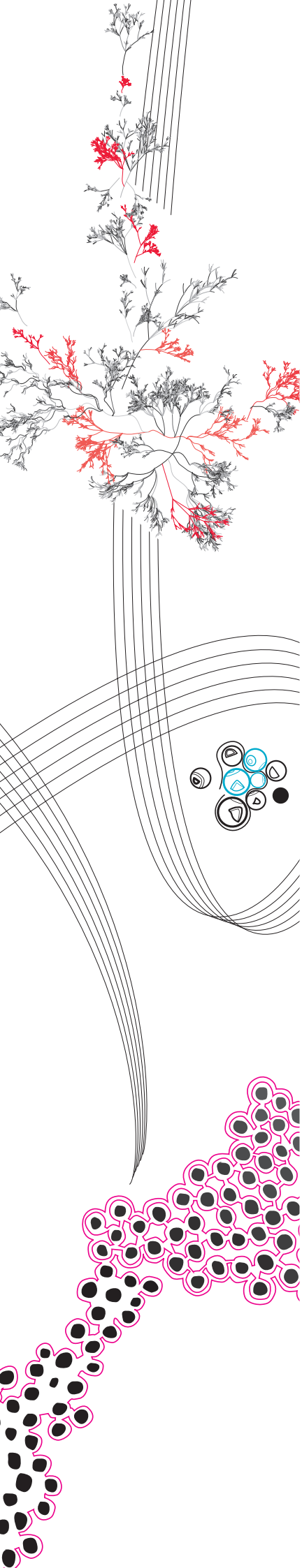

MASTER'S THESIS

IMPROVING BIOMEDICAL INFORMATION RETRIEVAL WITH PSEUDO AND EXPLICIT RELEVANCE FEEDBACK

S.W.R. Niesink

Computer Science (CSc): Data Science and Smart Services (DS3)

Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS)
Database Group (DB)

Exam committee:
dr. ir. Djoerd Hiemstra (University of Twente)
dr. Mariët Theune (University of Twente)
dr. ir. Dolf Trieschnigg (University of Twente)

August 18, 2017

UNIVERSITY OF TWENTE.

Bas Niesink: *Biomedical Publication Retrieval for the HERO Project,* Concept-based Retrieval with Pseudo and Explicit Relevance Feedback, August 18, 2017

ABSTRACT

The HERO project aims to increase the quality of supervised exercise during cancer treatment by making use of a clinical decision support system. In this research, concept-based information retrieval techniques to find relevant medical publications for such a system were developed and tested. These techniques were designed to search multiple document collections, without the need to store copies of the collections.

The influence of pseudo and explicit relevance feedback using the Rocchio algorithm were explored. The underlying retrieval models that were tested are TFIDF and BM25.

The tests were conducted using the TREC Clinical Decision Support datasets for the 2014 and 2015 editions. The TREC CDS relevance judgements were used to simulate explicit feedback. The NLM Medical Text Indexer was used to extract MeSH terms from the TREC CDS topics, to be able to conduct concept-based queries.

Furthermore, the difference in performance when using inverse document frequencies calculated on the entire PMC dataset, and on a collection of several thousand intermediate search results were measured.

The results show that both pseudo and explicit relevance feedback have a strong positive influence on the inferred NDCG. Additionally, the performance difference when using IDF values calculated on a very small document collection is limited.

To the memory of my beloved mother,
my wonderful father and brother,
and my dear friends.

Thanks for supporting me,
through peaks and valleys.

I could not have done this without you.

ACKNOWLEDGEMENTS

I want to thank Djoerd, Dolf, and Mariët for their support during this project. Especially, for their excellent feedback and ability to cope with my tendency to delay work till the last possible moment before a deadline. Furthermore, I'd like to thank them for giving inspiring courses and teaching many interesting subjects, which all led to this final project.

In addition, I would like to thank my family and close friends, present and past, for helping me during my final years in university, at both educational and personal level.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

1 INTRODUCTION

1.1 MOTIVATION

Patients undergoing cancer treatments may benefit from supervised exercise to limit fatigue, preserve their physical condition, improve their quality of life, and increase tolerance to their cancer treatments [8, 20, 26, 30, 31]. They need tailored exercise plans which take their treatment, health, and diseases or disorders into account. This is especially relevant when a patient suffers from *comorbidities*, which means that the patient has multiple diseases or disorders.

Creating exercise plans is a complex task, requiring knowledge about many influencing factors, such as side effects of treatments and exercise physiology. To aid medical professionals in this task and improve the quality of supervised exercise the HERO (*Health Rover*) project was established (section 2.8). HERO will consist of clinical decision support technology, an automated biomedical search engine providing continuous literature updates, and a system to derive knowledge from the retrieved literature. Experts review the papers found before these are added to the HERO knowledge system to make certain that only relevant publications are used.

The goal of this research is to design and test the automated retrieval engine for the HERO project. The system should be able to search in multiple online medical sources, given a search query containing numerous medical terms. To attempt to improve the quality of the search results, medical experts must be able to give feedback to the system, by judging the relevance of the retrieved publications.

The system's main source of medical publications will be the *PubMed* search engine, which searches in the *MEDLINE* and *PubMed Central* databases, in addition to other databases (section 2.1.1). MEDLINE and PubMed central are large online journal citation databases, created by the *U.S. National Library of Medicine*. They contain millions of medical citations.

1.2 PROBLEM STATEMENT

Existing medical retrieval engines, such as PubMed, operate on large sets of indexed publications. For a publication to be retrieved by the system, the publication must be present in its collection. In the case of HERO, it needs to be possible to search in multiple sources. It is impossible, however, to acquire copies of the collections used by the

various sources and unfeasible to store, update, and index these collections. Therefore, a method needs to be devised in which the collections can be searched, without requiring copies of the collections.

When a retrieval system utilizes the results of other search engines, it is common to *re-rank* the results. This means that the retrieved documents will be further analyzed to attempt to sort them from most relevant to least relevant. This is done using retrieval models, of which many kinds exist (section 2.2). For the HERO project it is important that the most relevant publications appear on top of the search results. Therefore, applying re-ranking may be useful, thus the top performing retrieval models for this task need to be found.

Another common approach to improve the quality of search results is to use the highest ranked documents to find additional important terms that can be added to the search query. This process is referred to as *Pseudo Relevance Feedback (PRF)* (section 2.4). This technique might be useful to increase the quality of the search query, and with that the quality of the search results. We need to determine if this approach is useful in the medical domain and in which way it can be implemented.

Since the users of the system must have the ability to give feedback to the system, it needs to be able to handle this feedback and adapt its search strategy accordingly. This concept is called *Explicit Relevance Feedback (ERF)* (section 2.4). A method needs to be found to implement ERF in this system. Furthermore, the influence this feedback has on the performance of the system needs to be measured, because it is highly undesirable to ask for feedback, when it does not influence the system's performance positively.

1.3 RESEARCH QUESTIONS

The problem statement results in four research questions which need to be answered:

REQ1 How can multiple document collections be searched effectively, without requiring copies of the collections?

REQ2 Which retrieval models are suitable to re-rank the results of biomedical search engines?

REQ3 How can pseudo relevance feedback be applied to the system and what is its influence on the quality of the search results?

REQ4 How can explicit relevance feedback be implemented and how does it affect the search result quality?

The research questions will be answered by implementing various prototypes and measuring their performance. In order to do so, a

dataset which contains a combination of medical queries and a list of relevance judgements for many publications is essential. This allows for more prototypes to be tested in a shorter time, than when medical experts were to test each prototype manually. The *Text REtrieval Conference's Clinical Decision Support* dataset (*TREC CDS*) seems to be suitable for this task (section 2.6).

The TREC CDS dataset contains relevance judgements for the PubMed Central database, which can be accessed through PubMed. Since PubMed is the most important data source for the HERO project, the prototype of the search system will be implemented specifically to search in PubMed. Other sources might be added to the system at a later point in time.

1.4 OUTLINE

The structure of this thesis is as follows. First, background terms required to understand this paper are discussed. These should give the reader a feel for the domain, existing techniques, and approaches used by other researchers. Next, setup of the experiments designed to answer the research questions are discussed. This includes details on the implementation of the prototype and existing retrieval systems. The results are presented and discussed in the following chapter, followed by the conclusion and proposed future work.

2 BACKGROUND

Some knowledge of medical libraries, retrieval techniques, the state-of-the-art in medical retrieval, and the HERO project is required to comprehend the further chapters of this thesis.

This chapter starts with a description of the most important library of medical publications, the way in which this library can be searched, and related software tools.

Next, information retrieval models are discussed, including evaluation metrics, ways in which user feedback can be taken into account, and ways to combine retrieval models.

After this, the state-of-the in medical retrieval is presented using the Text REtrieval Conference on Clinical Decision Support as a guideline.

Finally, the HERO project and its requirements are discussed.

2.1 MEDICAL

2.1.1 *Journal citation databases*

MEDLINE (MEDLARS Online) is the computerized journal citation database of the *U.S. National Library of Medicine (NLM)* [10]. It was introduced in 1971 and allows librarians to find bibliographic data on medical publications, using both regular keywords and a controlled vocabulary called *MeSH* (section 2.1.2). To make the latter possible, MeSH terms have been added to each citation by human annotators, which describe a publication's topics.

MEDLINE
NLM

The MEDLINE database can be searched using a free search engine called *PubMed*, which was introduced by the NLM in 1996[1]. In addition to the MEDLINE database, PubMed can search in several other databases.

PubMed

PubMed search queries can contain both free text and MeSH terms, all of which can be joined using *AND* and *OR* statements. If no statements were added, PubMed assumes that all terms need to be present in the retrieved documents, thus creating an AND-query. Furthermore, it tries to recognize the MeSH terms in queries and represent these as both MeSH and free text in the generated search queries.

Query construction

PubMed is accessible through a website, and through an API for automated requests. An example of the results produced by PubMed when using the website is shown in figure 1.

Interfaces

1 https://www.nlm.nih.gov/pubs/factsheets/dif_med_pub.html

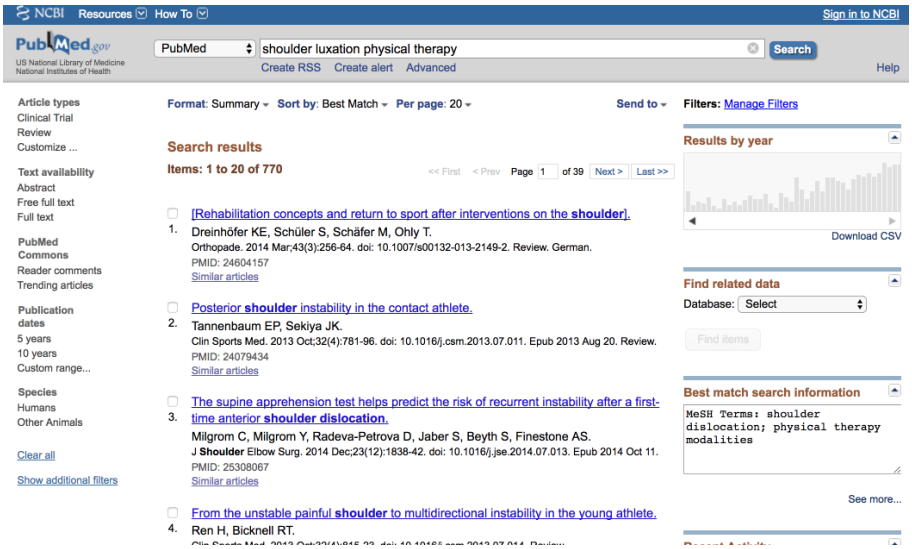

Figure 1: Example of query on PubMed

PubMed Central

Another database present in PubMed is *PubMed Central (PMC)*, which digitally archives full-text biomedical and life sciences publications. The full articles in the PMC database can be accessed free of charge[2]. This in contrast to MEDLINE, where only a fraction of the publications is free.

2.1.2 *Controlled medical vocabulary*

MeSH

Medical Subject Headings (MeSH) are used to describe and search for publications in the biomedical domain [10]. Each publication in MEDLINE typically contains ten to fifteen MeSH terms as keywords. MeSH is a controlled vocabulary and has a hierarchical structure. As a result of this structure, MeSH terms can be expanded to yield the terms of more specific subjects. For example, the general term of a disease will yield the variants of the disease after expansion.

UMLS

MeSH is included in the *Unified Medical Language System (UMLS)*, which is a collection of biomedical vocabularies and an ontology of those vocabularies' terms [7]. It holds medical concepts, names used for those concepts and the relationships between concepts. In November 2016 the repository contained over 3.4 millions concepts and 13.4 million names[3].

The UMLS Metathesaurus was developed to make searching for machine-readable information more feasible, by creating a standard format and describing medical concepts, their aliases and the connections between concepts. Terms used in biomedical literature are stored in MeSH, which was integrated in UMLS.

2 https://www.ncbi.nlm.nih.gov/pmc/about/faq/
3 https://www.nlm.nih.gov/research/umls/knowledge_sources/metathesaurus/release/statistics.html

2.1.3 *Related tools*

Numerous tools have been developed which can be used to extract medical concepts, in UMLS or MeSH format, from publications. These tools may assist while indexing a publication, or finding one. In this research, the tools may aid in the construction of search queries.

MetaMap

One tool to extract UMLS concepts from texts is called *MetaMap*. It combines natural language processing techniques with algorithms to find the most likely concepts given a text [3].

MedTagger

Another tool designed for this purpose is *MedTagger*. Similarly to MetaMap, it extracts UMLS concepts from texts, such as publications. It does this by using rules and patterns, and machine learning techniques [28]. Earlier research demonstrated a 94% accuracy and 77% recall when applying MedTagger [28].

MTI

A tool designed to index medical publications is the *NLM Medical Text Indexer (MTI)*. It suggests relevant UMLS terms using a complex processing flow, which can utilize both MetaMap and related citations[4]. It yields a list containing UMLS concepts and confidence scores, in addition to several other attributes.

The output of MTI can be restricted to MeSH terms, which makes it possible to transform a medical search query written in a natural language to one using MeSH terms.

2.2 RETRIEVAL MODELS

Retrieval models are used to find all relevant documents in a collection given a query. Additionally, the search results may be ranked, to make the most relevant results appear on top of the list. In this research, retrieval models are utilized to rank the search results, since it is important that highly relevant publications are presented to the user first. Furthermore, the models are required to incorporate user feedback in the system.

Boolean retrieval

The classical retrieval model is *Boolean retrieval*, in which documents are searched that contain the terms listed in the query. The query can contain multiple AND or OR statements, placed between the other tokens, which form a boolean expression of the query terms.

When using either AND or OR queries, either all terms from the query most be present in the publication (AND), or at least one term (OR).

A possible downside of the Boolean retrieval is the strict way in which queries are processed. If just one term of an AND-query is not present in a document, the document is not retrieved by the model. In many cases, this behaviour is undesirable, since users are often

4 https://ii.nlm.nih.gov/resource/Medical_Text_Indexer_Processing_Flow.pdf

interested in the most relevant documents, even if they do not contain all query terms.

Vector Space Model

One way to solve this problem is using a *Vector Space Model (VSM)*. In this model, documents and queries are represented as vectors, consisting of terms and their corresponding weights. The weights describe the importance of terms in documents or queries.

This representation makes it possible to compare a query and a document by calculating the angle between their vectors. The smaller the angle, the more likely it is for a document to be relevant.

In contrast to Boolean retrieval models, not all terms of a query need to be present in a document for the document to match.

TFIDF

The best known vector space model is called *TFIDF*. Many variants of this model exist, but it is usually given as the product of the *term frequency (TF)* and the *inverse document frequency (IDF)*.

TF

The term frequency represents the number of times a term occurs in a document. Often, this value is smoothed and scaled with a logarithmic function or square root, since it is unlikely for a term to be twice as important, when it occurs twice as much in a document.

IDF

The inverse document frequency represents the number of documents in a collection that contain the term. The greater the number of documents that have the term, the smaller the IDF. Usually, the IDF is smoothed and logarithmically scaled as well, for similar reasons.

