Information Retrieval

2021-2022

Marco Piazza 829588

Elisa Cazzaniga 829914

CORD-19

Introduction

Given a collection of documents, that are scholarly articles about coronavirus, and given a query, created by people with not no expertise in the medical domain, the objective of this project is to provide to the user a ranked list of the documents that are considered relevant for the specified query. The relevance assessment of a document with respect to the query is divided into three categories: not relevant, partially relevant and relevant.

The first part of this work is aimed to analyse the documents and the queries given. Since the documents in the repository may contain several kinds of coronavirus, some analysis has been done to compare the whole collection with documents that refer to covid19.

The second part of this work is aimed to analyze the performance of *pyterrier* using different types of index configurations, pre-processing steps and ranking models.

The third part of this project is aimed to try to improve the effectiveness of the approaches presented. Different types of analysis of queries expansion and queries reduction have been conducted.

Test Collection - Analysis of Queries and Documents

The first part of this project is aimed to analyse the documents and the queries given.

Some statistics about the documents are reported in the following table.

	Number of documents	Number of tokens*	Number of unique tokens*	Average number of tokens per document*	Average number of unique tokens per document*
All documents	192509	17083773	230904	88.74	62.63
COVID19 documents	83692	5800012	80969	69.3	50.75

*The number of tokens is calculated after the preprocessing step.

The preprocessing steps are finalized to prepare the content of a document in such a way that it can be easily managed and represented. The preprocessing steps that have been applied both on documents, on titles and on queries are the following:

- The text has been converted in **lowercase**.
- The - character has been substituted with the space character. By doing this operation at this time, with the following preprocess steps, we avoid maintaining numbers that are not relevant for our task and we avoid joining terms that initially are separated by - character.
- The **punctuation** has been **removed**.
- The **stopwords** have been **removed**.
- The **covid preprocessing** has been applied. A list of the synonyms and of the different writing ways of the covid19 concept has been created (`'covid 19'`,`'coronavirus disease 19'`, `'sars cov 2'`, ...]). These sequences of words have been searched in the texts and the space characters have been removed. Ex: `'sars cov 2'`--> `'sarscov2'`, `'covid 19'`--> `'covid19'`. In this way there are fewer variants of covid19 concept to handle.
- The **numbers** following the space character have been **removed**. By doing this operation at this time, we avoid removing the numbers that are important in order to identify the documents related to covid19.

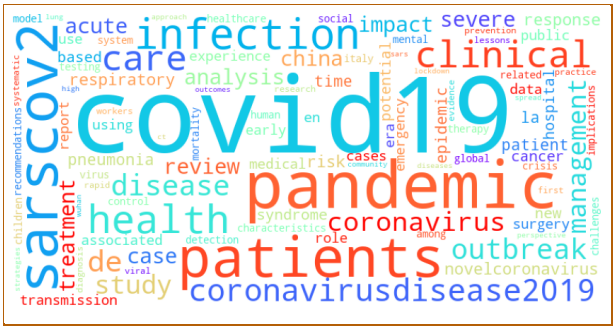
Example:

Meta-analysis investigating the relationship between clinical features, outcomes, and severity of severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) pneumonia

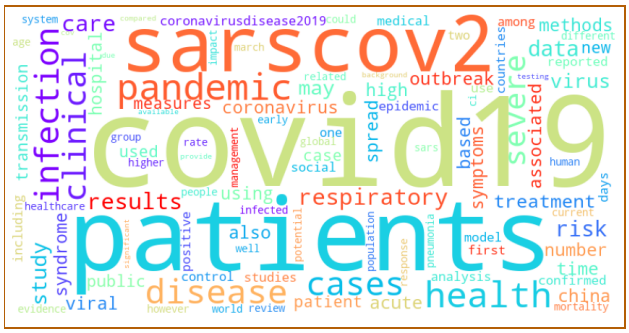
meta analysis investigating relationship clinical features outcomes severity severe acute respiratory syndrome coronavirus2 sarscov2 pneumonia

The documents that contain, in the abstract or in the title, at least one of the terms of the covid19 list are considered related to covid19.

