to Sub-word Unit

Decompose word to sub-word units, e.g., letter-trigram (LTG)

cat → #cat# → #-c-a, c-a-t, a-t-#

Unbounded variability (word) => bounded variability (sub-word)

E.g., only ~50K letter-trigrams in English \((37^3)\)

\[
W \rightarrow U \times V
\]

embedding vector

\(\text{dim} = 500\)

word embedding matrix:

\(500 \times 100M\)

1-hot word vector

\(\text{dim} = 100M\)

Could even go up to infinity

[Huang, He, Gao, Deng, Acero, Heck, CIKM2013]
Letter-trigram as the Sub-word Unit

Learn one vector per letter-trigram (LTG), the encoding matrix is a fixed matrix. Use the count of each LTG in the word for encoding.

Example: cat → #c-a, c-a-t, a-t# (w/ word boundary mark #)

- Address both the scalability and generalizability issues
Semantic Embedding: from Word to Phrase

The semantic intent is better defined at the phrase/sentence level rather than at the word level

- The meaning of a single word is often ambiguous
- A phrase/sentence/document contains rich contextual information that could be leveraged
DSSM for Semantic Embedding Learning

Deep structured semantic model/Deep semantic similarity model (DSSM)
The DSSM refers to a series of deep semantic models developed recently at MSR
With variations on model structures and training objectives

The DSSM is trained by an semantic similarity-driven objective
projecting semantically similar phrases to vectors close to each other
projecting semantically different phrases to vectors far apart

The DSSM uses the letter-trigram sub-word vector for the input word representation

[Huang, He, Gao, Deng, Acero, Heck, CIKM2013]
[Shen, He, Gao, Deng, Mesnil, WWW2014]
[Gao, He, Yih, Deng, ACL2014]
[Yih, He, Meek, ACL2014]
[He, Gao, Deng, ICASSP2014 Tutorial]
DSSM for Semantic Embedding Learning

**Initialization:**

Neural networks are initialized with random weights.

**Semantic vector**

- $v_s$
- $d=300$
- $W_4$
- $d=500$
- $W_3$
- $d=500$
- $W_2$
- $\text{dim} = 50K$
- $W_1$
- $\text{dim} = 100M$

**Letter-trigram embedding matrix**

- $s$: "racing car"

**Letter-trigram encoding matrix (fixed)**

**Bag-of-words vector**

**Input word/phrase**

- $v_{t^+}$
- $d=300$
- $d=500$
- $d=500$
- $\text{dim} = 100M$
- $t^+$: "formula one"

- $v_{t^-}$
- $d=300$
- $d=500$
- $d=500$
- $\text{dim} = 100M$
- $t^-$: "racing to me"

**Initialization:**

Neural networks are initialized with random weights.
DSSM for Semantic Embedding Learning

**Training:**
Compute Cosine similarity between semantic vectors

\[ \frac{\partial}{\partial W} \frac{\exp(\cos(v_s, v_{t^+}))}{\sum_{t'=t^+}^{t^-} \exp(\cos(v_s, v_{t'}))} \]

Semantic vector

- \( v_s \):
  - \( d = 300 \)
  - \( W_4 \):
    - \( d = 500 \)
  - \( W_3 \):
    - \( d = 500 \)
  - \( W_2 \):
    - \( \text{dim} = 50K \)
  - \( W_1 \):
    - \( \text{dim} = 100M \)

- \( v_{t^+} \):
  - \( d = 300 \)
  - \( \text{dim} = 50K \)

- \( v_{t^-} \):
  - \( d = 300 \)
  - \( \text{dim} = 100M \)

Letter-trigram embedding matrix
- \( W_1 \):
  - \( \text{dim} = 100M \)

Letter-trigram encoding matrix (fixed)

Bag-of-words vector
- \( s: \text{"racing car"} \)

Input word/phrase
- \( t^+: \text{"formula one"} \)
- \( t^-: \text{"racing to me"} \)

Compute Cosine similarity between semantic vectors

\[ \cos(v_s, v_{t^+}) \]

\[ \cos(v_s, v_{t^-}) \]

Compute gradients

\[ t' = \{t^+, t^-\} \]

Training:

1. Compute Cosine similarity between semantic vectors.
2. Compute gradients
3. Compute Cosine similarity between semantic vectors.