The TF and IDF values try to find a balance between the popularity of a term in a text and in the entire collection, to estimate the importance of the term in a particular document.

Apache Lucene

Two retrieval models used in this research were derived from implementations by *Apache Lucene*[5], which provides open-source search and indexing software.

Lucene Practical Scoring Function

The TFIDF implementation used in this project is based on the *Lucene Practical Scoring Function*. This scoring function is a twist on TFIDF with the ability to boost term scores, a coordination factor that gives a higher bonus when more terms of the query are present in a document, and a normalization factor which attempts to make it possible to compare the scores between queries.

When we are referring to TFIDF in this research, we are actually referring to the Lucene Practical Scoring Function.

The TFIDF term score for a document is calculated using the following equation:

$$\begin{aligned} \mathrm{score}(t, d) &= \mathrm{tf} \cdot \mathrm{idf} \cdot \mathrm{boost}(t) \\ &= \sqrt{\mathrm{freq}_d(t)} \cdot (1 + \log(\frac{|\mathrm{collection}|}{\mathrm{freq}_c(t) + 1}))^2 \cdot \mathrm{boost}(t) \end{aligned} \quad (1)$$

5 https://lucene.apache.org/

Here $freq_d$ and $freq_c$ represent the term frequency in the document and document collection respectively. The default boost value is equal to *1*, and thus does not influence the term score.

The query score for a document can be calculated by adding the term scores and multiplying the result with the coordination factor and norm:

$$\begin{aligned} score(q, d) &= coord \cdot norm \cdot \sum_{t \in q} score(t, d) \\ &= \frac{|\{t|t \in q, t \in d\}|}{|q|} \cdot \frac{1}{\sqrt{|d|}} \cdot \sum_{t \in q} score(t, d) \end{aligned} \tag{2}$$

Another well known retrieval model is *Okapi BM25*. It behaves largely similar to TFIDF, and utilizes the TF and IDF as well, but has a probabilistic foundation and has field-length normalization built into it. *Okapi BM25*

As with TFIDF, the BM25 implementation used by the system was derived from Lucene's implementation. The BM25 term score for a document *d* is given by the following formula:

$$\begin{aligned} score(t, d) &= boost(t) \cdot idf \cdot \frac{(k_1 + 1) \cdot tf}{tf + k_1 \cdot (1.0 - b + b \cdot \frac{|d|}{avg_{dl}})} \\ &= boost(t) \cdot idf \cdot \frac{(k_1 + 1) \cdot freq_d(t)}{freq_d(t) + k_1 \cdot (1.0 - b + b \cdot \frac{|d|}{avg_{dl}})} \end{aligned} \tag{3}$$

The IDF is the same as for the TFIDF implementation. The average document length is denoted with avg_{dl}.

Parameter k_1 controls the saturation of the term frequency and is usually set to *1.2*, this research included. Higher values for k_1 result in a slower saturation of TF, meaning that a term needs to be present in a document more often for the TF to approach its upper bound. k_1

The b parameter, which is usually set to *0.75* affects the field-length normalization. When b is set to zero field-length normalization is completely disabled. When set to one it is fully applied. b

The BM25 score for a query-document pair is attained by adding the BM25 scores for all term-document pairs:

$$score(q, d) = \sum_{t \in q} score(t, d) \tag{4}$$

Several variants of BM25 exist, such as *BM25F*, which is capable of searching in multiple fields of structured documents. When operating in this mode, weights can be given to each field to denote it's importance when being searched.

2.3 EVALUATION METRICS

There are several approaches to evaluate the performance of a search engine. In general, a distinction can be made between metrics that operate on *unordered* results, and metrics that take the *rank* of results into account.

Unordered

Two simple yet important metrics that operate on unordered sets are *precision* and *recall*.

Precision

Precision is defined as the fraction of relevant documents in the set of all documents that were retrieved:

$$\text{precision} = \frac{|\text{relevant} \cap \text{retrieved}|}{|\text{retrieved}|} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{5}$$

In the above equation *TP* stands for *true positive*, meaning the number of retrieved documents that were in fact relevant. The number of documents that were retrieved but were mistaken to be relevant are denoted with *FP*, or *false positive*.

Recall

Recall is a measure for the fraction of relevant documents in a collection that were retrieved by the system:

$$\text{recall} = \frac{|\text{relevant} \cap \text{retrieved}|}{|\text{relevant}|} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{6}$$

With *FN* being the number of *false negatives*, or the number of documents that were mistakenly judged to be non-relevant.

Accuracy

In natural languages the terms *precision* and *accuracy* often refer to the same concept. In the field of information retrieval, however, both terms point to different concepts. Whereas precision points to a search engine's capability to correctly classify relevant documents, accuracy also takes its ability to correctly classify non-relevant documents into account, resulting into the following equation:

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{7}$$

The new term in this equation is *TN*, which denotes the number of non-relevant documents which were successfully classified as being non-relevant.

In contrast to precision, accuracy is seldom used to evaluate search engines.

Ordered

Often it is desirable to take the order of search results into account. After all, users of search engines expect the results appearing on top to be more relevant than those at the bottom. Since precision and recall do not take the order of results into account, other metrics are needed.

Average Precision

Average Precision (AP) is one of those metrics. It is often calculated

by averaging the precision attained at the rank of each relevant document in a result set. Therefore, the more relevant documents appear on top of the list, the higher the average precision will be. When the relevant documents appear at the bottom of the list, the precision at each point will be low, thus the AP will also be low.

Mean Average Precision

The average precision applies to a single query. For the performance of a search engine on multiple queries the *Mean Average Precision (MAP)* is used. The MAP is determined by calculating the average AP on a group of queries. It is a widely used metric to evaluate search engines.

Discounted Cumulative Gain

Some documents can be more relevant than other relevant ones. While MAP includes the ranking of results in its evaluation, it does not account for relevance grades. The *Discounted Cumulative Gain (DCG)* takes both rank and relevance into account and is defined by the following formula:

$$DCG_p = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)} \tag{8}$$

For each retrieved document, it divides its relevance score by the logarithm of its rank. Therefore, documents with a higher relevance score result in a higher DCG. Furthermore, the lower the document is ranked, the smaller its influence on the DCG is, thus encouraging the most relevant documents to be placed on top.

The scores for all documents up to rank p are summed to yield the DCG_p. In this research, p-values of 100 and 1000 are used.

Normalized DCG

The attainable DCG depends on the grades used to judge the relevance of documents, making it hard to compare DCG scores. Therefore, the *Normalized Discounted Cumulative Gain (NDCG)* was introduced:

$$NDCG_p = \frac{DCG_p}{IDCG_p} \tag{9}$$

The NDCG is the fraction of the actual DCG and the DCG of a perfect ranking, thus normalizing the the score to a value between zero and one and making the scores comparable.

Inferred NDCG

Unfortunately, it is often not feasible or possible to attain a complete set of relevance judgements. A method was introduced by Yilmaz, Kanoulas, and Aslam to overcome this problem [32]. Their approach uses sampling to estimate the NDCG using incomplete relevance judgements. Their NDCG estimation is referred to as the *inferred NDCG*, or *infNDCG*. This metric was used to determine the performance of the TREC CDS participants.

2.4 RELEVANCE FEEDBACK

Relevance feedback incorporates techniques which take relevance judgements into account when performing a search query. This can be done by changing the terms of a query, or by changing the term weights.

Query expansion

The addition of new terms to a query is referred to as *query expansion*. Query expansion takes place when new terms have been found which are believed to be relevant for the retrieval task. These terms could be coming from, for instance, a thesaurus, or are present in the most relevant documents retrieved by a system.

Rocchio

A popular algorithm for handling relevance feedback, which can expand search queries and update term weights, is the *Rocchio* algorithm [25]. It uses a *vector space model* to adjust the weights of existing terms in a query and add new weighted terms, based on the query, the class of relevant documents, and the class of non-relevant documents.

To update a VSM query using Rocchio, the algorithm first calculates the centroids of the document vectors of each class. In other words, it determines which words are most important to those documents. Next, the centroids are multiplied with a class weight and are added to, or subtracted from, the original query. Terms with negative weights are removed from the query vector.

Explicit Relevance Feedback

There are two ways to determine the relevance of documents and apply feedback. The most straightforward one is *Explicit Relevance Feedback (ERF)*. With ERF a human judges the relevance of search results, either graded or binary. These judgments may be utilized to build a user profile describing his preferences. By taking this profile into account on subsequent search queries, the quality of the search results may be improved.

Pseudo Relevance Feedback

The second way is *Pseudo Relevance Feedback (PRF)*. With PRF, the top n documents of a search result are assumed to be relevant and utilized to modify the search query [17]. It applies a term-scoring function, using a retrieval model, to extract the top k most informative terms of the top documents. These terms are then added to the original query and the query is ran again.

In a medical setting, tools like MedTagger and MetaMap may be applied to find medical terms in the retrieved documents to expand the query. This can be done before the system handles explicit relevance feedback in order to reach a certain baseline performance.

2.5 COMBINING MODELS

Most retrieval models have strengths, weaknesses and are unlikely to result in a perfect document rankings on their own. These models can be combined to increase a search engine's overall performance. There

are numerous ways to do this, all of which operate on the rankings or scores produced by a model.

Score summation

A straightforward approach, in which the rankings themselves are not used, is to add all scores produced by the models for each document. The new ranking can then be constructed by sorting the documents an their combined score.

This method assumes, however, that the model scores have equal domains and importance, which does not need to be the case. Some models will yield scores between, for instance, zero and one, while others give scores greater than one hundred. To address this issue, the scores can be normalized for each model before summing them, thus making them operate on the same domain. When it is known which models perform best, the method can be improved further by multiplying the scores of each model with weights based on a model's performance. This results in the following document score:

Score normalization

Model weights

$$score(d) = \sum_{m \in M} weight_m \cdot \|score_m(d)\| \tag{10}$$

Another approach is *Borda-fuse*, which is based on *Borda Count* [4]. It is a voting system in which the document score depends on its rank. The top document receives a score equal to the number of documents returned by the models. At each lower rank, the score is reduced by one, to create a list of descending scores. Next, the scores are added and the documents sorted on this score.

Borda-fuse

Again, weights can be given to the models to increase the importance of better performing models. This method is referred to as *Weighted Borda-fuse*, and requires some knowledge of a model's performance [4, 13]. The document scores can be calculated as following:

Weighted Borda-fuse

$$score(d) = \sum_{m \in M} weight_m \cdot (|results| - rank_m(d)) \tag{11}$$

An advanced approach to combine results is *learning-to-rank*, which is based on *Machine Learning* [18]. Multiple variants exist, but generally it works by applying machine learning algorithms to numerous features related to a document. These features may be, for instance, scores generated by several retrieval models, the length of a document, or its publication date. The system needs to be trained to learn it how to reduce the error of the resulting rankings.

Learning to rank

2.6 TREC CDS

The *Text REtrieval Conference (TREC)* is a series of workshops organized by the *National Institute of Standard and Technology (NIST)*

TREC

to encourage research in text-based information retrieval for large datasets. Typically, a document collection, test queries and corresponding relevance judgements are supplied for each workshop, or track.

CDS

For the 2015 TREC on *Clinical Decision Support (CDS)*, participants had to retrieve medical publications. Given a short Electronic Health Record of a patient an offline PubMed Central dataset had to be searched [23]. Depending on the task, relevant publications helped diagnosing, testing, or treating a disease. The input for task A are a short description of a patient and investigation thereof, and a summary of the description. Task B uses the same input in addition to a concrete diagnosis. An example of an EHR for task B can be found in listing 1 of section 3.2.

infNDCG

The dataset was labeled on relevance by experts to make evaluation possible using the standard TREC evaluation approach for ad-hoc retrieval tasks, currently being the inferred NDCG metric (section 2.3). In this metric, the Normalized Discounted Cumulative Gain is estimated using a dataset which has not been completely judged on relevance [32].

There are three general techniques used by many TREC CDS participants: initial query expansion, pseudo relevance feedback, and applying several ranking algorithms.

Query expansion

With initial query expansion the search query was expanded before using it to search PubMed. Some teams added terms related to the MeSH terms in the query [1, 21], optionally with weighted terms. Others extracted MeSH terms from the top Google search results on the query and added these to the query [5, 27], or did the same using PubMed instead [33]. One group utilized DBpedia to expand queries [1]. DBpedia is a knowledge base containing structured machine readable information extracted from Wikipedia [6]. In addition to other information, it contains facts, relationships and hierarchy, and can be queried semantically.

Term extraction

To find medical terms in, for instance, the Google search results, some teams used tools like MedTagger and MetaMap to extract such terms. These tools may be used for the pseudo relevance feedback phase as well, in which the teams used the top k documents retrieved by PubMed to modify the query [5, 11, 21, 33]. Aside from adding or removing terms, there were teams who also added weights to the query terms.

In the final step, the teams ranked the retrieved documents. For this tasks, the teams used numerous techniques such as language models, the *Kullback-Leibler divergence* between such models, and machine learning algorithms.

The language and retrieval models used include BM25 [21, 24, 27], BO1 [2, 21], Markov Random Fields [5], TF-IDF [33], SPUD [11, 12], Hiemstra LM [1, 14], and LGD [1].

Examples of machine learning approaches include pointwise based random forest and pairwise based SVM learning-to-rank algorithms [27]. In these models the weighting scores of document-query pairs from retrieval models were used as features.

Top participants

The top participants of task A used either Markov Random Fields, DBPedia, or the SPUD language model to solve the task, with infNDCG scores ranging from 0.2823 to 0.2939 in the top three [23]. In task B the best scoring teams utilized an ensemble of information retrieval models, or concept-based search using MetaMap to extract medical terms. The infNDCG scores for the three best scoring approaches ranged from 0.3611 to 0.3821. The fact that the scores for task B are higher, could mean that the addition of a diagnosis in this task was beneficial for the search proces.

2.7 MEDICAL RETRIEVAL ADVANCEMENTS

While TREC shows a large fraction of the advancements in text retrieval, several other works are relevant to this research as well and are described in this section.

Text-based vs. concept-based

Trieschnigg compared the use of text-based search techniques and concept-based ones (MeSH and UMLS$_{++}$) for the use of biomedical information retrieval [29]. He concludes that on its own concept-based retrieval is outperformed by text-based retrieval, since it is unable to fully represent the information needs. However, combining concept-based retrieval with text-based retrieval improved performance substantially.

One of the problems with standalone concept-search is that biomedical papers often do not contain an exhaustive list of relevant MeSH terms, resulting in papers that are more difficult or impossible to find using concept-based retrieval.

Chang et al. compared the effectiveness of queries on PubMed using MeSH terms and text-words [9]. They conclude that the use of MeSH terms leads to a higher precision, but a lower recall than text word searches. By combining MeSH terms and text words the high precision of MeSH terms can be attained, while the recall is improved.

Jenuwine and Floyd also compared the use of MeSH and text-based queries on PubMed, using the MEDLINE database [15]. Their conclusions are the same as those from Chang et al. MeSH terms are precise but may miss relevant results, whereas text words result in a greater number of relevant results, but with lower precision. They too suggest the combination of both search strategies.

PubMed's term mapping

Richter and Austin explored methods to search on MEDLINE using MeSH and text words, for Evidence-Based Practice in Physical Therapy [22]. They entered queries into PubMed and analyzed

whether PubMed had been able to correctly translate or map the query terms to MeSH terms, which it does automatically (Automatic Term Mapping). Nearly half of the search terms were correctly mapped to MeSH terms. In 15% of the cases PubMed could be used to find the correct MeSH terms with manual interaction. For the remainder appropriate MeSH terms could not be found. Richter and Austin recommend that users of PubMed check to see whether the automated mapping to MeSH was successful and enhance their query if needed, since the automatic translation of PubMed is not perfect.

2.8 HERO

The aim of the HERO project is to generate customized physical therapy schemas for patients suffering from multiple diseases. An important element in this project is the automatic retrieval of relevant medical publications, given a search query. The queries HERO uses remain constant and are used over a long period of time, but may be modified by applying user feedback. The system needs to be able to search in multiple online sources, of which PubMed is the most important one. Medical professionals using the system should be able to give feedback to the system, which must be used to improve the system's performance. The precision of the system should be as high as possible, since reading medical publications is tedious work and the time the users can spend on this task is limited.