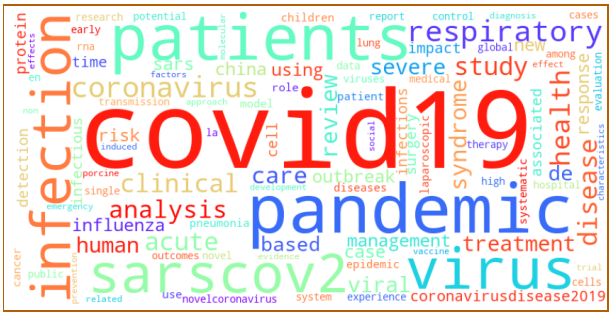
Some word-clouds, using the most common words in the abstracts and in the titles, both for coivd19 documents and for all documents, are created.



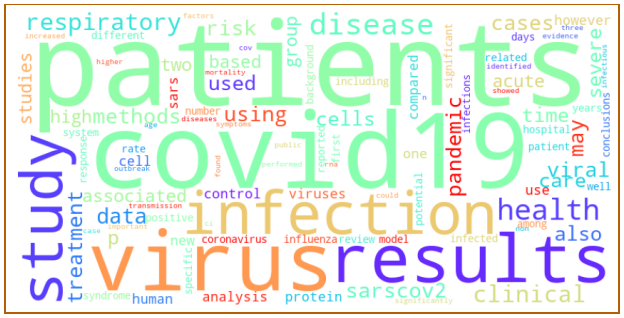
Most common words titles of covid19 documents



Most common words texts of covid19 documents



Most common words titles of general documents



Most common words texts of general documents

Most 10 common words in covid19 titles		
	Word	Frequency
0	covid19	65713
1	pandemic	14930
2	sarscov2	11573
3	patients	10659
4	infection	5586
5	health	4879
6	clinical	4384
7	care	4228
8	coronavirusdisease2019	4227
9	de	3901

Most 10 common words in covid19 abstracts		
	Word	Frequency
0	covid19	131314
1	patients	77563
2	sarscov2	44158
3	pandemic	36098
4	health	32636
5	disease	29300
6	infection	28271
7	cases	27768
8	clinical	24913
9	severe	23810

Most 10 common words in general titles		
	Word	Frequency
0	covid19	65713
1	pandemic	17163
2	patients	15653
3	virus	12361
4	sarscov2	11573
5	infection	11195
6	respiratory	11163
7	coronavirus	10843
8	health	8981
9	disease	8307

Most 10 common words in general abstracts		
	Word	Frequency
0	patients	162289
1	covid19	131314
2	virus	73122
3	results	68385
4	infection	68282
5	study	65769
6	disease	63260
7	health	60429
8	respiratory	56560
9	clinical	54019

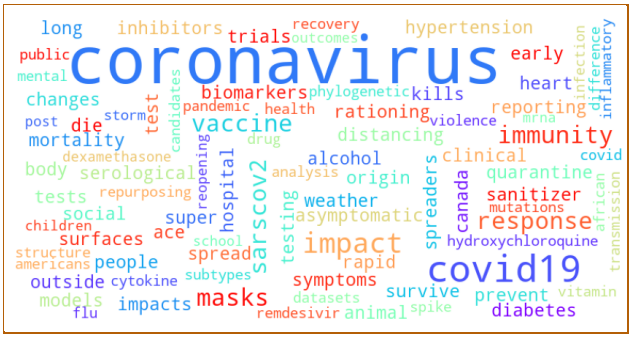
As shown in the word clouds and in the tables, the terms relating to covid19 also appear among the most common words of the documents of the original collection. This means that covid19 remains one of the central topics even considering documents that refer to coronavirus in general. Considering all the documents in the collection, we can notice that the word 'virus' is one of the most used. This word, on the other hand, is not present in the most frequent words in covid19 documents, in which there are more specific words relating to the particular type of virus considered.

Several other words like 'pandemic', 'patients', 'health', 'infection', ..., are common in the tables.

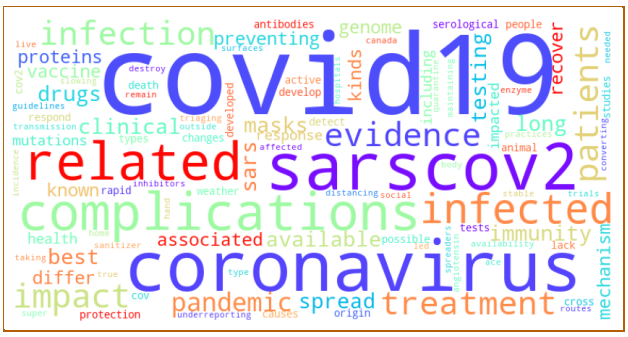
The number of the provided queries is 50. Each query is expressed in three different formats:

- **Ad-hoc**: a short keyword query.
- **Description**: a more precise natural language question.
- **Narrative**: longer description that further elaborates on the question.

