Deep Learning Models for Information Retrieval

Speaker : Semin Choi

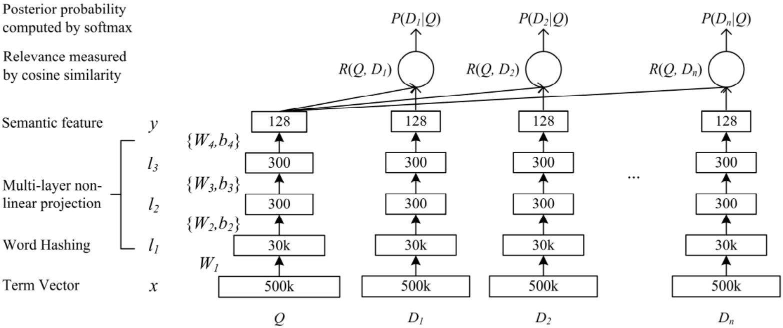
Department of Statistics, Seoul National University, South Korea

July 3, 2017

Outline

Deep Structured Semantic Model(DSSM)

- We strive to develop a series of new latent semantic models with a deep structure that project queries and documents into a common low-dimensional space.



Deep Structured Semantic Model(DSSM)

- Word Hashing
  - good : #go, goo, ood, od#
- The semantic relevance score between a query Q and a document D is measured as:

$$R(Q, D) = \cos(y_Q, y_D) = \frac{y_Q^T y_D}{\|y_Q\| \|y_D\|}.$$

- We compute the posterior probability of a document given a query :

$$P(D|Q) = \frac{\exp(\gamma R(Q, D))}{\sum_{D' \in \mathbf{D}} \exp(\gamma R(Q, D'))}$$

- We approximate $\mathbf{D}$ by including D^+ and four randomly selected unclicked documents, denote by $\{D_j^- : j = 1, ..., 4\}$.
- Learning the DSSM : we minimize the following loss function

$$L(\Lambda) = -\log \prod_{(Q, D^+)} P(D^+|Q)$$

Multi-task DNN for Semantic Classification and Information Retrieval

- Our multi-task model combines classification and ranking tasks:
  - Query Classification
    - In this study, we classify queries into four domains of interest: “Restaurant”, “Hotel”, “Flight”, “Nightlife”.
  - Web search
    - Given a query Q, we estimate $P(D|Q)$ for each document D.

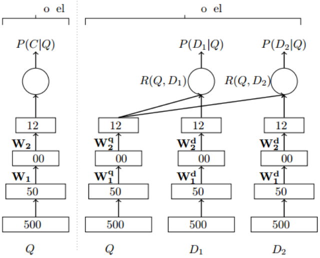
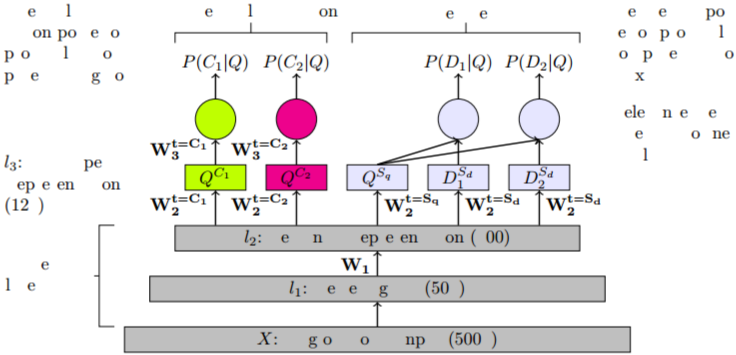


Figure 2: A DNN model for classification and a DSSM model (Huang et al., 2013) for ranking.

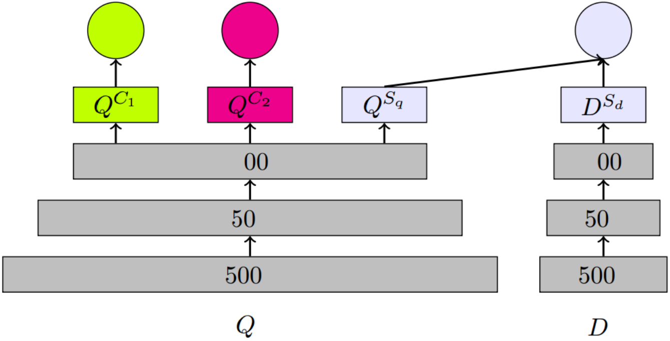
Multi-task DNN for Semantic Classification and Information Retrieval

- The Multi-task DNN for Semantic Classification and Information Retrieval :



Multi-task DNN for Semantic Classification and Information Retrieval

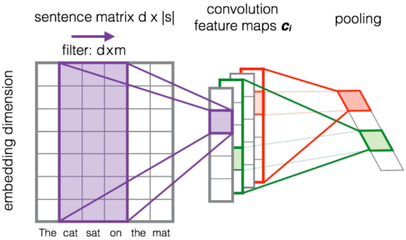
- Alternative Model :



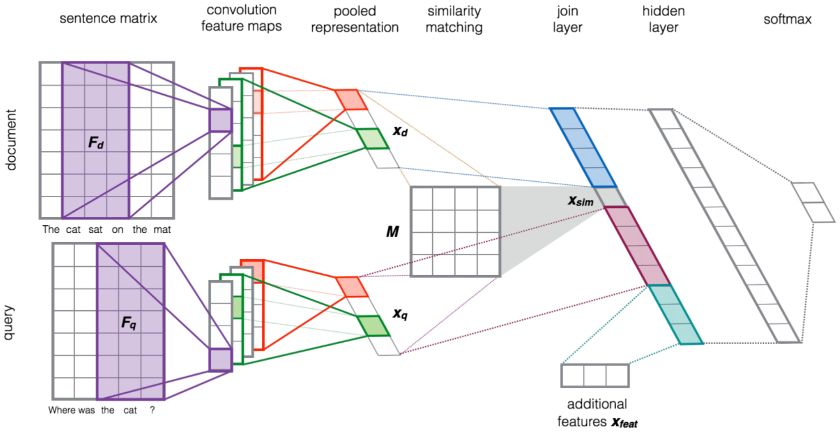
- It achieves good results in query classification at the expense of web search.