3 EXPERIMENT SETUP

3.1 GOAL EN GLOBAL SETUP

In this chapter the experiments needed to answer the research questions from section 1.3 are discussed. A prototype must be built to conduct the experiments. In addition, the prototype will need to contain the functionalities needed for the search engine to be usable in the HERO project. By measuring the performance of the system for all its parts, models, and settings, the research questions can be answered.

Firstly, the system needs to have the possibility to run search queries on existing search engines. The results of those queries need to be stored and the related publications downloaded. The publications can then be analyzed to be able to re-rank them, such that the most relevant publications end up higher in the search results than less relevant ones. Secondly, pseudo relevance feedback needs to be implemented, to attempt to increase the quality of the results. Finally, explicit relevance feedback has to be applied to make sure that the performance of the search engine keeps increasing, given a static query.

The entire retrieval flow to test the system using the TREC CDS dataset will be the following one:

1. Join topic summary and diagnosis
2. Extract MeSH terms with MTI
3. Filter terms and reduce the number of terms
4. Create queries for PubMed and the retrieval models
5. Run the query at PubMed and store the first 3000 results that are in the TREC CDS PMC collection
6. Apply pseudo relevance feedback for PubMed and the retrieval models on the top 10 results
7. Run the updated query on PubMed and add the top 1000 new results to the stored publications
8. Re-rank the stored publications using TFIDF, BM25, or a combination of both
9. Present the top 10 publications to the user and remove these from the stored publications

10. Receive and apply relevance feedback
11. Return to step 8.

When the system is used in the actual HERO project the input query already consists of MeSH terms, allowing the first three steps to be skipped. Additionally, the system will not be limited to the PMC collection in step 5 while being used within HERO.

3.2 DATA AND SEARCH ENGINE

TREC CDS Topics

The data of the 2014 and 2015 editions of TREC CDS (section 2.6) willl be used to evaluate the system. Together, the editions contain 60 topics from numerous medical domains. Each topic consists of a patient description, a summary of this description, and in 10 cases a concrete diagnosis as wel. The challenge of TREC CDS is to find publications to diagnose, test, or treat a patient. In practice, there is little need to adapt search engines for these tasks [5]. Therefore, no differentiation will be made between between the type of task in this research.

Diagnosis

Contrary to a topic's task, the presence of a diagnosis is influential for the difficulty of a topic [11]. However, to reduce the complexity of the system the decision was made to join the patient summary and diagnosis together, thus treating these as a single field. Since the full patient description adds little value to the system [11], only the joined field will be used in this prototype.

Listing 1 shows an example of a 2015 TREC CDS topic used for one of the tasks. The example clearly shows the big difference between the description field and the other two fields, in terms of length and information density. This example is only representative for TREC CDS, since the queries used in the HERO project will already be in the MeSH format.

Listing 1: Example of a 2015b TREC CDS topic

```
<topic number="27" type="treatment">
  <description>A 15 yo girl accompanied by her mother is
      referred for evaluation by the school. The girl has more
      than expected absences in the last three month, appears
      to be constantly tired and sleepy in class. Her mother
      assures the girl is well fed, and getting the proper
      sleep at night but admits the girls tires easily when
      they go out on weekend hikes. Physical examination: BP:
      90/60. HR 130/min the only remarkable findings are
      extremely pale skin and mucosae. Grade 3/6 systolic
      murmur. Lab tests report Hb: 4.2 g/dL, MCV 61.8 fL, serum
       iron &lt; 1.8 umol/L and ferritin of 2 ng/mL. Fecal
      occult blood is negative.</description>
  <summary>A 15 yo girl with fatigue, pale skin, low hemoglobin
       and ferritin.</summary>
```

```
  <diagnosis>Iron-Deficiency Anemia</diagnosis>
</topic>
```

Fold creation

Since it would not be fair if more diagnoses were used in one experiment than in another, the topics are divided over five stratified folds. This evenly distributes the diagnoses (and topic tasks) over the five folds. For each task a block of ten topics is present in the TREC dataset. The ten topics containing a diagnosis form a block as well. Since the topics within a block of ten topics have a random order and subject, it is a trivial task to generate the folds: the folds in which a topic belongs is equal to the topic id modulo 5.

PubMed Central

The document collection used in TREC CDS is the PubMed Central database, which is a subset of the entire PubMed collection. The latter is the most important source for the HERO project. Since PMC is a subset of PubMed, the system can be built such that all search queries are passed through PubMed. Next, the publications returned by PubMed that are not available in the PMC collection can be filtered, to make the results usable for TREC CS. That way, the system can be used in both TREC CDS and practice.

The disadvantage of this approach, however, is that publications can only be accessed using the PubMed search engine, whereas TREC CDS participants can index and apply retrieval models on the entire PMC dataset. Therefore, we need to make sure that as many relevant documents as possible are retrieved from PubMed for a query, since it is highly influential on the search quality.

Entrez

The National Center for Biotechnology Information (NCBI) offers an API, called Entrez, through which PubMed can be queried. In addition to searching, the API enables its users to retrieve meta data, download publications, and correct the spelling of search queries. Since no spelling errors are expected in the search queries used, the prototype online uses the search and download functionalities. The other functions are implemented by the prototype though, for possible future extensions.

The Entrez API allows for 100,000 search results to be returned for a single query call. Since there are roughly 1.25 million documents in the PMC collection, only a fraction of the collection will be relevant for a topic, and since the amount of data processed increases fast with the number of search results, only the top 20,000 results are used per query. After filtering out publications that are not present in the PMC collection, the top 3000 publications are stored for re-ranking.

MeSH recognition

PubMed is capable of recognizing all MeSH terms in a search query through its website and API and using these in a special way. Each MeSH term is added to the query as both a medical concept and normal text. This helps PubMed in determining the ranking of the search results. It is possible to run both boolean *AND* and *OR* on PubMed, but AND queries are the default setting. We expect that while AND queries are likely to return in a high precision, the

number of search results will be small, since each resulting publication must match all query terms. To verify this hypothesis, both AND and OR queries will be tested during the experiments.

3.3 QUERY CONSTRUCTION

PubMed

The search queries used in the HERO project will consist of several MeSH terms, which can be immediately used in PubMed search queries. For the TREC CDS topics, however, it is necessary to extract the important terms from the patient descriptions first, because otherwise the search queries would become too long to use and contain too many non-relevant terms, to receive useful results from PubMed. In the case of AND queries it is expected that long queries will not yield any results as all, whereas the recall for long OR queries is expected to be low, since only the first 20.000 PubMed search results are used by the prototype.

Extract important terms

Numerous methods exist to extract important terms from documents, such as the TFIDF scores of the terms, or a medical thesaurus. The latter exists, in the shape of MeSH and UMLS corpora, which contain lists of medical terms, their relation, and the preferred name for the concept. Such a list could be used to recognize medical concepts, extend queries with related medical terms, or replace terms with a better term for the same concept. However, it has proven to be difficult to attain and index a complete corpus of medical terms. Therefore, a different approach was chosen,

MTI

namely the use of the NLM Medical Text Indexer (MTI, section 2.1.3) to extract medical terms.

Given a publication, MTI is capable of extracting the most important MeSH concepts from the text. To do so, MTI looks at both the publication itself, related publications on PubMed, as well as a MeSH thesaurus. Since MTI also utilizes related publications in its process, MTI is applying a form of pseudo relevance feedback.

Since MTI is used to index medical publications, it expects an entire publication as its input. The system can then make a distinction between the title, abstract, and body in its process. In this case, however, the prototype should not extract terms from a publication, but from the TREC CDS topics, containing patient descriptions. To solve this problem, synthetic publications are generated, in which the entire patient summary and diagnosis are placed in a single section, the title field, of the synthetic publication. That way, MTI cannot add more weight to either the summary or the diagnosis.

We expect MTI to perform slightly better when the diagnosis is used as the title of the synthetic publication, with the summary used as the abstract, since the information density of both titles and diagnoses is generally high. However, since the choice was made to

join the summary and diagnosis together, this hypothesis will not be tested in this research.

Query length reduction

Since it is possible for MTI to return a greater number of MeSH terms than usable on PubMed, the maximum number of terms to be included in a search query needs to be found. In order to do so, the number of MeSH terms used in the PubMed query will be varied between 1 and 9, the latter included. This range was chosen arbitrarily, but may show the area in which the ideal value can be found.

To determine which concepts will make up a query while reducing the query length, the scores given to concepts by MTI will be used. These scores represent the rank of a concept within a publication. A preliminary investigation has shown that MTI may yield generic MeSH terms, such as *Human*, *Male*, and *Female*. Since it is unlikely for these terms to have a significant positive impact on the system's performance, during half of the experiments these common terms will be filtered from the query. Thus, the number of MeSH terms will be varied during the experiments, as well as filtering common MeSH terms.

The remaining terms form the basis of the search query and will be used in multiple query variants in the remaining parts of the system. Since only MeSH terms will be present in the search query, a form of *concept-based search* is used.

This approach has a few possible downsides, however. Firstly, the accuracy of MTI is unknown and it may yield medical concepts which are not relevant to the query. In addition, because only these MeSH terms are used, the original patient summary and diagnosis themselves are not used, which may affect the system's performance. The impact this omission has should be measured in future research.

A further disadvantage of MTI is that it can take up to 45 seconds for the process to be completed. For the HERO system this does not have to be a problem, however, since it will use a limited number of queries which only need to be processed by MTI once. Yet, for the evaluation of the prototype using TREC topics this may be problematic, since many experiments will need to be ran. Therefore, the MTI results will be stored in a cache, resulting in only a one time required MTI extraction per topic, even if the same topic is tested multiple times.

When processing the example topic with MTI it yields the terms displayed in listing 2.

Listing 2: MeSH terms collected from example using MTI, without filtering common terms

```
[ Humans, Female, Ferritins, Anemia, Iron-Deficiency, Pallor,
Iron, Hematologic Diseases, Fatigue, Hemoglobins ]
```

Query generation

The base query consists of the MeSH terms that were retrieved from the patient descriptions using MTI. This query cannot be used

straightaway in all parts of the retrieval system, however. The reason for this is that each part expects a different kind of query. For example, plain and unstemmed MeSH terms are needed to search on PubMed, but retrieval models such as TFIDF and BM25 expect stemmed unigrams and a corresponding list of term weights. These weights can then be updated by the relevance feedback models, to incorporate the feedback in the retrieval flow.

The query to search on PubMed can be easily derived from the base query by placing *AND* or *OR* tokens between the MeSH terms, thus describing the query type. PubMed can recognize the MeSH terms in this type of query flawlessly and use these to create the query that PubMed itself will use to search through its collection.

The listings 3 and 4 show how the example base query can be transformed to make it suitable for search on PubMed.

Listing 3: PubMed AND-query

```
Humans AND Female AND Ferritins AND  Anemia AND Iron-Deficiency
AND Pallor AND Iron AND Hematologic Diseases AND Fatigue AND
    Hemoglobins
```

Listing 4: PubMed OR-query

```
Humans OR Female OR Ferritins OR  Anemia OR Iron-Deficiency
OR Pallor OR Iron OR Hematologic Diseases OR Fatigue OR
    Hemoglobins
```

The re-ranking and relevance feedback models require a different query format. Firstly, these models operate on stemmed instead of unstemmed words. Secondly, these models only work on unigrams, whereas the PubMed query only contains MeSH terms, which may consist of multiple unigrams. Thirdly, weights are needed indicating the importance of a term in a query, since one term may be more important than another. The models use these weights to have more control when selecting relevant documents than a boolean retrieval model would have. When using a query with term weights, relevance feedback can be implemented as a modification of these weights, possibly adding or removing terms from the query in the process.

Vector Space Model

This form of representation, in which the query consists of a series of terms with corresponding weights, is called the *Vector Space Model* (section 2.2). Listing 5 demonstrates what our example query looks like, if doubled TFIDF values are used as weight for the stemmed unigrams. The reason that the TFIDF values are doubled is to prevent relevance feedback from modifying the original query too fast. After all, the original query remains important and sudden *query drifting*, in which the query moves in the wrong direction, needs to be prevented.

Listing 5: Representation of the example query in VSM with TFIDF-based weights

```
[ ('footbal', 99.17), ('pallor', 93.89), ('ferritin', 74.03), ('
    athlet', 68.02), ('fatigu', 39.41), ('muscl', 17.66), ('run',
     14.96), ('femal', 11.61), ('human', 6.18), ('perform', 3.69)
     ]
```

3.4 PUBLICATION RETRIEVAL AND STORAGE

Search engine

Usually existing search engine software is used in this types of research, such as *ElasticSearch* or *Apache Solr* (both based on *Apache Lucene*), or *Terrier*. The software indexes the documents and offers various retrieval models and methods to search through the indexed collection. This approach was used by many participants of TREC CDS as well. They used these existing search engines, but built additional models and algorithms on top, which implemented the ideas they wanted to test. The advantage of this method is that you start with a high quality and fast search engine, which contains a large functionality out of the box.

Nonetheless, in this research our own implementations of retrieval models and document storage engines will be used. The reason for this is twofold. Firstly, the existing retrieval software did not run properly on the hardware that was made available for this research. Secondly, implementing the prototype from scratch gives maximum freedom in the implementation of the system. Each parameter can be tweaked to desire in an attempt to raise the system's performance.

The major disadvantage of the chosen approach is that the prototype development has cost vast amounts of time, which were not spent on the optimization of the models and algorithms, but on the creation of a base system. In addition, our own software is much slower than existing retrieval software, which has the benefit of being optimized over the course of many years. Finally, our own software did not enable us to index the entire PMC collections, worsening the results on the TREC CDS dataset. On the other hand, this gives a more realistic view of the system's performance, since it would not have contained complete and indexed collections anyway, when used in the HERO system. The reason being that the system needs to be able to search multiple sources, and therefore utilizes existing medical search engines as a gateway to the publications.

Python
Klepto cache

The entire prototype was built in *Python* (version 3.6) and heavily relies on Klepto [19] for the persistent storage of data. All MTI results, PubMed search results and retrieved publications are stored in a Klepto cache for simple and fast access to the data on repeated runs of a query. We expect the software to be much faster than it would have been without caching on repeated queries. To determine whether this is actually the case, and measure the change in speed, the prototype will be tested with both caching enabled and disabled, and the time needed to complete the three tasks measured.

All the publications that are retrieved from PubMed need to be transformed to a format that can be used by the system and its model. In the case of the re-ranking and relevance feedback models, TFIDF and BM25, only the term frequencies are needed. Since the IDF values are calculated on the entire PMC dataset, the system could just store the term frequencies of each document and remove all other features. However, to make it possible to add more advanced features in the future, but still be able to use the existing caches, the raw texts and the ordered lists of tokens are stored as well.

Before this can be done several steps need to be followed. First, the publication needs to be downloaded, which is retrieved in the XML format. Next, the title, abstract, and main text (body) are extracted from the file. In addition, information such as the publication date, the journal in which the article was published and a list of authors are extracted. All information is stored structurally in a Python class.

Normalization

The title, abstract and body now need to be normalized. In this process the special characters are removed from the texts and texts are split into unigrams (tokenization). The procedure for normalization and tokenization is the one described by Jiang en Zhai [16], which was specifically created for medical publications. Their research showed that basic stemming has a positive impact on the performance of medical retrieval systems and is therefore implemented as well, using a SnowballStemmer. However, while the relevance and retrieval models operate on stemmed unigrams, the PubMed queries need to be performed using unstemmed terms. Therefore, both the stemmed and unstemmed variations of the normalized texts are stored.

To increase the speed of the evaluation process, the full PMC dataset used in TREC CDS will be inserted into the system, after which all documents will be normalized and stored in the persistent Klepto cache. Because of this, all publications needed for the TREC CDS tests can be pulled from the cache, and do not need to be downloaded and normalized again. However, it is still possible for the system to retrieve new documents, since documents are not required to be present in the cache.