**Semantic vector**

**Input word/phrase**

**DSSM for Semantic Embedding Learning**
DSSM for Semantic Embedding Learning

Runtime:

Semantic vector

Letter-trigram embedding matrix
Letter-trigram encoding matrix (fixed)
Bag-of-words vector
Input word/phrase

\[ \mathbf{v_s} \]

\[ d=300 \]

\[ W_4 \]

\[ d=500 \]

\[ W_3 \]

\[ d=500 \]

\[ W_2 \]

\[ \text{dim} = 50K \]

\[ W_1 \]

\[ \text{dim} = 100M \]

\( s: \text{"racing car"} \)

\[ \mathbf{v_{t1}} \]

\[ d=300 \]

\[ d=500 \]

\[ d=500 \]

\[ \text{dim} = 100M \]

\( t1: \text{"formula one"} \)

\[ \mathbf{v_{t2}} \]

\[ d=300 \]

\[ d=500 \]

\[ d=500 \]

\[ d=500 \]

\[ \text{dim} = 100M \]

\( t2: \text{"racing to me"} \)

\( similar \)

\( apart \)
Evaluation

Evaluated on an information retrieval task
Docs are ranked by the cosine similarity between semantic vectors of the query and the doc

<table>
<thead>
<tr>
<th>Model</th>
<th>Input dimension</th>
<th>NDCG@1 %</th>
</tr>
</thead>
<tbody>
<tr>
<td>BM25 baseline</td>
<td>--</td>
<td>30.8</td>
</tr>
<tr>
<td>Probabilistic LSA (PLSA)</td>
<td></td>
<td>29.5</td>
</tr>
<tr>
<td>Auto-Encoder (Word)</td>
<td>40K</td>
<td>31.0 (+0.2)</td>
</tr>
<tr>
<td>DSSM (Word)</td>
<td>40K</td>
<td>34.2 (+3.4)</td>
</tr>
<tr>
<td>DSSM (Random projection)</td>
<td>30K</td>
<td>35.1 (+4.3)</td>
</tr>
<tr>
<td>DSSM (Letter-trigram)</td>
<td>30K</td>
<td>36.2 (+5.4)</td>
</tr>
</tbody>
</table>

The higher the NDCG score the better, 1% NDCG difference is statistically significant.

DSSM-based embedding improves 5~7 pt NDCG over shallow models
Comparison: Auto-encoder vs. DSSM

**Auto-encoder**
- **Supervision:** AE: unsupervised (e.g., doc<->doc)  
  DSSM: weakly supervised (e.g., query<->doc search log)
- **Training objective:** AE: reconstruction error  
  DSSM: distance between embedding vectors
- **Input representation:** AE: 1-hot word vector  
  DSSM: letter-trigram

**DSSM**
- **Supervision:**  
  - AE: unsupervised (e.g., doc<->doc)  
  - DSSM: weakly supervised (e.g., query<->doc search log)
- **Training objective:** AE: reconstruction error  
  DSSM: distance between embedding vectors
- **Input representation:** AE: 1-hot word vector  
  DSSM: letter-trigram

The DSSM can be trained using a variety of signals without costly labeling effort (e.g., user behavior log data).
DSSM for Semantic Word Clustering and Analogy

Learn word embedding by means of its neighbors (context)

Construct context $\leftrightarrow$ word training pair for DSSM

Training Condition:
30K vocabulary size
10M words from Wikipedia
50-dimentional vector
Pure unsupervised training

$\text{dim} = 120K$

$d=300$

$d=500$

Similar

$s: \ "w(t-2) \ w(t-1) \ w(t+1) \ w(t+2)"$

$t: \ "w(t)"$

[Song et al. 2014]
Plotting 3K words in 2D
Plotting 3K words in 2D
Plotting 3K words in 2D
DSSM for Word Clustering and Analogy

