Team UVA_ART at TREC 2018 Precision Medicine Track: Graph-based Concept Expansion and NLP for Document Relevance Boosting

Sean Mullane, Kasi Vegesana, Valentina Baljak[1]

Abstract— **This paper describes the UVA_ART* team entries in the TREC 2018 workshop series Precision Medicine Track. We submitted 5 runs for the Scientific Abstracts task. Our approach used an exclusivity-based relatedness measure defined on the UMLS Metathesaurus ontologies to add context to our queries. We combined this with natural language processing using cTAKES for concept annotation to effect a graph-based query expansion on an enriched document corpus. We used Elasticsearch as our ranking and query engine with different query templates for each run. Our efforts demonstrate that the existing medical ontologies can be leveraged to achieve moderate results with little to no other clinical input.**

I. INTRODUCTION

Successfully treating cancer is made particularly difficult because of its high mutation rate and the many forms it can take. A treatment that is effective against one cancer variant may fail against another, even among cancers within the same patient. Thus it is paramount for a physician to choose the correct treatment for each patient [1].

At the same time, since there are a multitude of research papers and clinical trials that each may be relevant to the treatment of a given cancer, it is difficult for a physician to be aware of newer and potentially more effective treatments in each case. The TREC 2018 Precision Medicine track aims to encourage research into precision medicine, oncology in particular, to provide better solutions to physicians and researchers.

The TREC 2018 PM track is a continuation of the 2017 PM track with some modifications. This track is split into two components: the Scientific Abstracts task and the Clinical Trials task.

- Scientific Abstracts: Participants ranked and submitted articles from a corpus of bio-medical article abstracts, largely from MEDLINE/PubMed. The documents were ranked by relevance for the treatment, prevention, and prognosis of the disease given specific genetic and demographic information about the patient.
- Clinical Trials: Participants ranked and submitted clinical trials from a set of clinical trials listed on ClinicalTrials.gov. The trials were ranked by relevance and eligibility for the patient given their specific genetic and demographic information.

We chose to focus on the Scientific Abstracts task, for which we submitted 5 runs. This task is the most directly relevant to potential clinical care at our institution.

*This work was supported by the University of Virginia Health System

[1]All authors are with the Data Science group of the Analytics and Reporting Team at the UVA Health System, University of Virginia, Stacey Hall, Charlottesville, VA SPM9R@virginia.edu, KBV7C@virginia.edu, VB8N@virginia.edu

II. SYSTEM OVERVIEW

We worked under the usual time constraints, and wanted to utilize readily available tools for this task, so we chose to use several tools already in use at the UVA Health System: Apache cTAKES 4.0.0 and Elasticsearch 6.2.1. Apache cTAKES [2] is a natural language processing system designed for extraction of information from electronic medical record clinical free-text. Elasticsearch is an open-source document search and analytics engine.

A. Natural Language Parsing

We used a basic cTAKES dictionary look-up annotation pipeline to annotate the article abstracts and titles with identified concepts from the Unified Medical Language System (UMLS) Metathesaurus, a set of medical ontologies comprised of medically-relevant terms and relationships among them. The UMLS was provided by the National Library of Medicine [3]. This processing served two purposes:

1) Reduce noise and variation within terms by mapping multiple variations of a single concept to the single concept unique identifier (CUI).
2) Enable direct use of CUIs from query expansion tool to search no need to map CUIs back to natural language.

B. Query Expansion

The small amount of information available for each topic and the large number of articles that can be returned by a query make it difficult to distinguish among the highest-ranked articles. To better distinguish between more and less relevant articles, it can be useful to use query expansion to add terms to the query. Adding appropriate terms to the query can help surface more relevant papers over less relevant papers.

We used a largely hands-off approach to expand our queries. Our primary approach used a concept graph-based relatedness metric to find the most closely-related concepts to those associated with each topic. We also included a handful of terms that proved useful to distinguish queries in a similar approach to ours in the TREC 2017 PM task [4].

