IRIT @ TREC 2016 Clinical Decision Support Track

Gia-Hung Nguyen, Laure Soulier, Lynda Tamine, Nathalie Bricon-Souf

IRIT @ TREC 2016 Clinical Decision Support Track

Gia-Hung Nguyen*, Laure Soulier**, Lynda Tamine*, and Nathalie Bricon-Souf*

* IRIT, Université de Toulouse, CNRS, UPS,
France, 118 Route Narbonne, Toulouse, France
{gia-hung.nguyen,tamine,nathalie.souf}@irit.fr
** Sorbonne Universités, UPMC Univ Paris 06, CNRS, LIP6 UMR 7606,
4 place Jussieu 75005 Paris
laure.soulier@lip6.frhttp://www.irit.fr

Abstract. In this document, we describe our participation of the IRIT lab to the TREC 2016 Clinical Decision Support track. The goal of the Clinical Decision Support track is to develop the efficient systems to retrieve relevant biomedical articles given a form of patient medical record. To address this task, we propose a neural approach to match the document and the query, with the help of the MeSH thesaurus.

Keywords: Medical Information Retrieval, Neural Network, Evaluation

1 Introduction

The TREC Clinical Decision Support track is designed to encourage the formalization of retrieval models in order to anticipate needs of physicians by connecting medical cases with relevant information. The objective of the task is to retrieve relevant biomedical articles to answer topics that are generic clinical questions related to patient medical records. In order to simulate the actual information needs of physicians, each topic is annotated with one of three following categories: Diagnosis, Test, and Treatment. Participants are required to retrieve the biomedical articles which are useful for answering each topic question.

To address this challenge, we propose an end-to-end neural approach that learns the similarity of documents and queries using a raw text-based representation enhanced by a concept extraction from MeSH thesaurus. We describe in the next section our neural network model.

2 Our approach

In order to match document query pairs, we rely on the neural network architecture presented in [1]. While a simple raw text-based matching could lead to a semantic gap, we propose to consider information provided by an external resource in order to add conceptual semantics in our vector representation. Indeed, our intuition here is that the document-query matching could be enhanced by

exploiting the conceptual relations learned from an external resource through a hybrid representation of the distributional semantic (namely, plain text representation) and the symbolic semantic (namely, concept description).

Specifically, given a document or a query, our neural network aims to map the initial enhanced representation of the document/query into a low-dimensional feature in a semantic space. Then those latent semantic features are used to measure the relevance score between a query and a document. The architecture of the network is described in the following.

Input. We call the input layer x_{input} as the enhanced representation of a document or a query, which can be decomposed into two parts: the first part represents plain text in the document/query and the second part consists of the description of the concepts existing in the text:

- *Plain text representation.* This first part of input represents the textual content of the document or the query. We directly apply the *ParagraphVector* [2] model to learn the representation for each piece of text (document or query).
- *Description representation.* We propose to add a layer to capture expressions of document/query via the external concepts. To form this vector, we first extract the MeSH concepts existing in the document/query by using *Cxtractor*[1] relying on *MaxMatcher* [3]. Then, the description of extracted concepts in each document/query is gathered to build a conceptual-document description vector. Similarly to the plain text representation, we apply the *ParagraphVector* algorithm.

Hidden layers. For each network branch, the input vector is projected into a latent space by the L hidden layers l_i $(i = 1, \cdots, L)$ so as to obtain a latent semantic vector y. Each hidden layer l_i and the latent semantic vector y are respectively obtained by the following non-linear transformations:

$$\begin{gathered} l_0 = x_{input} \\ l_i = f(W_{i-1} \cdot l_{i-1} + b_{i-1}) \qquad i = 1, ..., L \\ y = f(W_L \cdot l_L + b_L) \end{gathered} \tag{1}$$

where W_i and b_i are respectively the weight matrix and bias term at the i^{th} layer. To perform the non-linear transformation, we use the ReLU activation function: $f(x) = max(0, x)$.

Similarity function. After obtaining the latent semantic vectors y_D and y_Q of document D and query Q through the non-linear transformation of hidden layers, the document-query cosine similarity score $sim(D|Q)$ is estimated between vectors y_D and y_Q. Following [1], the output of our model is calculated as a posterior probability of a document given a query, through a softmax function:

$$P(D|Q) = \frac{exp(sim(Q, D))}{\sum_{D' \in C} exp(sim(Q, D'))} \tag{2}$$

[1] https://sourceforge.net/projects/cxtractor/

where C is the set of candidate documents to be ranked for each query Q, approximated by including a relevant document D^+ and four random irrelevant documents D^-. The network is trained to minimize the cross-entropy loss function on the relevant pairs:

$$L = -\log \prod_{Q,D^+} P(D^+|Q). \quad (3)$$

We follow the configurations used in [1] to implement our network: two hidden layers of size 300 leading to an output layer of size 128. The model is trained using the stochastic gradient descent (SGD) regularized by a dropout layer before the output layer. The dropout value is set up to 0.4.

The training data is collected from previous TREC CDS tracks. We take the 60 topics and their relevant assessment provided in the 2014 and 2015 tracks to construct the set of relevant and irrelevant pairs for training and evaluating the neural network. To perform the retrieval result for this year track, we apply our trained model on the topics of this year, to obtain the ranked list of documents. For efficient computation reason, we perform a re-ranking technique over the preselected candidate documents. An initial candidate list is performed using the BM25 model to obtain the top 3,000 documents. Then we use our neural model to re-rank these candidates and retain the top 1,000 documents for submitting runs.

3 Runs and Results

We present in this section the results of our two runs submitted to the CDS track. *d2vDescIrit* represents the result obtained with our neural network scoring function, using topic description as query text. *d2vCombIrit*, also using the topic description as query text, is the result of a combination scoring between BM25 score and our neural score (where $\alpha = 0.4$):

$$score(Q, D) = \alpha \times score_{BM25}(Q, D) + (1 - \alpha) \times score_{neural}(Q, D). \quad (4)$$

Table 1. Results averaged over the 30 topics of our runs.

	Rprec	infAP	infNDCG	P@10
d2vDescIrit	0.0206	0.0017	0.0442	0.0433
d2vCombIrit	0.0215	0.0018	0.0463	0.0533
best	0.1860	0.0397	0.2751	0.4767
median	0.0648	0.0065	0.1043	0.1533
worst	0.0019	0.0004	0.0148	0.0233

Table 1 shows results of our runs, poor results were obtained. One possible explanation may be that our model scenarios are not optimal since we did not perform the parameter tuning. Moreover, the small amount of document-query pairs seems to lead to overlearning, avoiding the model to be generalized to any dataset. Since this work is a preliminary model, additional experiments should be conducted to analyze the model parameters, peculiarities, and performance.

References

[1] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. Learning deep structured semantic models for web search using clickthrough data. In *CIKM*, 2013.
[2] Quoc V Le and Tomas Mikolov. Distributed representations of sentences and documents. In *ICML*, 2014.
[3] Xiaohua Zhou, Xiaodan Zhang, and Xiaohua Hu. Maxmatcher: Biological concept extraction using approximate dictionary lookup. In *PRICAI*, 2006.