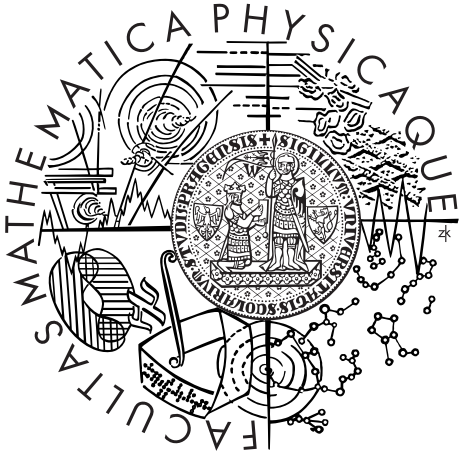
Charles University in Prague
Faculty of Mathematics and Physics

**MASTER THESIS**




Feraena Bibyna

# Query expansion for medical information retrieval

Institute of Formal and Applied Linguistics

Supervisors of the master thesis: RNDr. Pavel Pecina, Ph.D.
Günter Neumann, Ph.D.

Study programme: Master of Computer Science

Specialization: Mathematical Linguistics

Prague 2015

I would like to thank my supervisor in Prague, Pavel Pecina, as well as my supervisor in Saarbrücken, Günter Neumann for their invaluable help and guidance. I would also like to thank Shadi Saleh, for working with me in the CLEF task and sharing his knowledge with me. I would like to thank Ilana Rampula for our late night brainstorming sessions. Finally, I would like to thank my family for their continuous support and love.

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular the fact that the Charles University in Prague has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 paragraph 1 of the Copyright Act.

In Prague date 30 July 2015 signature of the author

Název práce: Rozšiřování dotazů pro vyhledávání medicínských informací

Autor: Feraena Bibyna

Katedra: Ústav formální a aplikované lingvistiky

Vedoucí diplomové práce: RNDr. Pavel Pecina, Ph.D., Günter Neumann, Ph.D.

Abstrakt: Jedním z problémů ve vyhledávání medicínských informací je terminologická "propast" mezi jazykem dokumentů (které jsou obvykle psané odborníky používajícími odbornou terminologii) a jazykem vyhledávacích dotazů (které jsou častěji tvořeny neodborníky používajícími především laické výrazy). V této diplomové práci zkoumáme možnosti řešení tohoto problému pomocí rozšiřování dotazů s využitím doménově specifických datových zdrojů. K tomuto používáme Unified Medical Language System (UMLS) obsahující sdružené biomedicínské názvosloví z několika zdrojů. Konkrétně používáme jeho metatezaurus a sémantickou síť. V experimentech používáme sadu dotazů z evaluační kampaně CLEF eHealth z let 2014 a 2015, které reprezentují dvě různé vyhledávací úlohy. Použité metody zahrnují rozšiřování dotazů pomocí synonymních i nesynonymních vztahů, metodu blind relevance feedback, vážení termů a také kombinování různých systémů pomocí lineární interpolace.

Klíčová slova: vyhledávání medicínských informací, ontologie, tezaurus, rozšiřování dotazů, UMLS

Title: Query expansion for medical information retrieval

Author: Feraena Bibyna

Department: Institute of Formal and Applied Linguistics

Supervisor: RNDr. Pavel Pecina, Ph.D., Günter Neumann, Ph.D.

Abstract: One of the challenges in medical information retrieval is the terminology gap between the documents (commonly written by medical professional, using medical jargons), and the queries (commonly composed by non professional, using layman terms). In this thesis, we investigate the effect of query expansion, using domain-specific knowledge resource, to deal with this challenge. We use the Unified Medical Language System (UMLS), a repository of biomedical vocabularies, and utilize two of its resources: the Metathesaurus and the Semantic Network. We use the query set and document set provided by CLEF eHealth organizer. The query sets, provided for the medical information retrieval shared task, represent two different use cases of medical information retrieval. We experiment with query expansion using synonymous terms and non-synonymous concepts, blind relevance feedback, field weighting, and linear interpolation of different systems.

Keywords: medical information retrieval, ontology, thesaurus, query expansion, UMLS

# Contents

**Introduction** **3**

**1 Background** **6**
1.1 Unified Medical Language System . . . . . 6
1.1.1 Metathesaurus . . . . . 6
1.1.2 Semantic Network . . . . . 7
1.2 Retrieval Models . . . . . 8
1.2.1 Inverse Document Frequency . . . . . 8
1.2.2 Language Model with Dirichlet Prior . . . . . 10
1.2.3 Per-Field Normalization . . . . . 11
1.2.4 LGD Weighting Model . . . . . 12
1.3 Query Reformulation . . . . . 12
1.3.1 Blind Relevance Feedback . . . . . 13
1.3.2 Query Expansion . . . . . 14
1.4 Evaluation Metrics . . . . . 15
1.4.1 Precision . . . . . 15
1.4.2 Normalized Discounted Cumulative Gain . . . . . 15
1.4.3 Relevant Documents and Unjudged Documents . . . . . 16
1.4.4 Wilcoxon Signed-Rank Test . . . . . 17

**2 Dataset** **19**
2.1 Document Collection . . . . . 19
2.2 Queries . . . . . 19
2.3 Annotation Process . . . . . 22

**3 Implementation** **24**
3.1 Terrier . . . . . 24
3.1.1 Indexing . . . . . 24
3.1.2 Retrieval . . . . . 25
3.1.3 Query Language . . . . . 26
3.2 Baseline System . . . . . 27
3.3 Expansion with Synonymous Terms . . . . . 27
3.3.1 Selecting Expansion Terms Using Inverse Document Frequency (idf) . . . . . 28
3.3.2 Selecting Expansion Terms Using Preferred Names . . . . 29
3.4 Expansion with Non-Synonymous Related Concepts . . . . . 30
3.5 Blind Relevance Feedback . . . . . 32
3.6 Field Weighting . . . . . 33
3.7 Utilizing Semantic Network . . . . . 34
3.8 Linear Interpolation . . . . . 35
3.9 CLEF eHealth 2015 Shared Task . . . . . 36

**4 Performance on Training Set** **37**
4.1 Using Original Query Terms . . . . . 37
4.2 Expansion Using Synonymous Terms . . . . . 37
4.2.1 Selecting Expansion Terms Using Inverse Document Frequency (idf) . . . . . 38
4.2.2 Selecting Expansion Terms Using Preferred Names . . . . 40
4.3 Query Expansion Using Non-Synonymous Related Concepts . . . 42
4.4 Blind Relevance Feedback . . . . . 44
4.5 Field Weighting Experiments . . . . . 47
4.5.1 Expansion using Synonymous Terms (idf) . . . . . 48
4.5.2 Expansion using Synonymous Terms (Preferred Names) . . 50
4.5.3 Expansion using Non-Synonymous Related Concepts . . . 52
4.5.4 Blind Relevance Feedback . . . . . 54
4.6 Utilizing Semantic Network . . . . . 55
4.7 Linear Interpolation . . . . . 57
4.8 Summary of Training Results . . . . . 60

**5 Performance on Test Sets** **62**
5.1 Systems' Performances on `test_14` . . . . . 62
5.2 Systems' Performances on `test_15` . . . . . 65
5.3 Discussion . . . . . 67

**Conclusion** **69**

**List of Tables** **77**

**List of Figures** **78**

# Introduction

The aim of this thesis is to investigate the effect of query expansion, using domain-specific knowledge resource, on medical information retrieval. We believe that using an external resource such as thesaurus can have an effect on the retrieval of relevant medical information.

Information retrieval can be defined in a very broad way. However, in the academic field, the task of information retrieval is defined as finding unstructured material within a large collection that satisfies an information need [Manning et al., 2008]. In most cases, the materials to be found are text documents that are a part of a collection stored in computers. The data is said to be unstructured in a way that it does not have a clear structure that can be easily parsed by machine, as opposed to structured data such as a relational database. The process of information retrieval starts by a user formulating their information need as a query. A query is a series of words that a user conveys to the system in an attempt to communicate their information need [Manning et al., 2008].

One of the information needs that has been commonly searched on the web is health information. In fact, nearly 70% of search engine users in the US have performed a search for information regarding a disease or health problem [Fox, 2011]. These searches are performed by a variety of users that ranges from laypeople without any medical training to medical health professional. This means that even though medical information retrieval is a domain specific task, it has a wide variety of information needs, which demands for medical information retrieval systems to be able to satisfy different type of health-related information needs for different kinds of users.

Can the current search engines satisfy this broad variety of information needs? Research showed that when users pose queries describing specific symptoms or general health information, the currently available search engines on the web can not effectively retrieve information relevant to their needs [Zhai and Lafferty, 2004]. If these users try to diagnose themselves using the result of the retrieval system, it could lead to dangerous consequences if these users try to treat themselves in case of misdiagnoses. One of the biggest challenges in medical information retrieval is that laypeople do not have sufficient medical knowledge to choose the correct medical terms which are relevant to their information needs. Laypeople users tend to use long, circumlocutory queries instead of precise medical terms to formulate their queries [Stanton et al., 2014].

Meanwhile, medical documents are often constructed by medical professionals who use a lot of medical jargon and abbreviations. This creates a "gap" between the terms in the query and terms that are available in the documents. For example, user might perform a search with the query "`blood spots on skin`". Based on a medical thesaurus, "`cherry haemangioma`" is one of the correct medical terms for this query phrase. However, document containing this medical phrase but not containing the query phrase will never be deemed relevant. To deal with this problem, we have to have a way to tell that the two phrases have the same meaning, or have the same semantic representation. A possible approach is to use domain-specific resources that contain semantic information of both medical and laypeople terms. Given some laypeople terms, we can this resource to get the

semantically related medical terms and add these terms to the original query, i.e. perform a query expansion, so that relevant documents containing these terms can also be retrieved.

Unified Medical Language System (UMLS)[1] is one of such resource. It is a repository of biomedical vocabularies developed by US National Library of Medicine [Bodenreider, 2004]. It aims to facilitate the interoperability between biomedical information systems. UMLS provided three knowledge sources: the Metathesaurus, the Semantic Network[2], and the SPECIALIST Lexicon. In this thesis, we make use of the first two systems. The Metathesaurus contains information about biomedical and health-related concepts, their various names, and relationships among them. It is organized by concept or meaning. This means that terms that are assigned to the same concept have the same meaning representation, i.e. they are synonymous. The Metahesaurus also defines relations between different concepst. The main relationship is the "`isa`" relation, but there are several other relations such as co-occurrences, causal, and location. Each concept in the Metathesaurus is assigned to at least one semantic type in the Semantic Network. The Semantic Network consists of a set of semantic types that provide a consistent categorization of all concepts represented in the UMLS Metathesaurus, and a set of useful and important relationships that exist between semantic types.

There has been previous research in utilizing UMLS and other thesaurus for query expansion. [Aronson and Rindflesch, 1997] compared their method of query expansion using the ULMS Metathesaurus to previous approach using statistically produces thesaurus [Srinivasan, 1996]. Their expanded queries consist of terms from the original queries and Metathesaurus phrases and concepts determined by MetaMap [Aronson, 2001]. They compared their system with a system without query expansion and gained some improvement, and the system using the UMLS Metathesaurus performed better compared to the system using statistically-produced thesaurus. [Koopman et al., 2012] utilized the SNOMED CT[3] to map queries and documents to their concept space. They also utilized the relations available in SNOMED CT to score documents, by giving weights to query concepts and related concepts. Their result showed that considering related concepts in addition to the original query concepts can improve retrieval effectiveness. However, they concluded that the selection of the relations to include and how to weight those relations is a challenging issue. [Shenwei et al., 2014] investigated concept-based approach in CLEF eHealth 2014 Task 3. They used UMLS Metathesaurus Release2012AB as their resource for mapping the documents and queries to concept space. They also performed query expansion using mutual information measure and Markov Random Field Model.

In this thesis, we try to tackle the problem of terminology gap by utilizing the UMLS Metathesaurus and Semantic Network to construct information retrieval systems that reformulate the original queries. We use Terrier[4] to build our systems. We have several ideas related to this that we think will improve on the performance of medical information retrieval systems that only use original

[1] (`http://www.nlm.nih.gov/research/umls/`

[2] `http://semanticnetwork.nlm.nih.gov/`

[3] `http://www.ihtsdo.org/snomed-ct`

[4] `http://terrier.org/`

queries terms. We use UMLS Metathesaurus to expand our original queries not only with the synonyms of their original terms, but also with non-synonymous related concepts. For query expansion using synonymous terms, we experiment with two different terms selection criteria: inverse document frequency (idf) and preferred names from Metathesaurus. As a comparison, we also experiment with another method of query reformulation: blind relevance feedback. For all of the aforementioned methods, we also experiment with weighting different fields differently. We also use UMLS Semantic Network to give different weights to terms with certain semantic types. Lastly, we combine our best systems using linear interpolation. We tune all of our systems on our training query set to find the optimal parameters and methods combination. Afterwards, we test the best systems on two sets of test queries, each representing different use cases in medical information retrieval.

As a prelude of this thesis, we participated in the CLEF eHealth 2015 Task 2 in User-Centered Health Information Retrieval[5]. This shared task aims to evaluate the effectiveness of information retrieval systems when searching for health content on the web. This shared task has been running since 2013, and this year's queries aims to mimic queries written by non-medical experts when presented with symptoms and sign and try to understand more about the condition that they might have. The document set provided by the organizers are crawled from medical websites. In other words, there will be a mismatch between query and document terms as we described before. In this thesis, we use the query set and document set provided by the organizers in the year 2014 and 2015.

We start by describing UMLS in more detail, and explaining the retrieval model that we use in this thesis in Chapter 1. In Chapter 2 we describe the document collection and queries that we use in this thesis. Chapter 3 covers the method of query expansion that we experimented with. We present the results of our experiments on our training set in Chapter 4. In Chapter 5, we present the results of our best system on our training sets, when tested on our test sets. Conclusion provides concluding remarks, suggestions, and some possibilities for future work.

[5] `https://sites.google.com/site/clefehealth2015/task-2`

# 1. Background

## 1.1 Unified Medical Language System

The Unified Medical Language System (UMLS) is a repository of biomedical vocabularies developed by US National Library of Medicine. It aims to integrate the various names used to expressed the same biomedical concept and to standardize the format for distributing terminologies. UMLS integrates more than 60 families of biomedical vocabularies. UMLS provides three knowledge sources: the Metathesaurus, the Semantic Network, and the SPECIALIST Lexicon. In this thesis, we make use of the first two systems. The distribution of UMLS is updated quarterly, and for the purpose of this thesis, we use the 2014AA distribution.

### 1.1.1 Metathesaurus

The Metathesaurus is the major component of the UMLS. The Metathesaurus is a large collection of biomedical terms and health related concepts, in fact the 2014AA release contains 2,973,458 concepts. It provides additional information about each term such as the variations in names for a term, the preferred name for the term, relationships between different terms, and links to semantic types in the Semantic Network. The Metathesaurus is built from several "source vocabularies", which are the electronic versions of various thesauri, classifications, code sets, and lists of controlled terms used in patient care, health service billing, public health statistics, indexing and cataloging biomedical literature, and/or basic, clinical, and health services research.

The Metathesaurus is organized by concept, which symbolizes a semantic concept or a meaning. A meaning can have many different surface realizations, which are the terms that are assigned to that particular concept in the Metathesaurus. Each concept in the Metathesaurus has a unique and permanent concept identifier (CUI). The different synonyms and abbreviations of this concept is called terms. A term is identified by a LUI (Lexical Unified Identifer).

The basic building blocks of the construction of the Metathesaurus are the concept names or strings from each of the source vocabularies. Every occurrence of a string in each source vocabulary is assigned a unique atom identifier (AUI). When the same string appears in multiple source vocabularies, it will have AUIs for every time it appears as a concept name in each of those sources. All of these AUIs will be linked to a single string identifier (SUI), since they represent occurrences of the same string. Each of these strings is then linked to all of its lexical variants or minor variations by means of the aforementioned LUI. Table 1.1 shows an example of difference between CUI, LUI, SUI, and AUI.

In addition to the synonymous relations described above, the Metathesaurus includes many relationships between different concepts. Most of these relations come from individual source vocabularies. There are two types of non-synonymous relationships in the Metathesaurus: non-synonymous relations between concepts from the same source vocabulary (intra-source vocabulary relations) and between concepts in different vocabularies (inter-source vocabulary relations). These relations are contained in the `MRREL` file in the UMLS distribu-

| Concept (CUI) | Terms (LUI) | Strings (SUI) | Atoms (AUI) |
|---|---|---|---|
| **C004238**<br>Atrial Fibrillation<br>(preferred)<br>Atrial Fibrillations<br>Auricular Fibrillation<br>Auricular Fibrillations | **L004238**<br>Atrial Fibrillation<br>(preferred)<br>Atrial Fibrillations | **S0016668**<br>Atrial Fibrillation<br>(preferred) | **A0027665**<br>Atrial Fibrillation<br>(from MSH)<br>**A0027667**<br>Atrial Fibrillation<br>(from PSY) |
| | | **S0016669**<br>(plural variant)<br>Atrial Fibrillations | **A0027668**<br>Atrial Fibrillations<br>(from MSH) |
| | **L004327**<br>(synonym)<br>Auricular Fibrillation<br>Auricular Fibrillations | **S0016899**<br>Auricular Fibrillation<br>(preferred) | **A0027930**<br>Auricular Fibrillation<br>(from PSY) |
| | | **S0016900**<br>(plural variant)<br>Auricular Fibrillations | **A0027932**<br>Auricular Fibrillations<br>(from MSH) |

**Table 1.1:** Concept, Term, String and Atom Identifiers

```
C0024109|A3154872|SCUI|RO|C0264408|A2957612|SCUI|has_finding_site|R14028961|
    994883025|SNOMEDCT_US|SNOMEDCT_US|0|Y|O||

C0231335|A2926532|SCUI|RO|C0264408|A2957612|SCUI|occurs_in|R123147138|
    1795540028|SNOMEDCT_US|SNOMEDCT_US|0|Y|N||

C0006255|A3104303|SCUI|RO|C0264408|A2957612|SCUI|has_finding_site|R98157815|
    3465258024|SNOMEDCT_US|SNOMEDCT_US|1|Y|N||

C0028778|A2873893|SCUI|RO|C0264408|A2957612|SCUI|has_associated_morphology|
    R98053314|3419439024|SNOMEDCT_US|SNOMEDCT_US|1|Y|N||
```

**Figure 1.1:** Example of relations in the `MRREL` file.

tion.

The primary intra-source relations in the Metathesaurus are "distance -1" hierarchical relations, i.e., immediate parent, immediate child, and immediate sibling relations. Some of the intra-source vocabulary relations are statistical relations, such as co-occurence relations. All of these relations carry a general label (REL), describing their basic nature and are identified by their source. About a quarter of these relations also carry an additional label (RELA), obtained from a source vocabulary, that explains the nature of the relationship more exactly, such as is_a, occurs_in, is_finding_of_disease. Figure 1.1 gives an example of relations in the `MRREL` file. Concept `C0264408` ("Childhood asthma") is related by the relation `has_finding_site` to concept `C0024109` ("lung structure"), which states that childhood asthma happens in lung. It is also related to concept `"childhood"` by `occurs_in` relations, which states that the disease happens in childhood.

### 1.1.2 Semantic Network

The purpose of the UMLS' Semantic Network [McCray, 2003] is to provide a consistent categorization of all concepts represented in the Metathesaurus and to provide a set of useful relations between these concepts. The Semantic Network

does not contain specific information about concepts that have already been described in the Metathesaurus. Rather, it provides information about the set of basic semantic types which may be assigned to these concepts. It also defines the set of relations that may hold between the semantic types. The Semantic Network contains 133 semantic types and 54 relations. The semantic types are the nodes in the Network, and the relations between them are the links.

Each Metathesaurus concept is assigned at least one semantic type. The major grouping of semantic types includes: organism, anatomical structure, biologic function, chemical, physical object, and idea or concept. Figure 1.2 illustrates the hierarchy of semantic type in the Semantic Network. The semantic type "Biological Function has two children, "Physiologic Function" and "Pathologic Function", and each of these in turn has several children and grandchildren. Each child in the hierarchy is linked to its parent by the `isa` link, which is the primary relation in the Semantic Network.

In addition to the `isa` relation, a set of non-hierarchical relations between the types has been identified. These are grouped into five major categories, which are themselves relations: `physically related to`, `spatially related to`, `temporally related to`, `functionally related to`, and `conceptually related to`. Figure 1.3 illustrates an example of a relation in the Semantic Network. The `affects` relationship has six children, including `manages`, `treats`, and `prevents`. Figure 1.4 shows a part of the relations that exist between semantic types in the Semantic Network.

## 1.2 Retrieval Models

We use several different information retrieval models for this thesis. We built our IR system using Terrier [1], an open source search engine developed by University of Glasgow [Ounis et al., 2006]. For all of the retrieval models that we used, we use Terrier's implementation of that model. More technical information about Terrier will be covered in Chapter 3. In this section, we will explain about different information retrieval models that we use in this thesis, either for a retrieval process or other purposes.

### 1.2.1 Inverse Document Frequency

Inverse document frequency is one of the term weighting model that is often used in information retrieval. It is related to the vector space retrieval model. We can represent documents and queries as vectors with each component corresponding to one term in the dictionary, together with the weight of the term given by our defined weighting model. This approach is known as the vector space model [Salton et al., 1975]. Vector space model views a document as a bag of words. By representing documents and query as vectors, we can then measure similarity between the document and the query.

The value of the element of the document vector can be many things. The simplest way to represent a document is by using the frequency of each term in the document as the value of its element. This weighting model is called the

[1] `http://terrier.org/`

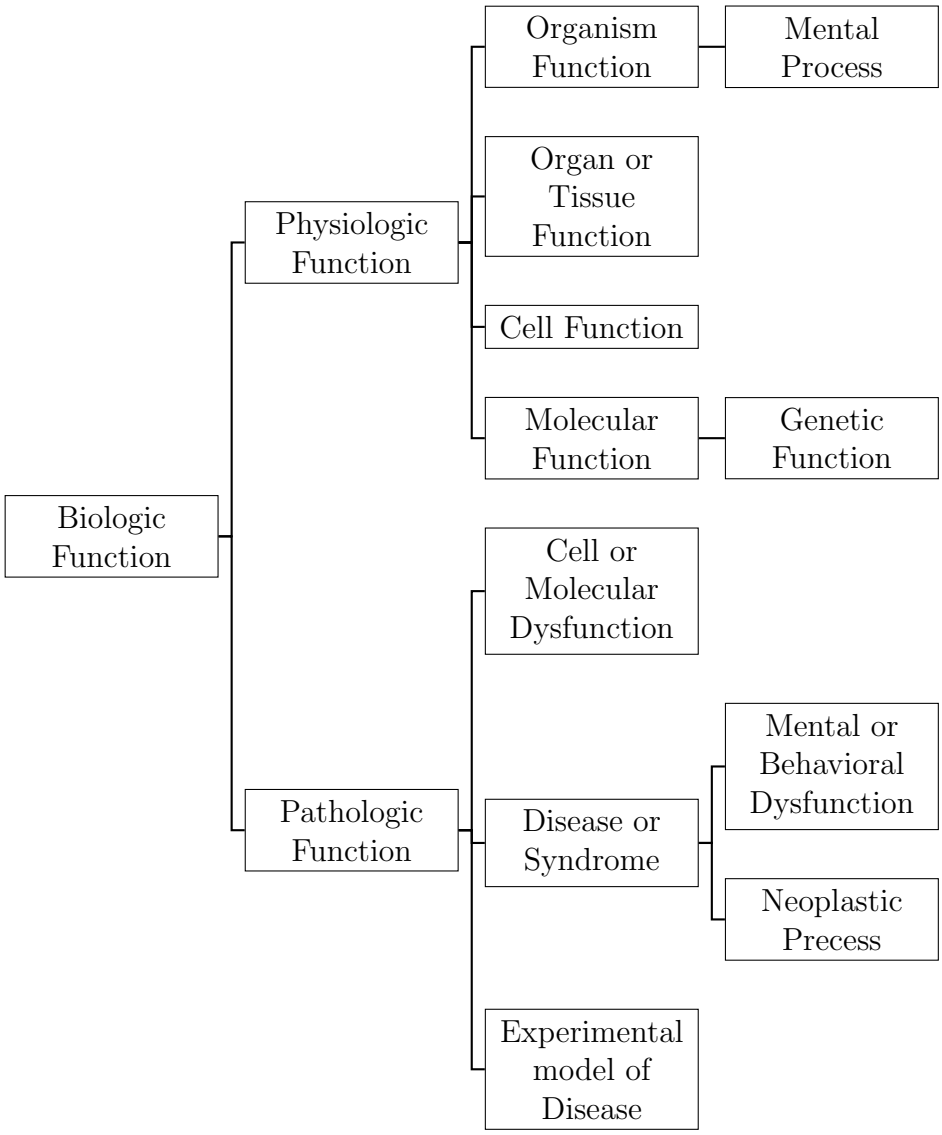



**Figure 1.2:** "Biologic Function" Hierarchy

term frequency (tf) model. Tf starts with an assumption that a document that mentions a query term more often is more relevant to the query, and therefore should receive a higher score. Tf model assigns to each term in a document a weight proportional to the number of occurrences of the term in the document [Luhn, 1957]. We then compute a score between a query term $t$ and a document $d$ by assigning the frequency of $t$ in $d$, denoted as $\mathrm{tf}_{t,d}$.