C. Elasticsearch Query Boosting

We used the Elasticsearch relevance ranking engine to score the articles in our corpus. All 5 runs used the default relevance scoring algorithm. Inspired by the approach in [4], we chose to create a set of templates for our 5 submissions, which included a variety of required terms and optional

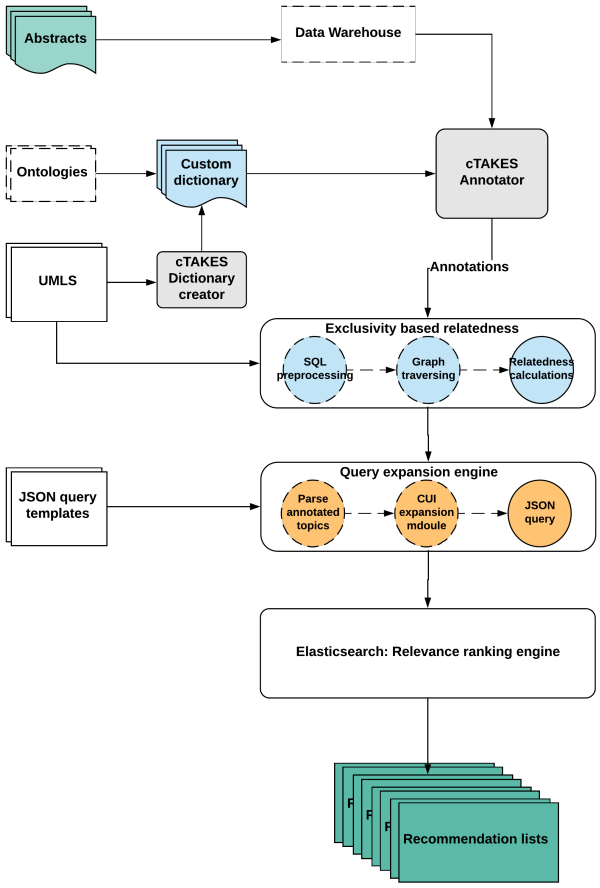


Fig. 1. System Overview

boosting terms to upweight or downweight the results. The 5 runs differed in which terms were included and what boosting values were applied.

III. METHODS

A. Raw Data Preprocessing

For our first step we wrote a Python script using the base Python xml package to parse the XML dataset. We extracted the most common elements of each document and wrote the resulting data into a database table in MS SQL Server 2016. We chose to use fields that were most frequently available for our analysis: PMID, Title, Abstract, and Journal. While other data e.g. MeSH tags were available, we chose to use only textual data for simplicity and because we already have a text-processing infrastructure available to us.

B. Article Pre-Filtering

Since PubMed includes many documents which are not relevant at all to cancer treatment, and to reduce the time required to annotate the documents, we filtered the corpus down to 2.7 million abstracts out of 15 million in our original data set. This count of articles was prohibitively large for any sort of meaningful processing.

To filter down the articles we used the tree-like nature of the ISA relation in the SNOMEDCT-US ontology to expand the concept for cancer (C0006826) to all of its descendants. We extracted the preferred natural text label of each concept, keeping 200 concepts that described types of cancer. Using an exact text search we kept in the data set all articles whose abstracts included at least 1 of these phrases or which was published in a journal composed of at least 10% of such articles.

C. Natural Language Processing

In order to include in our scope the search terms most relevant to the task, we created a custom dictionary using the cTAKES dictionary creator tool. We chose to use the NCI, RxNorm and SNOMED-CT US ontologies from the 2016 AB UMLS Metathesaurus as the basis of this dictionary and of the concept graph, selecting identified types (TUI) relevant to diseases and symptoms, cancer types, genes, demographics, medications and several other categories.

For the cTAKES dictionary look-up annotation, we used the fast dictionary look-up annotation pipeline. For this pipeline, cTAKES creates a set of phrase variants for each concept in the dictionary. Then for each word token and the tokens forming its surrounding context, phrases matching the set of tokens are returned. We chose the ”overlap” look-up annotator which allows for a limited number of tokens to be skipped to capture more phrase variants as matches. The fast look-up annotator was found to perform similarly to more complex methods of dictionary look-up [5].