Semantic clustering examples: top 3 neighbors of each word

<table>
<thead>
<tr>
<th>Word</th>
<th>Neighbor 1 (Similarity)</th>
<th>Neighbor 2 (Similarity)</th>
<th>Neighbor 3 (Similarity)</th>
</tr>
</thead>
<tbody>
<tr>
<td>king</td>
<td>earl (0.77)</td>
<td>pope (0.77)</td>
<td>lord (0.74)</td>
</tr>
<tr>
<td>woman</td>
<td>person (0.79)</td>
<td>girl (0.77)</td>
<td>man (0.76)</td>
</tr>
<tr>
<td>france</td>
<td>spain (0.94)</td>
<td>italy (0.93)</td>
<td>belgium (0.88)</td>
</tr>
<tr>
<td>rome</td>
<td>constantinople (0.81)</td>
<td>paris (0.79)</td>
<td>moscow (0.77)</td>
</tr>
<tr>
<td>winter</td>
<td>summer (0.83)</td>
<td>autumn (0.79)</td>
<td>spring (0.74)</td>
</tr>
<tr>
<td>rain</td>
<td>rainfall (0.76)</td>
<td>storm (0.73)</td>
<td>wet (0.72)</td>
</tr>
<tr>
<td>car</td>
<td>truck (0.8)</td>
<td>driver (0.73)</td>
<td>motorcycle (0.72)</td>
</tr>
</tbody>
</table>

Semantic analogy examples

\[ w_1 : w_2 = w_3 : ? \quad \Rightarrow \quad V_2 = V_3 - V_1 + V_2 \]

<table>
<thead>
<tr>
<th>Analogy</th>
<th>Top Neighbors</th>
</tr>
</thead>
<tbody>
<tr>
<td>summer : rain = winter : ?</td>
<td>snow (0.79) rainfall (0.73) wet (0.71)</td>
</tr>
<tr>
<td>italy : rome = france : ?</td>
<td>paris (0.78) constantinople (0.74) egypt (0.73)</td>
</tr>
<tr>
<td>man : eye = car : ?</td>
<td>motor (0.64) brake (0.58) overhead (0.58)</td>
</tr>
<tr>
<td>man : woman = king : ?</td>
<td>mary (0.70) prince (0.70) queen (0.68)</td>
</tr>
<tr>
<td>read : book = listen : ?</td>
<td>sequel (0.65) tale (0.63) song (0.60)</td>
</tr>
</tbody>
</table>
Broad impact on key text processing tasks

Semantic similarity modeling is critical in many text processing tasks
Deep Semantic Similarity Model (DSSM)

Compute semantic similarity between two text strings X and Y

- Map X and Y to feature vectors in a latent semantic space via deep neural net
- Compute the cosine similarity between the feature vectors

DSSM for ranking tasks

<table>
<thead>
<tr>
<th>Tasks</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web search</td>
<td>Search query</td>
<td>Web documents</td>
</tr>
<tr>
<td>Recommendation</td>
<td>Doc in reading</td>
<td>Interesting things in doc or other docs</td>
</tr>
<tr>
<td>Machine translation</td>
<td>Sentence in language A</td>
<td>Translations in language B</td>
</tr>
</tbody>
</table>
Learning DSSM on labeled X-Y pairs (clicked Q-D pairs)

- Map query (X) and docs (Y) into the same semantic space via deep neural net
Learning DSSM on labeled X-Y pairs (clicked Q-D pairs)

- Map query (X) and docs (Y) into the same semantic space via deep neural net
- Clicked (relevant) docs are closer to query than non-clicked (irrelevant) docs in that space
DSSM: compute X-Y similarity in semantic space

Learning: maximize the similarity between relevant queries and docs

DSSM combines three pieces of MSR research
- DNN structure follows deep auto-encoder (Hinton and Deng 2009)
- The use of search logs for translation model training (Gao, He, Nie, 2010)
- Parameter optimization uses the pairwise rank loss based on cosine similarity (Yih et al. 2011; Gao et al. 2011)

https://microsoft.sharepoint.com/teams/DSSM_Text_Processing
### Results on Web Search Ranking