However, using term frequency as the weight of a term causes a critical problem. All terms are considered equally important for assessing relevance of the document to a query. However, certain terms have little or no discerning power in a document, while some other are a better feature to decide a relevance of a document to a query. As an example, a collection of articles about animals, the term "animal" would most likely occur in every document, while the term "arachnid" does not. If we are looking for an article about spider, the term "arachnid" would be a very good indicator that the document is relevant to the query. However, the term "animal" would most likely to occur more often in the

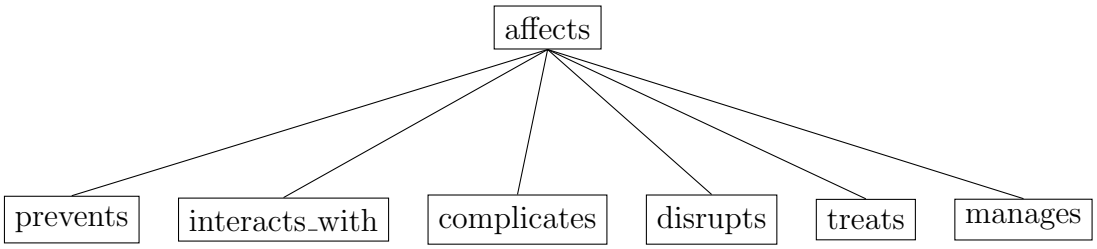



**Figure 1.3:** `affects` Hierarchy

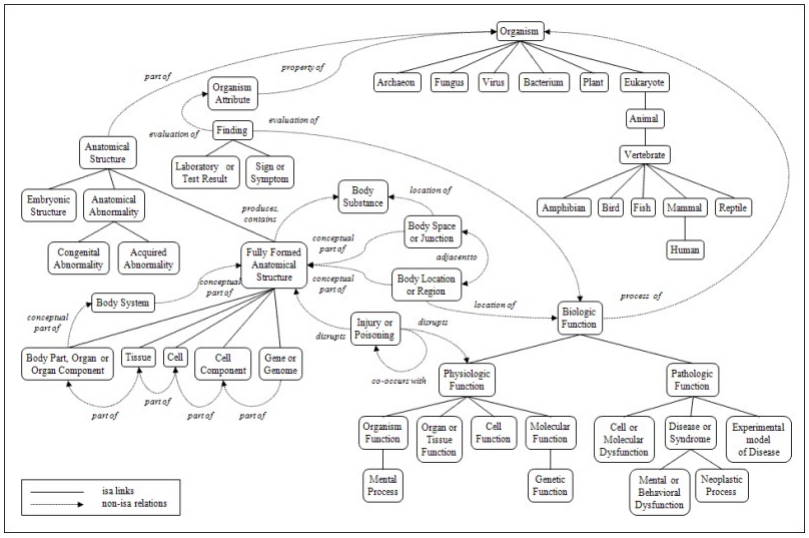



**Figure 1.4:** A part of relations between semantic types in the Semantic Network
*Source: [National Library of Medicine (US), 2009]*

document. Using only term frequency would give this term a higher weight, even though it is not a good indicator for relevance.

To avoid this problem, we have to introduce a mechanism for scaling down the effect of term frequency. This leads to the idea of inverse document frequency (idf). Document frequency $\mathrm{df}_t$ [Jones, 1972] is defined to be the number of documents in the collection where the term $t$ occurs. We then use this document frequency to give rare terms a higher value, by defining inverse document frequency

$$\mathrm{idf}_t = \log \frac{N}{\mathrm{df}_t}$$

where $N$ is the total number of documents in the collection.

In this thesis, we do not use tf, idf, or its combination as one of our retrieval models. However, we use idf as one of the selection criteria in our query expansion method. This will be explained in more detail in Subsection 3.3.1

### 1.2.2 Language Model with Dirichlet Prior

The basic idea behind language modeling approach to information retrieval is to estimate a language model for each document, and rank the documents by the

likelihood of the query based on the language model of the document. For a query $q$ and document $d$, the probability of the query being "generated" by the document's language model is denoted by $p(q|d)$. However, to rank the document, we want to obtain the posterior probability $p(d|q)$. By Bayes' formula, this probability can be obtained by

$$p(d|q) = \frac{p(q|d) \times p(d)}{p(q)}$$

where $p(d)$ is probability of $d$ is relevant to any query. $p(q)$ is probability of any query, and it is constant for all documents. This probability can therefore be ignored in the computation. Most of the works that have been done assumed that $p(d)$ is uniformly distributed. The language model used in most previous work is the unigram model

$$p(q|d) = \prod_{i=1}^{n} p(q_i|d)$$

where $q_i$ is the $i$-th query term. The unsmoothed unigram language model $p(w|d)$ is the maximum likelihood estimate, given by relative counts

$$p_m l(w|d) = \frac{c(w;d)}{\sum_{w' \in V} c(w';d)}$$

where $c(w;d)$ is the number of occurrences of word $w$ in document $d$, and $V$ is the set of all words in the vocabulary. However, this method will underestimate the probability of unseen words in the documents. The main purpose of smoothing is to assign non-zero probabilities to unseen words.

There are a many smoothing methods that have been proposed. In our task, we use Terrier's implementation of Bayesian smoothing using Dirichlet prior, a language model that uses Dirichlet distribution as its conjugate prior for Bayesian analysis [MacKay and Peto, 1994]. This retrieval performance of this implementation has been empirically verified to be similar to the one reported in [Zhai and Lafferty, 2004], where the model is given by

$$P_\mu(w|d) = \frac{c(w;d) + \mu p(w|C)}{\sum_{w' \in V} c(w';d) + \mu}$$

where $p(w|C)$ is the collection language model and $\mu > 0$. Terrier implementation set $\mu$ to 2500 by default.

### 1.2.3 Per-Field Normalization

Per-Field Normalization (PL2) is a model based on the Divergence From Randomness (DFR) framework [Amati and Van Rijsbergen, 2002]. It is based on the idea that the amount of information carried by a term $t$ in the document $d$ is proportional to the amount of divergence of the within-document term-frequency from its frequency within the collection. In other words,

$$\text{weight}(t|d) \propto -\log p_M(t \in d|C)$$

where $p_M(t \in d|C)$ is probability of term-frequency within the document d obtained by a model M of randomness.

DFR models are obtained by instantiating the three components of the framework: selecting a basic randomness model, applying the first normalization and normalizing the term frequencies. There are many ways to choose a basic DFR model. First normalization is including the risk of accepting a term as a descriptor in a document. If the term frequency is high, then the risk of not being informative is minimal. Second normalization principle is normalizing by the length of a document.

We used the Terrier's implementation of PL2F model [Macdonald et al., 2005]. In PL2, the frequencies from the different fields in the documents are normalized with respect to the statistics of lengths typical for that field. It is derived from a PL2 DFR model

$$score(d,q) = \sum_{t \in q} \frac{qtfn}{tfn+1}(tfn \times \log_2 \frac{tfn}{\lambda} + (\lambda - tfn) \times \log_2 e + 0.5 \times \log_2(2\pi \times tfn))$$

where $t$ is a query term in $q$, $\lambda$ is the mean and variance of a Poisson distribution, and $qtfn$ is the normalized query term frequency. The normalized term frequency $tfn$ is given by

$$tfn = tf \times \log_2(1 + c \times \frac{avg\_l}{l})$$

where $l$ is the document length and $avg\_l$ is the average document length in the whole collection, and $c > 0$.

### 1.2.4 LGD Weighting Model

In the LGD weighting model [Clinchant and Gaussier, 2009], DFR framework is used together with log-logistic distribution. It was proposed as a simplified DFR model based on only the first normalization principle and the log-logistic distribution. It can be defined by

$$score(d,q) = \sum_{w \in q \cap d} -c(w;q) \log(p(X \geq t(c(w;d), |d|)|r_w))$$

where $c(w;x)$ is the number of occurrences of word $w$ in $x$, $|d|$ is the length of document $d$, $t(c(w;d), |d|$ is the term frequency normalization, and $r_w$ is a parameter. In LGD model, $r_w$ is set to the document frequency of the word.

## 1.3 Query Reformulation

The same concept in a document or a query can be referred using different surface realizations. Different words that represent the same concept are synonymous. Synonymy can have a big impact on the performance of an information retrieval system. For example, when we are searching for "aircraft", we also want the documents that contains the word "plane" to be retrieved. However, we only want the documents that use "plane" as a reference to "airplane" to be retrieved, and not the one that refer to woodworking plane or other plane. Users often attempt to handle this problem themselves by redefining the query, but it is

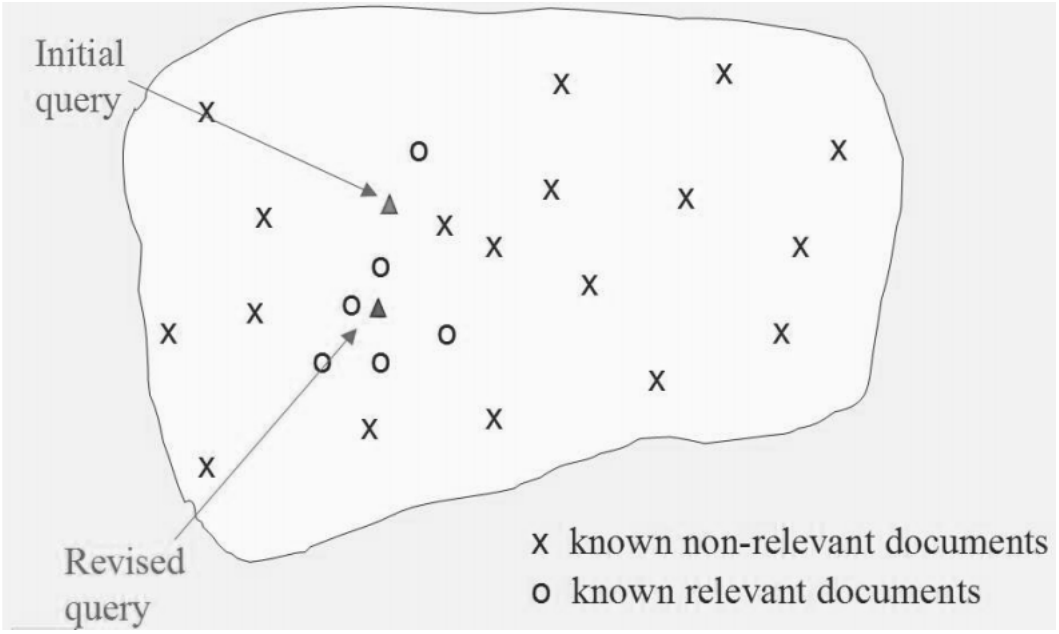



**Figure 1.5:** Illustration of the application of Rocchio's algoritm for relevance feedback.
*Source: [Manning et al., 2008]*

often difficult to formulate a good query, especially without knowing how the collection looks like. There are techniques for an information retrieval system to help with query reformulation.

The methods for this problem can be divided into two kinds: local and global methods. Local methods adjust the query relative to the documents that initially matched the query. One of the technique that we use in this thesis, blind relevance feedback, is a local method. Global methods, on the other hand, adjust the query without considering any relevant documents in the document collection. The other methods that we use in this thesis, query expansion, is a global method.

### 1.3.1 Blind Relevance Feedback

Relevance feedback is a method that tries to improve the retrieval result by involving user to give feedback on the relevance of documents in a set of results from an initial retrieval process. It came from the idea that while it might be difficult to formulate a query without knowing how the collection is like, judging whether a document is relevant or not given a query is relatively easy. Relevance feedback chooses important terms, or expressions in the pool of previously retrieved documents that have been judged as relevant by the users. The contribution of terms included in previously retrieved non-relevant documents can also be reduced, while terms included in relevant documents can be given more contribution. In a way, it "moves" the query nearer to the relevant documents space and further from the non-relevant documents space.

The Rocchio's algorithm is a well-known algorithm for relevance feedback. It incorporates relevance feedback into the vector space model. Given the original

query vector $\vec{q}_0$, we want to construct the modified query vector $\vec{q}_m$ as follows

$$\vec{q}_m = \alpha\vec{q}_o + \beta\frac{1}{D_r}\sum_{\vec{d}_j \in D_r}\vec{d}_j - \gamma\frac{1}{D_n r}\sum_{\vec{d}_j \in D_n r}\vec{d}_j$$

where $D_r$ and $D_n r$ are the set of relevant and non-relevant documents, respectively. $\alpha$, $\beta$, and $\gamma$ are the weights for the terms. Figure 1.5 illustrates the effect of Rocchio's algorithm in moving the query nearer the space of relevant documents [Manning et al., 2008].

Relevant feedback can be very useful to improve precision and recall in an IR system. However, it put the burden of giving feedback on users and users are often not willing to provide an explicit feedback, or are not willing to prolong the time it takes for the search operation. Blind relevance feedback, also known as pseudo relevance feedback, automates the manual part of relevance feedback by providing a method for local analysis. The system performs an initial retrieval process, and instead of asking users to judged the retrieved documents, it assumes that the top $k$ ranked documents are relevant, and perform the relevance feedback based on this assumption. This approach works, and in some cases work better than global methods. However, there is a danger of a bias. For example, in the case of a query about "rainforest", if the first $k$ queries in the initial retrieval is about the rain forest in Indonesia, the query might drift in the direction of documents about Indonesia.

### 1.3.2 Query Expansion

Query expansion is one method for query reformulation in information retrieval. Unlike relevance feedback, where users give additional input on documents, in query expansions users gives additional input on query terms. Most of web-based search engines nowadays give suggestions of related queries in response to users' queries. The most common way to do query expansion is by using some form of thesaurus. For each term $t$ in the query, the thesaurus can be used to automatically expand the query using synonyms or other related words. There are several methods for building a thesaurus for query expansion.

1. Maintaining a controlled vocabulary. For each concept, there is an assigned canonical term.

2. Manually constructing a thesaurus, where human editors have assign different names for concepts without any canonical terms. The UMLS Metathesaurus that we use in this thesis is an example of this method.

3. Automatically deriving a thesaurus, where word co-occurrence statistics over a collection is used to automatically build a thesaurus.

4. In case of web search, using query log where we utilize query reformulation from other users to make suggestion to new users. This require a large volume of queries, which is why this approach is more suitable for web-based systems.

Query expansion can also be combined with relevance feedback by adding candidate terms that appear in the relevant documents. It can also be combined with term weighting. Expansion terms can be weighted differently from the original terms. In Terrier, this can be done by utilizing Terrier's query language[2]. Terrier query language has several operators with different functions. In our experiment, we used the ^operator that is used to assign weights to words. `term1^2` means that the weight of `term1` is multiplied by 2. More about Terrier query language will be explained in Subsection 3.1.3.

## 1.4 Evaluation Metrics

We use three kinds of metrics for our evaluation purpose: precision, NDCG, and metrics related to the number of relevant documents retrieved.

### 1.4.1 Precision

The notion of precision is first defined for unranked retrieval. Precision measures the fraction of retrieved documents that are relevant. Given a set of $n$ retrieved documents, with $m$ retrieved documents among them, precision is simply

$$\text{Precision} = \frac{m}{n}$$

This metric of precision is computed over the whole set of retrieved documents. However, for a ranked-retrieval system with a lot of retrieved documents, the precision of the whole retrieved set does not matter. For example, in web search, user will perhaps only see the top $k$ documents on the first or second page. This means, we want to measure how precise is our retrieval process in the top $k$ retrieved results. This way of measuring precision at fixed low levels of retrieved result is referred to as precision at $k$. In our evaluation, we use precision at 5 (P@5) and precision at 10 (P@10) as one of our evaluation metrics.

Another metric that we use for our evaluation is the Mean Average Precision (MAP). For one single information need (represented by a query), average precision is the average of precision value for the set of top $k$ documents that exist after each relevant document is retrieved. This value is then averaged over all queries. In other words, for a set of queries $q_j \in Q$ with a set of relevant documents $\{d_1, ..., d_{m_j}\}$, and $R_{jk}$ is a set of ranked retrieval results from the top result until document $d_k$, MAP is defined as

$$\text{MAP}(Q) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_{k=1}^{m_j} \text{Precision}(R_{jk})$$

### 1.4.2 Normalized Discounted Cumulative Gain

Discounted Cumulative Gain (DCG) is designed for graded relevance, where every relevant document is given a non-binary scores according to its relevance to the query. It uses two assumptions:

[2] `http://terrier/org/docs/v4.0/querylanguage.html`

- highly relevant documents are more useful than marginally relevant documents (which are more useful than irrelevant documents).
- highly relevant documents are more useful when they are in the top ranks of the retrieval results.

Graded relevance is used as a measure of usefulness, or gain, from observing the document. Coming from the second assumption, this gain can be reduced or discounted at lower ranks. For a query, DCG at rank $k$ is defined as

$$\mathrm{DCG}_k = \sum_{m=1}^{k} \frac{2^{R(m)-1}}{\log(1+m)}$$

where $R(m)$ is the relevance score of the document in rank $m$. Normalized Discounted Cumulative Gain is a measure of DCG across queries. Let $R(j,d)$ be the relevance score of document $d$ for query $j$. NDCG at $k$ is defined as

$$\mathrm{NDCG}(Q,k) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} Z_k \sum_{m=1}^{k} \frac{2^{R(m)-1}}{\log(1+m)}$$

where $Z_k$ is a normalization factor which is calculated to make it so that a perfect ranking's NDCG at $k$, where every relevant documents are in the top $k$ rank, is 1.

### 1.4.3 Relevant Documents and Unjudged Documents

In our evaluation, we also show the number of relevant documents that are retrieved by the systems (rel_ret). However, the coverage of the assessment file that are provided by the CLEF eHealth organizer does not cover the entire data set. For every participants, only the top 10 documents from Run 1, Run 2, and Run 3 are judged. Therefore, there is a possibility that there are documents that are relevant to the query that are not judged yet. These documents can be located either beyond rank 10 in Run 1-3 or even in the top 10 retrieved documents in the other runs. These documents are therefore not considered as relevant when we are evaluating the systems

For this reason, it is important in our evaluation to know the coverage of the assessment file on the result of that particular run, especially when comparing two different runs. If there are a lot of unjudged documents in the retrieved list, there is a possibility that those documents might be relevant and will therefore affect the value of precision and NDCG. For example, if the documents that are not yet judged turns out to be relevant, we might obtain a higher precision and NDCG for that system. We define an evaluation metric, UNJ@10, as one of our metrics. This metrics measures the number of unjudged document of the first 10 retrieved documents for each query of a system, relative to the number of queries. The value of this metric is normalized by 10 so that it is between 0 and 1. A system has a 0 UNJ@10 score if all of the first 10 retrieved documents for all queries have their relevance judgment in the relevance assessment file. A system that has 1 UNJ@10 score does not have any documents in the first 10 retrieved documents for all queries evaluated in the relevance assessment file.

If UNJ@10 is high, this means that a system could have a higher perfomance if the unjudged documents are proven to be relevant. In an evaluation of a system, UNJ@10 goes down as the other metrics such as precision and NDCG improve. Lower UNJ@10 on a particular system means that the result of that system is coverging to the result of the best systems in our submission to CLEF. This means that the evaluation using the relevance assesment file has a bias towards this result. A system that has low scores on precision and NDCG but high score of UNJ@10 does not necessarily have a worse performance in the retrieval process compared to such system. It can be that this system actually has a better performance, but the result is very different from the results on the systems which results are evaluated for the relevance assessment file. We are aware of this bias towards similar system, and that the comparison of system might be unfair. However, the only solution for this problem is to do additional relevance assesment, which we did not perform for this thesis.

### 1.4.4 Wilcoxon Signed-Rank Test

When we observe two systems with differences in their evaluation scores, it is important to know whether that differences are really meaningful or simply due to chance. Statistical significance tests are a useful tool for this purpose. There are several different categories of significance test, and the most common methods are the parametric tests, where certain assumptions are made about the measurements of the distribution and their error. Another kind of significance tests are non-parametric tests, where no distributional assumptions are needed for the test to be valid.

When we perform a significance test to two different information retrieval systems, we define our preliminary assumption, or null hypothesis $H_0$, to be that all the retrieval methods being tested are equivalent in terms of performance [Hull, 1993]. The chosen significance test will attempt to disprove this hypothesis by determining a p-value, which is the probability that the differences could be merely by chance. We determine a significance level $\alpha$ for the testing, which is the probability of rejecting the null hypothesis when it is in fact true (Type I error). If the p-value is less than $\alpha$, we can conclude that the methods are significantly different. A smaller $\alpha$ means that the test is stricter and there is less chance of making a Type I error. On the other hand, the smaller the alpha level the bigger the chance of not rejecting the null hypothesis when it is in fact false (Type II) error. The common value of $\alpha$ is 0.05, and we use this value in our experiments.

We assume that the queries are independent. [Hull, 1993] states that since the performance differences between queries are greater than between methods, measurements should be viewed as matched pair where we analyze the difference between scores for each query. In this thesis, we use the paired Wilcoxon signed-rank test [Wilcoxon, 1945] for our significance tests. This method is one of the proposed method in [Hull, 1993]. The Wilcoxon test replaces each different between methods with the rank of its absolute value, and then multiply it with its sign. The sum of this value is then compared for each group to its expected value under the assumption that the two group is equal.

$$T = \frac{\sum R_i}{\sqrt{\sum R_i^2}}$$

where $R_i = \text{sign}(D_i) \times \text{rank}|D_i|$. In this thesis, we compare the methods on their precisions and NDCGs using this test.

# 2. Dataset

## 2.1 Document Collection

For this thesis, we use the document collection provided by the organizer of CLEF eHealth 2015 Task 2. This collection is formed by a large web crawl of one million document that has been made available to CLEF eHealth by the Khresmoi project [Aswani et al., 2012]. This collection consists of web pages that cover a broad variety of medical health topic. These web pages are targeted to both general public (laypeople) and health professionals. Most of the crawled domains are health and medicine websites that have been certified by the Health on the Net (HON) Foundation[1], as well as others popular health and medicine websites such as Drugbank[2] (a Canadian drugs database), Diagnosia[3] (a multilingual European drug database), and Tripanswers[4] (a collection of clinical questions and their answers).

The documents in the collection are provided as raw web pages including all the HTML (Hyper Text Markup Language) markup, CSS style, and Javascript codes. One file in the set could contains multiple pages from the web. Every page in the document starts with an ID, date, the URL of the page, and is followed by the content. The end of a page is marked with a line that contains `#EOR`. Figure 2.1 shows a snippet of a raw document file, containing one page from a website.

There is 2883673 unique terms in the collection, according to the indexing process using Terrier. Before indexing, we first clean the documents so that the system only indexes the content relevant to the retrieval process. In doing this, we utilize the HTML-Strip[5] Perl module to remove the noises in the document. This reduces the total size of the collection from 41,628 MB to 6,821 MB, which is about 16% of the original size. In the dataset, there are pages that contain binary files such as pdf files, ppt files, and zipped files. We were not able to parse this files, therefore we removed it from the test set. Other methods to clean the documents were previously tried, but HTML-Strip performed the best when the resulting documents are used for a retrieval task [Saleh and Pecina, 2014].

## 2.2 Queries

For the purpose of this thesis, we use test queries distributed by CLEF eHealth organizer for Task 3a of 2014 [Goeuriot et al., 2014] and Task 2 of 2015. There are 50 queries in the set of 2014 queries and 66 queries in the set of 2015 queries. Table 2.1 shows some basic statistic from the query set. Even though both of the query sets are aimed to mimic the use of medical information system by laypeople, the characteristic of the 2014 set and 2015 set is very different. This is because they represent different possible use cases for an information retrieval system.

[1] `http://www.healthonnet.org`
[2] `http://www.drugbank.ca/`
[3] `http://www.diagnosia.com/`
[4] `http://www.tripanswers.org/`
[5] `http://search.cpan.org/dist/HTML-Strip/Strip.pm`

```
#UID:publi0841_12_006379
#DATE:201209
#URL:http://publications.nice.org.uk/suction-diathermy-adenoidectomy-ipg328
#CONTENT:
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head></head>
  <body>
    &iuml;&raquo;&iquest;
    <title>IPG328 - Suction diathermy adenoidectomy- National Institute
    for Health and Clinical Excellence</title>
    <meta name="DC.title" content="IPG328 ... />
    ...
    ...
    ...
    ...
    <script type="text/javascript">//<![CDATA[var _tag=new WebTrends();
    _tag.dcsGetId();//]]></script>
    <script type="text/javascript">//<![CDATA[_tag.dcsCustom=function()
    {// Add custom parameters here.//_tag.DCSext.param_name=param_value;}
    _tag.dcsCollect();//]]></script>
    <noscript>
      <div>
        <img alt="DCSIMG" id="DCSIMG" width="1" height="1"
        src="http://statse.webtrendslive.com/dcs5qj9duvz5bdrrrup1evn7i_2c6p/
        njs.gif?dcsuri=/nojavascript&amp;WT.js=No&amp;WT.tv=9.3.0&amp;
        WT.dcssip=www.evidence.nhs.uk" />
      </div>
    </noscript>
  </body>
</html>
#EOR
```

**Figure 2.1:** A snippet of a raw document file.

| query set | queries | avg. title length | relevant documents |
|---|---|---|---|
| CLEF 2014 test set | 50 | 4.30 | 3,209 |
| CLEF 2015 test set | 66 | 5.03 | 1,972 |

**Table 2.1:** Statistics of the query sets

The 2014 queries model the queries used by laypeople who want to find more about their disorders, once they have examined their discharge summary. A discharge summary is a clinical report prepared by a physician or other health professionals at the conclusion of a hospital stay or series of treatments. It outlines the patient's chief complaint, the diagnostic findings, the therapy administered and the patient's response to it, and recommendations on discharge. Discharge summaries are a semi-structured document that can be considered as a description of the context in which the patient has been diagnosed with a given disorder. 85% of the discharge summaries used in the task contained the discharge diagnosis field. Figure 2.2 shows a sample of a discharge summary as appeared in [Goeuriot et al., 2014]. The queries are manually constructed by experts in the medical field given discharge summaries and discharge diagnosis. These experts select one disorder from the discharge diagnosis which patients might have questions about. This is why, the 2014 queries are centered around disorders or diseases. Figure 2.3 shows an example of a query provided in the 2014 shared task.

