A Study on Query Expansion with MeSH Terms and Elasticsearch. IMS Unipd at CLEF eHealth Task 3

Giorgio Maria Di Nunzio and Alexandru Moldovan

Dept. of Information Engineering – University of Padua
giorgiomaria.dinunzio@unipd.it,alexandru.moldovan@studenti.unipd.it

Abstract. In this paper, we describe the first participation of the Information Management Systems (IMS) group at CLEF eHealth 2018 Task 3, Consumer Health Search Task. In particular, we participated in the subtask IRTask 1: Ad-hoc Search which is a standard ad-hoc search task, aiming at retrieving information relevant to people seeking health advice on the web.

The goal of our work is to evaluate 1) different query expansion strategies based on the recognition of Medical Subject Headings (MeSH) terms present in the original query; 2) different approaches to combine multiple ranking lists given the query expansions. We used Elasticsearch as search engine and the indexes provided by the organizers of this task.

1 Introduction

In this paper, we report the experimental results of our first participation to the CLEF eHealth Lab Task 3 [4, 3]: "Consumer Health Search". This task investigates the problem of retrieving documents to support information needs of health consumers that are confronted with a health problem.

This work is part of a Master Degree thesis in Computer Engineering where the main goal is to test the effectiveness of some variants of query expansion approaches based on the recognition of MeSH terms present in the original query.

The contribution of our experiments to this task can be summarized as follows:

- A study of several query expansion approaches that takes into account the relationships between MeSH terms [6];
- An evaluation of different document scoring strategies given the multiple ranking list produced by the query expansions [1, 6].

The remainder of the paper will introduce the methodology and a brief summary of the experimental settings that we used in order to create the runs that we submitted for the task.

2 Methodology

In this section, we describe the query expansion approaches [6] as well as the document scoring strategies [1] that we used to create the expanded queries and the ranked lists.

2.1 Query expansion approaches

For the experiments of the query expansion approach, we used the English version of the information need. Before running the automatic expansion algorithms, we performed a manual check in order to correct spelling errors. For example, for topic 189001, the term *gonorrhea* is misspelt as *gonhrrea*.

Identification of MeSH terms After the manual cleaning process, we use the MeshOnDemand[1] API to identify the MeSH terms present in the query.

For example, for topic 188001 "caffeine high blood pressure" we obtain the following MeSH terms

- Caffeine[2]
- Hypertension[3]

plus one additional term

- Blood pressure[4]

Finding related MeSH terms For each MeSH term found in the previous step, we use the MeSHRDF[5] database to look for semantically related (MeSH) terms. See for example Figure 1 that shows a part of the structure of the tree of relations among concepts related to the MeSH term Caffeine.

We choose a subset of all the possible relations (predicates) between terms in the MeSHRDF database[6], and we use this subset of predicates for query expansion in the following way:

- Baseline: the original query is used without any additional term.
- Simple Expansion (SE): given a MeSH term identified in the first step, all the MesH entries related to that term are kept, except for the predicates 'meshv:Qualifier', 'meshv:seeAlso', 'meshv:broader' e 'meshv:broaderDescriptor'. Then we re-apply just once (not recursively) a SE for each 'child' node.
- SE broader: like SE, in addition, only for the original MeSH terms (those that appear in the query) we expand the MesH terms by adding the predicates 'meshv:broader' e 'meshv:broaderDescriptor'.

[1] `https://meshb.nlm.nih.gov/MeSHonDemand`
[2] `https://meshb.nlm.nih.gov/record/ui?name=Caffeine`
[3] `https://meshb.nlm.nih.gov/record/ui?name=Hypertension`
[4] `https://meshb.nlm.nih.gov/record/ui?name=Blood&#37;Pressure`
[5] `https://id.nlm.nih.gov/mesh/`
[6] `https://hhs.github.io/meshrdf/predicates`