<table>
<thead>
<tr>
<th>#</th>
<th>Models</th>
<th>NDCG@1</th>
<th>Improv.</th>
<th>NDCG@3</th>
<th>Improv.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Lexical Matching Models</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>BM25</td>
<td>30.5</td>
<td></td>
<td>32.8</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>ULM [Zhai and Lafferty 2001]</td>
<td>30.4</td>
<td>-0.1</td>
<td>32.7</td>
<td>-0.1</td>
</tr>
<tr>
<td></td>
<td><strong>Topic Models</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>PLSA [Hofmann 1999]</td>
<td>30.5</td>
<td>+0.0</td>
<td>33.5</td>
<td>+0.7</td>
</tr>
<tr>
<td>4</td>
<td>BLTM [Gao et al. 2011]</td>
<td>31.6</td>
<td>+1.0</td>
<td>34.4</td>
<td>+1.6</td>
</tr>
<tr>
<td></td>
<td><strong>Clickthrough-based Translation Models</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>WTM [Gao et al. 2010]</td>
<td>31.5</td>
<td>+1.0</td>
<td>34.2</td>
<td>+1.4</td>
</tr>
<tr>
<td>6</td>
<td>PTM [Gao et al. 2010]</td>
<td>31.9</td>
<td>+1.4</td>
<td>34.7</td>
<td>+1.9</td>
</tr>
<tr>
<td></td>
<td><strong>Deep Semantic Similarity Models</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>DSSM w/o convolutional layer</td>
<td>32.0</td>
<td>+1.5</td>
<td>35.5</td>
<td>+2.7</td>
</tr>
<tr>
<td>8</td>
<td>DSSM</td>
<td>34.2</td>
<td>+3.7</td>
<td>37.4</td>
<td>+4.6</td>
</tr>
</tbody>
</table>

DSSM is the new state-of-the-art
Modeling interestingness with DSSM

- Contextual entity search
  - Given a user-highlighted text span representing an entity of interest
  - Search for supplementary document for the entity

- Automatic highlighting
  - Given a document a user is reading
  - Discover the concepts/entities/topics that interest the user and highlight the corresponding text span

- Document prefetching
  - Given a document a user is reading
  - Prefetching a document that the user will be interested in next
DSSM for contextual entity ranking

- DSSM beats manually crafted text features
- +5 AUC gain over full ranker

<table>
<thead>
<tr>
<th>Ranker</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>BM25 (mention)</td>
<td>60%</td>
</tr>
<tr>
<td>Ranker (2306 features)</td>
<td>72%</td>
</tr>
<tr>
<td>DSSM (1 feature)</td>
<td>72%</td>
</tr>
<tr>
<td>Ranker+ DSSM</td>
<td>77%</td>
</tr>
</tbody>
</table>

- Mention
- Context

KB Entity (reference doc)
Highlighting: Interest Models Performance
NDCG @ Rank (EVAL) using src/tar Content

- **Features**
  - DSM: DSSM
  - WCAT: semantic labels (page categories) assigned by editors
  - JTT: LDA-style topic models
  - NSF: non-semantic features
- **DSSM learned features outperform the thousands of features coming from manually assigned labels (WCAT)**
Results on Machine Translation

- Map the sentences in source/target languages into the same, language-independent semantic space
- The semantic translation model leads up to 1.3 BLEU improvement
DSSM: learning semantic similarity between $X$ and $Y$

<table>
<thead>
<tr>
<th>Tasks</th>
<th>$X$</th>
<th>$Y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web search</td>
<td>Search query</td>
<td>Web documents</td>
</tr>
<tr>
<td>Ad selection</td>
<td>Search query</td>
<td>Ad keywords</td>
</tr>
<tr>
<td>Entity ranking</td>
<td>Mention (highlighted)</td>
<td>Entities</td>
</tr>
<tr>
<td>Recommendation</td>
<td>Doc in reading</td>
<td>Interesting things in doc or other docs</td>
</tr>
<tr>
<td>Machine translation</td>
<td>Sentence in language A</td>
<td>Translations in language B</td>
</tr>
<tr>
<td>Nature User Interface</td>
<td>Command (text/speech)</td>
<td>Action</td>
</tr>
<tr>
<td>Summarization</td>
<td>Document</td>
<td>Summary</td>
</tr>
<tr>
<td>Query rewriting</td>
<td>Query</td>
<td>Rewrite</td>
</tr>
<tr>
<td>Image retrieval</td>
<td>Text string</td>
<td>Images</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Save the planet and return your name badge before you leave (on Tuesday)