The 2015 queries are constructed differently. These queries aim to mimic the search behavior of laypeople who are confronted with signs, symptoms, and

```
Admission Date : [**2014-03-08**]
Discharge Date : [**2014-04-08**]
Date of Birth : [**1930-02-21**]
Sex : F
Service : CARDIOTHORACIC
Allergies :
        Patient recorded as having No Known Allergies to Drugs

Attending : [**Attending Info 565**]
Chief Complaint : Chest pain
Major Surgical or Invasive Procedure :
        Coronary artery bypass graft 4.
History of Present Illness :
        83 year-old woman, patient of Dr. [**First Name4 (NamePattern1)**]
        [**Last Name (NamePattern1) 500**], Dr. [**First Name ( STitle ) 5804**]
        [**Name ( STitle ) 2275**] , with increased SOB with activity,
        left shoulder blade/back pain at rest, + MIBI, referred for cardiac cath.
        This pleasant 83 year-old patient notes becoming SOB when walking up
        hills or inclines about one year ago. This SOB has progressively
        worsened and she is now SOB when walking [**01-19**] city block
        (flat surface).
        [...]

Past Medical History :
        arthritis ; carpal tunnel ; shingles right arm 2000; needs right knee
        replacement; left knee replacement in [**2010**]; thyroidectomy 1978;
        cholecystectomy in [**1981**]; hysterectomy 2001; h/o LGIB 2000-2001
        after taking baby ASA; 81 QOD
        [...]
```

**Figure 2.2:** Example of a discharge summary.

```
<topic><id>qtest2014.1</id>
    <discharge_summary>00211-027889-DISCHARGE_SUMMARY.txt</discharge_summary>
    <title>Coronary artery disease.</title>
    <desc>What does coronary artery disease mean? </desc>
    <narr>The documents should contain basic information about
    coronary artery disease and its care.</narr>
    <profile>This positive 83 year old woman has had problems with her heart with
    increased shortness of breath for a while. She has now received a diagnosis
    for these problems having visited a doctor. She and her daughter are seeking
    information from the internet related to the condition she has been diagnosed
    with. They have no knowledge about the disease.</profile>
  </topic>
```

**Figure 2.3:** Example of a query for 2014 shared task.

conditions and attempting to find out what kind of disease or disorder they might have. For example, when presented with a sign of chickenpox or varicella disease, a non expert might use queries like "red itchy spots on skin" to search for information that could allow them to diagnose themselves or better understand their condition. These queries often have a circumlocutory nature, where long, ambiguous wording or description of the condition is used instead of the medical name of the condition or disease. Research, such as [Zuccon et al., 2015], has shown that current search engines fail to effectively process such queries. Figure 2.4 shows an example of a query provided in the 2015 shared task.

We participated in CLEF eHealth 2015 Task 2 on medical information retrieval. For this participation, we were provided a list of queries from previous years and the current year. However, we were only provided the relevance assessment for the previous years' queries, and not for the current queries. For this reason, we used 2014's queries for our training set to tune our systems.We

```
<top>
<num>clef2015.test.2</num>
<query>lump with blood spots on nose</query>
</top>
```

**Figure 2.4:** Example of a query of 2015 shared task.

later tested the performance of our submissions on 2015's query set. However, as mentioned before, there is a big difference in the characteristic of the two sets. Because of this reason, there are differences on the performance of the configurations on the two system. For example, for the training set, the system that implemented the interpolation of Dirichlet Language Model, PL2F, and LGD model gave the best P@10 compared to the other systems. However, in the training set, the system that implemented query expansion using UMLS and interpolated PL2F and LGD gave the best performance. We will discuss more about our result on the shared task on Section 3.9.

Because of this characteristic difference, we decided to use a different division for training and test set for this thesis. Our training set contains 58 queries, with 25 queries randomly chosen from the 2014 queries set and 33 query randomly chosen from the 2015 query set. We use the rest of the queries to form two different test sets: one test set that contains the rest of the 2014 queries, and another set that contains the rest of the 2015 queries. We believe that by dividing the query sets in this manner, we would be able to measure the performance of the system for queries with different characteristic. This way, we would also be able to represent two different use cases of medical information retrieval in our training set and test sets.

## 2.3 Annotation Process

In order to be able to utilize the UMLS concepts in our retrieval process, we first have to annotate both the document set and query set by mapping the text to UMLS concept IDs. For this purpose, we utilize MetaMap[6], a tool to map biomedical text to the UMLS Metathesaurus [Aronson and Lang, 2010]. MetaMap is highly configurable, and has a lot of options that can be used to annotate the document. We mainly use two of those options: `-I` to output the CUI from each concept and and `-y` to enable word sense disambiguation. Word sense disambiguation is needed not only because of ambiguity of words, but also because of ambiguity of a scope of a concept. Some of the concepts in the UMLS Metathesaurus can actually be broken down into smaller concepts. For example, "lung cancer" can be taken as one concept, but it can also be taken as two individual concepts "lung" and "cancer". We choose to use MetaMap's default value for this option, that is to use the narrowest concept possible.

We perform annotation of both document and query set. For documents, the result of this annotation process is a structured document that has the following fields.

1. **docid** is the document id that was previously in the UID field in the raw

[6] `http://metamap.nlm.nih.gov/`

```
<doc>
  <docid>wiki.0842_12_009733</docid>
  <title>
    Testing for Celiac Disease..
  </title>
  <title_concepts>
    C0683443 C0007570 C0521125..
  </title_concepts>
  <text>
    Intestinal biopsy is the gold standard for diagnosing celiac..
  </text>
  <text_concepts>
   C1704732 C0036563 C0423896 ...
  </text_concepts>
</doc>
```

**Figure 2.5:** Eexample of an annotated document.

document.

2. **title** is the title of the document.
3. **title_concept** is the output of MetaMap mapping process on the title field.
4. **text** is the content of the document.
5. **text_concept** is the output of MetaMap mapping process on the text field.

Figure 2.5 shows an example of the cleaned document. Each of the annotated query contains the following fields.

1. **id** is the query id. In the 2015 queries, it was previously in the **num** field.
2. **title** is the query title. In the 2015 queries, it was previously in the **query** field.
3. **ctitle** is the result of MetaMap mapping on the title field.

# 3. Implementation

## 3.1 Terrier

Terrier (Terrabyte Retriever) is an open source information retrieval platform that has been designed to efficiently scale up with the size of document collections, operating in either a centralised or a distributed setting [Ounis et al., 2005]. There are two main components in the overall architecture of the Terrier platform: the indexing component and the retrieval component. In the indexing process, Terrier processes documents in the collection and represents the information in the collection in form of an index containing per-document and whole collection statistics of the terms frequency. In the retrieval process, Terrier uses the retrieval model defined to weigh each document term and use this information to calculate the relevance score of a document to a given query. Terrier also provides a query language that allows users to formulate specific preference or weight for terms in their query.

### 3.1.1 Indexing

Indexing is the first process in Terrier's retrieval architecture. Figure 3.1 illustrates the overview of the indexing process in Terrier as seen in [Ounis et al., 2006]. Each document in the collection is tokenized and parsed. Terrier comes with various document parsers, which allows users to parse different kind of document formats such as PDF, HTML, Plain-Text, and many more. During indexing, Terrier assign each term extracted from the document three fundamental properties: the actual String textual form of the term, the position at which the term occurs in the document, and the fields of the document in which the term occurs [Ounis et al., 2007].

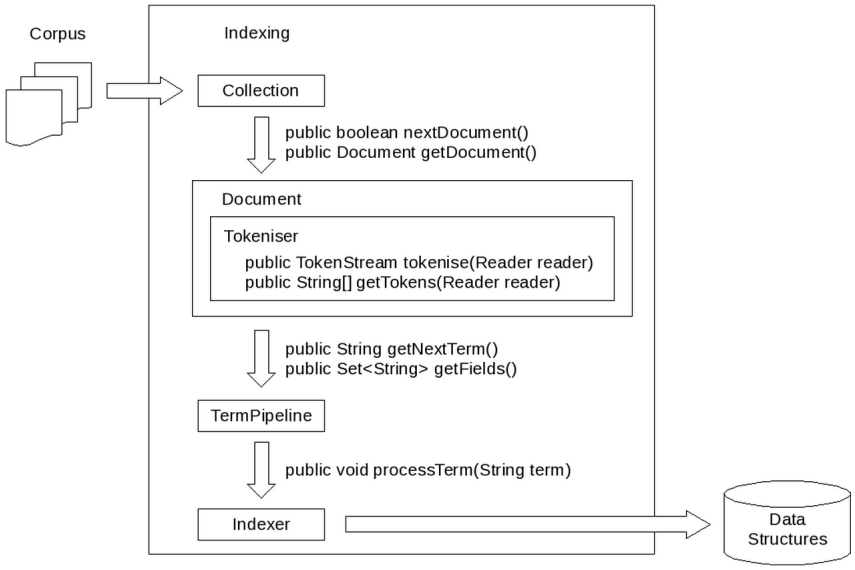



**Figure 3.1:** Overview of indexing process in Terrier.
*Source: [Ounis et al., 2006]*

During indexing, terms pass through the 'Term Pipeline' in Terrier. This pipeline is application-dependent, i.e. it is highly configurable. Term Pipeline allows terms to be transformed in various ways, using plug-ins such as n-gram indexing, stemming, removing stopwords in various languages, and so-on. The outcome of this process is passed to the Indexer, which writes the main data structures of the index. There are four data structures:

- **Lexicon** stores the term and its unique term ID, global statistics of the term which contain its collection frequency and document frequency, and the offsets of the posting list in the inverted index.
- **Inverted Index** stores the posting list of a term, that is the ID of the matching document and the term frequency of that term in the document.
- **Document Index** stores the unique document ID, the number of tokens in the document, and the offset of that document in the Direct Index
- **Direct Index** stores the terms and term frequencies of terms that appear in the document.

### 3.1.2 Retrieval

The retrieval module in Terrier offers flexibility to choose different weighting models as well as altering the scores of retrieved documents. Figure 3.4 shows an overview of the retrieval process in Terrier as shown in [Ounis et al., 2006]. The query is parsed and preprocessed before being passed to the Matching module. Terrier can parse query in a variety of formats. We used two different formats in our experiment: TREC format and Single Line format. For query in the TREC format, we defined four things: the tag that starts a query, the tag that contains the ID of the query, tags to be read when parsing the query, and tags to be ignored when parsing the query. Figure 3.2 shows an example of a query file in the TREC format. In the Single Line format, a single line is treated as an individual query. The advantage of using single line format is that we are able to use Terrier's query language capability (Subsection 3.1.3). However, we are not able to set up different kinds of fields in the query. We are able to indicate to Terrier whether a line starts with the query ID. If we do so, the string before the first whitespace of each line will be considered as the query id. Figure 3.3 shows an example of Single Line format query.

The Matching module employs a weighting model to estimate a relevance score of each document to that query. Each query term in the document is assigned a weight that measures the importance of that term to the document. Documents are matched to a query using the term weights, and documents are ranked according to their relevance scores. Terrier supports a wide range of weighting models, including Language Model with Dirichlet prior, and Divergence from Randomness (DFR) models, such as PL2F and LGD.

Terrier also allows scoring of documents to be altered at various stages of the retrieval to take into account additional types of retrieval evidence. The score of a term in a document can be altered by using TermScoreModifier. For example, users can ensure that query terms occur in a particular field in the document.

```
<topics>

<topic>
<id>qtest2014.4</id>
<title>Anoxic brain injury</title>
<ctitle>C0003132 </ctitle>
</topic>

<topic>
<id>qtest2014.8</id>
<title>Alcohol withdrawal seizures</title>
<ctitle>C0586323 </ctitle>
</topic>

<topic>
<id>clef2015.test.64</id>
<title>involuntary rapid left-right eye motion</title>
<ctitle>C2986385 C0439831 C0229090 C0205090 C0026597 C2986385
C0439831 C0205091 C0229089 C0026597 </ctitle>
</topic>

<topic>
<id>clef2015.test.65</id>
<title>weird brown patches on skin</title>
<ctitle>C0678579 C0991556 </ctitle>
</topic>
</topics>
```

**Figure 3.2:** Example of a query file in TREC format.

```
qtest2014.4 Anoxic brain injury C0003132
qtest2014.8 Alcohol withdrawal seizures C0586323
qtest2014.9 Right upper lobe pneumonia with cavitary lesion  C0585106 C0221198
```

**Figure 3.3:** Example of a query file in Single Line format.

Similarly, changing the score of a retrieved document can be achieved by applying DocumentScoreModifier.

### 3.1.3 Query Language

Terrier includes a query language that allows users to specify additional operations on top of a conventional query. Query language may specify that a query term should or should not appear in a document, appear in particular field, appear interchangeably with its synonym, and so on. An overview of available query language is as follows.

- `t1 t2` retrieves documents with either `t1` or `t2`.
- `t1^3.1` sets the weight of `t1` to 3.1.
- `+t1 -t2` retrieves documents that contain `t1` but not `t2`.
- "t1 t2" retrieves documents where both `t1` and `t2` appear next to each other.
- +(t1 t2) retrieves documents that contain both terms.
- `field:t1` specifies that `t1` must appear in a specific field.

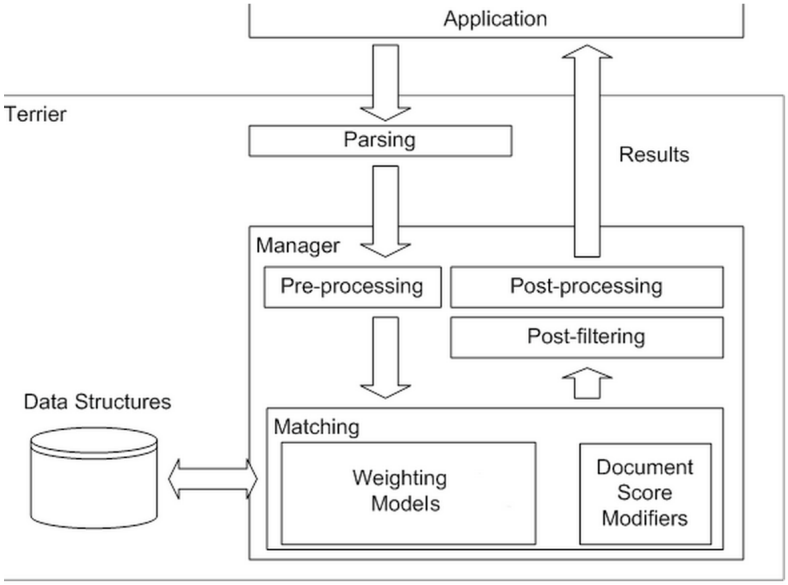



**Figure 3.4:** Overview of Retrieval process in Terrier.
*Source: [Ounis et al., 2006]*

The query language operations correspond to TermScoreModifier or DocumentScoreModifier modules, which are configured when the query is parsed. This query language functionality is applied after the indexing and retrieval stages.

## 3.2 Baseline System

We use the same baseline system that we used for our CLEF eHealth 2015 Task 2 Submission. In order to evaluate our query expansion systems, our baseline system only uses the original query terms in the retrieval process. We use Language model with the Dirichlet prior for the retrieval model of this system.

## 3.3 Expansion with Synonymous Terms

As if has been mentioned before, the UMLS Metathesaurus is organized by concepts, which represent meaning. Terms within the same concept are considered synonymous. We utilize this structure to expand our queries by expanding the terms with their synonyms. The information about concepts, their names, and their sources are located in the `MRCONSO` file in UMLS. Figure 3.5 shows an example of entries in the `MRCONSO` file. In our experiments, we mainly utilize the value of the first, seventh, and fifteenth column. The first column shows the CUI of the terms. Terms that are synonymous have the same value in their first column. The seventh column indicates whether a term is a preferred name of its concept. The value 'Y' indicates that a term is a preferred term, the value 'N' indicates otherwise. The fifteenth column is the String of the term.

In order to utilize this information for our query expansion, we first create a Python dictionary `concept_dict`. `concept_dict` has CUIs as its keys, and `lists` as its value. For every entry in the `MRCONSO` file, we add the String value in the

```
C0004886|ENG|S|L7167117|PF|S8463903|Y|A13748085||133313||MEDCIN|PT|133313|BCG
    vaccine, live, attenuated for bladder cancer (intravesical)|3|N||

C0004886|ENG|S|L7167117|VW|S10959622|Y|A16886102||133313||MEDCIN|SY|133313|live
    attenuated BCG vaccine for bladder cancer (intravesical)|3|N||

C0005684|ENG|S|L0266654|VC|S11862633|Y|A18664496|0000005336|0000001950||CHV|SY
    |0000001950|cancer of bladder|0|N|1536|

C0007107|ENG|S|L0266678|VC|S10932709|Y|A16873040||31744||MEDCIN|SY|31744|cancer
    of larynx|3|N||

C0007107|ENG|S|L0266677|VC|S0674962|N|A4349578|||CDR0000038962|PDQ|ET|
    CDR0000038962|Laryngeal cancer|0|N||
```

**Figure 3.5:** Example of entries in the `MRCONSO` file.

fifteenth column `concept_dict` under the entry's CUI that is found in the first column of the entry. Figure 3.6 shows an example of entries in `concept_dict`.

```
'C2051694': ["patient's impairment rating of thoracic spine", "patient's
    impairment rating of thoracic spine (diagnosis)"],

'C3680306': ['Eremophila alpestris adusta', 'Eremophila alpestris adusta (
    organism)'],

'C1180079': ['Sympathetic root of ciliary ganglion', 'Sympathetic root of
    ciliary ganglion', 'Branch of internal carotid plexus to ciliary ganglion',
    'Branch of internal carotid plexus to ciliary ganglion', 'Sympathetic branch
    of internal carotid plexus to ciliary ganglion', 'Sympathetic branch of
    internal carotid plexus to ciliary ganglion', 'Radix sympathica ganglii
    ciliaris (Plexus caroticus internus)', 'Radix sympathica ganglii ciliaris (
    Plexus caroticus internus)', 'Radix sympathica ganglii ciliaris'],
```

**Figure 3.6:** Example of entries in `concept_dict`.

We want to expand every query in our query sets with the synonyms of their terms. Every query in the query set is annotated with their terms' CUIs. For every concept, we want to generate a list of synonymous terms tp the original query terms as an expansion candidate and expand the query using some terms from the list. Therefore, we need to define a selection criterion to pick the expansion terms from the set of the expansion candidates. We experiment with two kind of selection criteria: the terms inverse document frequency (idf) in the document collection, and the concept preferred names.

### 3.3.1 Selecting Expansion Terms Using Inverse Document Frequency (idf)

One of the selection criteria that we choose is inverse document frequency (idf). The idea is that if a term happens very rarely in the document collection, it could be a good candidate of terms that has a discerning power in a document. In other words, a term with low document frequency, or high inverse document frequency, could be a good term to separate a document from other documents with a similar topic. Terrier stores document frequency in its Lexicon as one of the output of the indexing process.

For each CUI in each query, we retrieve a list of synonyms from `concept_dict`. We then tokenize the list of terms by word, and obtained a set of unique words expansion candidate for the concept. We first eliminate words that are already in the original query terms. We also eliminate words that are already among the current expansion candidates of that particular query as a whole, obtained from other CUIs in the query. We rank the candidates based on their idf, and choose $n$ terms with the highest idf. For concepts that have less than $n$ candidates, we add all of the candidates. We add these expansion terms to a new tag, `e_title`.

We experiment with the number of words per concept that we add as expansion terms. Figure 3.7 shows an example of queries expanded with $n = 5$.

```
<topic>
<id>qtest2014.4</id>
<title>anoxic brain injury</title>
<e_title> enceph dup encephalopathies hie </e_title>
<ctitle>C0003132 </ctitle>
</topic>

<topic>
<id>qtest2014.8</id>
<title>alcohol withdrawal seizures</title>
<e_title> alcoholrelated alcoh withdrawl ambiguous rum </e_title>
<ctitle>C0586323 </ctitle>
</topic>

<topic>
<id>clef2015.test.64</id>
<title>involuntary rapid left-right eye motion</title>
<e_title> orbital force qualifier dextro involuntary region od righting value
    sides motion lt levo agent motions quick os sided structure physical </
    e_title>
<ctitle>C2986385 C0439831 C0229090 C0205090 C0026597 C2986385 C0439831 C0205091
    C0229089 C0026597 </ctitle>
</topic>

<topic>
<id>clef2015.test.65</id>
<title>weird brown patches on skin</title>
<e_title> qualifier color colour patchs value transepidermal transdermalpatch </
    e_title>
<ctitle>C0678579 C0991556 </ctitle>
</topic>

</topics>
```

**Figure 3.7:** Example of queries expanded by terms with the highest idf score.

### 3.3.2 Selecting Expansion Terms Using Preferred Names

A word would have a maximum idf if it only occurs once in the collection. Some words that happen only once in he collection are actually misspelled words. This create a lot of noise in our approach of using idf as a selection criterion. For example, in the second query in Figure 3.7, the world 'alcoh' and 'withdrawl' appeared. For this reason, we experiment with utilizing one of the column in `MRCONSO` file, namely the preferred name flag. The idea is that if a term is a preferred name of a concept, then it is more likely to appear in a document that is talking about this concept than terms that are not a preferred name of the concept.

For a CUI in the query, instead of taking all the possible terms in Metathesaurus as its expansion candidate, we only take those that are the preferred names of the concept. We then use the same approach that we used before. We tokenize the terms by words to obtain a set of unique words, and then remove the words that are already in the original query terms or in the expansion candidate of the query as a whole. We choose $n$ terms with the highest idf. For concepts that have less than $n$ candidates, we add all of the candidates.

For this approach, we again experiment with the number of words per concept that we add as expansion terms. Figure 3.8 shows an example of queries expanded with $n = 5$. It can be seen in comparison to Figure 3.7 that even though some words are the same, there are some differences between the expansion terms chosen by the two methods.

```
<topic>
<id>qtest2014.4</id>
<title>Anoxic brain injury</title>
<e_title>encephalopathies syndrome damage anoxic encephalopathy </e_title>
<ctitle>C0003132 </ctitle>
</topic>

<topic>
<id>qtest2014.8</id>
<title>Alcohol withdrawal seizures</title>
<e_title>seizures fits epilepsy alcohol seizure </e_title>
<ctitle>C0586323 </ctitle>
</topic>

<topic>
<id>clef2015.test.64</id>
<title>involuntary rapid left-right eye motion</title>
<e_title>orbital eyes right force qualifier dextro sided righting value face lt
    part agent motions quick sides left side structure physical </e_title>
<ctitle>C2986385 C0439831 C0229090 C0205090 C0026597 C2986385 C0439831 C0205091
    C0229089 C0026597 </ctitle>
</topic>

<topic>
<id>clef2015.test.65</id>
<title>weird brown patches on skin</title>
<e_title>product transdermal qualifier color colour value patch transepidermal
    transdermalpatch </e_title>
<ctitle>C0678579 C0991556 </ctitle>
</topic>

</topics>
```

**Figure 3.8:** Example of queries expanded by preferred terms.

## 3.4 Expansion with Non-Synonymous Related Concepts

Aside from the synonymy relations from the concept structure, the UMLS Metathesaurus also contains different kind of non-synonymous relations between concepts. Most of these relations are unlabeled, where they only have relations ID but not the name that describes the relations. For the purpose of this thesis, we ignore the unlabeled relations and only focused on the labeled relations. There are 670

labeled relations in the Metathesaurus, most of them are `isa` and `inverse_isa` relations.

[Koopman et al., 2012] stated that even though empirical results show that considering related concepts alongside the original query can improve retrieval effectiveness, choosing which relations to consider and choosing the correct weight for them is a challenging issue. For our purpose, we choose the relations based on the characteristic of our query set. Most of our queries in our training set are queries that mimic the behavior of laypeople who are trying to find out about the disease that they might have based on some symptoms or signs. For this reasons, expanding our queries with concepts that are related to the original query concepts by relations that signifies findings or symptoms might prove beneficial.

There are several relations in the Metathesaurus that fit our purpose:

- `may_be_finding_of_disease`
- `disease_may_have_finding`
- `associated_finding_of`
- `disease_has_finding`
- `has_associated_finding`
- `is_finding_of_disease`

We ignore relations about findings that are not observable directly such as cellular findings, since laypeople are most likely to observe symptoms on physical level. We also make use of negative relations such as `is_not_finding_of_disease` and `disease_excludes_finding` to perform selection of the expansion candidates. Table 3.1 shows the number of occurrences of these relations in the UMLS.

| name | # occurrences |
|---|---|
| `may_be_finding_of_disease` | 12960 |
| `disease_may_have_finding` | 12960 |
| `associated_finding_of` | 12642 |
| `disease_has_finding` | 19141 |
| `has_associated_finding` | 12642 |
| `is_finding_of_disease` | 19141 |
| `is_not_finding_of_disease` | 9135 |
| `disease_excludes_finding` | 9135 |

**Table 3.1:** Number of occurrence of selected relations in UMLS.

As in expansion using synonyms, we first create a Python dictionary to store the information of the relations from the `MRREL` file. The difference is that in this dictionary, we store the CUI of the concept in the relations instead of the string of the terms. We want to avoid adding noise to the query by adding individual terms from the related concept. For every concept in a query, we get all related concepts from our dictionary. We exclude concept that has `is_not_finding_of_disease` and `disease_excludes_finding` relations with any concept in the query to avoid inconsistency by adding concepts that are the opposite of one of the query concepts.