3.5 INITIAL RETRIEVAL AND PUBMED PRF

In the previous sections we have described the various ways in which queries are generated from the TREC CDS topics, and in what way the publications are retrieved, processed, and stored. In this section, the process is described in which PubMed is searched and pseudo relevance feedback (section 2.4) is applied for a second search round on PubMed.

After the generation of queries, an AND or OR type query (listings 3 and 4) is send to PubMed to retrieve the top 20.000 most relevant

documents, according to PubMed. The type of query - AND or OR - depends on the retrieval configuration that is being tested. Both types will be tested for all other combinations of settings and models.

The result list is then filtered to remove any publications not present in the TREC CDS dataset. The year in with the TREC topic was used is required in this step, since the PMC collection that was used in the 2014 edition is smaller than in the 2015 edition. The top 3000 remaining publications are retrieved - either from PubMed or the cache - and stored in memory. These publications will be used in later phases of the retrieval flow to re-rank the results. The limit of 3000 publications was chosen while conducting preliminary research, to prevent the system from becoming too slow. After all, the time needed to process the publications scales linearly with the number of publications.

The downside of the limit on the number of search results is that this is limiting the number of documents the system can possibly return, since the entire collection the system uses consists of the publications from the PubMed search results. If this set contains few relevant documents, the system will never be able to yield many relevant documents, and substantially increasing the number of search results used is not an option because it is too slow. In an attempt to mitigate this problem, pseudo relevance feedback is applied on the top 10 search results. With this step the PubMed query is expanded (query expansion, section 2.4) with the most important terms from those ten results. After this, the expanded query is also run on PubMed and at most 1000 documents, after filtering TREC documents, are added to the set containing the 3000 already stored documents. By adding just 1000 new publications to the publication set the decrease in speed stays limited, while the chance of relevant documents increases. We expect this chance to be higher than if 4000 documents had been added to the set instead of 3000, without applying this pseudo relevance feedback step.

Query expansion

Which words are most important is decided using the TFIDF and BM25 scores of the words. As with the other parameters of the system, such as filtering MeSH terms and the type of PubMed query, both TFIDF and BM25 are tested for all other configuration options in the experiments.

Token score

The TFIDF and BM25 models use unstemmed tokens from the title and abstract fields of a publication. The rationale behind this is that these fields contain denser information on the topic of the publication than the full text body, thus increasing the chance of relevant terms being extracted. The same choice was made by some participants of TREC CDS [13]. In order to limit the total number of parameters that need to be tested, other combinations of text fields will not be tested.

To prevent query drifting at most 3 new terms are added to the PubMed query. Each term must occur at least 10 times in the set of 10

Token selection

documents PRF is applied on, to prevent infrequent terms with a high IDF from being added to the query. These words would not have been able to describe the topic of the documents properly and preliminary testing showed that words occurring only once were added to the query without this filter. For the same reason, stop words are filtered from the list of potential terms for query expansion as well.

3.6 MODEL PRF

In the previous step unstemmed tokens were used to expand the PubMed query using pseudo relevance feedback. The same concept could also be applied to the vector representation of the queries (listing 5) to allow the TFIDF and BM25 retrieval models to utilize the new terms and weights found in the PRF step.

The first step of this procedure is almost identical to the previous one. Again, BM25 and TFIDF scores are given to the terms of the title and abstract of the top 10 results from the initial PubMed search. This time, however, stemmed tokens are used to determine the scores. Next, the same token filtering procedure is applied, with the addition of the removal of short tokens, and tokens with low scores. During preliminary research, the minimum length was set at 3, the threshold for TFIDF scores at 2.0, and the one for BM25 scores at 0.3. The updated query vector consists of at most 20 terms.

The feedback is applied to the existing query vector using the Rocchio algorithm (section 2.4). In this algorithm, the weights of the original query and the center of weights attained in the previous step are added, with a factor being applied to the latter. A weight of 1.0 was chosen for the original query and 0.3 for the center of the PRF vector. The reason for this large difference in weight is that the original query remains the most important one and a high PRF quality cannot be assumed. By choosing a low weight for the documents in the PRF class query drifting is limited, since the influence of the documents on the query vector is limited as well. The parameters and filters chosen for pseudo relevance are not varied in the experiments.

Listing 6 shows what the example query vector looks like after applying pseudo relevance feedback.

Listing 6: The example query after PRF with TFIDF weights

```
[ ('footbal', 99.174855426903378), ('pallor', 93.890710748974712)
    , ('athlet', 75.885910260748801), ('ferritin',
    74.025234229394599), ('fatigu', 54.230805220080242), ('muscl
    ', 21.401200745501576), ('run', 14.959476346287717), ('femal
    ', 11.60798063015214), ('human', 6.177192896269271), ('fes',
    5.8725363999360018), \emph{('perform', 3.6906140457197805)},
    \emph{('exercis', 3.2689126725970277)}, \emph{('perceiv',
    2.5489173254668667)} ]
```

3.7 RETRIEVAL MODELS

The finish the basis of the retrieval system the results are now re-ranked using TFIDF, BM25, or a combination of both. Optionally, explicit relevance feedback can be applied afterwards, but this part will yield the first documents the user gets to see.

The input of these retrieval models consists of a set of publications - the set of at most 4000 documents collected in the earlier steps - and a query vector, which contains the query terms and corresponding weights. For each document the TFIDF or BM25 scores for terms occurring in both the query and document are calculated and multiplied by their respective weights. Roughly speaking, all the scores are added for each document and some compensation for the length of the document takes place. The reason for the latter is that when a term occurs the same number of times in a short text as in a long text, the short text is likely to be more relevant.

In this experiment no distinctions are made between the positions in the document in which a term occurs. The title, abstract and body of a document are all joined together and viewed as the representation of the document. It could be possible for a combination of publication fields to result in a greater performance when used, for instance only searching in a document's abstract, but this is not tested in this research.

TFIDF

The exact implementation of TFIDF is based on the way Lucene has implemented it, called the *Practical Scoring Function*. This adaption of TFIDF has the possibility to apply boosts factors to the scores of individual terms, which we use to let the weights in the query vectors affect the way TFIDF ranks the documents. Additionally, the implementation adds more weight to publications containing a higher fraction of terms matching the query. Furthermore, it calculates and utilizes the *norm* of the query, with the document length as its input, to make the scores comparable between queries. This does not affect the rankings the model produces, however. Section 2.2 contains more information on the TFIDF implementation.

BM25

For BM25 a Lucene variant of the algorithm was chosen as well. Lucene has a large number of users, meaning that their implementations have been tested rigorously over time, thus forming a robust basis for our system. To make it possible to boost the BM25 term scores as well, the entire score for a term is multiplied by the desired weight. That way the weights always have the same influence on the scores, regardless of the term frequency. This model is explained in more detail in section 2.2.

In both cases, TFIDF and BM25, the greater the document score, the more relevant the document is likely to be and the higher it is places in the rankings.

Combinations of models often perform better than single models. Therefore, in addition to TFIDF and BM25, combinations of both are tested as well. One of the best ways to combine models and other features is the use of a learning-to-rank machine learning model. This would make it possible to, for example, use the publication date as one of the re-ranking features. However, to keep the experiment size manageable two simpler approaches are tested, which may still be effective.

Normalized fusion

In the first approach, the document scores of the rankings received from the TFIDF and BM25 models are normalized, to make the highest score of a result list equal to 1. Next, the normalized scores of both result lists are added and sorted. The 1000 documents with the highest scores are then returned to the user, or used to apply explicit relevance feedback one (section 2.5).

Borda-fuse

In the second approach the document scores are not used, but only the ranks are. Scores are given to the documents depending on their rank, with the first document given the maximum score and the other documents scores that get 1 lower for each position further down the ranks. Next, the document scores are added and the list re-ranked on these scores. Again, the top 1000 documents are returned to the user, or sent to the next part of the retrieval system. Section 2.5 explains Borda-fuse in more detail.

When one of the models is known to be better, the Borda-fuse algorithm can be turned into Weighted Borda-fuse [4, 13] (section 2.5). This model multiplies the scores of each model with a certain weight, to increase or decrease the influence a model has. In this research we do not experiment with this variant and give both models equals weights.

3.8 INVERSE DOCUMENT FREQUENCY

Both TFIDF and BM25 make use of the *inverse document frequency* (section 2.2) to determine how special it is for a term to be present in a document. For instance, when a word such as *the* or *it* occurs frequently in a text, this is not very special, since these words are likely to be present in all documents. Thus the term frequency by itself does not give a proper indication of a text's subject. To compensate for words occurring in many documents the TFIDF and BM25 models multiply the term scores with that term's IDF value. Common words get a low IDF, whereas words occurring in a small number of documents attain high IDF scores.

Global IDF

The usual way to acquire IDF scores is to calculate these on a large collection of documents, to get a realistic view of the popularity of a

term. For this research, the IDF scores were calculated using the entire PMC dataset. As a result, the IDF values are only representative in the medical domain, or more specific, the PMC collection. To be able to use the system in other domains alternative IDF scores are needed, for instance scores calculated using a collection from that domain.

Local IDF

It would be pleasant if the IDF could be calculated using only the set of documents that need to be re-ranked. That way, the system remains flexible and agnostic of the domains it operates on. To determine the effect of such a locally calculated IDF an experiment will take place in which the performance of the systems using local IDF and global IDF are compared. This will be using the TFIDF and BM25 PRF models only, and not for combinations of these models or explicit relevance feedback.

3.9 EXPLICIT RELEVANCE FEEDBACK

For the users of the system high precision is of the utmost importance, since reading publications is time consuming and the available time limited. In an attempt to increase the precision explicit relevance feedback (section 2.4) will be taken into account. The feedback slightly modifies the query vector to make the query better match the type of documents that are requested.

To apply feedback, the user grades the relevance of each search result he received. For every 10 grades submitted, the ERF algorithm is ran and the feedback is applied to the query vector using Rocchio. This is done using a similar process as used to handle pseudo relevance feedback. The difference is that this time we know for certain how relevant each document is, instead of assuming the top 10 documents are all relevant. The maximum number of terms in the query vector was raised from 20 to 30 as a result.

The goal of the explicit relevance feedback experiment is to find the best weights for the Rocchio algorithm. If the relevance feedback system were to be tested with real persons giving feedback, either a large group of persons is needed, or they need to spend countless hours testing the system. Therefore, the TREC CDS data will be used in this experiment as well. The TREC data contains relevance judgements for many documents for each topic. These judgements can be used to simulate the user feedback. The judgements indicate whether a document was non-relevant, relevant, highly relevant, or left unjudged.

Graded relevance feedback

The Rocchio algorithm was designed to operate on binary feedback. Either a document is relevant, or it is not relevant. In this case, however, we possess graded relevance feedback. Therefore, the Rocchio algorithm needs to be modified to allow for graded feedback to be given. The algorithm will be modified by introducing another class of documents for the highly relevant publications, with

a weight that is a multiple of the weight for the regular relevant document class. This way, additional weight can be added for terms occurring in highly relevant publications.

Since feedback is processed for every 10 publications, to simulate real-world usage of the HERO system, the number of documents in each class is likely small. To account for this, the weights used in the Rocchio algorithm need to be substantially smaller than the usual weights used in Rocchio. If these weights would remain large a single document could have a large impact on the query vector, increasing the change of query drifting occurring.

If the Rocchio weights for explicit relevance feedback had to be tested for all other combinations of parameters, the number of test cases would explode, resulting in very large amounts of time needed to run all test cases. Therefore, the ERF parameters are only tested on top of the best system after applying pseudo relevance feedback. That way, only those parameters need to be varied during the experiment.

Similarly to the way PRF is handled, the ERF can be handled using the TFIDF model, BM25 model, or a combination of both to update the query vector and re-rank the documents. Therefore, different combinations of these models still need to be tested, but now only for the ERF stage instead of the entire retrieval flow.

The first ten documents the system with explicit feedback returns are the ones generated by the best mode using pseudo relevance feedback. After this, feedback is given to the publications in each batch of ten results. The feedback is used to update the query vector, after which the remaining documents are re-ranked and the new top 10 is returned. This proces repeats itself until there are no more documents left in the set, or the limit of 1000 documents is reached.

In addition to increased precision, the aim of explicit relevance feedback is to yield a higher infNDCG score, for the same number of documents returned.

3.10 EXPERIMENTS

To be able to answer the research questions, the influence each part of the retrieval system has on the infNDCG has to be known. Additionally, the combination of parameters resulting in the highest infNDCG for each part must be discovered.

The infNDCG was chosen as the main evaluation metric in this research since it is the metric used in TREC CDS, and takes the graded feedback into account, whereas the popular MAP metric (section 2.3) only works on binary relevance.

It may be the case that the parameters yielding the highest intermediate result in a single stage of the retrieval flow, do not result in the best performance when additional stages are added. After all, a high recall in the first stages of the process may result in

a low infNDCG, but it gives the models in later stages more potential to retrieve relevant documents. Therefore, up to the explicit feedback stage all parameters will be tested, even when the best parameters for the previous stage in the retrieval flow are known.

All possible combinations of the following settings will be tested in the different models:

1. Query construction
    - Number of MeSH terms used: [1..9]
    - Filter common MeSH terms: [yes, no]
    - PubMed query type: [and, or]
    - Weight boost of original query: [2.0]
2. PubMed retrieval
    - Maximum publications used: [3000]
    - Maximum number of query terms : [20]
    - Stemming: [yes]
3. Pseudo relevance feedback (for both PubMed and models)
    - Top documents: [10]
    - Scoring model: [TFIDF, BM25]
    - Document parts: ['title-abstract']
    - Weight of original query: [1.0]
    - Weight of feedback: [0.3]
    - Minimum term frequency: [10]
    - Filter stopwords: [yes]
4. PRF for PubMed
    - Maximum number of new terms: [3]
    - Stemming: [no]
    - Maximum publications added: [1000]
5. PRF for models
    - Maximum number of query terms : [20]
    - Stemming: [yes]
    - Minimum token length: [3]
    - Minimum BM25 score: [0.3]
    - Minimum TFIDF score: [2.0]
6. Model re-ranking
    - Scoring model: [TFIDF, BM25]

- Combination model: [None, Normalized-addition, Borda-fuse]
- Document parts: ['title-abstract-body']

7. Explicit relevance feedback
   - Feedback size: [10]
   - Scoring model: [TFIDF, BM25]
   - Document parts: ['title-abstract']
   - Weight of original query: [1.0]
   - Weight of relevant class: [0.06, 0.08, 0.10, 0.20, 0.30, 0.50, 0.70]
   - Weight of non-relevant class: [0.00, 0.01, 0.02, 0.06, 0.10]
   - Weight boost for highly relevant: [1.0, 1.5, 2.0, 2.5, 5.0]
   - Stemming: [yes[
   - Minimum token length: [3]
   - Minimum term frequency: [3]
   - Filter stopwords: [yes]
   - Minimum length: [3]
   - Minimum score: [0.0]

Evaluation scripts

After each stage of the retrieval flow, the rankings attained using all parameter combinations and folds are stored in files. Next, the measurements scripts of TREC are used to evaluate the test runs. The first script calculates the traditional metrics used in TREC, such as precision and recall, and does this with the first 1000 results of the test run. The second script determines the inferred measures, including the infNDCG, using the first 100 results of a run.

Since the results are stored for all parameter combinations and phases of the retrieval flow, all research questions can be answered after running the tests, but looking at the individual parameters, models, and scores.

As described earlier in this chapter, the TREC topics have been split into five stratified folds, each containing 12 topics. To answer the research questions, the first four folds are used to determine the best models and parameters. The fifth fold is only used to estimate the performance of the system in the real-world.

Cross-validation

Contrary to the usual procedures followed while conducting experiments like this one, cross-validation is not needed in this case. The reason for this is that no actual training of the system takes place. There are no hidden parameters learned and used between queries. All queries are independent and tested with all possible parameters. The results are calculated per query and averaged for all queries in the training set. If cross-validation were to be used, the

results of each query would be counted three times, but still result in the same average as in the current approach.