D. Query Expansion

There have been several approaches to query expansion in the TREC 2017 precision medicine track which performed well [8] [9]. We therefore chose to use query expansion but chose a different method to achieve this.

By using cTAKES to annotate the likely subset of articles and the topics themselves with identified concepts from the UMLS, we were able to directly make use of the concepts and their defined relationships in the UMLS ontologies to create a query expansion engine using graph analysis.

Our goal here was to add concepts that are not only related to the query concepts but nearly exclusively so. By using an exclusivity-based relatedness metric [7] we aimed to minimize the generality of the resulting expanded query. For example, expanding the Melanoma concept to the Cancer concept (C0006826) would return many results for other types of cancer. Expanding Melanoma to specific closely-related concepts, e.g. particular genes or cancer variants, instead expands the query to find articles that are still highly relevant to Melanoma in particular, so other more general articles that are not specific to Melanoma should be ranked lower.

E. UMLS Ontologies

We used the two default ontologies used by cTAKES (SNOMED-CT US and RxNorm) as well as NCI. These are part of the Metathesaurus:

> The UMLS includes the Metathesaurus, the Semantic Network, and the SPECIALIST Lexicon

and Lexical Tools. The Metathesaurus is the biggest component of the UMLS. It is a large biomedical thesaurus that is organized by concept, or meaning, and it links similar names for the same concept from nearly 200 different vocabularies. The Metathesaurus also identifies useful relationships between concepts and preserves the meanings, concept names, and relationships from each vocabulary. [6]

SNOMED-CT US and RxNorm are comprehensive repositories of general medical information and pharmaceuticals, respectively. The NCI ontology represents information from the National Cancer Institute.

F. Relatedness Metric

We adapted the relatedness metric defined in [7]. The UMLS by default has the necessary requirements for this: a set of concepts with relationships defined between them with types defined on the relationships. A further requirement, symmetry in the relationships, is not present by default. To make use of this it was necessary to alter pairs of relationships where a natural inverse exists (e.g. `is_part_of` and `has_part`) to a single relationship type. In other cases where a relationship is not paired with a natural inverse we pruned the relationships. We used MS SQL Server 2016 to preprocess the data and the NetworkX v1.11 library in Python 3.6.6 to create and traverse the graph. Our graph had 451,021 nodes with 3,134,917 edges and 162 relationship types. There were 632,634 nodes, 8,804,418 relationships and 397 relationship types in the 3 ontologies we used from the 2016AB UMLS Metathesaurus.

G. Relatedness Algorithm

Steps 1, 2, and 3 are quoted from the definition in [7]:

Definition *Given an edge e of type t between two adjacent nodes x and y, directed from x to y, we define the* ***exclusivity*** *of edge e as the probability that, if we randomly select an edge e' out of the set of all edges of type t that exit node x and all edges of type t entering node y, that edge e' is edge e. Formally,*

$$exclusivity(x \xrightarrow{\tau} y) = \frac{1}{\mid x \xrightarrow{\tau} * \mid + \mid * \xrightarrow{\tau} y \mid - 1} \quad (1)$$

where $|x \xrightarrow{\tau} *|$ denotes the number of relations of type $\tau \in T$ that exit node x, and $|* \xrightarrow{\tau} y|$ denotes the number of relations of type $\tau \in T$ that enter node y.

Given a path through G, $P = n_1 \xrightarrow{\tau 1} n_2 \xrightarrow{\tau 2}, ..., n_K$, with $\tau_i \in T^{\mp}$ its weight can be computed by Formula 2.

$$weight(P) = \frac{1}{\sum_i 1/exclusivity(n_i \xrightarrow{\tau} n_{i+1})} \quad (2)$$

Then, given a node x and the set $y_i \in N$ of nodes connected to x by paths of length k or less, we compute their relatedness as the sum of the path weights of all paths of length k or less between x and y. In order to give preference to shorter paths, we factor in a constant length decay factor, a.

$$rel^{(k)}_{Excl}(x, y) = \sum_{P_i \in P^{(k)}_{x,y}} \alpha^{length(P_i)} weight(P_i) \quad (3)$$

Finally we keep the top m nodes $y_i \in N$ defined for a constant c by:

$$\{y_i \mid rel^{(k)}_{Excl}(x, y_i) > c\} \quad (4)$$

We chose $\alpha = 0.5$ and $c \in [0.1, 0.2]$, with c chosen separately for each data type (Diagnosis, Gene) to return 5-10 CUIs on average.