We experiment with including the relations mentioned before individually. The amount of candidates from these individual relations are quite small, so we also use all the relations for our expansion. Figure 3.9 shows an example of queries expanded by this technique. The expanded concepts are contained within the `e_ctitle` tag. As can be seen, in some of the queries, there are no expansion candidate that were added.

```
<topics>
<topic>
<id>qtest2014.8</id>
<title>Alcohol withdrawal seizures</title>
<e_ctitle></e_ctitle>
<ctitle>C0586323 </ctitle>
</topic>

<topic>
<id>qtest2014.9</id>
<title>Right upper lobe pneumonia with cavitary lesion </title>
<e_ctitle></e_ctitle>
<ctitle>C0585106 C0221198 </ctitle>
</topic>

<topic>
<id>clef2015.test.64</id>
<title>involuntary rapid left-right eye motion</title>
<e_ctitle></e_ctitle>
<ctitle>C2986385 C0439831 C0229090 C0205090 C0026597 C2986385 C0439831 C0205091
    C0229089 C0026597 </ctitle>
</topic>

<topic>
<id>clef2015.test.65</id>
<title>weird brown patches on skin</title>
<e_ctitle></e_ctitle>
<ctitle>C0678579 C0991556 </ctitle>
</topic>
</topics>
```

**Figure 3.9:** Example of queries expanded by related concepts.

## 3.5 Blind Relevance Feedback

As a comparison to our method of query expansions using a thesaurus, we experiment with using blind relevance feedback for query reformulation. As opposed to query expansion using a thesaurus, blind relevance feedback is a local method of query reformulation. It adjust a query relative to the documents that match the query from an initial retrieval process.

In our implementation of blind relevance feedback, we first use our baseline system to do an initial retrieval process. We then take $n$ terms from the top $m$ documents that are retrieved by the initial retrieval. We use idf as our terms selection criterion, that is we select the top $n$ terms that are ranked by idf. From our participation in the CLEF eHealth 2015 Task 2 with the same data set, we found that using only the top 25 documents gave the best P@10. We experiment with different number of words that we add to the query. Figure 3.10 shows an example of queries expanded with this approach with $n = 5$.

```
<topics>
<topic>
<id>qtest2014.4</id>
<title>Anoxic brain injury</title>
<e_title>coma vegetative study hypothermia outcome </e_title>
<ctitle>C0003132 </ctitle>
</topic>

<topic>
<id>qtest2014.8</id>
<title>Alcohol withdrawal seizures</title>
<e_title>ciwa alcoholism chlordiazepoxide tremens delirium </e_title>
<ctitle>C0586323 </ctitle>
</topic>

<topic>
<id>clef2015.test.64</id>
<title>involuntary rapid left-right eye motion</title>
<e_title>dimensional galyfilcon sickness lens senofilcon </e_title>
<ctitle>C2986385 C0439831 C0229090 C0205090 C0026597 C2986385 C0439831 C0205091
    C0229089 C0026597 </ctitle>
</topic>

<topic>
<id>clef2015.test.65</id>
<title>weird brown patches on skin</title>
<e_title>melasma rash spots ppp psoriasis </e_title>
<ctitle>C0678579 C0991556 </ctitle>
</topic>

</topics>
```

**Figure 3.10:** Example of queries expanded by blind relevance feedback.

## 3.6 Field Weighting

For our query reformulation experiment, we have three different kinds of fields or zones that we can use for retrieval: original query terms, concept ids of original query terms, and expanded query terms or concepts. One way to perform retrieval with these three fields in Terrier is just to declare that some or all fields should be considered. This way, all of the different fields will be merge together as a bag of words and treated as equally valuable in the retrieval process.

Another way to use these fields for retrieval process is to give different weights to terms from different fields. Terms with higher weights will be given more importance in the retrieval process, i.e. those terms will get higher scores in the ranking process. For example, we could give original query terms more weight than their concepts. We could also give different weights to the original and expanded query terms . This bears some similarity to the Rocchio algorithm which gives different weight to the vector of original query terms, terms from documents judged relevant, and terms from documents judged irrelevant.

In Terrier, we can utilize its query language to achieve this. Terrier's query language have the ^operator that can be used to assign a weight to a term. t^y gives term t the weight y. However, the query language can only be used in Single Line query format, so we first have to convert our TREC-formatted query. Figure 3.11 shows an example of weighted Single Line query. In this example, the field `title`, `ctitle`, and `e_title` are given the weight 3.1, 2.6, and 1.1 respectively. For most of our query reformulation experiments, we have three fields that we experiment with: `title` is the original query terms, `ctitle` is the original query

```
qtest2014.26 gastrointestinal^3.1 bleed^3.1 gih^1.1 haemorrhag^1.1 gastroin^1.1
    haemorrh^1.1 gastrointestin^1.1 C0017181^2.6

qtest2014.27 aortic^3.1 valve^3.1 replacement^3.1 ARV^3.1 avr^1.1 replacements
    ^1.1 excision^1.1 replacement^1.1 procedure^1.1 C0003506^2.6

qtest2014.20 subdural^3.1 hematoma^3.1 dup^1.1 sdh^1.1 haematoma^1.1 hematomas
    ^1.1 hemorrhages^1.1 C0018946^2.6

clef2015.test.53 swollen^3.1 legs^3.1 extremity^1.1 feet^1.1 legs^1.1 swollen
    ^1.1 C0581394^2.6

clef2015.test.15 asthma^3.1 attack^3.1 attacks^1.1 nos^1.1 disorder^1.1
    asthmatic^1.1 C0347950^2.6

clef2015.test.41 eye^3.1 iris^3.1 large^3.1 greats^1.1 iris^1.1 qualifier^1.1
    big^1.1 larger^1.1 value^1.1 iri^1.1 structure^1.1 C0022077^2.6 C0549177^2.6
```

**Figure 3.11:** Example of weighted queries.

concepts, and `e_title` is the expanded query terms. An exception is for the case of `rel`, where we replace `e_title` with `e_ctitle`, which is the expanded query concepts.

## 3.7 Utilizing Semantic Network

One of the information that the UMLS Semantic Network provides is the semantic types of concepts in the Metathesaurus. This information could be useful to help with the retrieval process. Our idea is that in a query, terms with different semantic type could have different importance to retrieve relevant documents. As an example, in the query "swollen legs", user would be looking for documents that contain information about legs, that happen to be swollen. Documents that contain information about "swollen arms" should be considered as irrelevant even though they also contain the word "swollen", as they contain information about arms instead of legs. In other words, documents that are talking about legs should have a higher relevance score than documents that are talking about arms in this case.

There are two types of semantic types in the UMLS Semantic Network: Entity and Event. For this thesis, we only work with the Entity semantic type, in particular the semantic type related to anatomical structure or body part. The reason for this is that most of our training queries are describing physical symptoms. Figure 3.12 shows the hierarchy of the "Anatomical Structure" semantic type. From these types, we use "Fully Formed Anatomical Structure" and "Body Part, Organ, or Organ Component". We also use the semantic type "Body System", "Body Space or Junction", and "Body Location or Region" from the "Spatial Concept" hierarchy because these types also cover physical structure such as skin and joint.

We first create a list of terms of the concepts that are in the semantic types mentioned above. For each term in a query, if the term is defined as one of those semantic types in UMLS, we give a weight to that term using the ^operator. Figure 3.13 shows an example of queries after the weighting process.

Due to the number of relations available in the Semantic Network, a thorough investigation is needed to select which relations would be useful for query

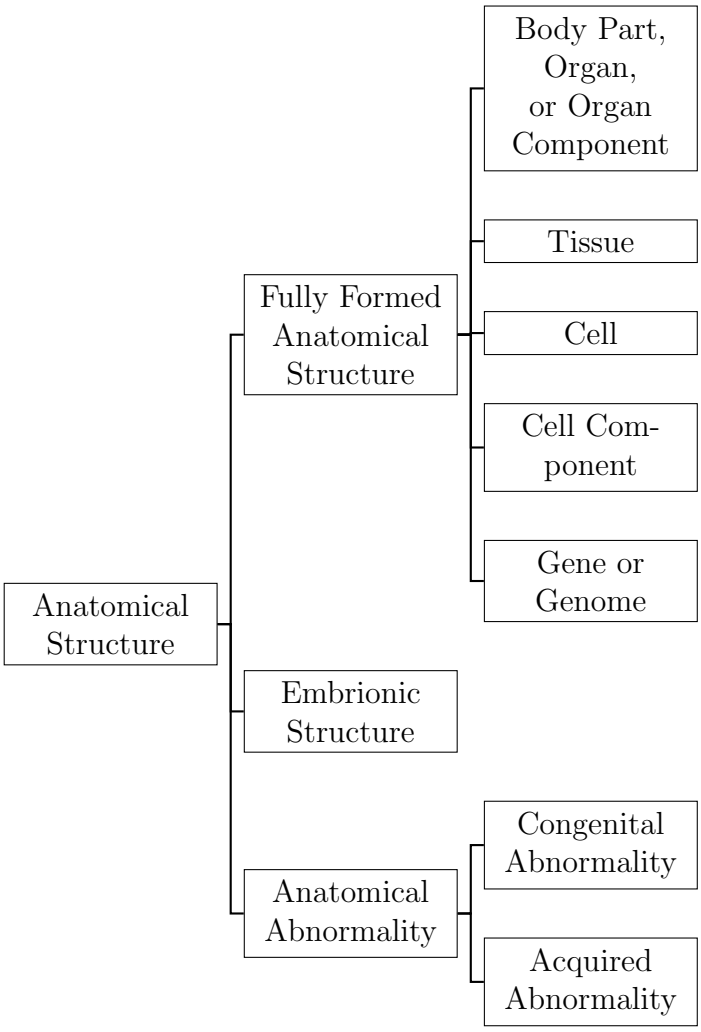



**Figure 3.12:** The "Anatomical Structure" hierarchy

expansion. We leave this for future work.

## 3.8 Linear Interpolation

In some of our experiments, we perform linear interpolation of the scores of two or three different systems. Given multiple ranked retrieval results from multiple different systems, we combine the scores given by individual systems to a new score, which generates a new ranked list of documents. We combine the scores using the following equation.

$$Score(D, Q) = \lambda \cdot Score_1(D, Q) + (1 - \lambda) \cdot Score_2(D, Q)$$

$\lambda$ are parameters that define how much weight does the score of one system involves the new scaring. These parameters add up to 1. We tune these parameters using our training list by iterating through combination of values.

```
qtest2014.24 Diabetes type 1^3.5 and heart^3.5 problems

qtest2014.23 coronary^3.5 artery^3.5 bypass

qtest2014.27 aortic^3.5 valve^3.5 replacement, ARV

clef2015.test.20 movement difficulty with involuntary hand^3.5 trembling

clef2015.test.8 cloudy cornea^3.5 and vision problem

clef2015.test.44 nail^3.5 getting dark
```

**Figure 3.13:** An example of queries with term weighted based on their semantic type.

## 3.9 CLEF eHealth 2015 Shared Task

Our experiments that are described in this thesis are preceded by our participation in CLEF eHealth 2015 Task 2 in medical information retrieval. We used part of the methods that we use in this thesis for our submission to this task. However, the experiments in our submission and the experiments in this thesis differ on the split of training query set and test query sets. In our submission to CLEF eHealth 2015, we used the 2014 query set as our training set, and 2015 query set as our test set. We use different split of queries for our training and test sets for the reason that has been explained in Section 2.2.

In our submission, we performed query expansion using synonymous terms with idf as the terms selection criterion. We also performed blind relevance feedback as a comparison. We also performed linear interpolation of scores of multiple retrieval model, by tuning the lambda value using our training queries. In the evaluation, we found that linear interpolation improve the performance of the system compared to the individual systems. We also found that some of the systems that used our query expansion method, while system using blind relevance feedback decreased the performance. Our submission to CLEF eHealth 2015 is described in [Saleh et al., 2015]. The official result of the entire task is described in [Palotti et al., 2015]. Our submissions performed above the median of the participants.

# 4. Performance on Training Set

In this chapter, we present the results of different experiments with different methods. For all of the tables, the best value for each metric is emphasized with bold. We perform paired Wilcoxon signed-rank test to the methods in each table, with $\alpha = 0.05$. The values which differences are not statistically significant with the best value are printed in italics.

## 4.1 Using Original Query Terms

In order to compare the effect of our query expansion system, we employ systems that only use original query terms in the retrieval process. We implement systems with three different retrieval model: Language model with Dirichlet prior (`dir`), Per-Field Normalization (`pl2f`), and the LGD weighting model (`lgd`). Table 4.1 shows performance of the systems on the training set.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| `dir` | **0.6207** | **0.5776** | **0.5568** | **0.5471** | 0.3412 | 2267 | 0.010 |
| `lgd` | *0.5862* | *0.5466* | *0.5297* | *0.5208* | 0.3283 | 2271 | 0.009 |
| `pl2f` | *0.5897* | *0.5517* | *0.5366* | *0.5288* | **0.3509** | 2289 | 0.007 |

**Table 4.1:** Performance of systems using original query terms on training set

The system using language model with Dirichlet prior has the highest performance across all of the evaluation metrics, except of MAP and number of relevant documents retrieved. This system also has the highest UNJ@10 score. as mentioned before, we perform paired Wilcoxon signed-rank test to the methods, with $\alpha = 0.05$. The differences of values in italics with the best values in bold are not statistically significant.

## 4.2 Expansion Using Synonymous Terms

In this section, we present the results of our experiments with our query expansion implementation using synonymous terms, varying on the number of additional expansion terms added to each query per concept. There are two kinds of systems that are presented in this section: systems which use idf as their term selection criteria, and systems which use preferred names as their term selection criteria. For each kind of system, we use the same three models that we use for our experiments on the unexpanded systems: Language model with Dirichlet prior (`dir`), Per-Field Normalization (`pl2f`), and the LGD weighting model (`lgd`). For each of the system with expanded queries, we compare its performance with that of the system using the same model that only uses original query terms. For each criteria and each retrieval model, we choose the system that has performance to be used later in our experiment with field weighting. For readability, we give each systems an id which has the format of [`model`].[`criteria`].[`# expansion terms`]

### 4.2.1 Selecting Expansion Terms Using Inverse Document Frequency (idf)

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| dir | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| dir.idf.1 | **0.5103** | **0.4603** | **0.4574** | **0.4330** | **0.2467** | 2024 | 0.217 |
| dir.idf.2 | *0.4966* | 0.4345 | 0.4362 | 0.4069 | 0.2322 | 2005 | 0.262 |
| dir.idf.3 | *0.4966* | *0.4500* | *0.4346* | *0.4159* | 0.2261 | 1970 | 0.281 |
| dir.idf.4 | *0.4690* | 0.4224 | 0.4135 | 0.3951 | 0.2066 | 1958 | 0.319 |
| dir.idf.5 | *0.4483* | 0.4172 | *0.4054* | *0.3932* | 0.2028 | 1957 | 0.334 |
| dir.idf.6 | *0.4517* | *0.4172* | *0.4042* | *0.3911* | 0.2015 | 1943 | 0.336 |
| dir.idf.7 | *0.4655* | *0.4207* | *0.4117* | *0.3924* | 0.2004 | 1940 | 0.352 |
| dir.idf.8 | *0.4621* | 0.4172 | *0.4152* | 0.3922 | 0.1989 | 1935 | 0.350 |
| dir.idf.9 | *0.4621* | 0.4121 | *0.4216* | *0.3944* | 0.1977 | 1930 | 0.364 |
| dir.idf.10 | 0.4034 | 0.3759 | 0.3623 | 0.3576 | 0.1802 | 1860 | 0.419 |

**Table 4.2:** Performance of query expansion system using synonyms with idf as their terms selection criteria on different numbers of words added as the expansion terms, implementing Dirichlet prior model.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| lgd | 0.5862 | 0.5466 | 0.5297 | 0.5208 | 0.3283 | 2271 | 0.009 |
| lgd.idf.1 | **0.4724** | **0.4155** | **0.4261** | **0.4040** | **0.2309** | 2022 | 0.266 |
| lgd.idf.2 | *0.4655* | *0.4052* | *0.4095* | *0.3796* | 0.2128 | 1996 | 0.305 |
| lgd.idf.3 | *0.4345* | *0.3897* | *0.3946* | *0.3686* | 0.1980 | 1940 | 0.319 |
| lgd.idf.4 | 0.4138 | *0.3655* | *0.3823* | *0.3551* | 0.1863 | 1905 | 0.366 |
| lgd.idf.5 | *0.4172* | *0.3569* | *0.3851* | *0.3551* | 0.1862 | 1892 | 0.398 |
| lgd.idf.6 | *0.4103* | *0.3586* | *0.3807* | *0.3554* | 0.1852 | 1883 | 0.390 |
| lgd.idf.7 | *0.4345* | *0.3914* | *0.4033* | *0.3836* | 0.1962 | 1943 | 0.350 |
| lgd.idf.8 | *0.4241* | *0.3897* | *0.4004* | *0.3835* | 0.1956 | 1928 | 0.357 |
| lgd.idf.9 | *0.4276* | *0.3810* | *0.4062* | *0.3814* | 0.1986 | 1918 | 0.372 |
| lgd.idf.10 | *0.4034* | *0.3724* | *0.3858* | *0.3674* | 0.1880 | 1881 | 0.419 |

**Table 4.3:** Performance of query expansion system using synonyms with idf as their terms selection criteria on different numbers of words added as the expansion terms, implementing LGD weighting model.

In this part, we present the result of our experiment on query expansion using idf as term selection criteria. Table 4.2 shows the results of our experiments using the language model with Dirichlet prior, compared with the system using the same model that only uses the original query terms. Adding only one expansion term gives the best performance in all of the metrics. Our evaluation metrics decrease as we add more expansion terms to the query, but it increases slightly when we add seven expansion terms.For this setting, our best system's performance is still lower than the performance of the performance of the same model when applied on queries without any expansion terms. However, our best system has a considerably higher UNJ@10 score.

Table 4.3 shows the results of our experiments using the LGD weighting model, compared with the system using the same model that only uses the original

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| pl2f | 0.5897 | 0.5517 | 0.5366 | 0.5288 | 0.3509 | 2289 | 0.007 |
| pl2f.idf.1 | **0.4276** | **0.4103** | **0.4005** | **0.3959** | **0.2389** | 2099 | 0.236 |
| pl2f.idf.2 | *0.4241* | *0.3914* | *0.3973* | *0.3820* | 0.2248 | 2079 | 0.276 |
| pl2f.idf.3 | *0.4000* | 0.3828 | *0.3734* | *0.3672* | 0.2111 | 2051 | 0.309 |
| pl2f.idf.4 | *0.3828* | *0.3638* | *0.3574* | 0.3518 | 0.2052 | 2042 | 0.352 |
| pl2f.idf.5 | *0.3862* | *0.3638* | *0.3628* | *0.3553* | 0.2078 | 1982 | 0.362 |
| pl2f.idf.6 | *0.3828* | 0.3517 | *0.3668* | *0.3487* | 0.2094 | 1989 | 0.379 |
| pl2f.idf.7 | *0.4000* | *0.3569* | *0.3703* | *0.3505* | 0.2147 | 2032 | 0.362 |
| pl2f.idf.8 | *0.4034* | *0.3741* | *0.3736* | *0.3618* | 0.2133 | 2017 | 0.362 |
| pl2f.idf.9 | *0.3931* | *0.3690* | *0.3729* | *0.3618* | *0.2175* | 2026 | 0.376 |
| pl2f.idf.10 | *0.4034* | *0.3638* | *0.3873* | *0.3658* | *0.2168* | 1990 | 0.383 |

**Table 4.4:** Performance of query expansion system using synonyms with idf as their terms selection criteria on different numbers of words added as the expansion terms, implementing PL2F model.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| baseline | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| dir.idf.1 | 0.4667 | 0.4815 | 0.4647 | 0.4639 | 0.2512 | 1241 | 0.162 |
| lgd.idf.1 | 0.5111 | 0.4741 | 0.5018 | 0.4753 | 0.2606 | 1209 | 0.162 |
| pl2f.idf.1 | 0.4276 | 0.4103 | 0.4005 | 0.3959 | 0.2389 | 2099 | 0.236 |

**Table 4.5:** Summary of best systems on query expansion using synonyms and idf as their terms selection criteria, with different models and number of words added as the expansion terms.

query terms. The results of this experiment is similar to the previous experiment using the language model with Dirichlet prior. We observe the same behavior with P@5 and P@10, which decreases with more addition of expansion terms except for adding seven terms and ten. Adding one expansion term gives the best performance across all other metrics. As previously, our best system for this setting has a lower performance compared to the system using the same model that is applied to the queries without any expansion. However, it has a high score of UNJ@10.

Table 4.4 shows the results of our experiments using the PL2F model, compared with the system using the same model that only uses the original query terms. The results using this model are similar to the results from the LGD weighting model. There is a decrease of the values of the metrics as more expansion terms are added, and the system where one expansion term is added result in the best performance on this model. Again, even though our best system for this setting has a lower performance compared the system using the same model that only uses the original query terms, it has a higher UNJ@10 score.

We summarizes the best systems across different models, compared with our baseline in Table 4.5. Among the systems that are discussed in this section, the system that uses PL2F model performs the best, although the other two systems have a higher UNJ@10 score. As can be seen, the query expansion systems with this setting do not improve on the baseline. However, the UNJ@10 on these three systems are substantially higher than the baseline's.

### 4.2.2 Selecting Expansion Terms Using Preferred Names

In this part, we present the results of our experiments on query expansion with synonymous terms, using preferred names as their term selection criteria.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| dir | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| dir.pt.1 | *0.5276* | **0.4862** | *0.4685* | **0.4591** | **0.2910** | 2108 | 0.184 |
| dir.pt.2 | **0.5345** | *0.4638* | **0.4779** | *0.4476* | 0.2690 | 2091 | 0.257 |
| dir.pt.3 | 0.4759 | 0.4103 | 0.4343 | 0.4075 | 0.2445 | 2055 | 0.366 |
| dir.pt.4 | 0.4345 | 0.3966 | 0.3921 | 0.3816 | 0.2137 | 1961 | 0.410 |
| dir.pt.5 | 0.3897 | 0.3672 | 0.3422 | 0.3462 | 0.1921 | 1919 | 0.462 |
| dir.pt.6 | 0.3862 | 0.3431 | 0.3535 | 0.3451 | 0.1861 | 1886 | 0.483 |
| dir.pt.7 | 0.3828 | 0.3362 | 0.3501 | 0.3381 | 0.1857 | 1866 | 0.483 |
| dir.pt.8 | 0.4034 | 0.3414 | 0.3702 | 0.3433 | 0.1844 | 1858 | 0.476 |
| dir.pt.9 | 0.4000 | 0.3362 | 0.3617 | 0.3342 | 0.1811 | 1844 | 0.479 |
| dir.pt.10 | 0.4000 | 0.3328 | 0.3584 | 0.3285 | 0.1811 | 1812 | 0.495 |

**Table 4.6:** Performance of query expansion system using synonyms with preferred names as their terms selection criteria on different number of words added as the expansion terms, implementing Dirichlet prior model.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| lgd | 0.5862 | 0.5466 | 0.5297 | 0.5208 | 0.3283 | 2271 | 0.009 |
| lgd.pt.1 | **0.5241** | **0.4793** | **0.4605** | **0.4579** | **0.2845** | 2118 | 0.224 |
| lgd.pt.2 | 0.4552 | 0.4224 | *0.4301* | 0.4109 | 0.2429 | 1996 | 0.307 |
| lgd.pt.3 | *0.4759* | 0.4293 | *0.4497* | *0.4274* | 0.2317 | 1930 | 0.328 |
| lgd.pt.4 | 0.4552 | 0.3897 | *0.4256* | 0.3889 | 0.2106 | 1918 | 0.381 |
| lgd.pt.5 | 0.4034 | 0.3621 | 0.3788 | 0.3592 | 0.1998 | 1875 | 0.422 |
| lgd.pt.6 | 0.4069 | 0.3483 | 0.3704 | 0.3421 | 0.1912 | 1873 | 0.417 |
| lgd.pt.7 | 0.3897 | 0.3534 | 0.3725 | 0.3500 | 0.1879 | 1854 | 0.419 |
| lgd.pt.8 | 0.4138 | 0.3638 | 0.3844 | 0.3538 | 0.1888 | 1840 | 0.424 |
| lgd.pt.9 | 0.3931 | 0.3534 | 0.3701 | 0.3427 | 0.1869 | 1839 | 0.438 |
| lgd.pt.10 | 0.3552 | 0.3414 | 0.3404 | 0.3258 | 0.1826 | 1832 | 0.459 |

**Table 4.7:** Performance of query expansion system using synonyms with preferred names as their terms selection criteria on different number of words added as the expansion terms, implementing LGD weighting model.