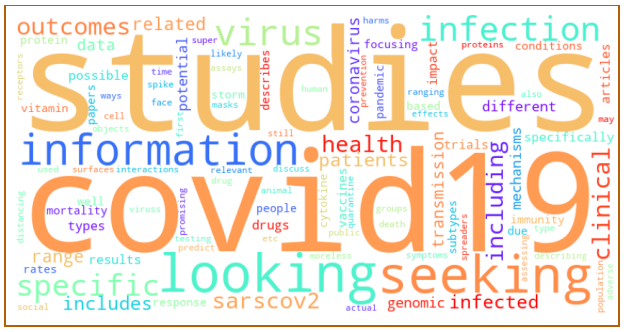
The word clouds of the most common words for the three different formats of queries are created.



Most common words ad-hoc queries



Most common words desc queries



Most common words nar queries

Most 10 common words adhoc queries		
	Word	Frequency
0	coronavirus	40
1	covid19	6
2	impact	3
3	response	2
4	immunity	2
5	masks	2
6	vaccine	2
7	sarscov2	2
8	origin	1
9	weather	1

Most 10 common words desc queries		
	Word	Frequency
0	covid19	31
1	coronavirus	8
2	sarscov2	8
3	related	4
4	complications	4
5	infected	3
6	impact	3
7	patients	3
8	evidence	3
9	infection	3

Most 10 common words nar queries		
	Word	Frequency
0	studies	30
1	covid19	29
2	looking	20
3	seeking	19
4	information	17
5	virus	12
6	infection	10
7	specific	10
8	sarscov2	9
9	outcomes	9

In ad-hoc queries there are few terms with a higher frequency and all the others appear a few times. This aspect is less present in desc queries and especially in nar queries, in which verbs appear among the most frequent words.

Two approaches have been tried to calculate the similarity between queries: in the first, tfidf vectorizer is used to represent the queries; in the second each query is represented using a doc2vec based approach. The models are built using documents collection to simulate a real situation in which the queries are not initially available and their representation is obtained using the model built from the documents.

Search Engine - Basic Search

The second part of the project is aimed to analyze the performance of *pyterrier* with different possible settings ingredients for building the pipeline. Different approaches have been tested and are presented in this paragraph within the comparison of the results for each model. In the end, the pipeline that performs best on this task is shown. Some preprocessing steps have been applied for all the experiments:

- Transformation to lower-case for all term of the queries/documents
- Punctuation removal

Some others have been tested to investigate which permits to improve the effectiveness of the system:

- Stemmer
- Stopwords removal

Finally, it has been decide to index all the elements of the document using different combinations of them:

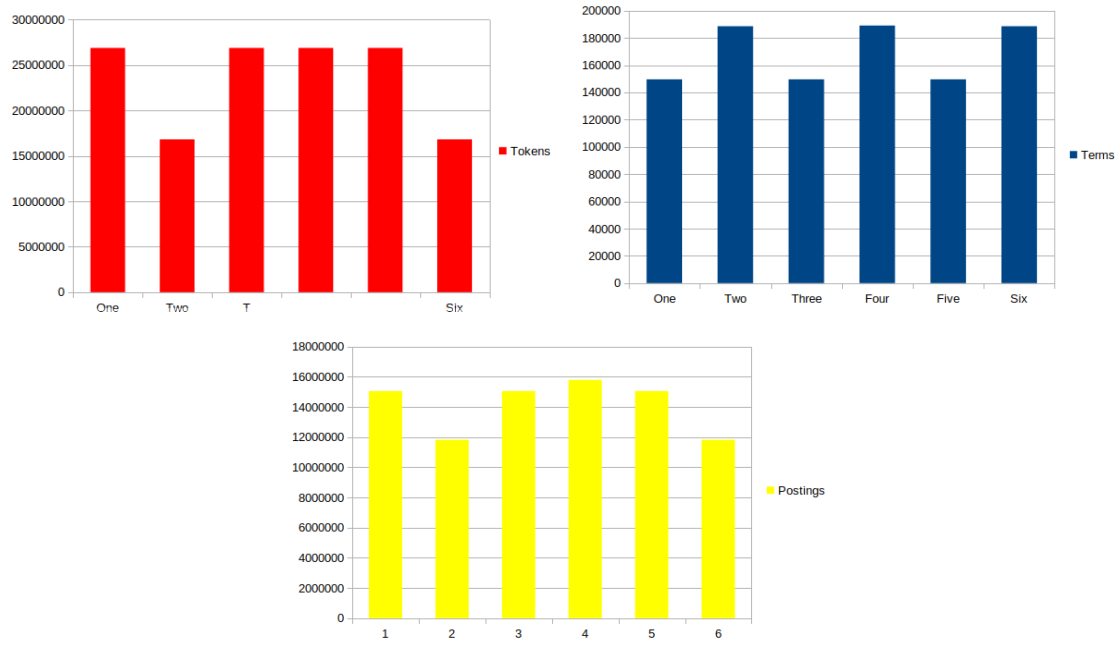
- Title
- Date
- Abstract

Some statistics have been collected from the indexing of documents and are presented, divided by the preprocessing steps applied before:

- One → stopwords removal and Porter stemmer for both documents and queries.
- Two → Stopwords removal for both categories.
- Three → Porter stemmer for both categories.
- Four → Neither stopwords removal nor stemmer.
- Five → Stopwords removal on queries and Porter Stemmer on documents.

- Six → Stopwords removal only on documents.

	Uno	Due	Tre	Quattro	Cinque	Sei
Documents	192509	192509	192509	192509	192509	192509
Terms	149557	188603	1499557	189070	149557	188603
Postings	15053000	11824971	15053000	15791700	15053000	11824971
Tokens	26884365	16819835	26884365	26884365	26884365	16819835



Comparison of the different values with different approaches for index creating.

It's possible to notice that the difference statistics are quite constant. The use of stemmer reduce the number of terms.

Then it's possible to evaluate the effectiveness of the different approaches presented above with four different models:

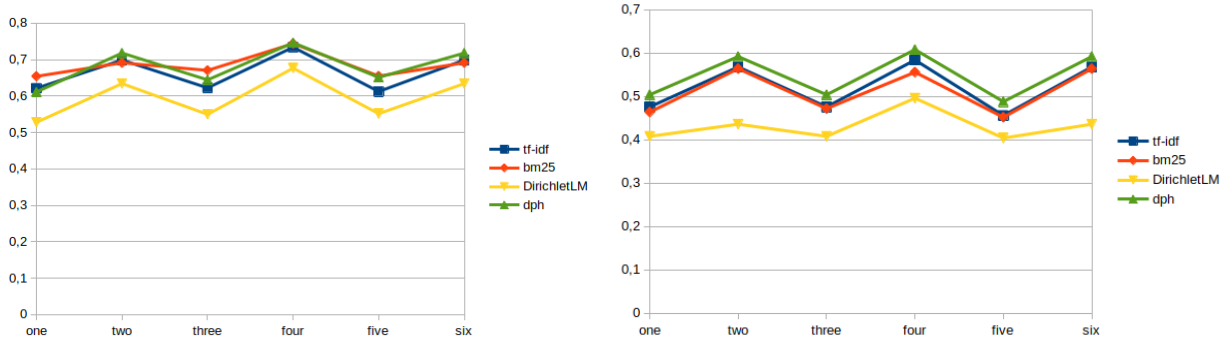
- tf-idf
- bm25
- DirichletLM
- Dph

In the table there are the results for the six different pre-processing approaches and the four different models.