CNN for ranking pairs of short texts

- We are given a set of retrieved lists, where each query $q_i \in \mathbf{Q}$ comes together with its list of candidate documents $\mathbf{D}_i = \{d_{i_1}, d_{i_2}, ..., d_{i_n}\}$.
- The candidate set comes with their relevancy judgements $\{y_{i_1}, y_{i_2}, ..., y_{i_n}\}$, where documents that are relevant have labels equal to 1 and 0 otherwise.
- The goal is to build a model that for each query q_i and its candidate list $\mathbf{D}_i$ generates an optimal ranking.



CNN for ranking pairs of short texts



CNN for ranking pairs of short texts

- The similarity between x_q and x_d vectors as follows:

$$sim(x_q, x_d) = x_q^T M x_d$$

  where $M \in \mathbb{R}^{d \times d}$ is a similarity matrix.

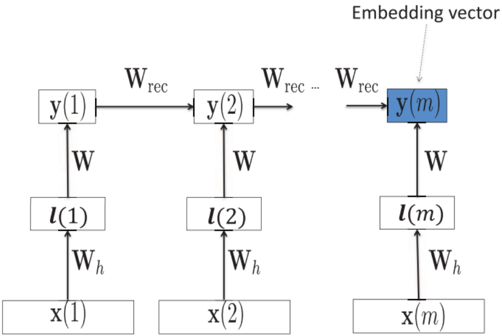
- The model is trained to minimize the cross-entropy function:

$$\begin{aligned} C &= -\log \prod_{i=1}^{N} p(y_i | q_i, d_i) + \lambda \|\theta\|_2^2 \\ &= -\sum_{i=1}^{N} [y_i \log a_i + (1 - y_i) \log(1 - a_i)] + \lambda \|\theta\|_2^2 \end{aligned}$$

  where a is the output from the softmax layer.

Deep Sentence Embedding Using LSTM

- The basic RNN structure



Deep Sentence Embedding Using LSTM

- We use the RNN with LSTM cells.
- We adopt the cosine similarity between the semantic vectors of two sentences as a measure for their similarity :

$$R(Q, D) = \frac{y_Q(T_Q)^T y_D(T_D)}{\|y_Q(T_Q)\| \|y_D(T_D)\|}.$$

  where T_Q and T_D are the lengths of the sentence Q and sentence D, respectively.
- Objective function :

$$L(\Lambda) = \min_{\Lambda} \left\{ -\log \prod_{r=1}^{N} P(D_r^+ | Q_r) \right\}$$

  where $P(D_r^+ | Q_r) = \frac{\exp(\gamma R(Q_r, D_r^+))}{\sum_{D' \in \mathbf{D}_r} \exp(\gamma R(Q_r, D'))}$ and $\mathbf{D}_r = \{D_r^+, D_{r,1}^-, ..., D_{r,n}^-\}$.

Deep Learning Based Query Expansion

- Each term t is represented by a vector of predefined dimension v_t.
  - e.g. Word2vec
- The similarity between two terms t_1 and t_2 is measured with the normalized cosine between their two vectors v_{t_1} and v_{t_2}.

$$SIM(t_1, t_2) = \widetilde{\cos}(v_{t_1}, v_{t_2})$$

- Based on this normalized cosine similarity between terms, we define the function that returns the k-most similar terms to a term t, $top_k(t)$.
- Building Expanded Query
  - The expanded query q' is defined as follows: $q' = q \cup_{t \in q} top_k(t)$.
  - The frequency of each $t \in q$ still the same in the expanded query q'.
  - The frequency of each expansion term $t' \in top_k(t)$ in the expanded query q' is given as follows :

$$\#(t', q') = \alpha \times \#(t, q') \times \widetilde{\cos}(v_t, v_{t'})$$

where $\alpha \in [0, 1]$ is a tuning parameter that determines the importance of expansion terms.

Reference

- Huang, Po-Sen, et al. "Learning deep structured semantic models for web search using clickthrough data." Proceedings of the 22nd ACM international conference on Conference on information & knowledge management. ACM, 2013.
- Liu, Xiaodong, et al. "Representation Learning Using Multi-Task Deep Neural Networks for Semantic Classification and Information Retrieval." HLT-NAACL. 2015.
- Severyn, Aliaksei, and Alessandro Moschitti. "Learning to rank short text pairs with convolutional deep neural networks." Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 2015.
- Palangi, Hamid, et al. "Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval." IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP) 24.4 (2016): 694-707.
- ALMasri, Mohannad, Catherine Berrut, and Jean-Pierre Chevallet. "A comparison of deep learning based query expansion with pseudo-relevance feedback and mutual information." European Conference on Information Retrieval. Springer International Publishing, 2016.