

TREC 2019 Precision Medicine - Medical University of Graz

Pilar López-Úbeda¹, José Antonio Vera-Ramos², Pablo López-García^{2,3*}

¹Computer Science Department, CEATIC, Universidad de Jaén (España)

²Institute for Medical Informatics, Statistics and Documentation, Medical University of Graz (Austria)

³CBmed GmbH (Austria)

*Corresponding author: pablo.lopez@medunigraz.at

ABSTRACT

In this paper we report on our participation in the TREC 2019 Precision Medicine track (team name: *imi_mug*). We submitted 5 fully automatic runs to the *biomedical articles* subtask, two of them with treatments. Our system was based on Elasticsearch, templates, and parameter grid search query generation, building heavily on our previous participation and the reference standards from 2017 and 2018. Our results are close to the mean for the biomedical articles subtask.

INTRODUCTION

The goal of the TREC Precision Medicine track is to improve search for clinicians treating cancer patients. As in previous editions, the challenge was divided into two subtasks that consisted of retrieving (1) *biomedical articles* relevant for treatment (from PubMed) and (2) *clinical trials* relevant for enrollment (from ClinicalTrials.gov). This year, 40 cases describing potential patients, termed *topics*, were provided as input and contained three features or dimensions: *disease*, *gene*, and *demographic*. The goal of retrieving scientific abstracts was to identify relevant articles for the treatment, prevention, and prognosis of the disease under the specific conditions for a given patient. A new subtask was introduced this year, where not only relevant PubMed articles had to be

identified, but also the specific treatments they mentioned. The description of the challenge, datasets, relevance judgment guidelines, etc. are available online¹.

SYSTEM OVERVIEW & GENERAL APPROACH

This year we refined the framework used in the two past editions and extended it to participate in the treatment identification subtask. In particular, (1) we reused our indexing infrastructure and extended it to include treatments, (2) we took advantage of the official reference standards from 2017 and 2018, and (3) we applied our learnt lessons from past editions. Reports on our previous participation^{2,3} and the overview papers of past edition of TREC Precision Medicine are useful to better understand this paper. This year we did not participate in the *clinical trials* subtask, but participated in the treatment identification subtask instead.

The overall aim of our system was to automatically generate Elasticsearch queries that produced good results using datasets from previous years for reference and training.

¹<http://www.trec-cds.org/2019.html>

²https://trec.nist.gov/pubs/trec26/papers/imi_mug-PM.pdf

³https://trec.nist.gov/pubs/trec27/papers/imi_mug-PM.pdf

QUERY TEMPLATES, EXPERIMENTS & EVALUATION

To find queries that provided good results, we reused our *query templates* from 2018. As a first step we analyzed our results from last year and reconsidered our strategies, most of which we kept: (1) exploring *must* and *should* clauses, (2) searching for *disease* and *gene* features in multiple fields and fine-tuning *multi_match type*, *tie_breaker* and *boost* parameters, and (3) adding keywords to improve the search. Strategies that we did not use include (1) synonym expansion for diseases and genes and (2) the use of each patient's demographics.

Strategies implemented this year were:

1. We kept the separation of the gene field into three fields but added a new one called *extra_inf* with separate information such as: mutations, amplifications, fusions, etc. These new strategies did not produce good results so we decided to use the gene field without expansion.
2. The use of different query parameters in Elastic such as: *best_fields*, *tie_breaker*, *boost*, etc.
3. Adding treatment-related keywords: “*surgery therapy treatment prognosis prognostic survival patient resistance recurrence targets malignancy study therapeutical outcome*”.
4. Indexing treatments separately⁴ using the extracted treatments provided by TREC. This allowed us to retrieve treatments mentioned in a scientific abstract efficiently by just searching by document *id*.

For convenience, we reused our experiments framework from 2018, based

⁴https://github.com/plopezgarcia/trec-2019-precision-medicine/blob/master/python-experiments/treatments_indexing/index_treatments.py

on python and pandas, which is available as open source in form of Jupyter Notebooks⁵. For evaluation, we relied on our custom modification of the *pytrec_eval* python module⁶.