	name	P@5	P@10	ndcg	RR
one	tf-idf	0,476	0,462	0,123527768	0,621230249
	bm25	0,464	0,456	0,123804214	0,653928429
	DirichletLM	0,408	0,396	0,111634835	0,527763288
	DPH	0,504	0,5	0,125042341	0,610859607
two	tf-idf	0,568	0,548	0,134185237	0,698583581
	bm25	0,564	0,536	0,13404099	0,690903122
	DirichletLM	0,436	0,464	0,121337926	0,633897235
	DPH	0,592	0,574	0,137243883	0,717690476
three	tf-idf	0,476	0,474	0,117591745	0,622155784
	bm25	0,472	0,452	0,117384657	0,670212454
	DirichletLM	0,408	0,394	0,106555983	0,549421239
	DPH	0,504	0,486	0,118802722	0,644094643
four	tf-idf	0,584	0,566	0,136130674	0,733696408
	bm25	0,556	0,562	0,134763775	0,743732766
	DirichletLM	0,496	0,482	0,126247966	0,676397375
	DPH	0,608	0,576	0,137425081	0,746154264
five	tf-idf	0,456	0,454	0,123242479	0,612157286
	bm25	0,452	0,434	0,12281494	0,654709707
	DirichletLM	0,404	0,396	0,112799233	0,551265272
	DPH	0,488	0,484	0,12572286	0,650900199
six	tf-idf	0,568	0,548	0,134185237	0,698583581
	bm25	0,564	0,536	0,13404099	0,690903122
	DirichletLM	0,436	0,464	0,121337926	0,633897235
	DPH	0,592	0,574	0,137243883	0,717690476

As it is possible to see, the fourth pre-processing approach (neither stopword nor stemmer) obtains the better results. We have observed how by using the stemmer the performances decrease (as can be seen from 5th approach). From the point of view of models it can be observed that DPH performs better, while DirichletLM is the worst choice.

These conclusions can be confirmed by observing the following graphs (in which precision@5 and reciprocal rank are reported):



The results reported are those obtained on *adhoc* queries, but the conclusions may be the

same for the three types of queries (the complete results are reported in the appendix).

Similar query - similar results

For analyzing the results of similar queries an approach based on clustering algorithm is proposed. Firstly the queries are transformed in vector-form via Doc2Vec, then k-means algorithm is applied on the vector and the results are 7 clusters (by optimized silhouette measure):

- Cluster 0 → 2 elements
- Cluster 1 → 9 elements
- Cluster 2 → 1 element
- Cluster 3 → 1 element
- Cluster 4 → 14 elements
- Cluster 5 → 22 elements
- Cluster 6 → 1 element

For the evaluation of the performance the improper clusters are removed (the ones with only one element). The evaluation includes the mean and the variance of the different effectiveness measures.

cluster			P@5	P@10	ncdg	RR
0	mean	tfidf	0,2	0,3	0,316639257	0,35
		bm25	0,3	0,3	0,323386221	0,375
		DirichletLM	0,4	0,25	0,208091533	0,291666667
		DPH	0,3	0,25	0,290444782	0,416666667
	variance	tfidf	0	0	0,016005787	0,013888889
		bm25	0,02	0	0,016952784	0,03125
		DirichletLM	0	0,005	0,008524917	0,003472222
		DPH	0,02	0,005	0,010266683	0
1	mean	tfidf	0,447966884	0,678372032	0,271418406	0,666666667
		bm25	0,511111111	0,477777778	0,271937687	0,666666667
		DirichletLM	0,244444444	0,3	0,237337551	0,340965208
		DPH	0,577777778	0,533333333	0,278931671	0,740740741
	variance	tfidf	0,161111111	0,116713655	0,271418406	0,157986111
		bm25	0,161111111	0,091944444	0,021700462	0,157986111
		DirichletLM	0,117777778	0,0925	0,01206764	0,145884058
		DPH	0,104444444	0,09	0,022989306	0,104938272
4	mean	tfidf	0,485714286	0,421428571	0,296916777	0,602324263
		bm25	0,471428571	0,421428571	0,299549704	0,512368584
		DirichletLM	0,428571429	0,478571429	0,279954759	0,56207483
		DPH	0,528571429	0,492857143	0,292290677	0,602324263
	variance	tfidf	0,176703297	0,132582418	0,01907739	0,145362566
		bm25	0,176043956	0,135659341	0,019356061	0,161382758
		DirichletLM	0,165274725	0,114120879	0,018443235	0,165580381
		DPH	0,145274725	0,108406593	0,017283004	0,145362566
5	mean	tfidf	0,381818182	0,445454545	0,279059565	0,51758658
		bm25	0,381818182	0,445454545	0,279485333	0,509294145
		DirichletLM	0,345454545	0,327272727	0,249041667	0,486572094
		DPH	0,445454545	0,427272727	0,273838826	0,519978191
	variance	tfidf	0,11012987	0,116883117	0,023332561	0,156849683
		bm25	0,11012987	0,111168831	0,023559974	0,14652249
		DirichletLM	0,145454545	0,10969697	0,025935826	0,202254195
		DPH	0,152121212	0,097316017	0,02214039	0,158249012

As it is possible to see, the variance is quite high for the different measures, so the hypothesis is that similar queries (based on our approach) have quite different results. A possible improvement can be the changing the clustering algorithm or the usage of some optimization techniques in this algorithm.