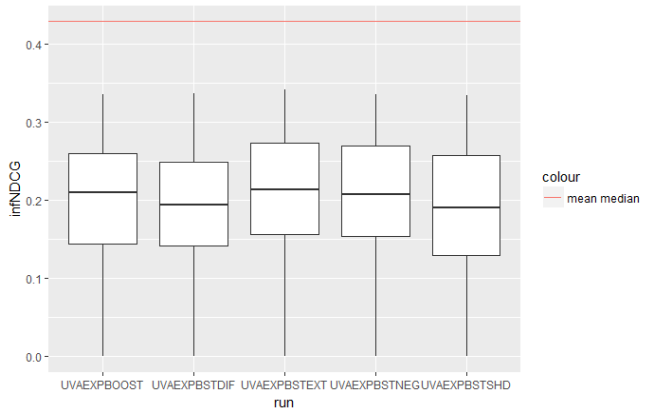
IV. RESULTS

A. Overview of Runs

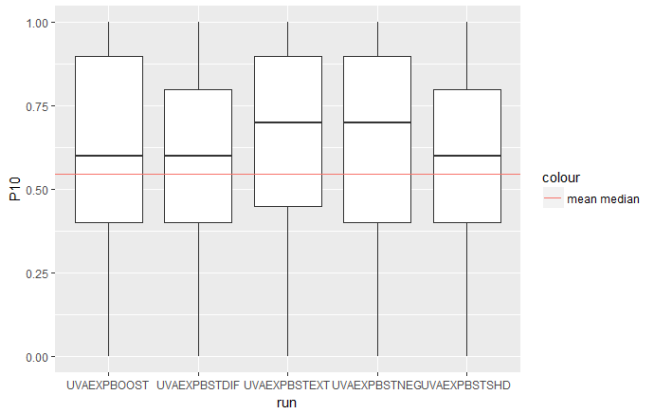
1) UVAEXPBOOST: Topic data were input in natural text and CUI-encoded form into a query for Elasticsearch that boosted exact CUI matches (especially disease and gene) and boosted results including terms that refer to treatment and prognosis, and negatively boosted some terms that refer to non-cancer or non-human entities.
2) UVAEXPBSTSHD: This was similar to UVAEXPBOOST but differs in that it left gene as a "should" field rather than a "must" field as with UVAEXPBOOST. It also included boosted results including CUIs that refer to treatment and clinical trial concepts and that refer to treatment and clinical trial concepts.
3) UVAEXPBSTEXT: This was similar to UVAEXPBOOST but included extra CUI terms for clinical trials and related to apoptosis and remission, which were boosted, and terms for screening and detection which were negatively boosted. It also included boosted results including CUIs that refer to treatment and clinical trial concepts.
4) UVAEXPBSTNEG: This run included the boost weighting from UVAEXPBOOST and the extra CUIs from runs 2 amd 3, but differed in that it negatively boosted some CUIs that refer to cancer screening and detection.
5) UVAEXPBSTDIF: This run was similar to UVAEXPBSTSHD but used different boosting ratios.