Furthermore, the aim is to find the best parameters for the system. When cross-validation is used and the best parameters are chosen for each set of training folds, the results are inconclusive.

This is why four out of five folds are used to find parameters and the corresponding performance, whereas the fifth fold is only used to estimate the performance if the system is used in the HERO project, or another real-world scenario.

Evaluation of explicit relevance feedback

In the case of explicit relevance feedback a slightly different testing procedure will be followed, since the effect of ERF on top of another model needs to be measured. The model ERF will be applied on is the best model that uses PRF. Therefore, before we can test the ERF performance, the best PRF model needs to be found.

When all measurements have taken place, the results can be analyzed and the research questions answered.

4 EXPERIMENT RESULTS

4.1 RESULTS

The prototype was tested at each stage of the retrieval process. The appendices A.1, A.2, A.3, and A.4 contain the full results of these tests. In this chapter those results will be used to answer the research questions.

4.1.1 *RQ1*

For the first research question we want to know which model yields the highest infNDCG score, without using explicit relevance feedback. This model will serve as the baseline for the explicit relevance feedback algorithm, which will be built on top of this.

The PRF variants resulting in the highest infNDCG scores are listed in table 1. Overall, the BM25 re-ranking models slightly outperform the TFIDF ones. A combination of both models with TFIDF scores as weights performed best. This system resulted in an infNDCG of 0.1548, when using the maximum amount of 9 MeSH terms and filtering common ones. On the test fold, which was not used to find the best parameters, these settings result in an infNDCG of 0.1858.

Two approaches were tested to combine the TFIDF and BM25 re-ranking models. However, while the produced scores were different from each other, the rankings the combinations yielded were equal. As a result, both combined systems resulted in the same scores and are, therefore, only listed once in the results.

The combination model that was chosen for the system is Borda-fuse, since it allows us to extend the system with more features in the future, and change the weights of the combined rankings.

4.1.2 *RQ2*

The second research question is about finding a method to search in multiple collections of medical publications, without the need to store copies of the entire collections.

This research has shown that it is possible to use existing search engines to retrieve a subset of a collection's documents and re-rank those documents.

Additionally, we wanted to determine whether it is necessary to calculate IDF values on large collections of documents, or whether

Table 1: Comparison of top PRF models

run	infNDCG	P@10	P@100	R@100
tfidf-prf-or-9-Y-tfidf-model	0.1492	0.1938	0.1373	0.1096
bm25-prf-or-9-Y-tfidf-model	0.1532	0.2042	0.1410	**0.1106**
tfidf-prf-or-9-Y-bm25-model	0.1540	0.2042	0.1373	0.1099
bm25-prf-or-9-Y-bm25-model	0.1535	**0.2167**	0.1358	0.1088
tfidf-prf-or-9-Y-both-models	**0.1548**	0.2146	0.1392	0.1092
bm25-prf-or-9-N-both-models	0.1519	0.2146	**0.1415**	0.1103

Table 2: Comparison of PRF models with local IDF variants

run	infNDCG	P@10	P@100	R@100
tfidf-prf-or-9-Y-tfidf-model	0.1492	0.1938	0.1373	**0.1096**
tfidf-prf-or-9-N-tfidf-model-local	**0.1505**	0.1938	**0.1396**	0.1067
bm25-prf-or-9-Y-tfidf-model	**0.1532**	**0.2042**	0.1410	**0.1106**
bm25-prf-or-9-Y-tfidf-model-local	0.1476	0.1979	**0.1423**	0.1071
tfidf-prf-or-9-Y-bm25-model	**0.1540**	**0.2042**	**0.1373**	**0.1099**
tfidf-prf-or-9-Y-bm25-model-local	0.1323	0.1792	0.1217	0.0969
bm25-prf-or-9-Y-bm25-model	**0.1535**	**0.2167**	**0.1358**	**0.1088**
bm25-prf-or-9-Y-bm25-model-local	0.1324	0.1854	0.1215	0.0976

the IDF calculated using a subset of documents collected for a single query is sufficient.

To answer this question, the results of the uncombined re-ranking models using PRF are compared. The scores of the models of each type are shown in table 2.

Overall, the systems using the standard IDF outperform the local IDF ones. In some cases however, the local IDF variants perform slightly better. The best standard IDF system yields an infNDCG of 0.1540, whereas the best local model yields an 0.1505 infNDCG.

4.1.3 *RQ3*

The third research question is about the influence of pseudo relevance feedback on the search quality of the system, which we measure in terms of recall, precision and infNDCG. To answer this question the results of the system with and without PRF are compared. The infNDCG is the main metric used in this research to estimate the ranking performance. Precision is important as well,

Table 3: Comparison of PubMed baseline and PRF models

run	infNDCG	P@10	P@100	R@100
baseline-or-9-N	0.1197	0.1500	0.1019	0.0817
tfidf-prf-or-9-Y-both-models	**0.1548**	0.2146	0.1392	0.1092
baseline-or-9-Y	0.0746	0.1667	0.0479	0.0327
tfidf-prf-or-9-Y-both-models	0.1548	**0.2146**	0.1392	0.1092
baseline-or-9-N	0.1197	0.1500	0.1019	0.0817
tfidf-prf-or-9-N-both-models	0.1525	0.2083	**0.1394**	0.1090
baseline-or-9-N	0.1197	0.1500	0.1019	0.0817
tfidf-prf-or-9-Y-both-models	0.1548	0.2146	0.1392	**0.1092**

since the medical experts that will be using the system have a limited amount of time available, thus their time should not be wasted by reading non-relevant publications.

For each metric we are interested in the best variants of the system with and without using PRF. Table 3 shows these results. The baseline runs contain the results as returned by PubMed, before having applied PRF.

Both the baseline and the PRF system work best when OR queries are used on PubMed, with the maximum number of 9 MeSH terms in the query.

The results clearly show that the PRF system is outperforming the baseline on all metrics. The highest recall attained at position 100 of the results is *0.1092* with the PRF system and *0.0817* for the baseline. Thus, the recall is over 33% higher than the baseline when PRF is used. The other metrics shown similar differences in performance.

Overall, pseudo relevance feedback seems to have had a positive impact on the system's performance.

4.1.4 *RQ4*

The fourth research question is about the use of explicit relevance feedback, and implicitly about the fact that explicit feedback may be graded.

For this prototype, the Rocchio feedback algorithm was adapted for graded relevance feedback by adding an additional class for highly relevant documents. The weight of this new class is the weight of the relevant class (β), multiplied by a factor (f).

The results in table 4 clearly show that explicit relevance feedback increases the performance of the system. The top PRF model attained an infNDCG score of 0.1548, whereas the top ERF models acquired an

Table 4: Comparison of PRF and ERF models

run	infNDCG	P@100	R@100
tfidf-prf-or-9-Y-both-models	0.1548	0.1392	0.1092
erf-a1.0-b0.30-c0.10-f5.0	0.1928	0.1737	0.1319
erf-a1.0-b0.50-c0.10-f5.0	0.1928	0.1700	0.1307

Table 5: Number of MeSH terms selected after MTI

Maximum length	1	2	3	4	5	6	7	8	9
Occurences	0	4	9	11	11	10	6	2	7

infNDCG of 0.1928. The precision at rank 100 increased from 0.1392 to 0.1737 and the recall increased as well. This answers the fourth and fifth research questions

4.1.5 *Query lengths*

The systems were tested with a maximum number of MeSH terms from 1 to 9. This is the number of MeSH terms that are selected at most, after the MTI process, to create the initial query. The results show that systems using PubMed OR-queries performed best and that they attained these results with a maximum of 9 MeSH terms, which is the highest limit that was tested.

To determine how many topics may have benefitted from an even higher limit the actual number of terms selected per query were recorded. The results are shown in table 5. Only 7 out of 60 topics used the maximum of 9 terms. Therefore, at most 7 topics could benefit from greater limits.

4.1.6 *PubMed query type*

To determine possible reasons why AND-queries performed worse than OR-queries, the number of results returned by the PubMed baseline was counted. Since there are 60 topics and 1000 results are returned at most, the maximum number of documents a system can retrieve is 60000. However, while OR-queries reach this limit, AND-queries yield far fewer results, as shown in table 6.

Since PubMed applies boolean retrieval, all terms in the AND-queries need to be present in a publication for it to be retrieved. When more terms are added to the query, fewer document will match the query, meaning that a smaller number of publications will be returned.

Table 6: Number of results returned by the PubMed baseline (max=60000)

max MeSH terms	type	filter MeSH terms	
		N	Y
1	and	58899	58899
2	and	42468	42468
3	and	26164	26703
4	and	16282	17065
5	and	10736	12540
6	and	9927	11822
7	and	9593	11591
8	and	9461	11540
9	and	9423	11525
1	or	58899	58899
2+	or	60000	60000

4.1.7 *Influence of caching on execution times*

Heavy caching was implemented in the prototype with the aim to reduce the execution times of the program. The run times to extract MeSH terms from a topic using MTI, execute a query on PubMed, and load 1000 publications were collected for 60 topics. The measurements are listed in table 7.

It is clearly visible that the use of caching drastically decreased the run times of the tasks. Since a typical run consists of one MeSH extraction tasks, two PubMed queries and the loading of 4000 publications, this results in an average time saved of 429.11 seconds per topic.

This speed-up is especially useful for the experiments, for which the retrieval process was followed over 27,000 times. However, the caching is also useful for the HERO-project, since the queries used in HERO are static, and therefore it is likely for publications to be present in the cache. In addition, the MeSH extraction only has to be done once for new static queries.

4.2 DISCUSSION

General performance

Overall, all steps of the retrieval process have shown to increase the inferred Normalized Discounted Cumulative Gain and other metrics. Each model benefits from a high number of MeSH terms in the initial query, and most perform better with the *Human*, *Male*, and *Female* MeSH terms removed. Additionally, the use of an OR-type query to

Table 7: Comparison of tasks with and without caching

task	caching	min time (s)	avg time (s)	max time (s)	n
MTI MeSH extraction	N	33.549	34.294	36.250	60
	Y	0.00039	0.00082	0.00142	60
PubMed search	N	1.5678	2.4229	4.5662	60
	Y	0.00339	0.005634	0.02530	60
Load 1k publications	N	26.793	99.936	186.50	60
	Y	1.4006	2.4399	3.9008	60

search on PubMed increased the performance, since the number of documents retrieved using AND-queries is small (section 4.1.6).

The baseline of the system, MTI and PubMed retrieval, yielded a maximum infNDCG of 0.1197. After applying pseudo relevance feedback, this score was increased to 0.1548, which is a substantial increase. After applying explicit relevance feedback the infNDCG was further increased to 0.1928.

Number of results used

Unfortunately, these results cannot be compared to those of the TREC CDS participants, since our results are measured with 100 publications and theirs with 1000. If we measure the infNDCG of our PubMed baseline with 1000 documents, the infNDCG is much higher at 0.2014.

However, we cannot simply measure the results of the entire prototype with 1000 documents in a fair way, since the system was optimized using 100 documents throughout the development process. The models that were selected to apply explicit relevance feedback on, and the static relevance feedback parameters that were chosen, are not necessarily the best ones for 1000 documents. If we do measure the infNDCG with 1000 documents though, we see that the highest score using only PRF will be at least 0.2470, and 0.2500 when explicit relevance feedback is applied as well.

4.2.1 *General limitations*

While the performance of the system did increase at each stage of the retrieval process, the performance is limited. There are several possible causes for this.

Firstly, the prototype uses the search results of PubMed as the collection to pick all its results from, instead of the entire PMC collection. The system can only re-rank a small number of publications from these results, since the process of retrieving and analyzing the documents is slow. Aside from the time needed to download the publications, the process to normalize the articles or fetch them from the cache is time consuming. We expect that the

performance could be increased drastically by allowing the system to process more publications per query. Even so, limitations would still apply to this system, since it should be able to search in multiple sources, and therefore does not have the possibility to copy the full collections of these sources.

Secondly, only basic retrieval models were tested in this research. More advanced models, such as BM25F, or the SPUD language model [11] may have performed better.

Thirdly, we believe that learning-to-rank (section 2.5) could have substantially improved the system's performance, since it allows the combination of many features to derive a document ranking. This would have made it possible to, for instance, use the publication date of a document as a feature, which is believed to be an important variable for determining the relevance of a medical publication [13].

Fourthly, the prototype does not treat the diagnosis field differently than the summary field of documents. Related work for TREC CDS has shown that the diagnosis fields has a large impact on performance, however. By having combined the fields, the diagnosis field was not used to its full potential.

Finally, since MTI is designed to derive the base query from the topics, the system is only capable of applying concept-based retrieval. However, related research has shown that a combination of text-based and concept-based retrieval outperforms concept-based retrieval. Therefore, we believe that this might have improved our system's performance as well, would it have been possible to implement text-based retrieval.

On a positive note, this research has shown that it is viable to calculate the IDF scores for the retrieval models using just the intermediate search results. This makes it possible to apply the system to new domains, without the need to update the IDF scores.

4.2.2 *Pseudo relevance feedback*

As expected, the use of pseudo relevance feedback increased the performance of the system, in terms of infNDCG, precision, and recall. However, the precision and recall are still low. The PRF system with the highest infNDCG has precisions of 0.2146 and 0.1392 at positions 10 and 100. This means that out of the first 10 returned publications, only 2 are expected to be relevant, and from the first 100 results only 14. The recall of 0.1092 at position 100 means that on average only 10.92 percent of the relevant documents are present in the top 100 results. At position 1000, the recall for this model is 0.3483, meaning that more relevant documents could have been present in the top 100 results if the re-ranking quality would have been better.

The tested versions of the pseudo relevance feedback models contained several parameters and filters which remained constant during the experiments. These parameters and filters where handpicked while developing the system, because they had demonstrated to increase the infNDCG scores. However, it is unlikely that these handpicked settings are the ideal ones. Therefore, the PRF quality may be improved by testing multiple variants of the parameters and filtering strategies.

The combined models that were tested resulted in the same document rankings, on the entire set of topics. This was unexpected, since both approaches work on a different basis. The Borda-fuse model uses the rank of documents as its input, whereas the second model normalizes and sums the document scores. While these models may yield the same ranking when the document scores are equally spread over the domain, we did not expect this to be the case for all topics.

4.2.3 *Explicit relevance feedback*

Explicit relevance feedback slightly improves precision and recall at position 100, to 0.1737 and 0.1319. These results are still poor, however, since the users of HERO require a very high precision because reading the return publications is very time consuming.

One way in which this problem could have been solved is to halt the retrieval of papers when a metric, such as the precision or utility score, drops below a certain treshold. We did not experiment with this approach in this research however.

Similar to the pseudo relevance feedback implementation, the explicit one also contains parameters and term filters that were handpicked and remained constant during the experiments. By varying these in the experiments better results may have been achieved.

Our version of Rocchio for graded relevance feedback has demonstrated that graded feedback resulted in a higher performance than binary relevance. The highest infNDCG that was attained with binary relevance is 0.1861. With graded relevance, however, the highest infNDCG achieved is 0.1928, with a weight that was five times as high for highly relevant documents as it was for normal relevant documents. This shows that our idea to incorporate the graded feedback was a successful one.

5

CONCLUSION

A prototype retrieval system for the HERO project was built in this research. It uses MTI to extract MeSH terms from patient descriptions to form concept-based queries. Both pseudo and explicit relevance feedback are applied to increase the quality of the search results.

Overall all these phases of the retrieval flow increased the system's performance, in terms of infNDCG, precision, and recall. The systems generally performed best on OR queries with the maximum of 9 MeSH terms used, after removing the MeSH terms *Human*, *Male*, and *Female*.

The recall of the PubMed baseline was increased from *0.0817* to *0.1092* after applying pseudo relevance feedback.

The best performing pseudo relevance feedback model is one using a combination of TFIDF and BM25 for re-ranking of the results, with TFIDF scores used in the relevance feedback process. This model attained an infNDCG of 0.1548.