Search Engine - Advanced Search

To improve the effectiveness of the approaches presented in the previous part, different types of analysis of queries expansion and queries reduction have been conducted.

Query expansion

Different types of expansion have been implemented to increase the size of *adhoc* queries. For this purpose several approaches has been implemented and tested:

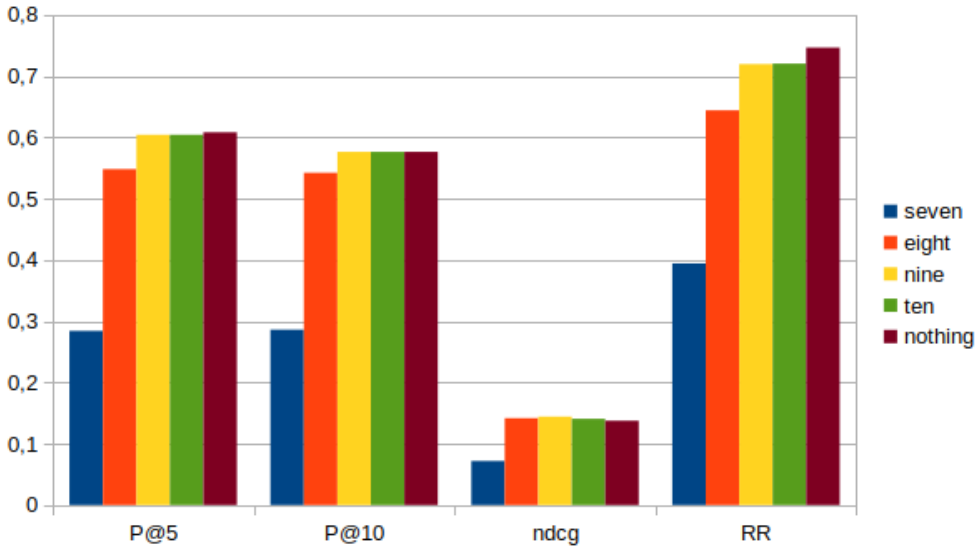
- Glove pre-trained approach (approach seven): queries have been expanded with the three most similar words extracted from Glove pre-trained on Gigaword for each word in the query.
- RM3 (approach eight): it uses RM3 implemented by pyterrier in combination with reranking with BERT and alone.
- Linear combination (approach ten): an approach has been implemented that makes a linear combination between the original queries and the expanded queries (RM3 was used as expanded query).

In the next table, the results obtained by different approaches are shown.

	name	P@5	P@10	ndcg	RR
seven	tf-idf	0,284	0,292	0,071183734	0,42287324
	bm25	0,272	0,28	0,070041526	0,391796289
	DirichletLM	0,236	0,21	0,057890967	0,278317474
	DPH	0,284	0,286	0,071791196	0,394142172
	tf-idf + BERT	0,38	0,372	0,075919051	0,531769522
	bm25 + BERT	0,384	0,372	0,075691242	0,526131178
	DirichletLM + BERT	0,36	0,344	0,066632478	0,491317708
	DPH + BERT	0,384	0,366	0,077291655	0,549860055
eight	tf-idf + BERT + RM3	0,576	0,536	0,139974364	0,680144653
	bm25 + BERT + RM3	0,568	0,518	0,139935591	0,704838691
	DirichletLM + BERT + RM3	0,284	0,252	0,069942969	0,40616384
	DPH + BERT + RM3	0,548	0,542	0,141536054	0,643866947
nine	tf-idf + RM3	0,58	0,544	0,140060216	0,661110824
	bm25 + RM3	556	0,54	0,139359476	0,66770034
	DirichletLM + RM3	0,364	0,328	0,073414534	0,548183613
	DPH + RM3	0,604	0,576	0,144159452	0,719318169
ten	tf-idf + linear	0,584	0,552	0,136749581	0,675062705
	bm25 + linear	0,576	0,536	0,136191856	0,699540571
	DirichletLM + linear	0,364	0,328	0,073414534	0,548183613
	DPH + linear	0,604	0,576	0,140449044	0,720005836