Table 4.6 shows the results of our experiments using the language model with Dirichlet prior, compared with the system using the same model that only uses the original query terms. Adding only one expansion term results in the best performance on most of our metrics, except for P@5 and NDCG@5 for which adding two expansion terms results in the best performance. As in our experiment using idf as term selection criterion, precisions and NDCGs mostly decrease as more expansion terms are added. An exception will be when we add eight expansion terms per concept to the original queries, where the precision and NDCGs increase slightly compared to the system with seven additional queries. We consider the system with one additional terms to be our best system for this setting, and while its performance is lower than the system with the same model

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| pl2f | 0.5897 | 0.5517 | 0.5366 | 0.5288 | 0.3509 | 2289 | 0.007 |
| pl2f.pt.1 | **0.5276** | **0.4862** | **0.4761** | **0.4676** | **0.3026** | 2158 | 0.186 |
| pl2f.pt.2 | 0.4310 | 0.4103 | 0.4016 | 0.3931 | 0.2513 | 2048 | 0.291 |
| pl2f.pt.3 | 0.4448 | 0.4121 | *0.4234* | 0.4046 | 0.2382 | 2014 | 0.338 |
| pl2f.pt.4 | 0.3966 | 0.3655 | 0.3838 | 0.3650 | 0.2245 | 2006 | 0.395 |
| pl2f.pt.5 | 0.4103 | 0.3690 | 0.3855 | 0.3665 | 0.2202 | 1996 | 0.393 |
| pl2f.pt.6 | 0.4034 | 0.3672 | 0.3642 | 0.3530 | 0.2177 | 2015 | 0.383 |
| pl2f.pt.7 | 0.3931 | 0.3534 | 0.3554 | 0.3432 | 0.2160 | 2022 | 0.397 |
| pl2f.pt.8 | 0.4069 | 0.3586 | 0.3709 | 0.3498 | 0.2167 | 2023 | 0.395 |
| pl2f.pt.9 | 0.4034 | 0.3672 | 0.3693 | 0.3553 | 0.2168 | 2025 | 0.390 |
| pl2f.pt.10 | 0.3862 | 0.3500 | 0.3481 | 0.3366 | 0.2131 | 2010 | 0.405 |

**Table 4.8:** Performance of query expansion system using synonyms with preferred names as their terms selection criteria on different number of words added as expansion terms, implementing PL2F model.

on unexpanded query, it has a higher score of UNJ@10. This system performs slightly better than the best system with the same model which use idf as its term selection criteria, although it has a lower UNJ@10.

Table 4.7 shows the results of our experiments using the LGD weighting model, compared with the system using the same model that only uses the original query terms. For this setting, adding one expansion term gives the best result on all the metrics. Similarly to the experiment using the language model with Dirichlet prior, the values of the metrics decrease as more expansion terms are added, except when it slightly increases on the system with eight additional terms. Our best system again gives lower performance than the system with the same model that are used on unexpanded query. However, the UNJ@10 score is considerably higher compared to the other system.

Table 4.8 shows the results of our experiments using the PL2F model, compared with the system using the same model that only uses the original query terms. As with the previous two retrieval models, in this setting system with only one expansion term shows the best performance on all metrics. Precisions and NDCGs decrease as more terms are added. However, when adding five or eight expansion terms, they slightly increase before decreasing again. As can be seen, our best system for this setting still has lower performance compared to the system using the same retrieval model when only applied to the original query terms. However, the UNJ@10 of our best system is considerably higher. This system performs only slightly better than the best system which uses idf as term selection criteria, and has a slightly lower UNJ@10.

We summarizes the best systems across different model for this experiment in Table 4.9. Unlike the systems that use idf as their selection criteria, adding less expansion terms in this setting gives the best performance. Among the systems that are discussed in this section, the system that uses the PL2F model performs the best, although the system with the language model with Dirichlet prior gets the same P@5 and P@10. Compared with our baseline, these three systems do not give any improvement. However, as in our experiment with using idf as terms selction criteria, the scores of UNJ@10 for these systems are higher compared to our baseline. This system has a lower performance compared to the best system

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| baseline | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| dir.pt.1 | 0.5276 | 0.4862 | 0.4685 | 0.4591 | 0.2910 | 2108 | 0.184 |
| lgd.pt.1 | 0.5241 | 0.4793 | 0.4605 | 0.4579 | 0.2845 | 2118 | 0.224 |
| pl2f.pt.1 | 0.5276 | 0.4862 | 0.4761 | 0.4676 | 0.3026 | 2158 | 0.186 |

**Table 4.9:** Summary of best systems on query expansion using synonyms and preferred names as their terms selection criteria, with different models and number of words added as expansion terms.

from our experiment of using idf as term selection criteria, although it has a substantially higher UNJ@10.

## 4.3 Query Expansion Using Non-Synonymous Related Concepts

In this section, we present the result of our experiments of our query expansion implementation using non-synonymous related concepts. In these experiments, we include different relations to expand our query, and also combine them together. The number of the candidates for this experiment is far smaller compared to those in our experiment with synonymous terms. First of all, this is because we are dealing with concepts instead of words. Secondly, the number of concepts connected with the synonymy relations are far greater than the number of concepts connected with the relations that we choose.

We use the same three models that we use for our experiments on the unexpanded systems: `dir`, `lgd`, and `pl2f`. For each of the system with expanded queries, we compare its performance with that of the system using the same model that only uses the original query terms. For each retrieval model, we choose the system that has performance to be used later in our experiment with field weighting. For readability, we give each systems an id which has the format of [`model`].[`id of included relations`].[`id of filtering relations`]. The second part of the system id is the id number of the relations that are used for the query expansion process. As mentioned before, we use some negative relations to filter our expansion candidates in order to avoid inconsistency by adding concepts that are the opposite of one of the query concepts. The third part of the system id is the id number of the relations that are used for this filtering process. Table 4.10 shows the id of the relations.

Table 4.11 shows the result of different inclusions of relations in a system that uses the language model with Dirichlet prior. Most of the systems perform similarly, with the exception of several systems that have lower performance. `dir`.R2_R9 is one of system that has the best performance. The system where we use all relations and use both negative relations as terms filters is the worse performing system, although having the highest UNJ@10. The best system in this setting is still outperformed by the system using Dirichlet prior on unexpanded queries. However, it has considerably higher UNJ@10.

Table 4.12 shows the result of different inclusions of relations in a system that uses the LGD weighting model. `lgd`.R6_R9 is the best performing system in this

| id | description |
|---|---|
| R1 | include `may_be_finding_of_disease` |
| R2 | include `disease_may_have_finding` |
| R3 | include `associated_finding_of` |
| R4 | include `disease_has_finding` |
| R5 | include `has_associated_finding` |
| R6 | include `is_finding_of_disease` |
| R7 | include all relations |
| R8 | filter with `is_not_finding_of_disease` |
| R9 | filter with `disease_excludes_finding` |
| R10 | filter with the relation in R8 and R9 |

**Table 4.10:** ID of relations inclusion and exclusion.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| `dir` | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| `dir`.R1_R9 | **0.5552** | **0.4914** | **0.5039** | **0.4704** | 0.2649 | 2083 | 0.172 |
| `dir`.R1_R8 | *0.5483* | *0.4810* | *0.4943* | *0.4593* | *0.2603* | 2080 | 0.188 |
| `dir`.R2_R9 | **0.5552** | **0.4914** | **0.5039** | **0.4704** | **0.2650** | 2084 | 0.172 |
| `dir`.R2_R8 | **0.5552** | **0.4914** | **0.5039** | **0.4704** | **0.2650** | 2084 | 0.172 |
| `dir`.R3_R9 | *0.5276* | *0.4690* | *0.4707* | *0.4427* | *0.2574* | 2081 | 0.195 |
| `dir`.R3_R8 | *0.5276* | *0.4690* | *0.4707* | *0.4427* | *0.2574* | 2081 | 0.195 |
| `dir`.R4_R9 | **0.5552** | **0.4914** | **0.5039** | **0.4704** | 0.2649 | 2083 | 0.172 |
| `dir`.R4_R8 | **0.5552** | **0.4914** | **0.5039** | **0.4704** | 0.2649 | 2083 | 0.172 |
| `dir`.R5_R9 | **0.5552** | **0.4914** | **0.5039** | **0.4704** | 0.2648 | 2083 | 0.172 |
| `dir`.R5_R8 | **0.5552** | **0.4914** | **0.5039** | **0.4704** | 0.2648 | 2083 | 0.172 |
| `dir`.R6_R9 | **0.5552** | **0.4914** | **0.5039** | **0.4704** | 0.2649 | 2083 | 0.172 |
| `dir`.R6_R8 | **0.5552** | **0.4914** | **0.5039** | **0.4704** | 0.2649 | 2083 | 0.172 |
| `dir`.R7_R10 | 0.5103 | 0.4638 | 0.4544 | 0.4324 | 0.2499 | 2081 | 0.203 |

**Table 4.11:** Performance on system using Dirichlet prior model with query expansion using non-synonymous related concepts with different relations.

setting. Again, the system that uses all relations and uses both negative relations as terms filters has the lowest performance. The best system in this setting is still outperformed by the system using Dirichlet prior on the unexpanded queries. However, it has considerably a higher score of UNJ@10

Table 4.13 shows the results of different inclusions of relations in a system that uses the PL2F model. `lgd`.R4_R9 is the best performing system in this setting. Unlike the other systems, the system that uses all relations and uses both negative relations as terms filters does not have the lowest performance in this setting. The best system in this setting is still outperformed by the system using Dirichlet prior on unexpanded queries. However, it has a considerably higher UNJ@10.

As we have seen above, the system that use all the relations and use all the negative relations to filter the relations are not the best performing systems over all the retrieval models that we use in our experiment. However, the pool of the expansion concept candidates that are provided by these relations turns out to be quite small, especially compared to the expansion candidates of terms that come from the synonymy relations. This is caused because not all the concepts

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| lgd | 0.5862 | 0.5466 | 0.5297 | 0.5208 | 0.3283 | 2271 | 0.009 |
| lgd.R1_R9 | *0.4931* | **0.4466** | *0.4564* | *0.4330* | *0.2415* | 2087 | 0.214 |
| lgd.R1_R8 | *0.4931* | **0.4466** | *0.4597* | *0.4356* | **0.2436** | 2086 | 0.214 |
| lgd.R2_R9 | 0.4759 | *0.4397* | *0.4462* | *0.4275* | *0.2388* | 2086 | 0229 |
| lgd.R2_R8 | 0.4759 | *0.4379* | *0.4462* | *0.4262* | *0.2381* | 2089 | 0229 |
| lgd.R3_R9 | 0.4690 | *0.4259* | 0.4310 | 0.4109 | *0.2378* | 2085 | 0.233 |
| lgd.R3_R8 | 0.4690 | *0.4259* | 0.4310 | 0.4109 | *0.2378* | 2085 | 0.233 |
| lgd.R4_R9 | *0.4931* | **0.4466** | 0.4597 | 0.4356 | **0.2436** | 2086 | 0.214 |
| lgd.R4_R8 | *0.4931* | **0.4466** | 0.4597 | 0.4356 | **0.2436** | 2086 | 0.214 |
| lgd.R5_R9 | *0.4828* | *0.4397* | *0.4531* | *0.4313* | 0.2358 | 2073 | 0.221 |
| lgd.R5_R8 | 0.4793 | *0.4414* | *0.4479* | 0.4297 | *0.2388* | 2075 | 0.221 |
| lgd.R6_R9 | **0.4966** | **0.4466** | **0.4619** | **0.4359** | *0.2420* | 2084 | 0.214 |
| lgd.R6_R8 | 0.4931 | **0.4466** | *0.4597* | *0.4356* | **0.2436** | 2086 | 0.214 |
| lgd.R7_R10 | 0.4448 | 0.4190 | 0.4167 | 0.4045 | 0.2278 | 2074 | 0.241 |

**Table 4.12:** Performance on system using LGD weighting model with query expansion using non-synonymous related concepts with different relations.

are involved in the relations that we choose above. We investigate the number of queries in our training set that are not expanded at all by this process of query expansion. The result can be seen in Table 4.14.

As it can be seen, including all relations result in the smallest number of unexpanded queries. We want to avoid the risk of having too many queries unexpanded in our small test sets, so we decide to included all of the relations in our queries. We use this system in our later experiment with field weighting. The summary of our selection system for this expansion method can be seen in Table 4.15. The systems are still outperformed by our baseline. However, they have a much higher score of UNJ@10.

## 4.4 Blind Relevance Feedback

In this section, we present the results of our experiments with our blind relevance feedback implementation. We experiment with different number of additional expansion terms that we add to the original query terms. Unlike the results in the previous sections, in experiment with the number of words added per query instead of per concept. We use the same three models that we use for our previous experiments: `dir`, `lgd`, and `pl2f`. For each of the system where we implement blind relevance feedback, we compare its performance with that of the system using the same model that only uses the original query terms. For each retrieval model, we choose the system that has the best performance to be used later in our experiment with field weighting. For readability, we give each system an id which has the format of [`model`].brf.[`# expansion terms`].

Table 4.16 shows the results of our experiments using language model with Dirichlet prior, compared with the system using the same model that only uses the original query terms. The system with only one additional expansion term performs the best among other systems. Similar to our experiment with query expansion, the precisions and NDCGs decrease as more terms are added as ex-

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| pl2f | 0.5897 | 0.5517 | 0.5366 | 0.5288 | 0.3509 | 2289 | 0.007 |
| pl2f.R1_R9 | **0.4448** | *0.4517* | *0.4185* | *0.4236* | *0.2462* | 2118 | 0.193 |
| pl.R1_R8 | **0.4448** | *0.4517* | *0.4185* | *0.4236* | *0.2462* | 2118 | 0.193 |
| pl2f.R2_R9 | *0.4483* | *0.4517* | *0.4176* | *0.4220* | *0.2445* | 2113 | 0.193 |
| pl2f.R2_R8 | *0.4483* | *0.4517* | *0.4176* | *0.4220* | *0.2445* | 2113 | 0.193 |
| pl2f.R3_R9 | 0.4276 | 0.4448 | 0.4005 | 0.4134 | 0.2437 | 2116 | 0.197 |
| pl2f.R3_R8 | 0.4276 | 0.4448 | 0.4005 | 0.4134 | 0.2437 | 2116 | 0.197 |
| pl2f.R4_R9 | **0.4448** | **0.4552** | *0.4185* | **0.4448** | **0.2470** | 2118 | 0.186 |
| pl2f.R4_R8 | **0.4448** | **0.4552** | *0.4185* | 0.4263 | **0.2470** | 2118 | 0.186 |
| pl2f.R5_R9 | *0.4414* | 0.4345 | *0.4134* | 0.4087 | 0.2328 | 2070 | 0.202 |
| pl2f.R5_R8 | *0.4414* | 0.4362 | **0.4200** | *0.4143* | 0.2368 | 2082 | 0.202 |
| pl2f.R6_R9 | **0.4448** | **0.4552** | *0.4144* | *0.4236* | *0.2441* | 2106 | 0.186 |
| pl2f.R6_R8 | **0.4448** | **0.4552** | *0.4185* | 0.4263 | **0.2470** | 2118 | 0.186 |
| pl2f.R7_R10 | *0.4414* | 0.4276 | 0.4107 | 0.4031 | 0.2314 | 2068 | 0.210 |

**Table 4.13:** Performance on system using PL2F model with query expansion using non-synonymous related concepts with different relations.

| id | # unexpanded queries |
|---|---|
| R1_R8 | 55 |
| R1_R9 | 54 |
| R2_R8 | 54 |
| R2_R9 | 54 |
| R3_R8 | 48 |
| R4_R8 | 57 |
| R4_R9 | 57 |
| R3_R9 | 48 |
| R5_R8 | 49 |
| R5_R8 | 48 |
| R6_R8 | 58 |
| R6_R9 | 57 |
| R7_R10 | 46 |

**Table 4.14:** Number of unexpanded queries.

pansion terms. There are some exceptions to this case, such as the slight increase to P@5 and NDCG@5 when adding eight terms, and the increase of NDCG@10 when adding ten terms to the expansion terms. Although the best system in this setting still has lower performance compared to the implementation of the same retrieval model on unexpanded query, the result for this experiment are better compared to the result of the best system with the same model in our implementation of query expansion using synonyms. There is also the matter of higher UNJ@10 compared to the unexpanded system, although lower compared to the systems using our query expansion implementation.

Table 4.17 shows the results of our experiments using the LGD weighting model, compared with the system using the same model that only uses the original query terms. Adding one or two expansion terms result in the same P@5, but the system with only one expansion term performs the best in terms of other metrics. Again, the scores of the metrics decrease as we add more expansion

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| baseline | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| dir.R7_R10 | 0.5103 | 0.4638 | 0.4544 | 0.4324 | 0.2499 | 2081 | 0.203 |
| lgd.R7_R10 | 0.4448 | 0.4190 | 0.4167 | 0.4045 | 0.2278 | 2074 | 0.241 |
| pl2f.R7_R10 | 0.4414 | 0.4276 | 0.4107 | 0.4031 | 0.2314 | 2068 | 0.210 |

**Table 4.15:** Summary of selected systems with query expansion using non-synonymous related concepts, using different models.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| dir | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| dir.brf.1 | **0.5897** | **0.5414** | **0.5370** | **0.5175** | **0.3123** | 2209 | 0.117 |
| dir.brf.2 | *0.5759* | *0.4914* | *0.5165* | *0.4785* | 0.2712 | 2122 | 0.162 |
| dir.brf.3 | *0.5655* | *0.5000* | *0.5153* | *0.4805* | 0.2582 | 2074 | 0.181 |
| dir.brf.4 | *0.5241* | 0.4707 | *0.4876* | 0.4585 | 0.2318 | 1982 | 0.200 |
| dir.brf.5 | *0.5448* | 0.4707 | *0.5038* | *0.4601* | 0.2247 | 1946 | 0.253 |
| dir.brf.6 | *0.5276* | 0.4500 | *0.4880* | 0.4396 | 0.2071 | 1901 | 0.253 |
| dir.brf.7 | *0.5138* | 0.4483 | *0.4694* | 0.4281 | 0.2046 | 1820 | 0.266 |
| dir.brf.8 | *0.5483* | 0.4379 | *0.4929* | 0.4295 | 0.2055 | 1847 | 0.319 |
| dir.brf.9 | *0.5310* | 0.4276 | *0.4883* | 0.4278 | 0.1992 | 1789 | 0.305 |
| dir.brf.10 | *0.5276* | 0.4552 | *0.4823* | 0.4395 | 0.2020 | 1789 | 0.286 |

**Table 4.16:** Performance of system with blind relevance feedback using idf as their terms selection criteria with different number of words added as expansion terms, implementing Dirichlet prior model.

terms, save for a few exceptions. The best system in this setting still has lower performance compare to its unexpanded equivalent. It performs better than the implementation of with query expansion using synonyms, although not as high as in the previous case of the language model with Dirichlet prior. It also has a considerably lower UNJ@10 compared to the system with query expansion using synonyms, although still higher than the unexpanded system.

Table 4.18 shows the results of our experiments using the PL2F model, compared with the system using the same model that only uses the original query terms. Again, adding only one expansion terms give the best performance on all metrics. The score for the metrics decreases as more terms are added to the expansion terms, except for a few exception such as adding nine additional terms to the query. The performance of this system is better than the performance of the best systems with the same model in the system with query expansion using synonyms. Its performance is still lower compared to the systems with the same model that uses only original query terms, but it has considerably higher UNJ@10, although still lower than the system with query expansion using synonyms.

In Table 4.19, we summarizes the best systems across different models. The three systems still do not improve on our baseline. They performs better than the previous experiments using query expansions, although they have considerably lower UNJ@10. The system that implemented the language model with Dirichlet prior performs the best compared to other systems, although it has the lowest number of relevant documents retrieved.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| lgd | 0.5862 | 0.5466 | 0.5297 | 0.5208 | 0.3283 | 2271 | 0.009 |
| lgd.brf.1 | **0.5517** | **0.5190** | *0.5110* | **0.4973** | **0.2970** | 2210 | 0.091 |
| lgd.brf.2 | **0.5517** | *0.5034* | *0.4951* | *0.4777* | 0.2574 | 2097 | 0.140 |
| lgd.brf.3 | *0.5483* | *0.5017* | **0.5156** | *0.4874* | 0.2452 | 2035 | 0.164 |
| lgd.brf.4 | *0.5172* | *0.4638* | *0.4837* | *0.4495* | 0.2205 | 1958 | 0.210 |
| lgd.brf.5 | *0.5172* | 0.4414 | *0.4769* | 0.4305 | 0.2068 | 1921 | 0.253 |
| lgd.brf.6 | *0.5069* | 0.4138 | 0.4616 | 0.4064 | 0.1912 | 1862 | 0.276 |
| lgd.brf.7 | *0.4931* | 0.4328 | 0.4534 | 0.4133 | 0.1924 | 1853 | 0.291 |
| lgd.brf.8 | *0.5034* | 0.4483 | 0.4557 | 0.4228 | 0.1959 | 1802 | 0.295 |
| lgd.brf.9 | *0.5207* | 0.4293 | *0.4716* | 0.4195 | 0.1923 | 1742 | 0.317 |
| lgd.brf.10 | *0.5138* | 0.4483 | *0.4766* | 0.4338 | 0.1973 | 1758 | 0.278 |

**Table 4.17:** Performance of system with blind relevance feedback using idf as their terms selection criteria with different number of words added as expansion terms, implementing LGD weighting model.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| pl2f | 0.5897 | 0.5517 | 0.5366 | 0.5288 | 0.3509 | 2289 | 0.007 |
| pl2f.brf.1 | **0.5552** | **0.5207** | **0.5216** | **0.5054** | **0.2954** | 2224 | 0.116 |
| pl2f.brf.2 | *0.5310* | *0.4759* | *0.4872* | *0.4633* | 0.2678 | 2164 | 0.181 |
| pl2f.brf.3 | *0.5241* | *0.4724* | *0.4820* | *0.4569* | 0.2494 | 2108 | 0.212 |
| pl2f.brf.4 | 0.4862 | 0.4431 | 0.4569 | 0.4303 | 0.2284 | 2057 | 0.233 |
| pl2f.brf.5 | *0.4931* | 0.4345 | 0.4564 | 0.4164 | 0.2228 | 2020 | 0.272 |
| pl2f.brf.6 | 0.4690 | 0.4034 | 0.4293 | 0.3870 | 0.2073 | 1962 | 0.281 |
| pl2f.brf.7 | *0.4862* | 0.4103 | 0.4397 | 0.3921 | 0.2090 | 1967 | 0.279 |
| pl2f.brf.8 | *0.4931* | 0.4276 | 0.4400 | 0.4040 | 0.2080 | 1911 | 0.297 |
| pl2f.brf.9 | 0.4759 | 0.4086 | 0.4316 | 0.3928 | 0.1994 | 1902 | 0.329 |
| pl2f.brf.10 | 0.4655 | 0.4103 | 0.4344 | 0.3962 | 0.2008 | 1876 | 0.302 |

**Table 4.18:** Performance of system with blind relevance feedback using idf as their terms selection criteria with different number of words added as expansion terms, implementing PL2F model.

## 4.5 Field Weighting Experiments

In this section, we present the results of our experiments with field weighting. For all methods and retrieval models in our experiment with the number of expansion terms added to the query(Section 4.2 - 4.4), we take the best system and use it for our experiments with field weighting. To review, we use three models in our experiment: language model with Dirichlet prior (`dir`), the LGD weighting model (`lgd`), and the PL2F model (`pl2f`). We experiment with our different implementation of query reformulation: query expansion with synonymous terms using idf (`idf`) and preferred terms (`pt`) as term selection criteria, query expansion using non-synonymous related concepts (`rel`), and blind relation feedback (`brf`). For each of the parameters, we iterate from 0.1 to 4.6 with the step of 0.5. We choose this range based on our previous experiment during our participation in CLEF eHealth 2015, where we found that giving the fields values less than five gives the best performance. From our previous experiment, we also know that giving the original query terms a greater weight than the concepts, which in turn

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| `baseline` | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| `dir`.brf.1 | 0.5897 | 0.5414 | 0.5370 | 0.5175 | 0.3123 | 2209 | 0.117 |
| `lgd`.brf.1 | 0.5517 | 0.5190 | 0.5110 | 0.4973 | 0.2970 | 2210 | 0.091 |
| `pl2f`.brf.1 | 0.5552 | 0.5207 | 0.5216 | 0.5054 | 0.2954 | 2224 | 0.116 |

**Table 4.19:** Summary of best systems with blind relevance feedback using idf as their terms selection criteria, with different models and number of terms added as expansion terms.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| `dir` | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| `dir`.idf.0.6_0.1_0.1 | **0.5276** | **0.4897** | **0.4642** | **0.4568** | **0.3009** | 2216 | 0.15 |
| `dir`.idf.* | **0.5276** | **0.4897** | **0.4642** | **0.4568** | **0.3009** | 2216 | 0.15 |

**Table 4.20:** Performance of field weighting on system with query expansion using synonyms (using idf as their terms selection criteria), implementing Dirichlet prior model.

are given a greater or equal weight than any expansion terms or concepts, result in the best performance. Therefore, we use this restriction when iterating our parameters.

For readability, we give each experiment in field weighting an id with the format of `[model]`.[`method`].[weight of `title`].[`weight of ctitle`].[`weight of e_title`]. An exception is for the case of `rel`, where we replace `e_title` with `e_ctitle`, which contains the expanded query concepts. For each of the method and model, we have 219 experiments to run. The result is visually difficult to present, so we only shows the best 15 results in this section.

### 4.5.1 Expansion using Synonymous Terms (idf)

In this part, we present the result of our field weighting experiments on the systems on query expansion using synonymous terms that use idf as their term selection criteria. Table 4.20 shows the results of our experiments with field weighting, using the language model with Dirichlet prior, compared with the system with the same model without any expansion or term weighting. In our experiments with term weighting using this model, the results are the same across all weights. We present only the system using the lowest weight configuration. The results of the experiments are still lower than the result that we get from the system that we have from the unexpanded system using the same model. However, it has a higher UNJ@10. In this experiment, we observe that the systems give lower performances compared to the system where we use the same number of expansion terms without any weighting. They have substantially llower scores of UNJ@10, which might be the cause of this difference in performance.