V. DISCUSSION

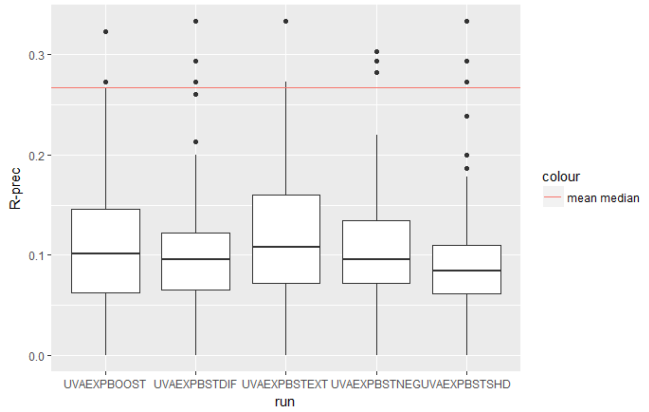
Overall all 5 runs had decidedly mixed results. All were below the mean of TREC participant median topic performance in infNDCG and R-Prec, but above mean median in the P10 measure. We saw a sharp drop-off in article count during 2014-2015 which could possibly explain some of the issues we faced with prediction accuracy. However we discovered this too late to correct it. We also observed that the journals with the highest number of articles pre-and-post filtering were significantly different. This was to be expected given that we selected specifically for cancer-related articles.



(a) All Topics infNDCG: UVA_ART vs Median



(b) All Topics P10: UVA_ART vs Median



(c) All Topics R-Prec: UVA_ART vs Median

Fig. 2. UVA_ART overall results

The UVAEXPBSTEXT run performed best of the 5 runs. The boosting of results including CUIs that refer to treatment and clinical trial concepts and that refer to treatment and clinical trial concepts caused an improvement in infNDCG and particularly P10 precision over the more basic UVAEXPBOOST. We note that the requirement that the Gene CUI match caused an improvement over the UVAEXPBSTSHD run, which was similar except for Gene being optional.

Interestingly the addition of extra negation terms in the UVAEXPBSTNEG run did not improve precision. Based on

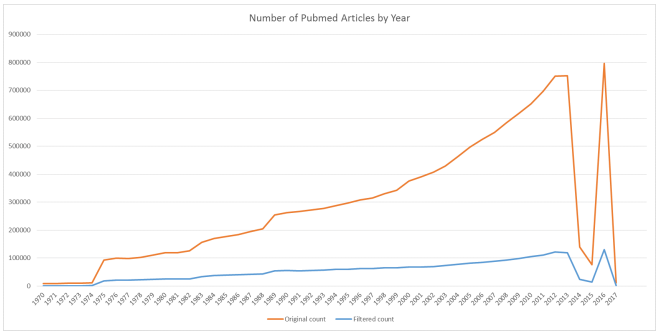


Fig. 3. Article count by year

results from other teams in TREC 2017 PM, we believe that tuning the boosting terms with clinical judgment of article relevance would provide a significant improvement.

The moderate overall performance of this method demonstrates that even an untuned method that makes no direct use of clinical judgment can perform adequately on this task.

VI. Follow-up Analysis

After the results had been published, we looked further into the reasons for the performance of our model. We were interested to determine the effect of several factors to the performance across the three main TREC metrics. New runs included the full set of 1000 documents per topic, instead of top 20 we submitted for the conference. Further, we decided not to add missing abstracts for 2014 and 2015 in order to isolate effects of changes to the model, and to compare results to the original runs.

We wanted to evaluate key aspects of our approach to compare the relative gain in performance from each: concept embedding of query terms, relatedness graph-based expansion of query terms, Elasticsearch custom query boosting, and extended general text terms and CUIs. To do this we ran a series of additional queries using different combinations of query elements with the best-performing query from the original submission for comparison.

1) **must_diseasegene_should_extracuis** (UVAEXPBSTEXT): This run was our best performing run originally submitted to TREC and it included full abstract text, exact CUIs, and extended general CUIs as "should" field in Elasticsearch query.
2) **new_basic_noCUIs**: As an input, uses only natural text of abstracts.
3) **new_basic_exactCUIs**: Includes the text and exact CUIs, but does not utilize any type of query expansion or boosting
4) **new_basic_extCUI**: Includes text, and extended general CUIs in "should" query clause.
5) **new_must_exactCUIs_expCUIs_extCUIs**: This run includes previous terms and adds exact CUIs and expanded CUIs for diagnosis and gene as "must".
6) **new_must_exactCUIs_expCUIs_extCUIs_boost**: In addition to previous run, this boosted exact CUIs for diagnosis and gene.