It's possible to make some observations on these results: Glove is the worst approach, maybe because it is too general for this task, the use of Scibert, or the use of fine-tuning

with our documents can improve the performance. The use of RM3 obtain better results, only if it is used without BERT reranking. The use of linear combination doesn't improve the results. But as it is possible to see in the next plot the performances are still without any query expansion technique.



The results reported are the ones obtained on *adhoc* queries, but the conclusions may be the same for the three types of queries (complete results are reported in the appendix).

Query reduction

Different types of reduction have been implemented to reduce the size of the narrative and description queries.

Several experiments using different types of index configurations, pre-processing steps and ranking models have been run to compare the results of the reduced queries with respect to the original models performances. The different indexing configurations that have been tried are: to index only the document's texts, to index only the document's titles and to index both. The pre-processing steps that have been tried are: none, stopwords removal and stopwords removal + stemmer. The different ranking models used are: "tf-idf", "bm25", "DirichletLM", "DPH".

By indexing only the texts or both the texts and the titles, the same results are obtained (best index configuration). The best pre-processing technique is the stopwords removal + stemmer.

For the sake of brevity, only the techniques and experiments that gave the best results are

reported. The best approach of queries reduction is to identify the different parts of speech in the query and remove the words that are associated with specific TAGS. The TAGS that are removed differ from the descriptive to the narrative queries and are the following: RB (adverb), MD (modal), PRP (personal pronoun), CD (cardinal digit), FW (foreign word), DT (determiner), JJS (adjective, superlative), JJR (adjective, comparative), VBG (verb, gerund/present participle), VBD (verb, past tense), VB (verb, base form).

Base Description			
	P@5	P@10	ndcg
tf-idf	0.688	0.654	0.3982146987
bm25	0.680	0.634	0.4010538833
DirichletLM	0.512	0.526	0.3581683159
DPH	0.672	0.636	0.3804839297

Base Narrative			
	P@5	P@10	ndcg
tf-idf	0.6	0.562	0.3050868355
bm25	0.608	0.544	0.3111949108
DirichletLM	0.38	0.348	0.2367194071
DPH	0.556	0.524	0.2843633022

RB - MD - PRP - CD - FW - DT - JJS - JJR Removal Description			
	P@5	P@10	ndcg
tf-idf	0.696	0.668	0.4006264525
bm25	0.688	0.654	0.4037545809
DirichletLM	0.52	0.538	0.3598487431
DPH	0.676	0.652	0.383700856

RB - MD - PRP - CD - FW - DT - VBG - VBD - VB Removal Narrative			
	P@5	P@10	ndcg
tf-idf	0.628	0.572	0.3095844327
bm25	0.616	0.566	0.3158299373
DirichletLM	0.432	0.422	0.2678106271
DPH	0.608	0.568	0.2906203838

Example of query reduction: (Narrative)

Original query:

index	qid	query
47	48	possibility schools opening covid19 pandemic still ongoing topic looking evidence projections potential implications terms covid19 cases hospitalizations deaths well benefits harms opening schools includes impact students teachers families wider community

TAGs:

```
('possibility', 'NN')   ('looking', 'VBG')      ('hospitalizations', 'NNS')
('schools', 'NNS')      ('evidence', 'NN')      ('deaths', 'VBP')            ('impact', 'JJ')
('opening', 'VBG')      ('projections', 'NNS')  ('well', 'RB')               ('students', 'NNS')
('covid19', 'NN')       ('potential', 'JJ')     ('benefits', 'NNS')          ('teachers', 'NNS')
('pandemic', 'NN')      ('implications', 'NNS') ('harms', 'NNS')             ('families', 'NNS')
('still', 'RB')         ('terms', 'NNS')        ('opening', 'VBG')           ('wider', 'VBP')
('ongoing', 'VBG')      ('covid19', 'VBP')      ('schools', 'NNS')           ('community', 'NN')
('topic', 'NN')         ('cases', 'NNS')        ('includes', 'VBZ')
```