When IDF values calculated over a large collection of documents were used, the maximum infNDCG was higher than with IDF values calculated locally. The difference is quite small however, with an infNDCG of 0.1540 for global IDF and 0.1505 for local IDF. Thus while the global IDF performed better on the TREC CDS dataset than the local one, for different domains the use of local IDF may provide a viable alternative for the collection of large datasets and the calculations of the IDFs.

The explicit relevance feedback results show that ERF increases the system's performance and our graded Rocchio algorithm performs better than the classic binary version. The best ERF model multiplied the influence of highly relevant documents by five, while compared to the influence of regular relevant documents. This shows how much information is lost when not distinguishing highly relevant documents from relevant ones. The highest infNDCG that was attained is 0.1928, with a precision of 0.1737 and recall of 0.1319 at position 100.

These results cannot be compared to those of the TREC CDS contestants, however, since our infNDCG was calculated using the first 100 documents returned by the system, whereas they used the first 1000 documents. Limited testing shows that the infNDCG of our system will increase substantially when switching to the top 1000 documents as well.

While the prototype has several limitations and disadvantages, it showed that it is possible to use IDF values that were calculated on

small document sets and that both methods of feedback increase the system's performance.

5.1 FUTURE WORK

Many ideas were not implemented in this research, to reduce the size of the experiments, and because there was no time left to conduct experiments. These ideas can be tested during future research.

The current system works on unigrams, while many MeSH terms consist of multiple words. Therefore, the use of n-grams for medical retrieval should be investigated.

During preliminary research BM25F was implemented and tested on the TREC CDS collection. This did not increase the performance of the system straightaway, after which the model was removed from the prototype. Further research is needed to determine whether the model can be tweaked to increase the system's performance.

Currently, the results of the TFIDF and BM25 models are combined using Borda-fuse. The use of learning-to-rank for this case should be investigated. In addition, the use of weighted Borda-fuse as an easier alternative should be considered.

The current relevance feedback models used the title and article of a publication, and the re-ranking modules used the entire document. The best fields for these models need to be determined.

The performance of MTI might be increased when the diagnosis is placed in the title of the synthetic article and the summary in the abstract. This hypothesis needs to be investigated.

A combination of concept-based and text-based search should be investigated. This is not possible with the current prototype implementation.

Finally, the optimal point to stop returning results was not investigated for this research, while it could well fit the HERO use case. Additional research is needed to find proper measures and techniques to stop yielding documents when the performance has dropped significantly.

BIBLIOGRAPHY

[1] Asma Ben Abacha and Saoussen Khelifi. "LIST at TREC 2015 Clinical Decision Support Track: Question Analysis and Unsupervised Result Fusion." In: *Proceedings of the 2015 Text Retrieval Conference*. 2015.

[2] G Amati and C J Van Rijsbergen. "Probabilistic models of information retrieval based on measuring the divergence from randomness." In: *ACM Transactions on Information Systems (TOIS)* 20.4 (Oct. 2002), pp. 357–389.

[3] Alan R Aronson and François-Michel Lang. "An overview of MetaMap - historical perspective and recent advances." In: *JAMIA* 17.3 (2010), pp. 229–236.

[4] Javed A Aslam and Mark Montague. "Models for metasearch." In: *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM. 2001, pp. 276–284.

[5] S Balaneshinkordan, A Kotov, and R Xisto. "WSU-IR at TREC 2015 Clinical Decision Support Track: Joint Weighting of Explicit and Latent Medical Query Concepts from Diverse Sources." In: *Proceedings of the 2015 Text Retrieval Conference* (2015).

[6] Christian Bizer et al. "DBpedia - A crystallization point for the Web of Data." In: *Web Semantics: Science, Services and Agents on the World Wide Web* 7.3 (Sept. 2009), pp. 154–165.

[7] Olivier Bodenreider. "The Unified Medical Language System (UMLS) - integrating biomedical terminology." In: *Nucleic Acids Research* 32.suppl 1 (2004), pp. D267–D270.

[8] L M Buffart et al. "Evidence-based physical activity guidelines for cancer survivors: Current guidelines, knowledge gaps and future research directions." In: *Cancer Treatment Reviews* 40.2 (Mar. 2014), pp. 327–340.

[9] Angela A Chang, Karen M Heskett, and Terence M Davidson. "Searching the Literature Using Medical Subject Headings versus Text Word with PubMed." In: *The Laryngoscope* 116.2 (Feb. 2006), pp. 336–340.

[10] Margaret H Coletti and Howard L Bleich. "Medical Subject Headings Used to Search the Biomedical Literature." In: *Journal of the American Medical Informatics Association* 8.4 (July 2001), pp. 317–323.

[11] Ronan Cummins. "Clinical Decision Support with the SPUD Language Model." In: *Proceedings of the 2015 Text Retrieval Conference* (2015).

[12] Ronan Cummins, Jiaul H Paik, and Yuanhua Lv. "A Pólya Urn Document Language Model for Improved Information Retrieval." In: *ACM Transactions on Information Systems (TOIS)* 33.4 (May 2015), pp. 21–34.

[13] E Dhondt, B Grau, and P Zweigenbaum. "LIMSI @ 2015 Clinical Decision Support Track." In: *Proceedings of the 2015 Text Retrieval Conference*. Feb. 2016, pp. 1–11.

[14] Djoerd Hiemstra. *Using language models for information retrieval*. PhD thesis, Centre for Telematics and Information Technology, University of Twente, 2001.

[15] Elizabeth S Jenuwine and Judith A Floyd. "Comparison of Medical Subject Headings and text-word searches in MEDLINE to retrieve studies on sleep in healthy individuals." In: *Journal of the Medical Library Association* 92.3 (2004), p. 349.

[16] Jing Jiang and ChengXiang Zhai. "An empirical study of tokenization strategies for biomedical information retrieval." In: *Information Retrieval* 10.4-5 (2007), pp. 341–363.

[17] Ilyes Khennak and Habiba Drias. "A Firefly Algorithm-based Approach for Pseudo-Relevance Feedback: Application to Medical Database." In: *Journal of Medical Systems* (Oct. 2016), pp. 1–15.

[18] Tie-Yan Liu et al. "Learning to rank for information retrieval." In: *Foundations and Trends® in Information Retrieval* 3.3 (2009), pp. 225–331.

[19] M McKerns and M Aivazis. *pathos: a framework for heterogeneous computing, 2010*. http://dev.danse.us/trac/pathos.

[20] Shiraz I Mishra et al. "Exercise interventions on health-related quality of life for people with cancer during active treatment." In: *The Cochrane database of systematic reviews* 8 (Aug. 2012), p. CD008465.

[21] J Palotti and A Hanbury. "TUW@ TREC Clinical Decision Support Track 2015." In: *Proceedings of the 2015 Text Retrieval Conference*. 2015.

[22] Randy R Richter and Tricia M Austin. "Using MeSH (medical subject headings) to enhance PubMed search strategies for evidence-based practice in physical therapy." In: *Physical Therapy* 92.1 (Jan. 2012), pp. 124–132.

[23] K Roberts et al. "Overview of the TREC 2015 Clinical Decision Support Track." In: *Proceedings of the 2015 Text Retrieval Conference* (2015).

[24] Stephen Robertson. "The Probabilistic Relevance Framework: BM25 and Beyond." In: *Foundations and Trends® in Information Retrieval* 3.4 (2009), pp. 333–389.

[25] J J Rocchio. *Relevance Feedback in Information Retrieval, Report No*. ISR-9 to the National Science Foundation, 1965.

[26] Kathryn H Schmitz et al. "American College of Sports Medicine Roundtable on Exercise Guidelines for Cancer Survivors." In: *Medicine and Science in Sports and Exercise* 42.7 (July 2010), pp. 1409–1426.

[27] Y Song et al. "ECNU at 2015 CDS Track: Two Re-ranking Methods in Medical Information Retrieval." In: *Proceedings of the 2015 Text Retrieval Conference*. 2015.

[28] Jakob Stöber et al. "Concept based Information Retrieval for Clinical Case Summaries." In: *Proceedings of the 2015 Text Retrieval Conference* (2015).

[29] Rudolf Berend Trieschnigg. *Proof of concept: concept-based biomedical information retrieval*. 10-176. PhD Thesis, Centre for Telematics and Information Technology, University of Twente, 2010.

[30] M J Velthuis et al. "The Effect of Physical Exercise on Cancer-related Fatigue during Cancer Treatment: a Meta-analysis of Randomised Controlled Trials." In: *Clinical Oncology* 22.3 (Apr. 2010), pp. 208–221.

[31] Hanna van Waart et al. "Effect of Low-Intensity Physical Activity and Moderate-to High-Intensity Physical Exercise During Adjuvant Chemotherapy on Physical Fitness, Fatigue, and Chemotherapy Completion Rates: Results of the PACES Randomized Clinical Trial." In: *Journal of Clinical Oncology* 33.17 (2015), 1918–U90.

[32] Emine Yilmaz, Evangelos Kanoulas, and Javed A Aslam. "A simple and efficient sampling method for estimating AP and NDCG." In: *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM. 2008, pp. 603–610.

[33] R You et al. "FDUMedSearch at TREC 2015 Clinical Decision Support Track." In: *Proceedings of the 2015 Text Retrieval Conference*. 2015.

APPENDIX

A.1 PUBMED BASELINE RESULTS

The results of the PubMed baseline are listed in table 8. The baseline shows the performance of the system when no re-ranking models or relevance feedback is used. The results that are retrieved from PubMed are passed straight to the user.

A.2 PSEUDO RELEVANCE FEEDBACK RESULTS

This section shows the results of the system after applying pseudo relevance feedback. Tables 9 and 10 show the performance of the system while using the TFIDF model, with TFIDF and BM25 used to generate term weights. In tables 11 and 11 this is shown with BM25 as the model. In tables 13 and 14 the TFIDF and BM25 models were combined. Since the results for normalized score addition and Borda-fuse are equal, they are only shown once.

A.3 EXPLICIT RELEVANCE FEEDBACK RESULTS

Table 15 shows the explicit relevance feedback results. The feedback cycle was started after applying the best pseudo relevance feedback models to the data and returning the top 10 results. This is the model using OR-type queries, uses 9 MeSH terms, filters common MeSH terms, uses TFIDF weight for pseudo relevance feedback and combines the results of the TFIDF and BM25 models.

Since the feedback cycle started after returning the top 10 publications, the P@10 score is the same as in the best PRF model and therefore omitted from this table.

A.4 LOCAL IDF RESULTS

Tables 16, 17, 18, and 19 the system's performance when the IDF scores calculated on the intermediate results, instead of the entire document collection.

Table 8: Results of the PubMed MTI baseline

type	num	filter	infNDCG	P@10	P@100	R@100	Test infNDCG
and	1	N	0.0302	0.0396	0.0248	0.0121	0.0140
and	1	Y	0.0302	0.0396	0.0248	0.0121	0.0140
and	2	N	0.0405	0.0479	0.0396	0.0244	0.0616
and	2	Y	0.0405	0.0479	0.0396	0.0244	0.0616
and	3	N	0.0573	0.1089	0.0427	0.0301	0.0609
and	3	Y	0.0573	0.1089	0.0427	0.0301	0.0659
and	4	N	0.0609	0.1143	0.0476	0.0275	0.0487
and	4	Y	0.0680	0.1190	0.0495	0.0321	0.0532
and	5	N	0.0601	0.1220	0.0444	0.0246	0.0418
and	5	Y	0.0725	0.1375	0.0518	0.0328	0.0463
and	6	N	0.0594	0.1225	0.0420	0.0242	0.0386
and	6	Y	0.0744	0.1487	0.0515	0.0339	0.0537
and	7	N	0.0573	0.1375	0.0378	0.0227	0.0335
and	7	Y	0.0734	0.1641	0.0482	0.0329	0.0537
and	8	N	0.0509	0.1300	0.0333	0.0206	0.0283
and	8	Y	0.0743	0.1667	0.0479	0.0327	0.0537
and	9	N	0.0499	0.1205	0.0326	0.0206	0.0283
and	9	Y	0.0746	0.1667	0.0479	0.0327	0.0537
or	1	N	0.0302	0.0396	0.0248	0.0121	0.0140
or	1	Y	0.0302	0.0396	0.0248	0.0121	0.0140
or	2	N	0.0268	0.0312	0.0275	0.0119	0.0159
or	2	Y	0.0268	0.0312	0.0275	0.0119	0.0159
or	3	N	0.0586	0.0646	0.0510	0.0338	0.0559
or	3	Y	0.0586	0.0646	0.0510	0.0338	0.0559
or	4	N	0.0795	0.0854	0.0675	0.0496	0.0853
or	4	Y	0.0794	0.0854	0.0673	0.0489	0.0842
or	5	N	0.0948	0.1021	0.0800	0.0594	0.0896
or	5	Y	0.0951	0.1042	0.0794	0.0591	0.0885
or	6	N	0.1023	0.1167	0.0877	0.0640	0.1205
or	6	Y	0.1015	0.1208	0.0865	0.0635	0.1190
or	7	N	0.1059	0.1250	0.0912	0.0659	0.1290
or	7	Y	0.1050	0.1292	0.0904	0.0659	0.1286
or	8	N	0.1079	0.1313	0.0938	0.0681	0.1308
or	8	Y	0.1068	0.1333	0.0925	0.0680	0.1303
or	9	N	0.1197	0.1500	0.1019	0.0817	0.1336
or	9	Y	0.1187	0.1521	0.1008	0.0817	0.1330

Table 9: Results TFIDF model with TFIDF PRF

type	num	filter	infNDCG	P@10	P@100	R@100	Test infNDCG
and	1	N	0.0502	0.0583	0.0371	0.0161	0.0105
and	1	Y	0.0502	0.0583	0.0371	0.0161	0.0105
and	2	N	0.0486	0.0500	0.0440	0.0302	0.0627
and	2	Y	0.0486	0.0500	0.0440	0.0302	0.0627
and	3	N	0.0609	0.0911	0.0480	0.0341	0.0658
and	3	Y	0.0621	0.0911	0.0482	0.0343	0.0678
and	4	N	0.0618	0.1024	0.0507	0.0294	0.0613
and	4	Y	0.0662	0.0952	0.0531	0.0350	0.0665
and	5	N	0.0544	0.1049	0.0437	0.0243	0.0537
and	5	Y	0.0651	0.0975	0.0510	0.0343	0.0589
and	6	N	0.0544	0.1200	0.0408	0.0237	0.0469
and	6	Y	0.0686	0.1256	0.0497	0.0350	0.0656
and	7	N	0.0517	0.1325	0.0380	0.0228	0.0434
and	7	Y	0.0670	0.1333	0.0474	0.0344	0.0656
and	8	N	0.0435	0.1100	0.0330	0.0204	0.0351
and	8	Y	0.0671	0.1308	0.0467	0.0341	0.0656
and	9	N	0.0412	0.1000	0.0323	0.0205	0.0351
and	9	Y	0.0677	0.1308	0.0467	0.0341	0.0656
or	1	N	0.0512	0.0583	0.0379	0.0177	0.0105
or	1	Y	0.0512	0.0583	0.0379	0.0177	0.0105
or	2	N	0.0445	0.0542	0.0375	0.0220	0.0285
or	2	Y	0.0445	0.0542	0.0375	0.0220	0.0285
or	3	N	0.0756	0.0833	0.0660	0.0453	0.0743
or	3	Y	0.0756	0.0833	0.0660	0.0453	0.0732
or	4	N	0.1026	0.1292	0.0952	0.0673	0.0796
or	4	Y	0.1044	0.1292	0.0956	0.0682	0.0796
or	5	N	0.1139	0.1500	0.1048	0.0783	0.0958
or	5	Y	0.1166	0.1396	0.1069	0.0800	0.0958
or	6	N	0.1238	0.1562	0.1187	0.0896	0.1670
or	6	Y	0.1256	0.1521	0.1204	0.0911	0.1647
or	7	N	0.1331	0.1625	0.1260	0.0950	0.1698
or	7	Y	0.1349	0.1688	0.1285	0.0976	0.1661
or	8	N	0.1370	0.1771	0.1306	0.0994	0.1767
or	8	Y	0.1389	0.1771	0.1317	0.1001	0.1729
or	9	N	0.1473	0.1917	0.1356	0.1083	0.1860
or	9	Y	0.1492	0.1938	0.1373	0.1096	0.1821