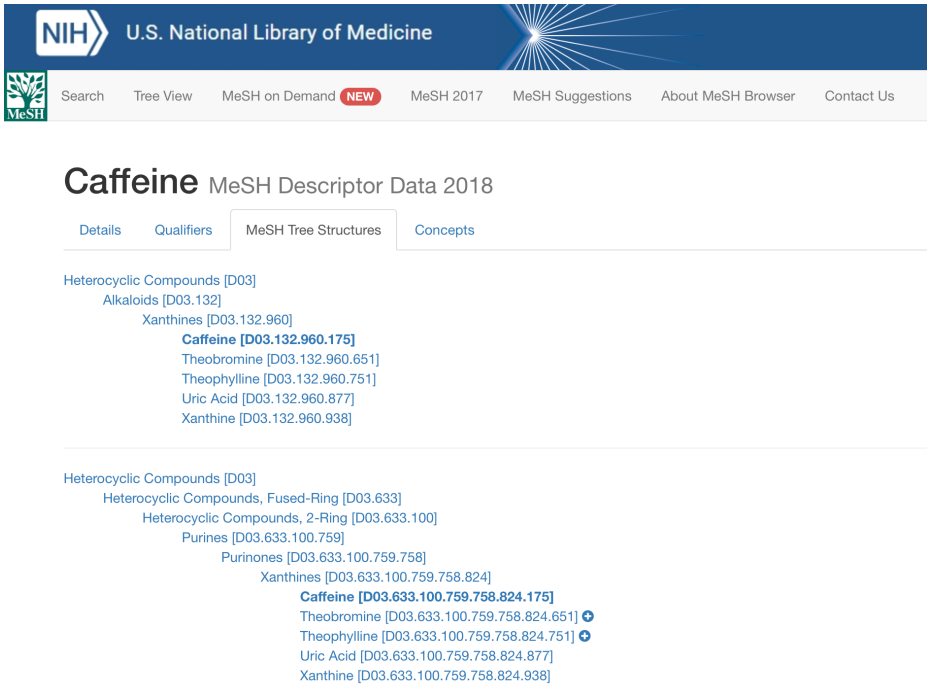

Fig. 1. Partial view of the tree structure of the MeSH term Caffeine.

- SE also: like SE broader, instead of adding the predicates 'broader', we add the MesH terms related with the predicate 'meshv:seeAlso'.
- SE broader also: a combination of SE broader and SE also.
- SE child broader: first, apply SE, then for each 'child' node search for 'parents' different from the original MeSH terms.
- SE recursive down: like SE, for each 'child' we recursively apply SE until leaf nodes are found (no more recursions).
- All in one: all the approaches at once.

At the end of a query expansion process, we have three main data objects:

- the original query q;
- a vector $m = (m_1, m_2, \ldots, m_n)$ of MeSH terms associated with the original query;
- a list t of expanded terms of n elements where each element t_i is another vector of terms resulting from the iteration of the expansion approach, $t_i = (t_{i1}, t_{i2}, \ldots, t_{ij})$.

Given these three objects, we create a set of expanded queries that will be used to create a number of ranked lists, as explained in the following subsections.

Building expanded query Give the vector of MesH terms m and the list of expanded terms t, we create a set of expanded queries by means of the following procedure:

- For each term term m_i of the vector of MeSH terms associated with the original query,
- we substitute m_i with one of the terms in t_i, for example t_{i1},
- then, we build the expanded query by joining the original query with the new vector $m_i^* = (m_1, \ldots, t_{i1}, \ldots, m_n)$, $q^* = q \bigcup m_i^*$.

Therefore, at the end of the process we generate a set V of vectors of expanded queries where the cardinality $|V|$ is the sum of all the elements in the vectors of the list t. For each vector of $v_k \in V$ we obtain a list l_k of ranked documents.

Merging ranked lists We use different approaches to merge the $|V|$ ranked list into a single list based on the combination of scores of the documents.

- Average: given a document present in one or more lists, the scores associated to the document are averaged. Then the documents are ordered in decreasing order on the basis of this new score.
- Sum: given a document present in one or more lists, we sum the scores associated to the documents.
- Normalized sum: like Sum but the sum of the scores are normalized by the highest score in the ranked lists.
- Round robin: for each rank r, we take the document of each list l_k at r and add it to the new ranked list if it has not already been seen.

3 Experiments

For the experiments, we used the Elasticsearch search engine[7] and the indexes provided by the organizers of the task. We used the BM25 ranking function with default parameters.