TAGs that are removed:

```
('opening', 'VBG')
('still', 'RB')
('ongoing', 'VBG')
('looking', 'VBG')
('well', 'RB')
('opening', 'VBG')
```

Final query:

index	qid	query
47	48	possibility schools covid19 pandemic topic evidence projections potential implications terms covid19 cases hospitalizations deaths benefits harms schools includes impact students teachers families wider community

Re-ranking documents using the date

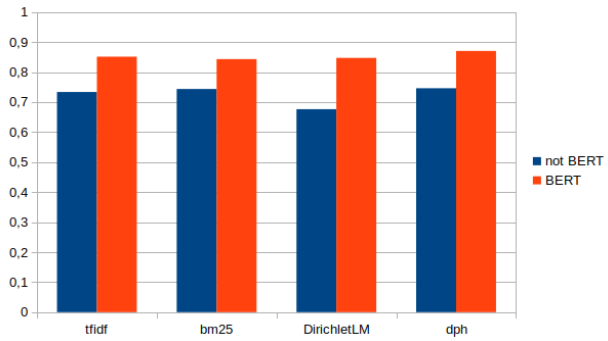
A specific function has been implemented to re-rank documents using the date. Given the ranked list of the retrieved documents for a query the documents have been sorted by grouping them by integer score and in each group by descending date. There is also a function that re-ranks documents by date for all the queries.

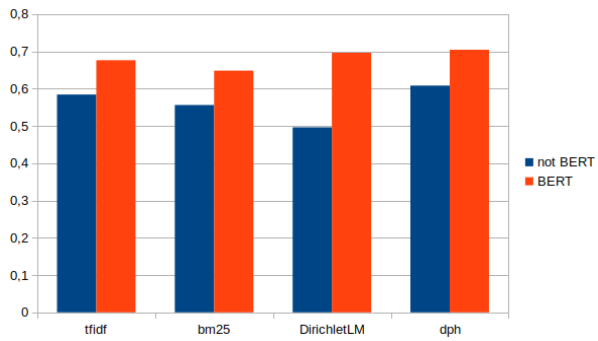
Re-ranking using Neural Approach

For the reranking based on Neural Approach has been used the title of the documents. It used a version of BERT pre-trained on scientific articles (SciBERT), because the version trained on regular text performed worse. It has been observed that this approach improve the performance of the retrieval, as it shown in the next table (the comparison between reranking and not reranking with the two best approaches of the second section is shown):

	name	P@5	P@10	ndcg	RR
Four	tf-idf	0,584	0,566	0,136130674	0,733696407862247
	bm25	0,556	0,562	0,134763775	0,743732766259082
	DirichletLM	0,496	0,482	0,126247966	0,676397374847375
	DPH	0,608	0,576	0,137425081	0,746154264399362
	tf-idf + BERT	0,676	0,638	0,141954305	0,851339285714286
	bm25 + BERT	0,648	0,628	0,141116626	0,843061224489796
	DirichletLM + BERT	0,696	0,656	0,138880319	0,847259085783676
	DPH + BERT	0,704	0,626	0,142016396	0,870215053763441
Six	tf-idf	0,568	0,548	0,134185237	0,698583581021661
	bm25	0,564	0,536	0,13404099	0,690903121526751
	DirichletLM	0,436	0,464	0,121337926	0,633897234901032
	DPH	0,592	0,574	0,137243883	0,717690476190476
	tf-idf + BERT	0,68	0,642	0,141834032	0,841047619047619
	bm25 + BERT	0,672	0,644	0,143355577	0,876699134199134
	DirichletLM + BERT	0,692	0,644	0,13611329	0,878830172999924
	DPH + BERT	0,688	0,642	0,142474817	0,855393374741201

The results for the fourth approach with and without reranking are shown in the following plots. As it is possible to see, the Bert approach for reranking improves the performance.


1
0,9
0,8
0,7
0,6
0,5
0,4
0,3
0,2
0,1
0
tfidf
bm25
DirichletLM
dph
not BERT
BERT


0,8
0,7
0,6
0,5
0,4
0,3
0,2
0,1
0
tfidf
bm25
DirichletLM
dph
not BERT
BERT