Table 10: Results TFIDF model with BM25 PRF

type	num	filter	infNDCG	P@10	P@100	R@100	Test infNDCG
and	1	N	0.0479	0.0521	0.0377	0.0155	0.0133
and	1	Y	0.0479	0.0521	0.0377	0.0155	0.0133
and	2	N	0.0474	0.0542	0.0433	0.0299	0.0688
and	2	Y	0.0474	0.0542	0.0433	0.0299	0.0688
and	3	N	0.0620	0.0956	0.0484	0.0353	0.0643
and	3	Y	0.0620	0.0956	0.0484	0.0353	0.0708
and	4	N	0.0625	0.1048	0.0505	0.0293	0.0606
and	4	Y	0.0649	0.0952	0.0529	0.0349	0.0687
and	5	N	0.0540	0.1098	0.0437	0.0243	0.0537
and	5	Y	0.0628	0.1000	0.0510	0.0341	0.0617
and	6	N	0.0549	0.1350	0.0410	0.0238	0.0470
and	6	Y	0.0670	0.1333	0.0500	0.0349	0.0685
and	7	N	0.0529	0.1425	0.0380	0.0228	0.0421
and	7	Y	0.0659	0.1385	0.0474	0.0343	0.0685
and	8	N	0.0441	0.1200	0.0330	0.0204	0.0353
and	8	Y	0.0655	0.1359	0.0467	0.0339	0.0685
and	9	N	0.0420	0.1103	0.0323	0.0205	0.0353
and	9	Y	0.0662	0.1359	0.0467	0.0339	0.0685
or	1	N	0.0490	0.0542	0.0383	0.0158	0.0133
or	1	Y	0.0490	0.0542	0.0383	0.0158	0.0133
or	2	N	0.0459	0.0521	0.0381	0.0221	0.0299
or	2	Y	0.0459	0.0521	0.0381	0.0221	0.0299
or	3	N	0.0791	0.0896	0.0690	0.0477	0.0814
or	3	Y	0.0791	0.0896	0.0690	0.0477	0.0771
or	4	N	0.1057	0.1437	0.0965	0.0682	0.0883
or	4	Y	0.1080	0.1396	0.0973	0.0699	0.0870
or	5	N	0.1183	0.1542	0.1079	0.0810	0.1102
or	5	Y	0.1213	0.1500	0.1094	0.0826	0.1088
or	6	N	0.1274	0.1563	0.1227	0.0909	0.1811
or	6	Y	0.1306	0.1562	0.1240	0.0925	0.1788
or	7	N	0.1348	0.1688	0.1298	0.0958	0.1795
or	7	Y	0.1373	0.1750	0.1310	0.0977	0.1777
or	8	N	0.1389	0.1813	0.1333	0.0991	0.1892
or	8	Y	0.1430	0.1854	0.1358	0.1018	0.1855
or	9	N	0.1490	0.2042	0.1377	0.1085	0.1986
or	9	Y	0.1532	0.2042	0.1410	0.1106	0.1952

Table 11: Results BM25 model with TFIDF PRF

type	num	filter	infNDCG	P@10	P@100	R@100	Test infNDCG
and	1	Y	0.0478	0.0521	0.0369	0.0170	0.0076
and	2	Y	0.0497	0.0542	0.0450	0.0315	0.0767
and	3	N	0.0582	0.0933	0.0451	0.0330	0.0628
and	3	Y	0.0582	0.0933	0.0449	0.0328	0.0706
and	4	N	0.0629	0.1119	0.0510	0.0303	0.0575
and	4	Y	0.0659	0.1095	0.0526	0.0354	0.0701
and	5	N	0.0544	0.1098	0.0432	0.0241	0.0555
and	5	Y	0.0649	0.1175	0.0505	0.0340	0.0680
and	6	N	0.0540	0.1350	0.0405	0.0236	0.0468
and	6	Y	0.0680	0.1487	0.0495	0.0348	0.0716
and	7	N	0.0511	0.1350	0.0378	0.0227	0.0462
and	7	Y	0.0660	0.1487	0.0472	0.0343	0.0716
and	8	N	0.0426	0.1125	0.0328	0.0203	0.0356
and	8	Y	0.0663	0.1487	0.0464	0.0339	0.0716
and	9	N	0.0407	0.1026	0.0321	0.0204	0.0356
and	9	Y	0.0671	0.1487	0.0464	0.0339	0.0716
or	1	Y	0.0511	0.0542	0.0394	0.0192	0.0076
or	2	Y	0.0471	0.0500	0.0398	0.0261	0.0504
or	3	N	0.0788	0.0979	0.0685	0.0476	0.0838
or	3	Y	0.0788	0.0979	0.0685	0.0476	0.0850
or	4	N	0.1044	0.1396	0.0973	0.0716	0.0837
or	4	Y	0.1090	0.1375	0.0973	0.0719	0.0850
or	5	N	0.1153	0.1604	0.1096	0.0811	0.1079
or	5	Y	0.1211	0.1521	0.1108	0.0833	0.1092
or	6	N	0.1277	0.1833	0.1212	0.0921	0.1585
or	6	Y	0.1340	0.1729	0.1233	0.0949	0.1631
or	7	N	0.1370	0.1917	0.1298	0.1001	0.1555
or	7	Y	0.1433	0.1833	0.1317	0.1027	0.1603
or	8	N	0.1411	0.1958	0.1315	0.1016	0.1670
or	8	Y	0.1469	0.1875	0.1337	0.1046	0.1712
or	9	N	0.1489	0.2125	0.1352	0.1077	0.1756
or	9	Y	0.1540	0.2042	0.1373	0.1099	0.1798

Table 12: Results BM25 model with BM25 PRF

type	num	filter	infNDCG	P@10	P@100	R@100	Test infNDCG
and	1	N	0.0492	0.0542	0.0385	0.0174	0.0097
and	1	Y	0.0492	0.0542	0.0385	0.0174	0.0097
and	2	N	0.0497	0.0583	0.0467	0.0320	0.0726
and	2	Y	0.0497	0.0583	0.0467	0.0320	0.0726
and	3	N	0.0602	0.0956	0.0458	0.0336	0.0603
and	3	Y	0.0602	0.0956	0.0456	0.0333	0.0673
and	4	N	0.0637	0.1143	0.0514	0.0306	0.0556
and	4	Y	0.0673	0.1095	0.0529	0.0355	0.0702
and	5	N	0.0529	0.1049	0.0429	0.0240	0.0536
and	5	Y	0.0661	0.1125	0.0508	0.0341	0.0682
and	6	N	0.0530	0.1300	0.0405	0.0236	0.0448
and	6	Y	0.0693	0.1436	0.0497	0.0349	0.0714
and	7	N	0.0505	0.1325	0.0375	0.0226	0.0425
and	7	Y	0.0680	0.1462	0.0474	0.0344	0.0714
and	8	N	0.0417	0.1100	0.0325	0.0202	0.0340
and	8	Y	0.0679	0.1462	0.0467	0.0340	0.0714
and	9	N	0.0397	0.1000	0.0318	0.0202	0.0340
and	9	Y	0.0686	0.1462	0.0467	0.0340	0.0714
or	1	N	0.0488	0.0521	0.0383	0.0179	0.0097
or	1	Y	0.0488	0.0521	0.0383	0.0179	0.0097
or	2	N	0.0479	0.0604	0.0419	0.0296	0.0584
or	2	Y	0.0479	0.0604	0.0419	0.0296	0.0584
or	3	N	0.0816	0.1000	0.0694	0.0473	0.0863
or	3	Y	0.0815	0.1000	0.0692	0.0471	0.0855
or	4	N	0.1033	0.1479	0.0962	0.0724	0.0857
or	4	Y	0.1065	0.1479	0.0965	0.0719	0.0858
or	5	N	0.1144	0.1563	0.1083	0.0809	0.1137
or	5	Y	0.1193	0.1542	0.1088	0.0822	0.1137
or	6	N	0.1279	0.1896	0.1225	0.0960	0.1642
or	6	Y	0.1330	0.1854	0.1237	0.0970	0.1651
or	7	N	0.1384	0.1938	0.1300	0.1009	0.1592
or	7	Y	0.1429	0.1917	0.1310	0.1022	0.1602
or	8	N	0.1411	0.2083	0.1310	0.1021	0.1703
or	8	Y	0.1462	0.2042	0.1329	0.1041	0.1702
or	9	N	0.1488	0.2208	0.1346	0.1075	0.1797
or	9	Y	0.1535	0.2167	0.1358	0.1088	0.1796

Table 13: Results of both models with TFIDF PRF

type	num	filter	infNDCG	P@10	P@100	R@100	Test infNDCG
and	1	N	0.0506	0.0604	0.0377	0.0166	0.0106
and	1	Y	0.0506	0.0604	0.0377	0.0166	0.0106
and	2	N	0.0494	0.0521	0.0454	0.0316	0.0718
and	2	Y	0.0494	0.0521	0.0454	0.0316	0.0718
and	3	N	0.0603	0.0889	0.0471	0.0343	0.0634
and	3	Y	0.0602	0.0889	0.0469	0.0340	0.0700
and	4	N	0.0631	0.1048	0.0507	0.0299	0.0608
and	4	Y	0.0657	0.0976	0.0526	0.0351	0.0702
and	5	N	0.0537	0.1049	0.0429	0.0240	0.0555
and	5	Y	0.0636	0.1050	0.0503	0.0336	0.0648
and	6	N	0.0532	0.1275	0.0400	0.0234	0.0477
and	6	Y	0.0667	0.1359	0.0492	0.0343	0.0706
and	7	N	0.0510	0.1250	0.0373	0.0225	0.0467
and	7	Y	0.0656	0.1359	0.0469	0.0338	0.0706
and	8	N	0.0424	0.1075	0.0323	0.0201	0.0364
and	8	Y	0.0658	0.1359	0.0462	0.0334	0.0706
and	9	N	0.0413	0.0974	0.0324	0.0206	0.0364
and	9	Y	0.0665	0.1359	0.0462	0.0334	0.0706
or	1	N	0.0521	0.0625	0.0388	0.0181	0.0106
or	1	Y	0.0521	0.0625	0.0388	0.0181	0.0106
or	2	N	0.0452	0.0500	0.0402	0.0245	0.0409
or	2	Y	0.0452	0.0500	0.0402	0.0245	0.0409
or	3	N	0.0794	0.0833	0.0683	0.0451	0.0871
or	3	Y	0.0793	0.0833	0.0681	0.0449	0.0872
or	4	N	0.1050	0.1417	0.0944	0.0671	0.0838
or	4	Y	0.1077	0.1375	0.0946	0.0680	0.0828
or	5	N	0.1180	0.1542	0.1100	0.0805	0.1149
or	5	Y	0.1200	0.1521	0.1098	0.0813	0.1138
or	6	N	0.1296	0.1708	0.1250	0.0924	0.1749
or	6	Y	0.1322	0.1708	0.1240	0.0924	0.1735
or	7	N	0.1387	0.1896	0.1329	0.0993	0.1683
or	7	Y	0.1411	0.1938	0.1323	0.0995	0.1660
or	8	N	0.1423	0.1875	0.1358	0.1022	0.1781
or	8	Y	0.1455	0.1979	0.1354	0.1026	0.1753
or	9	N	0.1525	0.2083	0.1394	0.1090	0.1886
or	9	Y	0.1548	0.2146	0.1392	0.1092	0.1858

Table 14: Results of both models with BM25 PRF

type	num	filter	infNDCG	P@10	P@100	R@100	Test infNDCG
and	1	N	0.0485	0.0542	0.0379	0.0161	0.0132
and	1	Y	0.0485	0.0542	0.0379	0.0161	0.0132
and	2	N	0.0498	0.0521	0.0442	0.0303	0.0709
and	2	Y	0.0498	0.0521	0.0442	0.0303	0.0709
and	3	N	0.0624	0.0956	0.0476	0.0348	0.0607
and	3	Y	0.0623	0.0956	0.0473	0.0346	0.0687
and	4	N	0.0642	0.1119	0.0502	0.0295	0.0616
and	4	Y	0.0654	0.1071	0.0521	0.0348	0.0698
and	5	N	0.0538	0.1073	0.0427	0.0239	0.0567
and	5	Y	0.0632	0.1150	0.0503	0.0338	0.0649
and	6	N	0.0538	0.1300	0.0400	0.0234	0.0485
and	6	Y	0.0665	0.1462	0.0492	0.0346	0.0701
and	7	N	0.0520	0.1325	0.0373	0.0225	0.0454
and	7	Y	0.0656	0.1487	0.0469	0.0341	0.0701
and	8	N	0.0429	0.1125	0.0323	0.0201	0.0372
and	8	Y	0.0653	0.1462	0.0462	0.0337	0.0701
and	9	N	0.0417	0.1053	0.0324	0.0206	0.0372
and	9	Y	0.0660	0.1462	0.0462	0.0337	0.0701
or	1	N	0.0504	0.0542	0.0394	0.0173	0.0132
or	1	Y	0.0504	0.0542	0.0394	0.0173	0.0132
or	2	N	0.0432	0.0479	0.0394	0.0253	0.0458
or	2	Y	0.0432	0.0479	0.0394	0.0253	0.0458
or	3	N	0.0790	0.0896	0.0685	0.0465	0.0936
or	3	Y	0.0789	0.0896	0.0683	0.0463	0.0927
or	4	N	0.1098	0.1438	0.0965	0.0679	0.0919
or	4	Y	0.1110	0.1396	0.0969	0.0688	0.0912
or	5	N	0.1199	0.1583	0.1125	0.0834	0.1183
or	5	Y	0.1203	0.1562	0.1115	0.0818	0.1176
or	6	N	0.1282	0.1792	0.1258	0.0936	0.1853
or	6	Y	0.1293	0.1813	0.1250	0.0930	0.1819
or	7	N	0.1379	0.1875	0.1348	0.1008	0.1767
or	7	Y	0.1378	0.1979	0.1335	0.1000	0.1728
or	8	N	0.1418	0.1938	0.1375	0.1029	0.1855
or	8	Y	0.1423	0.2000	0.1367	0.1026	0.1799
or	9	N	0.1519	0.2146	0.1415	0.1103	0.1953
or	9	Y	0.1514	0.2229	0.1396	0.1082	0.1896