Our algorithm to find those optimal queries via experiments was an iterative version of the one used last year⁷:

1. Pick a query template.
2. Build and perform the associated Elasticsearch queries.
3. Calculate the aggregated evaluation measures for all the topics of 2018 using the official reference standard.
4. Check *infNDCG*, *R-precision*, and *precision at 10* using the reference standard from 2018.
5. Save the template and optimized parameters as an Elasticsearch *json* query.
6. Perform the query for this year's topics.
7. Append the previously indexed treatments (in two of the five submitted runs).
8. Export results in standard *trec_eval* format as a run for submission.

SUBMITTED RUNS & RESULTS

We submitted 5 fully automatic runs to the *biomedical articles* subtask, where we tried to capture different strategies and evaluation measures. Two of these runs were repeated to include treatments extracted using MetaMapLite as provided by the TREC organization. Our submissions, the exact steps we took, and the evaluation results for 2018 topics used

⁵<http://jupyter.org/>

⁶https://github.com/cvangysel/pytrec_eval

⁷https://github.com/plopezgarcia/trec-2019-precision-medicine/blob/master/python-experiments/trec_utils/running.py

for training are available online as Jupyter Notebooks for the biomedical articles⁸.

They can be summarized as follows:

Biomedical Articles

1. Baseline well-performing query from 2018 with strict *multi_match* (*must*) of *disease* and *gene* in *title*, *abstract* and *MeSH tags* and grid-search optimized parameters *tie_breaker*, *type*, and *boost*.
2. Same as (1), but eliminating the search in *MeSH tags* field.
3. Same as (2), plus the treatment identifier codes on each returned document.
4. Same as (2), plus keywords related to treatments to the query (*should*).
5. Same as (4), plus the treatment identifier codes on each returned document.

A comprehensive and more illustrative description of the results and how we obtained them can be found in the appendices.

Appendix A summarizes the submitted runs in a detailed table, and shows the exact Elasticsearch *json* queries that produced our submitted runs.

Appendix B and C show our results of *biomedical articles* subtask and *treatments* results as graphs and compares them with the average and best results.

CONCLUSION

In this notebook we reported on our participation in the TREC 2019 Precision Medicine track and described our approach, strategies, results, and lessons learnt.

As last year, we confirmed that many promising strategies (like expanding *disease* and *gene*) did not produce the expected results when evaluated with past gold standards, so we had to remove them. Grid search was again useful to fine-tune complex Elasticsearch query parameters, such as *multi_match type*, *tie_breaker* and *boost*, but required considerable experimentation time.

Our historical results show a steady decline in performance. From being among the top teams in 2017, we only got results slightly above average in 2019. This outcome suggests that we revisit our strategy, which might include developing a new one from scratch and abandon Elastisearch-based query generation. A partial explanation for our declining results is that we open sourced our framework and we were happy to witness it was used by other teams in 2018 (possibly in 2019 too). This proves the positive aspects of open source and the impact of our participation in TREC Precision Medicine.

Acknowledgments

This work was partially supported by the Spanish government via the projects RTI2018-094653-B-C21 (LIVING-LANG) and TIN2015-65136-C2-1-R (REDES).

⁸https://github.com/plopezgarcia/trec-2019-precision-medicine/blob/master/python-experiments/abstracts/abstracts_submissions.ipynb

APPENDIX A: Submitted Runs (Queries) 2019

Biomedical Articles - Summary

		RUN		
FEATURE	QUERY	1	2	3
disease	fields	title abstract meshTag	title abstract	
	match type	multi_match best_fields		
	bool	must		
	tie_breaker	0.5		
	boost	1.5		
gene	fields	title abstract meshTag	title abstract	
	match type	multi_match best_fields		multi_match cross_fields
	bool	must		
	tie_breaker	0.5		
	boost	1		
treatment keywords	fields	-		title abstract
	match type	-		multi_match
	bool	-		should
FEATURE	QUERY	1	2	3
RUN				