Table 4.21 shows the best 15 results of our experiments with field weighting, using language model with the LGD weighting model, compared with the system with the same model without any expansion or term weighting. Unlike the previous system, the performance of this system varies across weight configurations, even though the best 15 systems have the same results. We observe that the

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| lgd | 0.5862 | 0.5466 | 0.5297 | 0.5208 | 0.3283 | 2271 | 0.009 |
| lgd.idf.2.6_0.1_0.1 | **0.5828** | **0.5517** | **0.5241** | **0.5218** | **0.3309** | 2280 | 0.009 |
| lgd.idf.2.6_0.6_0.1 | **0.5828** | **0.5517** | **0.5241** | **0.5218** | **0.3309** | 2280 | 0.009 |
| lgd.idf.2.6_1.1_0.1 | **0.5828** | **0.5517** | **0.5241** | **0.5218** | **0.3309** | 2280 | 0.009 |
| lgd.idf.2.6_1.6_0.1 | **0.5828** | **0.5517** | **0.5241** | **0.5218** | **0.3309** | 2280 | 0.009 |
| lgd.idf.2.6_2.1_0.1 | **0.5828** | **0.5517** | **0.5241** | **0.5218** | **0.3309** | 2280 | 0.009 |
| lgd.idf.2.6_2.6_0.1 | **0.5828** | **0.5517** | **0.5241** | **0.5218** | **0.3309** | 2280 | 0.009 |
| lgd.idf.3.1_0.1_0.1 | **0.5828** | **0.5517** | **0.5241** | **0.5218** | **0.3309** | 2280 | 0.009 |
| lgd.idf.3.1_0.6_0.1 | **0.5828** | **0.5517** | **0.5241** | **0.5218** | **0.3309** | 2280 | 0.009 |
| lgd.idf.3.1_1.1_0.1 | **0.5828** | **0.5517** | **0.5241** | **0.5218** | **0.3309** | 2280 | 0.009 |
| lgd.idf.3.1_1.6_0.1 | **0.5828** | **0.5517** | **0.5241** | **0.5218** | **0.3309** | 2280 | 0.009 |
| lgd.idf.3.1_2.1_0.1 | **0.5828** | **0.5517** | **0.5241** | **0.5218** | **0.3309** | 2280 | 0.009 |
| lgd.idf.3.1_2.6_0.1 | **0.5828** | **0.5517** | **0.5241** | **0.5218** | **0.3309** | 2280 | 0.009 |
| lgd.idf.3.1_3.1_0.1 | **0.5828** | **0.5517** | **0.5241** | **0.5218** | **0.3309** | 2280 | 0.009 |
| lgd.idf.3.6_0.1_0.1 | **0.5828** | **0.5517** | **0.5241** | **0.5218** | **0.3309** | 2280 | 0.009 |
| lgd.idf.3.6_0.6_0.1 | **0.5828** | **0.5517** | **0.5241** | **0.5218** | **0.3309** | 2280 | 0.009 |

**Table 4.21:** Performance of field weighting on system with query expansion using synonyms (using idf as their terms selection criteria), implementing LGD weighting model.

values of the metrics tend to be the same within a cluster weight configuration. The best system in this setting performs substantially better than the system on unexpanded queries using the same retrieval model. It also has a slightly higher UNJ@10. This system does not give any improvement to similar system without any weighting, although it has a lower UNJ@10.

Table 4.22 shows the best 15 results of our experiments with field weighting, using language model with the PL2F model, compared with the system with the same model without any expansion or term weighting. The performance varies across weight configurations, but the best 15 systems all have the same performance. The systems greatly improves the performance compared to the system using the same retrieval model on unexpanded queries. It also has the same scores of UNJ@10, that suggest that this system is compared fairly to the other system. All the systems in the best 15 tend to cluster on two different weight configuration, similar to what we observe on the previous experiments using LGD language model. Our result in this setting is slightly better than the result of the best system using LGD language model. It also improves on the result of the system with similar setting using no term weighting.

Table 4.23 summarizes the best systems across different models, compared with our baseline. The only system that does not perform better than our baseline is the system using the language model with Dirichlet prior. The system using the PL2F model performs the best in this experiment, with the system using the LGD weighting model as a close second. We use these system later for our testing purpose in Chapter 5.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| pl2f | 0.5897 | 0.5517 | 0.5366 | 0.5288 | 0.3509 | 2289 | 0.007 |
| pl.idf.2.6_0.6_0.6 | **0.6000** | **0.5621** | **0.5372** | **0.5327** | **0.3539** | 2302 | 0.017 |
| pl.idf.2.6_1.1_0.6 | **0.6000** | **0.5621** | **0.5372** | **0.5327** | **0.3539** | 2302 | 0.017 |
| pl.idf.2.6_1.6_0.6 | **0.6000** | **0.5621** | **0.5372** | **0.5327** | **0.3539** | 2302 | 0.017 |
| pl.idf.2.6_2.1_0.6 | **0.6000** | **0.5621** | **0.5372** | **0.5327** | **0.3539** | 2302 | 0.017 |
| pl.idf.2.6_2.6_0.6 | **0.6000** | **0.5621** | **0.5372** | **0.5327** | **0.3539** | 2302 | 0.017 |
| pl.idf.4.6_1.1_1.1 | **0.6000** | **0.5621** | **0.5372** | **0.5327** | **0.3539** | 2302 | 0.017 |
| pl.idf.4.6_1.6_1.1 | **0.6000** | **0.5621** | **0.5372** | **0.5327** | **0.3539** | 2302 | 0.017 |
| pl.idf.4.6_2.1_1.1 | **0.6000** | **0.5621** | **0.5372** | **0.5327** | **0.3539** | 2302 | 0.017 |
| pl.idf.4.6_2.6_1.1 | **0.6000** | **0.5621** | **0.5372** | **0.5327** | **0.3539** | 2302 | 0.017 |
| pl.idf.4.6_3.1_1.1 | **0.6000** | **0.5621** | **0.5372** | **0.5327** | **0.3539** | 2302 | 0.017 |
| pl.idf.4.6_3.6_1.1 | **0.6000** | **0.5621** | **0.5372** | **0.5327** | **0.3539** | 2302 | 0.017 |
| pl.idf.4.6_4.1_1.1 | **0.6000** | **0.5621** | **0.5372** | **0.5327** | **0.3539** | 2302 | 0.017 |
| pl.idf.4.6_4.6_1.1 | **0.6000** | **0.5621** | **0.5372** | **0.5327** | **0.3539** | 2302 | 0.017 |
| pl.idf.0.6_0.1_0.1 | **0.6000** | **0.5621** | **0.5372** | **0.5327** | **0.3539** | 2302 | 0.017 |
| pl.idf.0.6_0.6_0.1 | **0.6000** | **0.5621** | **0.5372** | **0.5327** | **0.3539** | 2302 | 0.017 |

**Table 4.22:** Performance of field weighting on system with query expansion using synonyms (using idf as their terms selection criteria), implementing PL2F model.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| baseline | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| dir.idf.0.6_0.1_0.1 | 0.5276 | 0.4897 | 0.4642 | 0.4568 | 0.3009 | 2216 | 0.15 |
| lgd.idf.2.6_0.1_0.1 | 0.5828 | 0.5517 | 0.5241 | 0.5218 | 0.3309 | 2280 | 0.009 |
| pl2f.idf.0.6_0.1_0.1 | 0.6000 | 0.5621 | 0.5372 | 0.5327 | 0.3539 | 2302 | 0.017 |

**Table 4.23:** Summary of best systems with field weighting on system with query expansion using synonyms using idf as their terms selection criteria, with different models and weight.

### 4.5.2 Expansion using Synonymous Terms (Preferred Names)

In this part, we present the results of our experiments with the systems with query expansion using synonymous terms with preferred names as their term selection criteria.

Table 4.24 shows the results of our experiments with field weighting, using language model with the language model with Dirichlet prior, compared with the system with the same model without any expansion or term weighting. Similar to our previous experiment using the same model, the performance of the systems are similar across all weighting configurations. We present only the system using the lowest weight configuration. The result from this experiments are lower than the systems using the same retrieval model on original query terms, although with higher number of UNJ@10. The systems give the same performance compared to the similar system that does not use any term weighting.

Table 4.25 shows the best 15 results of our experiments with field weighting, using language model with the LGD weighting model, compared with the system with the same model without any expansion or term weighting. The system with (0.6, 0.1, 0.1) and a few other systems give the highest performance for most

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| dir | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| dir.pt.0.6_0.1_0.1 | **0.5276** | **0.4862** | **0.4685** | **0.4591** | **0.2910** | 2108 | 0.185 |
| dir.pt.* | **0.5276** | **0.4862** | **0.4685** | **0.4591** | **0.2910** | 2108 | 0.185 |

**Table 4.24:** Performance of field weighting on system with query expansion using synonyms (using preferred names as their terms selection criteria), implementing Dirichlet prior model.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| lgd | 0.5862 | 0.5466 | 0.5297 | 0.5208 | 0.3283 | 2271 | 0.009 |
| lgd.pt.0.6_0.1_0.1 | 0.6034 | **0.5638** | **0.5419** | **0.5371** | **0.3376** | 2284 | 0.022 |
| lgd.pt.0.6_0.6_0.1 | *0.6034* | **0.5638** | **0.5419** | **0.5371** | **0.3376** | 2284 | 0.022 |
| lgd.pt.3.6_0.6_0.6 | *0.6034* | **0.5638** | **0.5419** | **0.5371** | **0.3376** | 2284 | 0.022 |
| lgd.pt.3.6_1.1_0.6 | *0.6034* | **0.5638** | **0.5419** | **0.5371** | **0.3376** | 2284 | 0.022 |
| lgd.pt.3.6_1.6_0.6 | *0.6034* | **0.5638** | **0.5419** | **0.5371** | **0.3376** | 2284 | 0.022 |
| lgd.pt.3.6_2.1_0.6 | *0.6034* | **0.5638** | **0.5419** | **0.5371** | **0.3376** | 2284 | 0.022 |
| lgd.pt.3.6_2.6_0.6 | *0.6034* | **0.5638** | **0.5419** | **0.5371** | **0.3376** | 2284 | 0.022 |
| lgd.pt.3.6_3.1_0.6 | *0.6034* | **0.5638** | **0.5419** | **0.5371** | **0.3376** | 2284 | 0.022 |
| lgd.pt.3.6_3.6_0.6 | *0.6034*: | **0.5638** | **0.5419** | **0.5371** | **0.3376** | 2284 | 0.022 |
| lgd.pt.2.1_0.6_0.6 | **0.6069** | *0.562*1 | *0.5411* | *0.5340* | 0.3341 | 2282 | 0.038 |
| lgd.pt.2.1_1.1_0.6 | **0.6069** | *0.5621* | *0.5411* | *0.5340* | *0.3341:* | 2282 | 0.038 |
| lgd.pt.2.1_1.6_0.6 | **0.6069** | *0.5621* | *0.5411* | *0.5340* | *0.3341* | 2282 | 0.038 |
| lgd.pt.2.1_2.1_0.6 | **0.6069** | *0.5621* | *0.5411* | *0.5340* | *0.3341* | 2282 | 0.038 |
| lgd.pt.2.6_0.6_0.6 | **0.6069** | *0.5621* | *0.5409* | *0.5345* | *0.3353* | 2283 | 0.036 |
| lgd.pt.2.6_1.1_0.6 | **0.6069** | *0.5621* | *0.5409* | *0.5345* | *0.3353* | 2283 | 0.036 |

**Table 4.25:** Performance of field weighting on system with query expansion using synonyms (using preferred names as their terms selection criteria), implementing LGD weighting model.

metrics except P@5. The system with (2.1, 0.6, 0.6) gives the best result for this metric. As has been observed before on different systems, the value of the metrics tend to be the same within certain clusters of weight configurations. The best system in this setting performs better than the system that uses the same retrieval model but only uses the original query terms. It also have a higher UNJ@10. The results are also better than the results from the system that uses the same retrieval model and number of expansion terms, but without any weighting.

Table 4.26 shows the best 15 results of our experiments with field weighting, using language model with the PL2F model, compared with the system with the same model without any expansion or term weighting. The system with weight configuration (4.6, 1.1, 1.1) has the best performance on P@10 and NDCG@10, while the system with weight configuration (1.1, 0.1, 0.1) has the best performance on P@5, NDCG@5, and MAP. We choose the first system as our best system for this result because it has a higher UNJ@10. The result of this system is better compared to the system that uses the same model but only uses the original query terms, and also compared to the system that uses query expansion with the same number of expansion terms but without any weighting. Similar results also tend

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| pl2f | 0.5897 | 0.5517 | 0.5366 | 0.5288 | 0.3509 | 2289 | 0.007 |
| pl2f.pt.4.6_1.1_1.1 | *0.5862* | **0.5690** | *0.5315* | **0.5377** | *0.3496* | 2293 | 0.022 |
| pl2f.pt.4.6_1.6_1.1 | *0.5862* | **0.5690** | *0.5315* | **0.5377** | *0.3496* | 2293 | 0.022 |
| pl2f.pt.4.6_2.1_1.1 | *0.5862* | **0.5690** | *0.5315* | **0.5377** | *0.3496* | 2293 | 0.022 |
| pl2f.pt.4.6_2.6_1.1 | *0.5862* | **0.5690** | *0.5315* | **0.5377** | *0.3496* | 2293 | 0.022 |
| pl2f.pt.4.6_3.1_1.1 | *0.5862* | **0.5690** | *0.5315* | **0.5377** | *0.3496* | 2293 | 0.022 |
| pl2f.pt.4.6_3.6_1.1 | *0.5862* | **0.5690** | *0.5315* | **0.5377** | *0.3496* | 2293 | 0.022 |
| pl2f.pt.4.6_4.1_1.1 | *0.5862* | **0.5690** | 0.5315 | **0.5377** | *0.3496* | 2293 | 0.022 |
| pl2f.pt.4.6_4.6_1.1 | *0.5862* | **0.5690** | 0.5315 | **0.5377** | *0.3496* | 2293 | 0.022 |
| pl2f.pt.1.1_0.1_0.1 | **0.6034** | *0.5672* | **0.5405** | 0.5367 | **0.3546** | 2293 | 0.012 |
| pl2f.pt.1.1_0.6_0.1 | **0.6034** | *0.5672* | **0.5405** | 0.5367 | **0.3546** | 2293 | 0.012 |
| pl2f.pt.1.1_1.1_0.1 | **0.6034** | *0.5672* | **0.5405** | *0.5367* | **0.3546** | 2293 | 0.012 |
| pl2f.pt.2.6_0.6_0.6 | *0.5897* | *0.5672* | *0.5331* | *0.5362* | *0.3497* | 2293 | 0.022 |
| pl2f.pt.2.6_1.1_0.6 | *0.5897* | *0.5672* | *0.5331* | *0.5362* | *0.3497* | 2293 | 0.022 |
| pl2f.pt.2.6_1.6_0.6 | *0.5897* | *0.5672* | *0.5331* | *0.5362* | *0.3497* | 2293 | 0.022 |
| pl2f.pt.2.6_2.1_0.6 | *0.5897* | *0.5672* | *0.5331* | *0.5362* | *0.3497* | 2293 | 0.022 |

**Table 4.26:** Performance of field weighting on system with query expansion using synonyms (using preferred names as their terms selection criteria), implementing PL2F model.

to cluster on similar weight configurations in this setting.

In Table 4.27, we summarize the best systems across different models, compared with our baseline. The system with the LGD weighting model perform the best compared to the two other systems. However, the performance of this system is still lower than the performance of our baseline. It has a higher UNJ@10, although not by much. We use these system later for our testing purpose in Chapter 5.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| baseline | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| dir.pt.0.6_0.1_0.1 | 0.5276 | 0.4862 | 0.4685 | 0.4591 | 0.2910 | 2108 | 0.185 |
| lgd.pt.0.6_0.1_0.1 | 0.6034 | 0.5638 | 0.5419 | 0.5371 | 0.3376 | 2284 | 0.022 |
| pl2f.pt.4.6_1.1_1.1 | 0.5862 | 0.5690 | 0.5315 | 0.5377 | 0.3496 | 2293 | 0.022 |

**Table 4.27:** Summary of best systems with field weighting on system with query expansion using synonyms using preferred names as their terms selection criteria, with different models and weight.

### 4.5.3 Expansion using Non-Synonymous Related Concepts

In this part, we present the result of our field weighting experiment on the systems with query expansion using non-synonymous related concepts. For our experiments on term weighting using this model, the results are the same across all weights for all of the retrieval models that we use. This is perhaps because the number of queries that are successfully expanded using this method are really low, as we shows in table 4.14. We present only the system using the lowest weight configuration.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| dir | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| dir.rel.0.6_0.1_0.1 | **0.6207** | **0.5828** | **0.5570** | **0.5494** | **0.3434** | 2266 | 0.012 |
| dir.rel.* | **0.6207** | **0.5828** | **0.5570** | **0.5494** | **0.3434** | 2266 | 0.012 |

**Table 4.28:** Performance of field weighting on system with query expansion using non-synonymous concepts, using Dirichlet prior model

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| lgd | 0.5862 | 0.5466 | 0.5297 | 0.5208 | 0.3283 | 2271 | 0.009 |
| lgd.rel.0.6_0.1_0.1 | **0.5897** | **0.5483** | **0.5315** | **0.5223** | **0.3310** | 2277 | 0.009 |
| lgd.rel.* | **0.5897** | **0.5483** | **0.5315** | **0.5223** | **0.3310** | 2277 | 0.009 |

**Table 4.29:** Performance of system using blind relevance feedback with field weighting, using LGD weighting model.

Table 4.28 shows the results of our experiments with field weighting, using language model with the language model with Dirichlet prior, compared with the system with the same model without any expansion or term weighting. The result of the experiments are higher than the result that we get from a system that we have from the unexpanded system using the same model, except for P@5 and number of relevant documents retrieved. In this experiment, we observe that the systems give slightly higher performances compared to the system where we do not use any weighting on the expanded concepts. They have a lower UNJ@10, which might be the cause of this difference in performance.

Table 4.29 shows the results of our experiments with field weighting, using the LGD weighting model, compared with the system with the same model without any expansion or term weighting. This system gives a slightly higher performance compared to the system that use the same retrieval model but only use the original query terms. Like in the case for system with the language model with Dirichlet prior, this system give slightly higher performance compared to the system where we do not use any weighting on the expanded concepts. Again, it has a lower UNJ@10.

Table 4.30 shows the results of our experiments wuth field weighting, using the PL2F model, compared with the system with the same model without any expansion or term weighting. This system also gives a higher performance compared to the system that use the same retrieval model but only use the original query terms. As in the case for system with the two other retrieval model, this system gives slightly higher performances compared to the system where we do not use any weighting on the expanded concepts. Although, it has a lower UNJ@10.

Table 4.31 summarizes the system that we choose from our experiment with term weighting on system with query expansion using non-synonymous related concepts. The language model with Dirichlet prior gives the best performance among the system. It also performs slightly better compared to our baseline, although they have the same P@5. We use these systems for our testing purpose in Chapter 5.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| pl2f | 0.5897 | 0.5517 | 0.5366 | 0.5288 | 0.3509 | 2289 | 0.007 |
| pl2f.rel.0.6_0.1_0.1 | **0.6000** | **0.5586** | **0.5404** | **0.5325** | **0.3529** | 2296 | 0.009 |
| pl2f.rel.* | **0.6000** | **0.5586** | **0.5404** | **0.5325** | **0.3529** | 2296 | 0.009 |

**Table 4.30:** Performance of system using blind relevance feedback with field weighting, using PL2F model.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| baseline | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| dir.rel.0.6_0.1_0.1 | 0.6207 | 0.5828 | 0.5570 | 0.5494 | 0.3434 | 2266 | 0.012 |
| lgd.rel.0.6_0.1_0.1 | 0.5897 | 0.5483 | 0.5315 | 0.5223 | 0.3310 | 2277 | 0.009 |
| pl2f.rel.0.6_0.1_0.1 | 0.6000 | 0.5586 | 0.5404 | 0.5325 | 0.3529 | 2296 | 0.009 |

**Table 4.31:** Summary of best systems with field weighting on systems with query expansion using non-synonymous terms, with differen models and weight.

### 4.5.4 Blind Relevance Feedback

In this part, we present the results of our experiments of field weighting for systems with blind relevance feedback.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| dir | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| dir.brf.0.6_0.1_0.1 | **0.5897** | **0.5414** | **0.5370** | **0.5175** | **0.3123** | 2209 | 0.117 |
| dir.brf.* | **0.5897** | **0.5414** | **0.5370** | **0.5175** | **0.3123** | 2209 | 0.117 |

**Table 4.32:** Performance of system using blind relevance feedback with field weighting, using Dirichlet prior model.

Table 4.32 shows the results of our experiments with field weighting, using language model with Dirichlet prior, compared with the system with the same model without any expansion or term weighting. As in our previous experiments with term weighting, the results using this model are the same across all weight. We present only the system using the lowest weight configuration. The result from this experiments is lower than the systems using the same retrieval model on original query terms, although with higher nUNJ@10. The systems give the same performance compared to the similar system that does not use any term weighting.

Table 4.33 shows the best results of our experiments on field weighting, using language model with the LGD weighting model, compared with the system with the same model without any expansion or term weighting. The system with (4.6, 1.1, 1.1) and a few other systems give the highest performance for most metrics except for the number of relevant documents returned. The system with (4.1, 1.6, 1.6) gives the best result for this metric. As has been observed before on different systems, the value of the metrics tend to be the same within certain clusters of weight configurations. The best system in this setting performs better than the system that uses the same retrieval model but only uses the original query terms. It also have a higher UNJ@10. The results are also better than the results from the system that uses the same retrieval model and number of expansion terms,

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| lgd | 0.5862 | 0.5466 | 0.5297 | 0.5208 | 0.3283 | 2271 | 0.009 |
| lgd.brf.4.6_1.1_1.1 | **0.6034** | **0.5638** | **0.5513** | **0.5362** | **0.3375** | 2269 | 0.010 |
| lgd.brf.4.6_1.6_1.1 | **0.6034** | **0.5638** | **0.5513** | **0.5362** | **0.3375** | 2269 | 0.010 |
| lgd.brf.4.6_2.1_1.1 | **0.6034** | **0.5638** | **0.5513** | **0.5362** | **0.3375** | 2269 | 0.010 |
| lgd.brf.4.6_2.6_1.1 | **0.6034** | **0.5638** | **0.5513** | **0.5362** | **0.3375** | 2269 | 0.010 |
| lgd.brf.4.6_3.1_1.1 | **0.6034** | **0.5638** | **0.5513** | **0.5362** | **0.3375** | 2269 | 0.010 |
| lgd.brf.4.6_3.6_1.1 | **0.6034** | **0.5638** | **0.5513** | **0.5362** | **0.3375** | 2269 | 0.010 |
| lgd.brf.4.6_4.1_1.1 | **0.6034** | **0.5638** | **0.5513** | **0.5362** | **0.3375** | 2269 | 0.010 |
| lgd.brf.4.6_4.6_1.1 | **0.6034** | **0.5638** | **0.5513** | **0.5362** | **0.3375** | 2269 | 0.010 |
| lgd.brf.4.1_1.6_1.6 | *0.6000* | **0.5638** | *0.5410* | *0.5288* | *0.3297* | 2273 | 0.033 |
| lgd.brf.4.1_2.1_1.6 | *0.6000* | **0.5638** | *0.5410* | *0.5288* | *0.3297* | 2273 | 0.033 |
| lgd.brf.4.1_2.6_1.6 | *0.6000* | **0.5638** | *0.5410* | *0.5288* | *0.3297* | 2273 | 0.033 |
| lgd.brf.4.1_3.1_1.6 | *0.6000* | **0.5638** | *0.5410* | *0.5288* | *0.3297* | 2273 | 0.033 |
| lgd.brf.4.1_3.6_1.6 | *0.6000* | **0.5638** | *0.5410* | *0.5288* | *0.3297* | 2273 | 0.033 |
| lgd.brf.4.1_4.1_1.6 | *0.6000* | **0.5638** | *0.5410* | *0.5288* | *0.3297* | 2273 | 0.033 |
| lgd.brf.1.6_0.6_0.6 | *0.5966* | *0.5621* | *0.5402* | *0.5285* | *0.3307* | 2272 | 0.029 |

**Table 4.33:** Performance of system using blind relevance feedback with field weighting, using LGD weighting model.

but without any weighting.

Table 4.34 shows the best results of our experiments with field weighting, using language model with the PL2F model, compared with the system with the same model without any expansion or term weighting. The system with weight configuration (1.1, 0.1, 0.1) has the best performance on every metrics but P@10. System with weight configuration (2.6, 1.1, 0.6) has a slightly better P@10, and a higher UNJ@10. Similar results also tend to cluster around similar weight configuration. The best system in this setting performs better than the system that uses the same retrieval model but only uses the original query terms. It also have a higher UNJ@10. The results are also better than the results from the system that uses the same retrieval model and number of expansion terms, but without any weighting.

Table 4.35 summarizes the best systems across different models, compared with our baseline. The system with the PL2F model perform the best compared to the two other systems. However, the performance of this system is still lower than the performance of our baseline. It has a higher UNJ@10, although not by much. We use these system later for our testing purpose in Chapter 5.

## 4.6 Utilizing Semantic Network

In this section, we present the results of our experiment where we weight terms with the semantic type "body part" differently than the rest of the terms. Again, we use three models in our experiment: language model with Dirichlet prior (`dir`), the LGD weighting model (`lgd`), and the PL2F model (`pl2f`). We experiment with weight of the terms, between 1.5 to 10 with a step of 0.5. For readability, we give each of the system an id with the format of [`model`].bp.[`weight`].