Table 15: Results explicit relevance feedback

α	β	γ	f	infNDCG	P@100	R@100	Test infNDCG
1.0	0.06	0.00	1.0	0.1705	0.1583	0.1251	0.2038
1.0	0.06	0.00	1.5	0.1761	0.1660	0.1290	0.2104
1.0	0.06	0.00	2.0	0.1779	0.1679	0.1291	0.2162
1.0	0.06	0.00	2.5	0.1791	0.1665	0.1280	0.2203
1.0	0.06	0.01	1.0	0.1720	0.1650	0.1299	0.2120
1.0	0.06	0.01	1.5	0.1768	0.1685	0.1314	0.2213
1.0	0.06	0.01	2.0	0.1820	0.1712	0.1328	0.2269
1.0	0.06	0.01	2.5	0.1805	0.1694	0.1318	0.2285
1.0	0.06	0.02	1.0	0.1734	0.1652	0.1304	0.2111
1.0	0.06	0.02	1.5	0.1793	0.1708	0.1334	0.2238
1.0	0.06	0.02	2.0	0.1845	0.1731	0.1344	0.2238
1.0	0.06	0.02	2.5	0.1803	0.1706	0.1324	0.2301
1.0	0.08	0.00	1.0	0.1751	0.1640	0.1278	0.2121
1.0	0.08	0.00	1.5	0.1779	0.1679	0.1291	0.2162
1.0	0.08	0.00	2.0	0.1828	0.1687	0.1311	0.2208
1.0	0.08	0.00	2.5	0.1830	0.1692	0.1312	0.2200
1.0	0.08	0.01	1.0	0.1753	0.1671	0.1313	0.2248
1.0	0.08	0.01	1.5	0.1839	0.1715	0.1339	0.2279
1.0	0.08	0.01	2.0	0.1837	0.1717	0.1337	0.2307
1.0	0.08	0.01	2.5	0.1885	0.1742	0.1366	0.2336
1.0	0.08	0.02	1.0	0.1764	0.1694	0.1328	0.2256
1.0	0.08	0.02	1.5	0.1837	0.1731	0.1352	0.2258
1.0	0.08	0.02	2.0	0.1839	0.1721	0.1344	0.2359
1.0	0.08	0.02	2.5	0.1880	0.1740	0.1353	0.2343
1.0	0.10	0.00	1.0	0.1756	0.1667	0.1294	0.2139
1.0	0.10	0.00	1.5	0.1791	0.1665	0.1280	0.2203
1.0	0.10	0.00	2.0	0.1830	0.1692	0.1312	0.2200
1.0	0.10	0.00	2.5	0.1855	0.1698	0.1316	0.2192
1.0	0.10	0.01	1.0	0.1796	0.1679	0.1336	0.2313
1.0	0.10	0.01	1.5	0.1826	0.1715	0.1329	0.2320
1.0	0.10	0.01	2.0	0.1851	0.1725	0.1350	0.2344
1.0	0.10	0.01	2.5	0.1866	0.1725	0.1342	0.2308
1.0	0.10	0.02	1.0	0.1794	0.1685	0.1335	0.2264
1.0	0.10	0.02	1.5	0.1832	0.1725	0.1336	0.2323
1.0	0.10	0.02	2.0	0.1864	0.1731	0.1350	0.2313
1.0	0.10	0.02	2.5	0.1871	0.1719	0.1343	0.2274

α	β	γ	f	infNDCG	P@100	R@100	Test infNDCG
1.0	0.20	0.00	1.0	0.1830	0.1692	0.1312	0.2200
1.0	0.20	0.00	1.5	0.1830	0.1700	0.1319	0.2176
1.0	0.20	0.00	2.0	0.1836	0.1681	0.1296	0.2211
1.0	0.20	0.00	2.5	0.1861	0.1710	0.1322	0.2388
1.0	0.20	0.01	1.0	0.1854	0.1710	0.1340	0.2413
1.0	0.20	0.01	1.5	0.1875	0.1715	0.1335	0.2362
1.0	0.20	0.01	2.0	0.1856	0.1721	0.1341	0.2349
1.0	0.20	0.01	2.5	0.1889	0.1733	0.1339	0.2545
1.0	0.20	0.02	1.0	0.1861	0.1731	0.1357	0.2432
1.0	0.20	0.02	1.5	0.1877	0.1717	0.1341	0.2351
1.0	0.20	0.02	2.0	0.1869	0.1717	0.1337	0.2263
1.0	0.20	0.02	2.5	0.1915	0.1754	0.1352	0.2523
1.0	0.30	0.02	2.5	0.1882	0.1731	0.1332	0.2540
1.0	0.30	0.02	5.0	0.1895	0.1708	0.1291	0.2625
1.0	0.30	0.06	2.5	0.1885	0.1715	0.1319	0.2555
1.0	0.30	0.06	5.0	0.1905	0.1708	0.1285	0.2629
1.0	0.30	0.10	2.5	0.1903	0.1744	0.1354	0.2546
1.0	0.30	0.10	5.0	0.1928	0.1737	0.1319	0.2611
1.0	0.50	0.02	2.5	0.1875	0.1696	0.1290	0.2678
1.0	0.50	0.02	5.0	0.1918	0.1683	0.1278	0.2576
1.0	0.50	0.06	2.5	0.1892	0.1715	0.1302	0.2621
1.0	0.50	0.06	5.0	0.1900	0.1694	0.1289	0.2536
1.0	0.50	0.10	2.5	0.1898	0.1725	0.1318	0.2588
1.0	0.50	0.10	5.0	0.1928	0.1700	0.1307	0.2568
1.0	0.70	0.02	2.5	0.1882	0.1690	0.1284	0.2584
1.0	0.70	0.02	5.0	0.1912	0.1708	0.1292	0.2540
1.0	0.70	0.06	2.5	0.1890	0.1692	0.1280	0.2597
1.0	0.70	0.06	5.0	0.1899	0.1698	0.1282	0.2543
1.0	0.70	0.10	2.5	0.1880	0.1702	0.1298	0.2592
1.0	0.70	0.10	5.0	0.1920	0.1717	0.1322	0.2554

Table 16: Results TFIDF model with TFIDF PRF and local IDF

type	num	filter	infNDCG	P@10	P@100	R@100	Test infNDCG
and	1	N	0.0459	0.0500	0.0323	0.0164	0.0040
and	1	Y	0.0459	0.0500	0.0323	0.0164	0.0040
and	2	N	0.0459	0.0521	0.0450	0.0320	0.0635
and	2	Y	0.0459	0.0521	0.0450	0.0320	0.0635
and	3	N	0.0610	0.0956	0.0469	0.0335	0.0607
and	3	Y	0.0621	0.0956	0.0471	0.0337	0.0679
and	4	N	0.0627	0.1024	0.0502	0.0293	0.0595
and	4	Y	0.0672	0.1000	0.0526	0.0350	0.0654
and	5	N	0.0529	0.1024	0.0437	0.0243	0.0533
and	5	Y	0.0653	0.1050	0.0508	0.0342	0.0592
and	6	N	0.0544	0.1300	0.0413	0.0239	0.0478
and	6	Y	0.0696	0.1359	0.0500	0.0351	0.0656
and	7	N	0.0522	0.1325	0.0380	0.0227	0.0420
and	7	Y	0.0684	0.1385	0.0472	0.0344	0.0656
and	8	N	0.0433	0.1075	0.0330	0.0204	0.0335
and	8	Y	0.0680	0.1333	0.0464	0.0340	0.0656
and	9	N	0.0413	0.0974	0.0323	0.0204	0.0335
and	9	Y	0.0688	0.1333	0.0464	0.0340	0.0656
or	1	N	0.0486	0.0521	0.0348	0.0171	0.0058
or	1	Y	0.0486	0.0521	0.0348	0.0171	0.0058
or	2	N	0.0443	0.0479	0.0363	0.0232	0.0391
or	2	Y	0.0443	0.0479	0.0363	0.0232	0.0391
or	3	N	0.0703	0.0896	0.0660	0.0397	0.0802
or	3	Y	0.0702	0.0896	0.0658	0.0395	0.0804
or	4	N	0.1023	0.1313	0.0954	0.0658	0.0838
or	4	Y	0.1024	0.1292	0.0954	0.0655	0.0838
or	5	N	0.1146	0.1396	0.1096	0.0801	0.1058
or	5	Y	0.1136	0.1333	0.1083	0.0780	0.1058
or	6	N	0.1300	0.1750	0.1250	0.0890	0.1583
or	6	Y	0.1264	0.1687	0.1240	0.0880	0.1619
or	7	N	0.1388	0.1729	0.1352	0.0979	0.1599
or	7	Y	0.1337	0.1708	0.1335	0.0966	0.1632
or	8	N	0.1405	0.1813	0.1348	0.0987	0.1687
or	8	Y	0.1371	0.1813	0.1348	0.0976	0.1711
or	9	N	0.1505	0.1938	0.1396	0.1067	0.1779
or	9	Y	0.1462	0.1938	0.1400	0.1062	0.1802

Table 17: Results TFIDF model with BM25 PRF and local IDF

type	num	filter	infNDCG	P@10	P@100	R@100	Test infNDCG
and	1	N	0.0480	0.0479	0.0346	0.0158	0.0125
and	1	Y	0.0480	0.0479	0.0346	0.0158	0.0125
and	2	N	0.0468	0.0542	0.0435	0.0306	0.0654
and	2	Y	0.0468	0.0542	0.0435	0.0306	0.0654
and	3	N	0.0605	0.0933	0.0462	0.0336	0.0613
and	3	Y	0.0640	0.0933	0.0464	0.0338	0.0695
and	4	N	0.0630	0.1095	0.0502	0.0294	0.0598
and	4	Y	0.0703	0.1071	0.0524	0.0348	0.0696
and	5	N	0.0534	0.1024	0.0437	0.0244	0.0533
and	5	Y	0.0684	0.1125	0.0508	0.0341	0.0631
and	6	N	0.0552	0.1300	0.0413	0.0240	0.0449
and	6	Y	0.0732	0.1436	0.0500	0.0350	0.0691
and	7	N	0.0533	0.1300	0.0380	0.0229	0.0388
and	7	Y	0.0720	0.1462	0.0472	0.0342	0.0691
and	8	N	0.0438	0.1050	0.0330	0.0205	0.0310
and	8	Y	0.0713	0.1410	0.0464	0.0339	0.0691
and	9	N	0.0418	0.0949	0.0323	0.0205	0.0310
and	9	Y	0.0721	0.1410	0.0464	0.0339	0.0691
or	1	N	0.0462	0.0458	0.0350	0.0155	0.0126
or	1	Y	0.0462	0.0458	0.0350	0.0155	0.0126
or	2	N	0.0466	0.0479	0.0383	0.0247	0.0453
or	2	Y	0.0466	0.0479	0.0383	0.0247	0.0453
or	3	N	0.0714	0.0938	0.0679	0.0424	0.0712
or	3	Y	0.0713	0.0938	0.0677	0.0422	0.0703
or	4	N	0.1040	0.1375	0.0967	0.0667	0.0860
or	4	Y	0.1040	0.1354	0.0967	0.0664	0.0830
or	5	N	0.1190	0.1542	0.1115	0.0811	0.1090
or	5	Y	0.1181	0.1521	0.1108	0.0807	0.1059
or	6	N	0.1303	0.1729	0.1256	0.0903	0.1650
or	6	Y	0.1300	0.1708	0.1267	0.0907	0.1597
or	7	N	0.1358	0.1646	0.1337	0.0958	0.1690
or	7	Y	0.1358	0.1729	0.1362	0.0981	0.1632
or	8	N	0.1385	0.1771	0.1350	0.0976	0.1793
or	8	Y	0.1395	0.1813	0.1373	0.0989	0.1709
or	9	N	0.1471	0.1917	0.1400	0.1063	0.1883
or	9	Y	0.1476	0.1979	0.1423	0.1071	0.1793

Table 18: Results BM25 model with TFIDF PRF and local IDF

type	num	filter	infNDCG	P@10	P@100	R@100	Test infNDCG
and	1	N	0.0366	0.0417	0.0292	0.0144	0.0023
and	1	Y	0.0366	0.0417	0.0292	0.0144	0.0023
and	2	N	0.0502	0.0500	0.0392	0.0307	0.0688
and	2	Y	0.0502	0.0500	0.0392	0.0307	0.0688
and	3	N	0.0529	0.0822	0.0436	0.0313	0.0599
and	3	Y	0.0543	0.0822	0.0438	0.0316	0.0680
and	4	N	0.0582	0.1000	0.0460	0.0256	0.0547
and	4	Y	0.0635	0.0976	0.0486	0.0329	0.0622
and	5	N	0.0516	0.1000	0.0427	0.0246	0.0480
and	5	Y	0.0638	0.1125	0.0480	0.0334	0.0555
and	6	N	0.0505	0.1175	0.0400	0.0233	0.0457
and	6	Y	0.0670	0.1410	0.0472	0.0335	0.0637
and	7	N	0.0489	0.1150	0.0375	0.0225	0.0410
and	7	Y	0.0660	0.1385	0.0449	0.0330	0.0637
and	8	N	0.0393	0.0925	0.0325	0.0201	0.0291
and	8	Y	0.0655	0.1385	0.0441	0.0326	0.0637
and	9	N	0.0375	0.0821	0.0318	0.0201	0.0291
and	9	Y	0.0664	0.1385	0.0441	0.0326	0.0637
or	1	N	0.0401	0.0438	0.0300	0.0153	0.0038
or	2	N	0.0436	0.0479	0.0342	0.0249	0.0405
or	2	Y	0.0436	0.0479	0.0342	0.0249	0.0405
or	3	N	0.0620	0.0813	0.0581	0.0390	0.0696
or	3	Y	0.0619	0.0813	0.0579	0.0388	0.0673
or	4	N	0.0898	0.1146	0.0875	0.0629	0.0685
or	4	Y	0.0913	0.1146	0.0877	0.0636	0.0671
or	5	N	0.0989	0.1208	0.0952	0.0718	0.0846
or	5	Y	0.1007	0.1208	0.0977	0.0735	0.0832
or	6	N	0.1142	0.1500	0.1073	0.0809	0.1303
or	6	Y	0.1150	0.1479	0.1108	0.0829	0.1318
or	7	N	0.1213	0.1646	0.1142	0.0880	0.1343
or	7	Y	0.1230	0.1625	0.1175	0.0902	0.1359
or	8	N	0.1242	0.1646	0.1156	0.0903	0.1389
or	8	Y	0.1262	0.1646	0.1187	0.0923	0.1401
or	9	N	0.1315	0.1792	0.1187	0.0956	0.1432
or	9	Y	0.1323	0.1792	0.1217	0.0969	0.1445

Table 19: Results BM25 model with BM25 PRF and local IDF

type	num	filter	infNDCG	P@10	P@100	R@100	Test infNDCG
and	1	N	0.0383	0.0500	0.0275	0.0133	0.0033
and	1	Y	0.0383	0.0500	0.0275	0.0133	0.0033
and	2	N	0.0472	0.0521	0.0404	0.0285	0.0742
and	2	Y	0.0472	0.0521	0.0404	0.0285	0.0742
and	3	N	0.0548	0.0800	0.0429	0.0307	0.0599
and	3	Y	0.0563	0.0800	0.0431	0.0310	0.0655
and	4	N	0.0592	0.1000	0.0474	0.0265	0.0532
and	4	Y	0.0640	0.1000	0.0495	0.0328	0.0657
and	5	N	0.0505	0.0927	0.0420	0.0235	0.0463
and	5	Y	0.0636	0.1075	0.0478	0.0319	0.0588
and	6	N	0.0504	0.1175	0.0403	0.0234	0.0401
and	6	Y	0.0677	0.1385	0.0477	0.0330	0.0665
and	7	N	0.0480	0.1125	0.0373	0.0223	0.0352
and	7	Y	0.0661	0.1359	0.0449	0.0322	0.0665
and	8	N	0.0383	0.0900	0.0323	0.0199	0.0240
and	8	Y	0.0658	0.1359	0.0441	0.0318	0.0665
and	9	N	0.0366	0.0795	0.0315	0.0199	0.0240
and	9	Y	0.0666	0.1359	0.0441	0.0318	0.0665
or	1	N	0.0396	0.0458	0.0310	0.0147	0.0052
or	1	Y	0.0396	0.0458	0.0310	0.0147	0.0052
or	2	N	0.0449	0.0521	0.0404	0.0301	0.0439
or	2	Y	0.0449	0.0521	0.0404	0.0301	0.0439
or	3	N	0.0636	0.0896	0.0571	0.0399	0.0674
or	3	Y	0.0635	0.0896	0.0569	0.0397	0.0620
or	4	N	0.0931	0.1167	0.0852	0.0619	0.0652
or	4	Y	0.0930	0.1146	0.0856	0.0631	0.0681
or	5	N	0.0967	0.1229	0.0923	0.0679	0.0817
or	5	Y	0.0990	0.1188	0.0946	0.0710	0.0846
or	6	N	0.1138	0.1562	0.1075	0.0804	0.1267
or	6	Y	0.1152	0.1542	0.1100	0.0838	0.1311
or	7	N	0.1216	0.1687	0.1148	0.0859	0.1312
or	7	Y	0.1229	0.1687	0.1165	0.0893	0.1348
or	8	N	0.1237	0.1708	0.1137	0.0873	0.1370
or	8	Y	0.1257	0.1708	0.1171	0.0911	0.1405
or	9	N	0.1310	0.1812	0.1183	0.0942	0.1465
or	9	Y	0.1324	0.1854	0.1215	0.0976	0.1500