Biomedical Articles - Queries for Submitted Runs

imi_mug_run_1.json

```
{
  "from":0,
  "size":{{output_size}},
  "query": {
    "bool": {
      "must": [
        {
          "multi_match": {
            "query": "{{disease}}",
            "fields": [
              "title",
              "abstract",
              "meshTags"
            ],
            "tie_breaker": 0.5,
            "type": "best_fields",
            "boost": 1.5
          }
        },
        {
          "multi_match": {
            "query": "{{gene}}",
            "fields": [
              "title",
              "abstract",
              "meshTags"
            ],
            "tie_breaker": 0.5,
            "type": "best_fields",
            "boost": 1
          }
        }
      ]
    }
  }
}
```

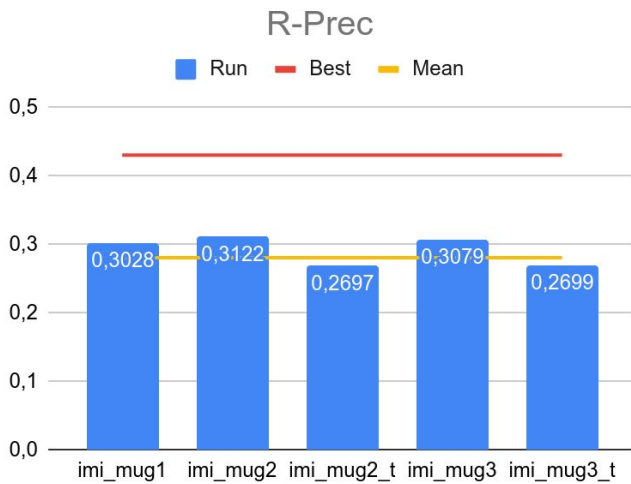
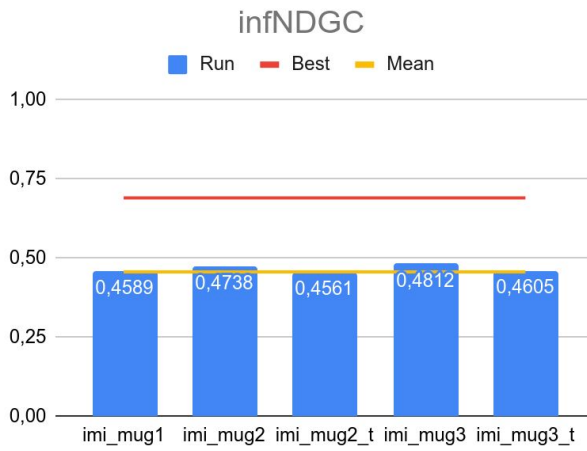
imi_mug_run_2.json

```
{
  "from": 0,
  "size": {{output_size}},
  "query": {
    "bool": {
      "must": [
        {
          "multi_match": {
            "query": "{{disease}}",
            "fields": [
              "title",
              "abstract"
            ],
            "tie_breaker": 0.5,
            "type": "best_fields",
            "boost": 1.5
          }
        },
        {
          "multi_match": {
            "query": "{{gene}}",
            "fields": [
              "title",
              "abstract"
            ],
            "tie_breaker": 0.5,
            "type": "best_fields",
            "boost": 1
          }
        }
      ]
    }
  }
}
```

imi_mug_run_3.json

```
{
  "from":0,
  "size": {{output_size}},
  "query": {
    "bool": {
      "must": [
        {
          "multi_match": {
            "query": "{{disease}}",
            "fields": [
              "title",
              "abstract"
            ],
            "tie_breaker": 0.5,
            "type": "best_fields",
            "boost": 1.5
          }
        },
        {
          "multi_match": {
            "query": "{{gene}}",
            "fields": [
              "title",
              "abstract"
            ],
            "tie_breaker": 0.5,
            "type": "cross_fields",
            "boost": 1
          }
        }
      ],
      "should": [
        {
          "multi_match": {
            "query": "surgery therapy treatment prognosis
            prognostic survival patient resistance
            recurrence targets malignancy study
            therapeutical outcome",
            "fields": [
              "title",
              "abstract"
            ]
          }
        }
      ]
    }
  }
}
```

APPENDIX B: Biomedical Articles Results



APPENDIX C: Treatments Results