Given the constraints on the number of runs, four in total, that could be submitted to the task, we submitted one baseline run and three query expansion variant with the same scoring approach. We will evaluate all the other combinations as soon as the qrels will be made available.

In particular, we submitted the following runs that use the Sum (of the scores) as the document scoring approach:

- baseline.exp, a baseline run (plain BM25 with no expansion),
- sum_score_simple.exp, simple expansion,
- sum_score_broader_also, simple expansion plus the two predicates broader and also,
- sum_score_recursive.exp, a recursive down approach.

In Table **??**, we show the number of query variants that are generated per topic by three approaches.

[7] `https://www.elastic.co/products/elasticsearch`

4 Final remarks and Future Work

The aim of our first participation to the CLEF eHealth Task 3 was to test the effectiveness of different query expansion approaches that use the MeSH term RDF graph as well as different merging approaches of the ranking lists produced by the query expansion approach. As future work, we will study the combination of the MesH term expansion with the help of medical terminological records [5] for technologically assisted systematic reviews [2].

References

1. Nick J. Belkin, Paul Kantor, Edward A. Fox, and Joseph A. Shaw. Combining the evidence of multiple query representations for information retrieval. *Information Processing & Management*, 31(3):431 – 448, 1995. The Second Text Retrieval Conference (TREC-2).
2. Giorgio Maria Di Nunzio. A study of an automatic stopping strategy for technologically assisted medical reviews. In *Advances in Information Retrieval - 40th European Conference on IR Research, ECIR 2018, Grenoble, France, March 26-29, 2018, Proceedings*, pages 672–677, 2018.
3. Jimmy, Guido Zuccon, Joao Palotti, Lorraine Goeuriot, and Liadh. Kelly, editors. *Overview of the CLEF 2018 Consumer Health Search Task. CLEF 2018 Evaluation Labs and Workshop: Online Working Notes*. CEUR-WS, September 2018.
4. Hanna Suominen, Liadh Kelly, Lorraine Goeuriot, Evangelos Kanoulas, Leif Azzopardi, Rene Spijker, Dan Li, Aurélie Névéol, Lionel Ramadier, Aude Robert, Joao Palotti, Jimmy, and Guido Zuccon, editors. *Overview of the CLEF eHealth Evaluation Lab 2018. CLEF 2018 - 8th Conference and Labs of the Evaluation Forum*, volume Lecture Notes in Computer Science (LNCS). Springer, September 2018.
5. Federica Vezzani, Giorgio Maria Di Nunzio, and Geneviève Henrot. Trimed: A multilingual terminological database. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018.*, 2018.
6. Theodore B Wright, David Ball, and William Hersh. Query expansion using mesh terms for dataset retrieval: Ohsu at the biocaddie 2016 dataset retrieval challenge. *Database*, 2017:bax065, 2017.

Table 1. Number of query variants per approach

topic	simple	broader_also	recursive
151001	290	412	524
152001	310	330	1402
153001	71	117	127
154001	90	128	139
155001	74	123	184
156001	281	406	547
157001	222	226	671
158001	85	152	85
159001	298	404	1260
160001	84	191	119
161001	104	134	129
162001	42	118	42
163001	168	286	216
164001	38	60	231
165001	181	296	439
166001	253	323	436
167001	120	126	312
168001	241	339	241
169001	16	80	16
170001	144	169	169
171001	22	61	22
172001	96	135	96
173001	15	20	15
174001	112	186	166
175001	82	124	102
176001	42	101	42
177001	76	169	92
178001	236	336	388
179001	100	179	146
180001	83	105	89
181001	28	75	28
182001	257	279	423
183001	85	157	116
184001	117	166	117
185001	96	175	96
186001	245	308	562
187001	85	204	169
188001	125	164	239
189001	18	38	18
190001	377	428	678
191001	6	19	6
192001	169	245	270
193001	53	88	104
194001	95	147	95
195001	47	62	47
196001	13	104	13
197001	109	201	185
198001	51	78	51
199001	131	209	131
200001	129	185	202
average	124.24	183.36	239.94