Table 4.36 shows the results of our experiments using language model with

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| pl2f | 0.5897 | 0.5517 | 0.5366 | 0.5288 | 0.3509 | 2289 | 0.007 |
| pl2f.brf.1.1_0.1_0.1 | **0.6069** | *0.5655* | **0.5501** | **0.5414** | **0.3553** | 2304 | 0.014 |
| pl2f.brf.1.1_0.6_0.1 | **0.6069** | *0.5655* | **0.5501** | **0.5414** | **0.3553** | 2304 | 0.014 |
| pl2f.brf.1.1_1.6_0.1 | **0.6069** | *0.5655* | **0.5501** | **0.5414** | **0.3553** | 2304 | 0.014 |
| pl2f.brf.1.1_1.6_0.1 | *0.6034* | *0.5569* | *0.5453* | *0.5309* | *0.3410* | 2285 | 0.057 |
| pl2f.brf.2.1_1.1_1.1 | *0.6000* | *0.5517* | *0.5457* | *0.5281* | *0.3306* | 2273 | 0.071 |
| pl2f.brf.2.1_1.6_1.1 | *0.6000* | *0.5517* | *0.5457* | *0.5281* | *0.3306* | 2273 | 0.071 |
| pl2f.brf.2.6_1.1_1.1 | *0.6000* | *0.5517* | *0.5457* | *0.5281* | *0.3306* | 2273 | 0.071 |
| pl2f.brf.2.6_1.1_1.1 | *0.6000* | *0.5517* | *0.5457* | *0.5281* | *0.3306* | 2273 | 0.071 |
| pl2f.brf.2.6_0.6_0.6 | *0.5931* | **0.5672** | *0.5373* | *0.5373* | *0.3488* | 2298 | 0.034 |
| pl2f.brf.2.6_1.1_0.6 | *0.5931* | **0.5672** | *0.5373* | *0.5373* | *0.3488* | 2298 | 0.034 |
| pl2f.brf.2.6_1.6_0.6 | *0.5931* | **0.5672** | *0.5373* | *0.5373* | *0.3488* | 2298 | 0.034 |
| pl2f.brf.2.6_2.1_0.6 | *0.5931* | **0.5672** | *0.5373* | *0.5373* | *0.3488* | 2298 | 0.034 |
| pl2f.brf.2.6_2.6_0.6 | *0.5931* | **0.5672** | *0.5373* | *0.5373* | *0.3488* | 2298 | 0.034 |
| pl2f.brf.4.6_1.1_1.1 | *0.5931* | **0.5672** | *0.5373* | *0.5373* | *0.3488* | 2298 | 0.034 |
| pl2f.brf.4.6_1.6_1.1 | *0.5931* | **0.5672** | *0.5373* | *0.5373* | *0.3488* | 2298 | 0.034 |

**Table 4.34:** Performance of system using blind relevance feedback with field weighting, using PL2F model.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| baseline | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| dir.brf.0.6_0.1_0.1 | 0.5897 | 0.5414 | 0.5370 | 0.5175 | 0.3123 | 2209 | 0.117 |
| lgd.brf.4.6_1.1_1.1 | 0.6034 | 0.5638 | 0.5513 | 0.5362 | 0.3375 | 2269 | 0.010 |
| pl2f.brf.1.1_0.1_0.1 | 0.6069 | 0.5655 | 0.5501 | 0.5414 | 0.3553 | 2304 | 0.014 |

**Table 4.35:** Summary of best systems using blind relevance feedback with field weighting, with different models and weight.

Dirichlet prior, compared with the system using the same model without weighting. As in our previous experiments with term weighting, the results using this model is the same across all weight. We display only the first result of the experiments. The systems perform better compared to the system where body part terms are not weighted. They also have the same scores of UNJ@10, which is quite small. This can be an indicator that these systems' performances are evaluated fairly.

Table 4.37 shows the results of our experiments using the LGD weighting model, compared with the system using the same model without weighting. The best performing system is the system where each body part term is given the 1.5 weight. The only metric where this system does not achieve the best performance is NDCG@10, for which weighting the terms with 2.0 weight gives the highest performance. We observe that the performance slightly decreases as we given more weight to the body part terms. However, the UNJ@10 increases at the same time, which might be one of the reasons of the decrease in performance. The best system in this setting outperform the system with the same model where body part terms are not given any weighting.

Table 4.38 shows the results of our experiments using the PL2F model, compared with the system using the same model without weighting. As in the system

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| dir | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| dir.bp.1.5 | **0.6276** | **0.5879** | **0.5608** | **0.5527** | **0.3411** | 2249 | 0.010 |
| dir.bp.* | **0.6276** | **0.5879** | **0.5608** | **0.5527** | **0.3411** | 2249 | 0.010 |

**Table 4.36:** Performance of system where body part terms are weighted differently, implementing Dirichlet prior model.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| lgd | 0.5862 | 0.5466 | 0.5297 | 0.5208 | 0.3283 | 2271 | 0.009 |
| lgd.bp.1.5 | **0.5931** | **0.5483** | **0.5312** | *0.5230* | **0.3310** | 2256 | 0.026 |
| lgd.bp.2.0 | *0.5828* | **0.5483** | **0.5312** | **0.5253** | 0.3294 | 2247 | 0.033 |
| lgd.bp.2.5 | 0.5862 | *0.5328* | *0.5358* | 0.5168 | 0.3242 | 2229 | 0.079 |
| lgd.bp.3.0 | *0.5655* | *0.5259* | *0.5166* | 0.5038 | 0.3129 | 2207 | 0.103 |
| lgd.bp.3.5 | 0.5655 | 0.5138 | *0.5086* | 0.4906 | 0.3018 | 2174 | 0.121 |
| lgd.bp.4.0 | 0.5483 | 0.5000 | *0.4943* | 0.4774 | 0.2903 | 2159 | 0.152 |
| lgd.bp.4.5 | *0.5448* | *0.4966* | 0.4838 | 0.4698 | 0.2768 | 2131 | 0.166 |
| lgd.bp.5.0 | 0.5379 | 0.4707 | 0.4780 | 0.4510 | 0.2674 | 2098 | 0.195 |
| lgd.bp.5.5 | 0.5345 | 0.4655 | 0.4729 | 0.4445 | 0.2603 | 2080 | 0.203 |
| lgd.bp.6.0 | 0.5241 | 0.4586 | 0.4617 | 0.4376 | 0.2548 | 2049 | 0.217 |
| lgd.bp.6.5 | 0.5207 | 0.4517 | 0.4586 | 0.4326 | 0.2503 | 2003 | 0.238 |
| lgd.bp.7.0 | 0.5138 | 0.4448 | 0.4529 | 0.4258 | 0.2457 | 1973 | 0.259 |
| lgd.bp.7.5 | 0.5069 | 0.4397 | 0.4481 | 0.4215 | 0.2414 | 1945 | 0.272 |
| lgd.bp.8.0 | 0.5000 | 0.4293 | 0.4420 | 0.4153 | 0.2372 | 1929 | 0.291 |
| lgd.bp.8.5 | 0.4897 | 0.4293 | 0.4347 | 0.4134 | 0.2334 | 1917 | 0.298 |
| lgd.bp.9.0 | 0.4828 | 0.4224 | 0.4234 | 0.4044 | 0.2295 | 1902 | 0.310 |
| lgd.bp.9.5 | 0.4724 | 0.4172 | 0.4176 | 0.3997 | 0.2260 | 1893 | 0.319 |

**Table 4.37:** Performance of system where body part terms are weighted differently, implementing LGD weighting model.

with the LGD weighting model, the best performing system is the system where each body part term are given a 1.5 weight. We observe the same tendency of decrease in performance as we given more weight to the body part terms, and also the increase of UNJ@10 at the same time. This might be one of the reasons of the decrease in performance. The best system in this setting perform better than the system with the same model where body part terms are not given any weighting.

Table 4.39 summarizes the best systems across different models, compared with our baseline. The system with the language model with Dirichlet prior performs best, although the other systems have higher UNJ@10. The best system performs slightly better than our baseline system, and they have the exact same scores of UNJ@10.

## 4.7 Linear Interpolation

In this section, we present the results of our experiments with linear interpolation of different systems into new systems. For each method that we use above, we experiment with linear interpolation of different combinations of retrieval models.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| pl2f | 0.5897 | 0.5517 | 0.5366 | 0.5288 | 0.3509 | 2289 | 0.007 |
| pl2f.bp.1.5 | **0.5931** | **0.5552** | **0.5338** | **0.5228** | **0.3457** | 2264 | 0.040 |
| pl2f.bp.2.0 | *0.5759* | 0.5379 | *0.5218* | *0.5089* | 0.3288 | 2221 | 0.071 |
| pl2f.bp.2.5 | 0.5621 | 0.5207 | *0.5131* | 0.4962 | 0.3116 | 2161 | 0.116 |
| pl2f.bp.3.0 | *0.5586* | 0.5103 | *0.4993* | 0.4826 | 0.2936 | 2092 | 0.134 |
| pl2f.bp.3.5 | 0.5414 | 0.4931 | *0.4886* | 0.4673 | 0.2795 | 2053 | 0.167 |
| pl2f.bp.4.0 | 0.5310 | 0.4759 | *0.4806* | 0.4546 | 0.2691 | 2018 | 0.212 |
| pl2f.bp.4.5 | *0.5379* | 0.4793 | *0.4821* | 0.4564 | 0.2636 | 1992 | 0.231 |
| pl2f.bp.5.0 | 0.5069 | 0.4707 | 0.4583 | 0.4442 | 0.2533 | 1959 | 0.248 |
| pl2f.bp.5.5 | 0.5000 | 0.4500 | 0.4500 | 0.4288 | 0.2457 | 1924 | 0.279 |
| pl2f.bp.6.0 | 0.4793 | 0.4276 | 0.4331 | 0.4108 | 0.2383 | 1894 | 0.314 |
| pl2f.bp.6.5 | 0.4724 | 0.4241 | 0.4248 | 0.4050 | 0.2328 | 1869 | 0.324 |
| pl2f.bp.7.0 | 0.4621 | 0.4224 | 0.4162 | 0.4017 | 0.2291 | 1859 | 0.333 |
| pl2f.bp.7.5 | 0.4552 | 0.4069 | 0.4105 | 0.3900 | 0.2254 | 1842 | 0.348 |
| pl2f.bp.8.0 | 0.4448 | 0.3983 | 0.4029 | 0.3827 | 0.2220 | 1829 | 0.360 |
| pl2f.bp.8.5 | 0.4276 | 0.3914 | 0.3945 | 0.3770 | 0.2187 | 1810 | 0.376 |
| pl2f.bp.9.0 | 0.4345 | 0.3897 | 0.3970 | 0.3757 | 0.2163 | 1795 | 0.381 |
| pl2f.bp.9.5 | 0.4310 | 0.3862 | 0.3947 | 0.3732 | 0.2141 | 1778 | 0.384 |

**Table 4.38:** Performance of system where body part terms are weighted differently, implementing PL2F model.

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| baseline | 0.6207 | 0.5776 | 0.5568 | 0.5471 | 0.3412 | 2267 | 0.010 |
| dir.bp.1.5 | 0.6276 | 0.5879 | 0.5608 | 0.5527 | 0.3411 | 2249 | 0.010 |
| lgd.bp.1.5 | 0.5931 | 0.5483 | 0.5312 | 0.5230 | 0.3310 | 2256 | 0.026 |
| pl2f.bp.1.5 | 0.5931 | 0.5552 | 0.5338 | 0.5228 | 0.3457 | 2264 | 0.040 |

**Table 4.39:** Summary of best systems where body part terms are weighted differently, with different models and weight.

For each experiment, we iterate through values between 0 and 1 to find the best parameter for our linear interpolation. We use the parameters that give us the best P@10. For readability, we give our systems different ids with the format of [method].[model 1][model 2][model 3]. The first part of the id is the method of query manipulation that we use on the original systems: query expansion with synonymous terms using idf (idf) or preferred names (pt) as their term selection criteria, query expansion using non-synonymous related concepts (rel), blind relevance feedback (brf), or weighting of terms that has "body part" as their semantic type (bp). For systems that use only original query terms, this part is absent. The second to last part of the id are the list of the models that the original systems use: language model with Dirichlet prior (dir), LGD language model (lgd), or PL2F weighting model (pl2f). For example, a system called idf.pl.lgd is a combination of systems using the PL2F model and the LGD weighting model, with query expansion using synonymous terms with idf as its term selection criterion

Table 4.40 shows the result of linear interpolation for all methods. For systems that only use the original query expansion terms, the combination of all three retrieval models gives the best performance. This system outperforms all of

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| dir.lgd | *0.6034* | *0.5914* | *0.5491* | *0.5557* | *0.3503* | 2267 | 0.003 |
| pl2f.lgd | *0.5897* | *0.5621* | *0.5379* | *0.5351* | *0.3543* | 2289 | 0.003 |
| pl2f.dir | *0.6172* | *0.5879* | *0.5590* | *0.5534* | *0.3510:* | 2267 | 0.003 |
| pl2f.dir.lgd | **0.6345** | **0.5879** | **0.5698** | **0.5540** | **0.3414** | 2267 | 0.012 |
| idf.dir.lgd | **0.6000** | 0.5690 | 0.5337 | 0.5324 | 0.3434 | 2280 | 0.029 |
| idf.pl2f.lgd | *0.5897* | 0.5655 | 0.5344 | 0.5357 | 0.3569 | 2304 | 0.002 |
| idf.pl2f.dir | 0.5862 | **0.5759** | 0.5327 | 0.5406 | 0.3586 | 2304 | 0.014 |
| idf.pl2f.dir.lgd | 0.5897 | 0.5741 | **0.5373** | **0.5409** | **0.3586** | 2304 | 0.014 |
| pt.pl2f.dir | *0.5897* | *0.5707* | *0.5343* | *0.5386* | *0.3499:* | 2293 | 0.024 |
| pt.pl2f.lgd | **0.5931** | **0.5741** | **0.5345** | **0.5401** | **0.3538** | 2293 | 0.014 |
| pt.pl2f.dir | *0.5897* | *0.5707* | *0.5343* | *0.5386* | 0.3499 | 2293 | 0.024 |
| pt.pl2f.dir.lgd | *0.5862* | *0.5724* | *0.5322* | 0.5392 | 0.3481 | 2293 | 0.024 |
| brf.dir.lgd | *0.6000* | *0.5776* | *0.5497* | *0.5418* | 0.3431 | 2269 | 0.045 |
| brf.pl2f.lgd | *0.5931* | *0.5776* | *0.5465* | *0.5510* | *0.3564:* | 2304 | 0.016 |
| brf.pl2f.dir | **0.6345** | **0.5966** | **0.5748** | **0.5655** | **0.3621** | 2304 | 0.026 |
| brf.pl2f.dir.lgd | *0.6276* | *0.5948* | *0.5724* | *0.5646* | 0.3623 | 2304 | 0.028 |
| rel.dir.lgd | *0.6069* | *0.5983* | *0.5508* | *0.5589* | 0.3527 | 2265 | 0.005 |
| rel.pl2f.lgd | *0.6034* | *0.5690* | *0.5461* | *0.5405* | *0.3568* | 2296 | 0.005 |
| rel.pl2f.dir | *0.6207* | *0.5966* | *0.5616* | *0.5579* | 0.3534 | 2266 | 0.003 |
| rel.pl2f.dir.lgd | **0.6448** | **0.5966** | **0.5771** | **0.5593** | **0.3466** | 2266 | 0.008 |
| bp.dir.lgd | *0.6207* | *0.5983* | *0.5619* | *0.5634* | *0.3533* | 2249 | 0.003 |
| bp.pl2f.lgd | *0.6138* | *0.5707* | *0.5586* | *0.5470* | *0.3552* | 2271 | 0.013 |
| bp.pl2f.dir | **0.6310** | **0.6000** | **0.5666** | **0.5643** | **0.3526** | 2249 | 0.005 |
| bp.pl2f.dir.lgd | *0.6310* | *0.6000* | *0.5666* | *0.5643* | *0.3526* | 2249 | 0.005 |

**Table 4.40:** Performance of linear interpolation of various systems.

the individual systems, and also outperforms our baseline. There are almost no difference on the scores of UNJ@10. For systems that use synonyms for query expansion and use idf as their term selection criteria, combination of all of the retrieval models used gives the best performance for almost all metrics. This system outperforms all of the individual systems that it combines in all metrics but P@5.

The combination of the PL2F model and the LGD weighting models gives the best performance on systems that use synonyms for query expansion and use preferred names as their term selection criteria. This system outperforms one of its individual systems, the system using the PL2F model. However, it does not outperforms the system using the LGD weighting model. It also gives a lower performance compared to our baseline. The system on query expansion using non-synonymous related concepts gets the best result from combining all of the retrieval models. For systems with blind relevance feedback, the combination of the PL2F model and the language model with Dirichlet prior brings the best performance. It also outperforms both of its individual systems and our baseline. The same combination of retrieval model also brings the best performance on the systems weighting body part terms. This system outperforms its individual systems and the baseline system.

Table 4.41 gives the summary of the best systems out of all the linear interpolation systems. The parameters that we use to generate these systems can be

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| pl2f.dir.lgd | 0.6345 | 0.5879 | 0.5698 | 0.5540 | 0.3414 | 2267 | 0.012 |
| idf.pl2f.dir.lgd | 0.5897 | 0.5741 | 0.5373 | 0.5409 | 0.3586 | 2304 | 0.014 |
| pt.pl2f.lgd | 0.5931 | 0.5741 | 0.5345 | 0.5401 | 0.3538 | 2293 | 0.013 |
| brf.pl2f.dir | 0.6345 | 0.5966 | 0.5748 | 0.5655 | 0.3621 | 2304 | 0.043 |
| rel.pl2f.dir.lgd | 0.6448 | 0.5966 | 0.5771 | 0.5593 | 0.3466 | 2266 | 0.008 |
| bp.pl2f.dir | 0.6310 | 0.6000 | 0.5666 | 0.5643 | 0.3526 | 2249 | 0.005 |

**Table 4.41:** Summary of best linearly interpolated systems.

| id | $\lambda_1$ | $\lambda_2$ | $\lambda_3$ |
|---|---|---|---|
| pl2f.dir.lgd | 0.15 | 0.94 | -0.09 |
| idf.pl2f.dir.lgd | 0.81 | 0.2 | -0.01 |
| pt.pl2f.lgd | 0.86 | 0.14 | - |
| brf.pl2f.dir | 0.73 | 0.27 | - |
| rel.pl2f.dir.lgd | 0.18 | 0.91 | -0.09 |
| bp.pl2f.dir | 0.12 | 0.88 | - |

**Table 4.42:** Parameters for the best sytems for linear interpolation.

seen in 4.42. We use these systems for our testing purpose in Chapter 5.

## 4.8 Summary of Training Results

Table 4.43 summarizes the selected results of our experiments on the training query set. We separate the table into sections of methods of query reformulation. Among the systems of unexpanded queries, the system that uses the language model with Dirichlet prior performs the best. We set this system as our baseline. the LGD weighting model gives the best performance for systems on query expansion with synonyms, both for those that use idf or as their term selection criteria. This model also gives the best performance among the systems that implement blind relevance feedback.

In our experiment using non-synonymous relation for query expansion, we decide to use all of the relations that we choose even though those systems do not have the best performance. The reason for this is that there are too many unexpanded queries, and we are trying to minimize the number of them on our small test sets. The language model with Dirichlet prior performs the best among these systems, and also among the systems where we give weights to body part terms. In our experiment with linear interpolation of different retrieval systems, we find that using the combination of all the retrieval systems used performs very well on the best system in our experiment with query expansion using synonymous terms and idf as term selection criterion.

In all the cases where we use term weighting with systems that use the language model with Dirichlet prior, we find that all the systems with different weights give the same performance. This is perhaps related on the implementation of the language model with Dirichlet prior itself. We also find this case on all the retrieval model in our experiment with query expansion with non-synonymous concepts. This might be because of the small size of the expansion pool, and also

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| dir | **0.6207** | **0.5776** | **0.5568** | **0.5471** | *0.3412* | 2267 | 0.010 |
| lgd | *0.5862* | *0.5466* | *0.5297* | *0.5208* | *0.3283* | 2271 | 0.009 |
| pl2f | *0.5897* | *0.5517* | *0.5366* | *0.5288* | **0.3509** | 2289 | 0.007 |
| dir.idf.0.6_0.1_0.1 | 0.5276 | 0.4897 | *0.4642* | 0.4568 | 0.3009 | 2216 | 0.15 |
| lgd.idf.2.6_0.1_0.1 | *0.5828* | *0.5517* | *0.5258* | *0.5233* | *0.331* | 2279 | 0.009 |
| pl2f.idf.0.6_0.1_0.1 | **0.6000** | **0.5621** | **0.5372** | **0.5327** | **0.3539** | 2302 | 0.017 |
| dir.pt.0.6_0.1_0.1 | 0.5276 | 0.4862 | *0.4685* | 0.4591 | 0.2910 | 2108 | 0.185 |
| lgd.pt.0.6_0.1_0.1 | **0.6034** | *0.5638* | **0.5419** | *0.5371* | *0.3376.* | 2284 | 0.022 |
| pl2f.pt.4.6_1.1_1.1 | *0.5862* | **0.5690** | *0.5315* | **0.5377** | **0.3496** | 2293 | 0.022 |
| dir.brf.0.6_0.1_0.1 | *0.5897* | *0.5414* | *0.5370* | *0.5175* | *0.3123* | 2209 | 0.117 |
| lgd.brf.4.6_1.1_1.1 | *0.6034* | 0.5638 | *0.5513* | 0.5362 | 0.3375 | 2269 | 0.10 |
| pl2f.brf.1.1_0.1_0.1 | **0.6069** | **0.5655** | **0.5501** | **0.5414** | **0.3553** | 2304 | 0.014 |
| dir.rel.0.6_0.1_0.1 | **0.6207** | **0.5828** | **0.5570** | **0.5494** | **0.3434** | 2266 | 0.010 |
| lgd.rel.0.6_0.1_0.1 | *0.5897* | *0.5483* | *0.5315* | *0.5223* | *0.3310* | 2277 | 0.009 |
| pl2f.rel.0.6_0.1_0.1 | *0.6000* | *0.5586* | *0.5404* | *0.5325* | *0.3529* | 2296 | 0.009 |
| dir.bp.1.5 | **0.6276** | **0.5879** | **0.5608** | **0.5527** | *0.3411* | 2249 | 0.010 |
| lgd.bp.1.5 | *0.5931* | *0.5483* | *0.5312* | *0.5230* | 0.3310 | 2256 | 0.026 |
| pl2f.bp.1.5 | *0.5931* | *0.5552* | *0.5338* | *0.5228* | **0.3457** | 2264 | 0.040 |
| pl2f.dir.lgd | 0.6345 | 0.5879 | 0.5698 | 0.5540 | 0.3414 | 2267 | 0.012 |
| idf.pl2f.dir.lgd | 0.5897 | 0.5741 | 0.5373 | 0.5409 | 0.3586 | 2304 | 0.014 |
| pt.pl2r.lgd | 0.5931 | 0.5741 | 0.5345 | 0.5401 | 0.3538 | 2293 | 0.013 |
| brf.pl2f.dir | 0.6345 | 0.5966 | 0.5748 | 0.5655 | 0.3621 | 2304 | 0.026 |
| rel.pl2f.dir.lgd | 0.6448 | 0.5966 | 0.5771 | 0.5593 | 0.3466 | 2266 | 0.009 |
| bp.pl2f.dir | 0.6310 | 0.6000 | 0.5666 | 0.5643 | 0.3526 | 2249 | 0.005 |

**Table 4.43:** Summary of performance of selected systems on training set. These systems are to be used on test set.

because there are a lot of queries that are left unexpanded.

From our experiments on the training data, we found that our method of query expansion with synonymous terms with idf as term selection criterion greatly improves on the baseline. The use of preferred terms as term selection criterion and blind relevance feedback do not give any improvement on the baseline. We cannot really observe the effect of adding non-synonymous related concepts to the original queries because of the amount of unexpanded query, and our systems on this method perform similarly to the unexpanded systems. Giving a specific semantic type more weight also improve on the baseline, although not as much as the query expansion systems. Our experiments with linear interpolation shows that the combined systems mostly perform better compared to their individual systems. We will evaluate all the systems in Table 4.43 on our test sets to see whether this effects hold.

# 5. Performance on Test Sets

In this chapter, we discuss the performance of our selected systems (Table 4.43) on our test sets. As we mentioned in Section 2.2, we use two different test sets in this thesis. The first set is a set of 25 queries taken from the query set provided for CLEF eHealth 2014. From here on, we refer to this set as `test_14`. The second set is a set of 34 queries taken from the query set provided for CLEF eHealth 2015. From here on, we refer to this set as `test_15`. These test sets represent two different use cases of medical information retrieval systems. `test_14` is centered around diseases or disorders. It mimics the behavior of users who already know the name of the diseases or disorders that they have, and are attempting to find more information about them. On the other hand, `test_15` is centered around symptoms. It mimics the behavior of users that observe some symptoms that they are having, and are attempting to find out what diseases or disorders they might have.

In the first section of this chapter, we are going to discuss performances of the systems on the `test_14` query set. The next section will discuss the performances of the systems on the `test_15` query set. Lastly, we will discuss and compare the performances on the two query set in the last section. As in our experiments on the training set, for all of the tables, the best value for each metric is emphasized with bold. We perform paired Wilcoxon signed-rank test with the methods in each table, and the values which differences are not statistically significant with the best value are printed in italics.

## 5.1 Systems' Performances on `test_14`

In this section, we discuss the performance of our selected systems on `test_14` test set. Table 5.1 shows the performance of our selected systems on the test set. We divided the results into several parts to differentiate between the kinds of methods that we use with the systems. The first part contains the results of systems that only use the original query terms. As in our experiment with the training query set, the model using the language model with Dirichlet prior (our baseline system) is the best performing system for systems that use only original query terms. The second best performance is achieved by the system using the PL2F model, also similar with our result from training. As in training, this system also has a higher MAP and number of relevant documents retrieved compared to our baseline system. However, for this test set the system using the LGD weighting model performs better on P@10 compared with this system. These systems have relatively low scores of UNJ@10.