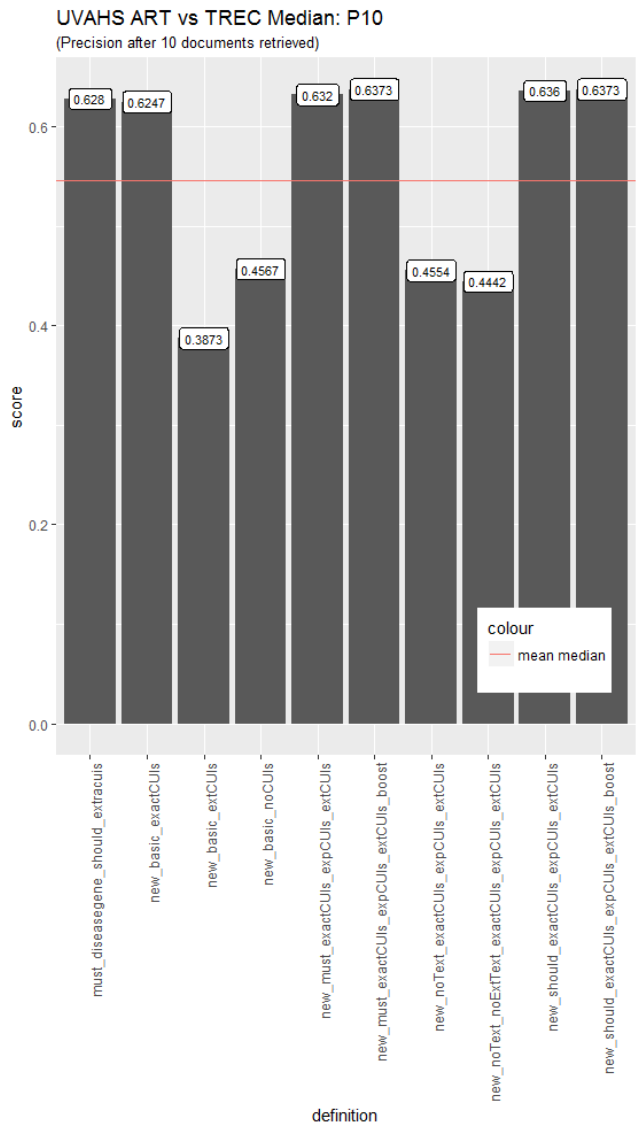


Fig. 4. Updated UVAHS ART vs TREC median:: P10

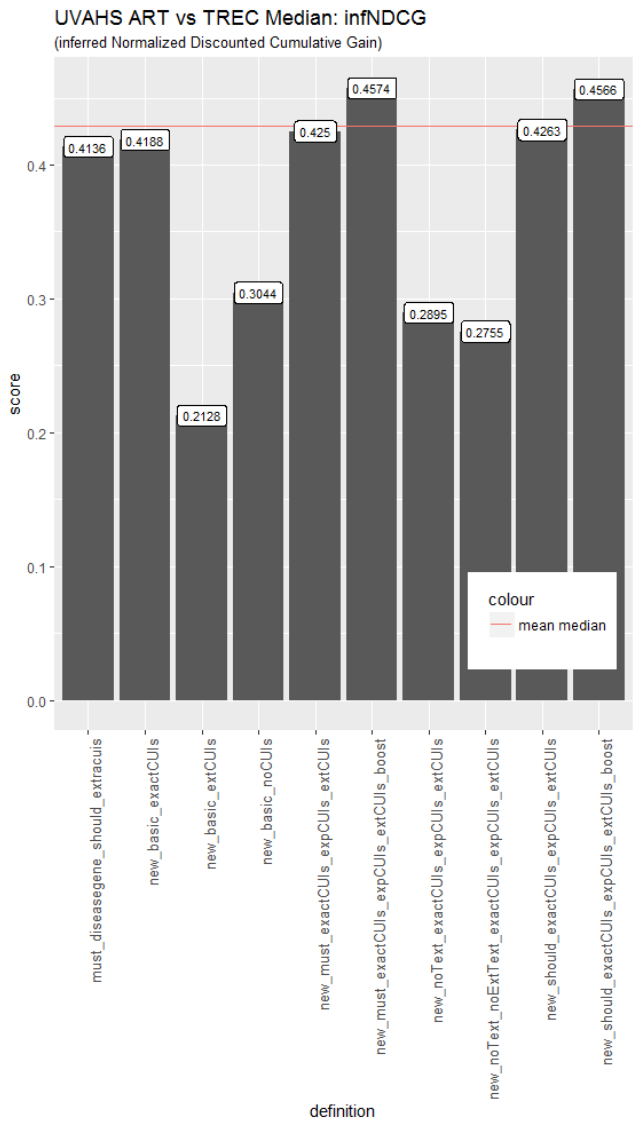


Fig. 5. Updated UVAHS ART vs TREC median: infNDCG

7) **new_should_exactCUIs_expCUIs_extCUIs**: This run includes text, extended general CUIs as ”should”, exact CUIs and expanded CUIs for diagnoses and gene in a ”should” clause.
8) **new_should_exactCUIs_expCUIs_extCUIs_boost**: Similar to the previous run, this run adds exact CUI boosting.
9) **new_noText_exactCUIs_expCUIs_extCUIs**: This run excludes diagnosis and gene text query terms, but includes expanded general text terms; it uses exact CUIs, extended general CUIs in ”should” field, and expanded CUIs for diagnosis and gene in a ”should” clause.
10) **new_noText_noExtText_exactCUIs_expCUIs_extCUIs**: In this run, text is not included at all, and instead uses exact CUIs, extended general CUIs as ”should”, and expanded CUIs for diagnosis and gene in a ”should” clause.

Overall, new runs show improved results compared to the original submission in the metrics where additional documents are beneficial. P10 remains our strongest metric category, with 6 out 0f 10 runs beating the TREC median performance, as shown in Figure 4. We have seen the most significant improvement, with inclusion of all 1000 articles per topic, in infNDCG metric, with 2 runs beating the median, Figure 5. In R-prec metrics, we have seen some significant increase, Figure 6, with several runs coming close to the median, but none surpassing it. We believe that missing articles had the most significant impact on the performance under this metric, given the high probability that a number of relevant articles have been published in the time frame where we missed documents.

Using ”must” vs ”should” section in Elasticsearch query did not significantly affect results. cTakes CUI embbeding provided the highest precision gain, while relatedness helps long tail cumulative gain, but doesn’t improve precision.

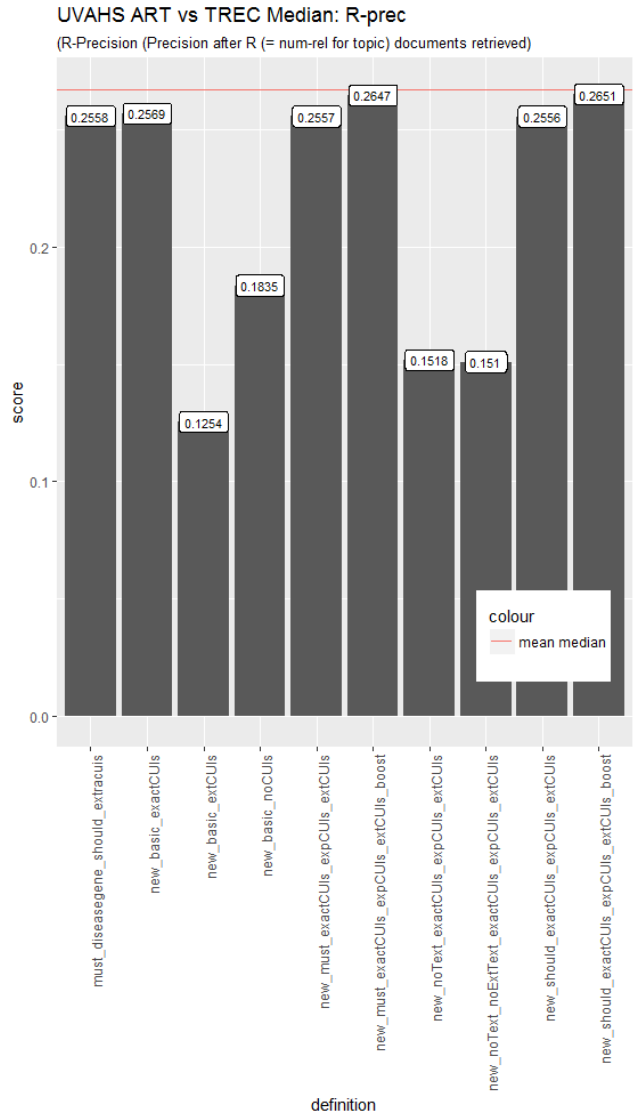


Fig. 6. Updated UVAHS ART vs TREC median: R-prec

Similarly, Elasticsearch boosting helps total cumulative gain, without significant increase of precision. However, pure embedding doesn't perform as well as a combined approach with inclusion of full abstract text. Also, at some point, additional expanded CUIs show a decline in gain, most likely due to widening the search scope to include less relevant articles.

Overall best performance across the metrics was achieved in a run named new_should_exactCUIs_expCUIs_extCUIs_boost. This run uses the most complete query with inclusion of full text, extended general CUIs as "should", exact CUIs, expanded CUIs for diagnoses and gene in "extrashould" field, and Elasticsearch boosting. While some components of the approach affected performance in certain metrics more than others, the most complete combination of concept embedding of query terms, relatedness graph-based expansion of query terms, Elasticsearch custom query boosting, and expanded general text terms and CUIs, has proven to be the best performing overall.

We believe that with further refinement of CUIs and additional expert knowledge we can improve the performance of this approach. The relative simplicity of the system lends itself well to practical use in health care institutions, and for the wide variety of topics.

REFERENCES

[1] K. Roberts et al., Overview of the TREC 2017 Precision Medicine Track, p. 13.

[2] Savova, Guergana; Masanz, James; Ogren, Philip; Zheng, Jiaping; Sohn, Sunghwan; Kipper-Schuler, Karin and Chute, Christopher. 2010. Mayo Clinic Clinical Text Analysis and Knowledge Extraction System (cTAKES): architecture, component evaluation and applications. JAMIA 2010;17:507-513 doi:10.1136/jamia.2009.001560

[3] UMLS Release File Archives: 2004-2017AB. [Online]. Available: https://www.nlm.nih.gov/research/umls/licensedcontent/umlsarchives04.html. [Accessed: 31-Oct-2018].

[4] P. Lpez-Garca, M. Oleynik, Z. Kas, and S. Schulz, TREC 2017 Precision Medicine - Medical University of Graz, p. 12.

[5] cTAKES 4.0 - Fast Dictionary Lookup - Apache cTAKES - Apache Software Foundation. [Online]. Available: https://cwiki.apache.org/confluence/display/CTAKES/cTAKES+4.0+-+Fast+Dictionary+Lookup. [Accessed: 31-Oct-2018].

[6] Metathesaurus. [Online]. Available: https://www.nlm.nih.gov/research/umls/knowledge_sources/metathesaurus/. [Accessed: 31-Oct-2018].

[7] I. Hulpu, N. Prangnawarat, and C. Hayes, Path-Based Semantic Relatedness on Linked Data and Its Use to Word and Entity Disambiguation, in The Semantic Web - ISWC 2015, vol. 9366, M. Arenas, O. Corcho, E. Simperl, M. Strohmaier, M. dAquin, K. Srinivas, P. Groth, M. Dumontier, J. Heflin, K. Thirunarayan, K. Thirunarayan, and S. Staab, Eds. Cham: Springer International Publishing, 2015, pp. 442457.

[8] T. R. Goodwin, M. A. Skinner, and S. M. Harabagiu, UTD HLTRI at TREC 2017: Precision Medicine Track, p. 9.

[9] Team UKNLP at TREC 2017 Precision Medicine Track: A Knowledge-Based IR System with Tuned Query-Time Boosting.