The second part of the table contains the results of systems with query expansions with synonymous terms using idf as their term selection criteria, where each field of the queries is weighted differently. On our training set, the system using the LGD weighting model gives the best performance for every metrics but P@10, MAP, and number of relevant documents retrieved. The system using PL2F, which is the second best performing system for this method on the training model, gives the best performance for these metrics. However, on `test_14` query set, the system using PL2F only has a higher performance on MAP and number

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| dir | **0.6960** | **0.6240** | **0.7147** | **0.6670** | *0.3147* | 1205 | 0.032 |
| lgd | *0.5920* | *0.6080* | 0.6129 | 0.6219 | 0.3107 | 1174 | 0.028 |
| pl2f | *0.6000* | *0.6000* | 0.6199 | 0.6240 | **0.3382** | 1225 | 0.02 |
| dir.idf.0.6_0.1_0.1 | *0.5120* | 0.4360 | *0.5234* | 0.4741 | 0.2306 | 1115 | 0.356 |
| lgd.idf.2.6_0.1_0.1 | **0.5920** | **0.6080** | **0.6129** | **0.6221** | *0.3114* | 1180 | 0.032 |
| pl2f.idf.0.6_0.1_0.1 | *0.5840* | *0.5720* | *0.6001* | *0.5990* | **0.3287** | 1216 | 0.044 |
| dir.pt.0.6_0.1_0.1 | *0.5200* | *0.4840* | *0.5507* | *0.5231* | 0.2320 | 1067 | 0.28 |
| lgd.pt.0.6_0.1_0.1 | **0.6320** | **0.5960** | **0.6368** | *0.6143* | 0.3056 | 1176 | 0.052 |
| pl2f.pt.4.6_1.1_1.1 | *0.6160* | *0.5920* | *0.6267* | **0.6192** | **0.3358** | 1213 | 0.056 |
| dir.brf.0.6_0.1_0.1 | *0.6480* | *0.6080* | *0.6930* | **0.6595** | *0.3084* | 1161 | 0.12 |
| lgd.brf.4.6_1.1_1.1 | **0.6560** | **0.6160** | **0.6589** | *0.6432* | *0.3332* | 1195 | 0.02 |
| pl2f.brf.1.1_0.1_0.1 | *0.6240* | *0.6080* | *0.6455* | *0.6367* | **0.3531** | 1244 | 0.012 |
| dir.rel.0.6_0.1_0.1 | **0.6960** | **0.6240** | **0.7147** | **0.6670** | *0.3147* | 1205 | 0.032 |
| lgd.rel.0.6_0.1_0.1 | *0.5920* | *0.6080* | 0.6129 | 0.6219 | 0.3107 | 1174 | 0.028 |
| pl2f.rel.0.6_0.1_0.1 | *0.6000* | *0.6000* | 0.6199 | 0.6240 | **0.3382** | 1225 | 0.02 |
| dir.bp.1.5 | **0.6960** | **0.6240** | **0.7147** | **0.6670** | *0.3147* | 1205 | 0.032 |
| lgd.bp.1.5 | *0.6320* | *0.6160* | 0.6337 | 0.6277 | *0.3111* | 1164 | 0.032 |
| pl2f.bp.1.5 | *0.5840* | *0.5840* | 0.6006 | 0.6024 | **0.3233** | 1219 | 0.032 |
| pl2f.dir.lgd | 0.6640 | 0.5920 | 0.6879 | 0.6413 | 0.3121 | 1205 | 0.052 |
| idf.pl2f.dir.lgd | 0.6480 | 0.6000 | 0.6478 | 0.6244 | 0.3352 | 1214 | 0.048 |
| pt.pl2f.lgd | 0.6160 | 0.6000 | 0.6299 | 0.6269 | 0.3369 | 1214 | 0.048 |
| brf.pl2f.dir | 0.6640 | 0.6240 | 0.6791 | 0.6514 | 0.3596 | 1245 | 0.032 |
| rel.pl2f.dir.lgd | 0.6560 | 0.6000 | 0.6873 | 0.6461 | 0.3144 | 1205 | 0.048 |
| bp.pl2f.dir | 0.6720 | 0.6240 | 0.7001 | 0.6646 | 0.3200 | 1205 | 0.02 |

**Table 5.1:** Performance of selected systems on test set test_14.

of relevant documents retrieved when compared to the system using the LGD weighting model. Similar to the results of our experiments on the training set, the system using language model with Dirichlet prior is the only system that has a high UNJ@10. The other two systems have quit low scores UNJ@10. However, the system using the PL2F model has a higher score on this metric compared to its score on the training set. No systems in this method outperform our baseline system.

The third part of the table contains the results of systems similar to the systems in the previous part. However, the systems in this part use preferred names as their term selection criteria. The performances of the systems in these experiments are a little bit different compared to the performances of the same systems on training set. On the training set, the system using the LGD weighting model has the highest performance on all but two metrics: P@5 and NDCG@5. These two metrics are higher on the system using PL2F weighting model. However, on the test_14 query set, the system using LGD also has the highest P@10 among all the systems. The system using language model for Dirichlet prior have the highest score of UNJ@10. However, the score is less than the score gotten by the same system on the training set. There are no systems from our experiments with this method that outperform our baseline system, not unlike our result from our experiments on the training set.

The fourth part of the table contains the results of systems using blind relevance feedback, where we also perform field weighting. The results from these experiments are not very similar to the results of the experiments with the same setting on our training set. On our training set, the best performance is given by the system using the PL2F model. However, in the experiments on `test_14` set, the LGD weighting model gives the best performance among other systems, except for MAP and number of relevant documents retrieved for which the PL2F model still has the best performance. The model using language model with Dirichlet prior still has the highest UNJ@10, although not as high as in the experiment on the training set. Like our result on the training set, there are also no systems that outperform the baseline system.

The next part of the table contains the results of systems using non-synonymous related concepts for query expansion. Interestingly, the performances of the systems for the experiments are the same with the results of our experiments using only original query terms. On the training set, the results of the systems are slightly different compared to the results of the systems using only the original query terms. We investigate whether the cause for this is because there are no expansion concepts added to the query set at all. We found this out of 25 queries in the `test_14` set, there are 15 queries for which no expansion terms are added at all. Even though this is a high number, we found a similar case on our experiments on the training data and still the result was different. Perhaps in this case the number of queries are too small to affect the end results. As in our experiments using only original query terms, the model using language model with Dirichlet prior performs the best. This is different compared to our experiments using the same method on the training set, where system with the LGD weighting model performs the best.

The second to last part of the table is the results of our experiments on systems utilizing the semantic network, where body part terms are weighted. Again, the performance of this system is the same with the performance of the systems on our systems that only use original query terms. We investigate if there are no body parts terms in the system. We find that there are 15 queries out of 25 for which there are at least one body part terms that are weighted. Although this is quite a large part of the training set, perhaps as in the previous experiments, the result of the query set is too small to see the difference. The only difference is that for two last systems, the number of relevant query retrieved are smaller. On our experiments using the same method on the training data, the system using the LGD weighting model have the best performance. In this case, the system using language model with Dirichlet prior performs the best.

The last part of the table is the results of our experiments on linear interpolation of different systems. In this case, we do not pick the best system because it would not be a fair comparison, as the systems of this part are linear interpolation on different methods. On the training process, most of the systems have better result than the best individual system on the method. The only exception to this is the system on query expansion with preferred names. We observe a similar result in this case, although the system on expansion of non-synonymous related concept and body part weighting also do not have a better performance compared to the individual systems.

To summarize, unlike the results on the training set, there are no systems

| id | P@5 | P@10 | NDCG@5 | NDCG@10 | MAP | rel_ret | UNJ@10 |
|---|---|---|---|---|---|---|---|
| dir | *0.3394* | *0.3394* | *0.2895* | *0.3095* | **0.2262** | 819 | 0 |
| lgd | **0.3939** | **0.3667** | **0.3257** | **0.3190** | *0.2153* : | 796 | 0 |
| pl2f | *0.3576* | *0.3303* | *0.2868:* | 0.2853 | 0.1975 | 781 | 0.012 |
| dir.idf.0.6_0.1_0.1 | 0.2750 | 0.2000 | 0.2582 | 0.2301 | 0.1402 | 187 | 0.109 |
| lgd.idf.2.6_0.1_0.1 | **0.4000** | **0.3625** | **0.3547** | **0.3371** | **0.2049** | 205 | 0 |
| pl2f.idf.0.6_0.1_0.1 | *0.3500* | 0.3125 | *0.3086* | 0.2997 | *0.1646* | 197 | 0.003 |
| dir.pt.0.6_0.1_0.1 | 0.3333 | 0.2939 | *0.2849* | *0.2804* | *0.2089* | 763 | 0.306 |
| lgd.pt.0.6_0.1_0.1 | **0.4545** | **0.3879** | **0.3756** | **0.3524** | **0.2400** | **792** | 0.039 |
| pl2f.pt.4.6_1.1_1.1 | 0.3818 | *0.3606* | 0.3206 | 0.3157 | 0.2178 | 791 | 0.030 |
| dir.brf.0.6_0.1_0.1 | *0.3576* | *0.3303* | *0.2979* | *0.2983* | *0.1966* | 777 | 0.158 |
| lgd.brf.4.6_1.1_1.1 | *0.3576* | 0.3303 | *0.2960* | 0.2872 | 0.1995 | 784 | 0.024 |
| pl2f.brf.1.1_0.1_0.1 | **0.4061** | **0.3758** | **0.3445** | **0.3489** | **0.2283** | 800 | 0.060 |
| dir.rel.0.6_0.1_0.1 | *0.3333* | *0.3364* | *0.2870* | *0.3060* | **0.2238** | 817 | 0 |
| lgd.rel.0.6_0.1_0.1 | **0.3879** | **0.3636** | **0.3237** | **0.3170** | *0.2120* | 794 | 0 |
| pl2f.rel.0.6_0.1_0.1 | *0.3515* | *0.3242* | *0.2846* | 0.2821 | 0.1947 | 780 | 0.015 |
| dir.bp.1.5 | *0.3394* | *0.3394* | *0.2895* | *0.3097* | **0.2271** | 819 | 0.003 |
| lgd.bp.1.5 | **0.3636** | **0.3364** | **0.3005** | **0.2983** | *0.2088* | 804 | 0.046 |
| pl2f.bp.1.5 | *0.3515* | *0.3091* | *0.2880* | *0.2726* | 0.1906 | 777 | 0.067 |
| pl2f.dir.lgd | 0.3394 | 0.3182 | 0.2947 | 0.3033 | 0.2093 | 819 | 0.033 |
| idf.pl2f.dir.lgd | 0.3576 | 0.3242 | 0.2869 | 0.2892 | 0.2048 | 783 | 0.076 |
| pt.pl2f.lgd | 0.4182 | 0.3758 | 0.3445 | 0.3299 | 0.2294 | 791 | 0.021 |
| brf.pl2f.dir | 0.3879 | 0.3576 | 0.3293 | 0.3225 | 0.2304 | 785 | 0.055 |
| rel.pl2f.dir.lgd | 0.3394 | 0.3091 | 0.2984 | 0.2977 | 0.2087 | 817 | 0.036 |
| bp.pl2f.dir | 0.3515 | 0.3394 | 0.3030 | 0.3180 | 0.2294 | 820 | 0.006 |

**Table 5.2:** Performance of selected systems on test set test_15.

that improve on the performance of the baseline system on test_14 test set. For query expansion using synonymous terms, systems that use preferred names as their term selection criteria perform better than systems that use idf as their term selection criteria. On this test set, blind relevance feedback gives a higher performance compared to query expansion using synonyms. On our experiments with query expansion using non-synonymous related terms and weighting body part term, the results are the same with the results from unexpanded systems. Combining systems using linear interpolation generally gives higher performances compared to individual systems.

## 5.2 Systems' Performances on test_15

In this section, we discuss performances of our selected systems with test_15 test set. Table 5.2 shows the results of our experiments. We divided the results into several parts to differentiate between the kinds of methods that we use with the systems. The first part contains the results of systems that only use the original query terms. Unlike the case of our experiments on the training set and test_14 set, the model using the LGD weighting model is the best performing system for this setting. It has the highest scores for all metrics except for MAP and number of relevant documents retrieved, for which the systems using language model with

Dirichlet prior has the highest scores. The second best performance is achieved by the system using Dirichlet model, similar with our result from training. These systems have relatively low scores of UNJ@10.

The second part of the table contains the results of systems with query expansions with synonymous terms using idf as their term selection criteria, where each field of the queries is weighted differently. As in our experiments on the training set, the system with the LGD weighting model gives the best performance for this method. It even has the best performance on all metrics in this case. The second best system is the system using the PL2F model, which on the training set has the best number of relevant documents retrieved and MAP score. As in the results from the training set, the best result of this method outperforms the result of our baseline system. It also has zero UNJ@10 in the retrieved set, so we can say that the comparison is fair and that the result of this system is very similar to the result of the baseline system. However, the difference in performance is not as high as in our experiment on the training set.

The third part of the table contains the results of systems similar to the systems in the previous part. However, the systems in this part use preferred names as their term selection criteria. Unlike in our experiments on the training set where the system using the PL2F model has the best performance on most metrics, on `test_15` set the system with LGD model gives the best performance. It is followed by the system with the PL2F model. The system with language model with Dirichlet prior still has a high UNJ@10. As in our results from training, the best result of this method outperforms the result of the baseline system. In this case, it even has a higher difference of performance to the result of the training. It has a higher UNJ@10.

The fourth part of the table contains the result of systems with blind relevance feedback with field weighting. As in our result on the training set, the system using the PL2F model has a higher performance among all the systems for this method. The performance of two other systems are similar, except that the system using language model with Dirichlet prior has a higher UNJ@10. Unlike the results on the training set, our best model for this method outperforms our baseline system.

The next part of the table is the result of systems using non-synonymous related concepts with field weighting. Unlike the results of our experiments on the training set where the system with language model with Dirichlet prior gives the best performance, in this experiment we find that the system using LGD weighting system gives the best performance. The UNJ@10 for these systems are quite low. We find that out of 34 queries in the query set, 24 queries are not expanded. This ratio is somehow similar to `test_14` query set. However, unlike in the results on `test_14` the results of the systems for this method has some differences with the result from the systems that only use the original query expansion. The best system of this method does not outperform our baseline system.

The second to last part of the table contains the results of our experiments on systems utilizing the semantic network, where body part terms are weighted. Unlike what we find in our experiments on `test_14` query set, the performances in the systems of this method differ from the performances of systems that only use the original query terms. We find that out of 34 queries in the query set,

25 queries have at least one body part terms. Unlike in our experiments on the training set, we find that the model using the LGD weighting model gives the best performance except for MAP and number ot relevant documents retrieved. For these two metrics, the system using language model with Dirichlet prior gives the highest performance. There are some unjudged documents on the last two systems of this method. Unlike in our experiment on `test_14`, these scores are different compared to the scores of UNJ@10 on the systems that only uses the original query terms. The best result from this method does not outperform our baseline model.

The last part of the table is the results of our experiments on linear interpolation of different systems. Again, we do not pick the best system in this case because it would not be a fair comparison, as the systems of this part are linear interpolation on different methods. Interestingly, we find that unlike the results on the training set and `test_14` set, the results of the linear interpolation of different systems on `test_15` do not outperform the best individual systems on each method.

To summarize, on our experiment on `test_15` test set, query expansions using synonymous terms improves the performance of the systems on the baseline. Systems that use preferred names as their term selection criteria give the highest improvement. Blind relevance feedback also gives some improvement on the baseline, although lower than the systems that use preferred names. Adding non-synonymous related concepts do not improve the performance, nor does weighting the body part terms in the query. Linear interpolations of different systems do not improve on their individual systems.

## 5.3 Discussion

There are some observations that we make on the results of the experiments on the training set and also the two test sets. First of all, the scores of the experiments on `test_15` set are lower than the experiments on `test_14` set, and the systems have lower number of relevant documents retrieved. We think that this is because the queries from 2015 set are harder compared to the queries from 2014 set. If we are given what the diseases are, it is possible to find the information on it in a relatively easy way. However, given a set of symptoms there could be multiple possibilities of diseases that have those symptoms. For the training set where the queries are a mix of 2014 and 2015 queries, we do an evaluation per query for some of the systems and find that the queries from the 2015 test set always have a lower performance on the metrics compared to the queries from 2014 test set.

While this is not true for our result on the training set, we find that on both of test sets systems that use preferred names as their term selection criteria for query expansion using synonymous terms gives a better performance compared to systems that uses idf as their term selection criteria. We think that this shows even though idf reflects how discerning a term is in the collection, preferred name is a better criteria on showing how likely a document is talking about a certain concept because it brings additional information from the Metathesaurus. However, idf might be a better indicator in some cases if we are only considering a certain set of documents that most likely are relevant instead of the entire collection. The results of our experiments on systems using blind relevance feedback are better

than the query expansion systems using preferred names on the training set and `test_14`. We think that this is because using blind relevance feedback gives the systems more information about the characteristic of the document collection, while using preferred names does not give this information at all.

However, this is not the case for our experiments on `test_15` test set. We think that the difference here is the characteristic of the queries. As we said before, the queries from 2015 query set are more difficult than the queries from 2014 query set. Because of this, when we are doing blind relevance feedback it might be the case that the top n documents retrieved in the initial retrieval are actually not as relevant as the systems think. Some of the documents might be related to other diseases that contain the same symptoms. As mentioned in Subsection 1.3.1, blind relevance feedback has a danger of bias. If we are adding terms from these documents, the next retrieval might drift to the direction of these other diseases. In this case, adding preferred names are more precise because it will only add other names of the symptoms.

In our experiments on query expansion using non-synonymous related concepts, we have difficulties of selecting the correct relations. It seems that even though our selected relations theoretically fit the characteristic of the queries, the low number of occurrences in the query set caused a lot of queries to not be expanded at all. This makes it difficult to see the effect of this approach. The results of the systems are very similar to the results of the unexpanded systems for this case. It will perhaps be better to do a thorough observation on the relations and add only relations that have a lot of occurrences in the query sets and document collection. However, as [Koopman et al., 2012] shows, choosing what relations to add are not an easy task and a more sophisticated way to choose relations that will help us improve the retrieval results is needed.

In our experiment with the body part terms weighting, while we find that it gives some improvement on the training set it does not give any improvement on both test sets. This method is also very dependent on the characteristic of the queries. The queries that describes physical symptoms naturally have more body part terms compared to query that only mention diseases. We have to consider other semantic types in the semantic network and experiments with the effect of treating them differently in the query, e.g. by giving them different weights.

While our experiments on linear interpolation of systems perform well on training set and `test_14` set, they does not give a boost to the performances on `test_15`. Choosing the correct parameter that will fit different characteristics of queries are not an easy task. It will be interesting to combine the best systems from different methods of query reformulation to see whether it gives any improvement on the retrieval result.

# Conclusion

In this thesis, we have described our experiments on query expansion in medical information retrieval. We used the Unified Medical Language System (UMLS), a repository of biomedical vocabularies, for our purpose. We utilized two of the resources that UMLS has: the Metathesaurus and Semantic Network. We explored on several ideas that we thought would improve on the performance of medical information retrieval systems that only use original queries terms. We used UMLS Metathesaurus to expand our original queries not only with the synonyms of their original terms, but also with non-synonymous related concepts. For query expansion using synonymous terms, we experimented with two different term selection criteria: inverse document frequency (idf) and preferred names from Metathesaurus. As a comparison, we also experimented with another method of query reformulation: blind relevance feedback. For all of the aforementioned methods, we also experimented with weighting different fields differently. We also used UMLS Semantic Network to give different weights to terms with certain semantic types. Lastly, we combined our best systems using linear interpolation.

Our experiments that were described in this thesis were preceded by our participation in CLEF eHealth 2015 Task 2 in medical information retrieval. We used part of the methods that we used in this thesis for our submission to this task. However, the experiments in our submission and the experiments in this thesis differed on the split of training query set and test query sets. In our submission to CLEF eHealth 2015, we used the 2014 query set as our training set, and 2015 query set as our test set. Our submission to CLEF eHealth 2015 was described in [Saleh et al., 2015]. The official result of the entire task was described in [Palotti et al., 2015]. Our submissions performed above the median of the participants.

We used the query set and document set provided by CLEF eHealth organizer for 2014 and 2015 medical information retrieval shared task. The characteristics of both sets are quite different. While the 2014 queries models the queries used by laypeople who want to find more about their known diseases, the 2015 queries models the queries used by laypeople who observe certain symptoms and want to know what conditions they might have. Our training set contained 58 queries, with 25 queries randomly chosen from 2014 queries set (`test_14`) and 33 query randomly chosen from 2015 query set (`test_15`). We used the rest of the queries to form two different test sets: one test set that contains the rest of the 2014 queries, and another set that contains the rest of the 2015 queries. We tuned all of our systems on our training query set to find the optimal parameters and methods combination. Afterwards, we tested the best systems on the two sets of test queries. We performed paired Wilcoxon signed-rank test to determine whether the performance differences of the systems are statistically significant to the results of the best systems.

For our query expansion systems using synonymous terms, we found that on both test sets systems that used preferred names as their term selection criteria gave better performances compared to systems that used idf. Our experiments' results showed that preferred names was a better criteria on showing how likely a document is talking about a certain concept, because by using idf the systems

missed the additional information from the Metathesaurus about the concepts. However, idf might be a good criteria if we first do an initial retrieval using blind relevance feedback because instead of looking at the whole document set, we are now only looking at documents that are most likely relevant to certain concepts. However, this might depend on the characteristic of the queries as it did not work well on our `test_15` set. It might be the case that some of the top n documents on the initial retrieval might be related to other diseases that contain the same symptoms, which caused a bias on the next retrieval process.

We had difficulties of selecting the correct relations to add for our experiments with query expansion using non-synonymous related concepts. Even though our selected relations theoretically fit the characteristic of the queries, the low number of occurrences in the query set caused a lot of queries to not be expanded at all. This made it difficult to see the effect of this approach, since the results of the systems were very similar to the results of the unexpanded systems. In our experiments with the body part terms weighting, while we found that it gave some improvement on the training set it did not gave any improvement on both test sets. This method was also very dependent on the characteristic of the queries. Our experiments on linear interpolation of systems performed well on training set and `test_14` set, but it did not give a better performance compared to the individual systems on `test_15`.

The scores of the experiments on our `test_15` set were lower than the experiments on `test_14` set, and the systems have lower number of relevant documents retrieved. We believe that this was because the queries in that set are harder compared to the queries from 2014 set. Given the diseases, it is more possible to find the information on it in a relatively easy way compare to using a set of symptoms to find one disease, as there could be multiple possibilities of diseases that have those symptoms.

For our future works, we would like to experiment with combining information of preferred names with the application of blind relevance feedback. We believe that by doing this we would reduce the risk of bias on the initial document retrieval. It would also gives the information of the characteristic of the collection that the systems that only consider the preferred terms are missing. We would also like experiment with different terms selection criteria. We would also like to do a thorough observation on the non-synonymous relations to decide which relation to add in order to improve the performance of the retrieval task. It would be useful to experiment with adding relations while considering the characteristic of document collections and query sets, e.g. add only relations that have a lot of occurrences in the query sets and document collection. We would need to define a sophisticated way on selecting the relations from our set of candidates.

Due to the size of the relations in the semantic network, we did not perform query expansion using relation in the semantic network. We need to do a thorough investigation on which relations to add, like in our experiment with non-synonymous relations. We have to consider other semantic types in the semantic network and experiments with the effect of treating the differently in the query, e.g. by giving them different weights. In this thesis, we only experimented with combining systems that used the same query reformulation method. It might be interesting to combine the best systems from different methods of query reformulation to see whether it gives any improvement on the retrieval result.

Most importantly, we plan to deal with the problem of unjudged document that is shown by our UNJ@10 metrics. As has been said before, systems that have lower performance on other metrics but have a higher UNJ@10 has a chance of actually being the better performing systems compared to other systems. This will be the case if the unjudged documents are actually relevant to the queries. This means that the results of the systems differ greatly to the results of the systems that are used for relevance assesment. This made the evaluation biased and the comparison not fair. It will be interesting to perform a relevance judgment to these documents in order to have a fairer comparison between systems.

# Bibliography

[Amati and Van Rijsbergen, 2002] Amati, G. and Van Rijsbergen, C. J. (2002). Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Trans. Inf. Syst.*, 20(4):357–389.

[Aronson and Rindflesch, 1997] Aronson, A. and Rindflesch, T. (1997). Query expansion using the UMLS Metathesaurus. In Masys, D., editor, *Proceedings of the 1997 AMIA Annual Fall Symposium*, pages 485–489.

[Aronson, 2001] Aronson, A. R. (2001). Effective mapping of biomedical text to the UMLS Metathesaurus: the MetaMap program. *Proc AMIA Symp*, pages 17–21.

[Aronson and Lang, 2010] Aronson, A. R. and Lang, F.-M. (2010). An overview of MetaMap: historical perspective and recent advances. *JAMIA*, 17:229–236.

[Aswani et al., 2012] Aswani, N., Beckers, T., Birngruber, E., Boyer, C., Burner, A., Bystroň, J., Choukri, K., Cruchet, S., Cunningham, H., Dědek, J., Dolamic, L., Donner, R., Dungs, S., Eggel, I., Foncubierta-Rodríguez, A., Fuhr, N., Funk, A., García Seco de Herrera, A., Gaudinat, A., Georgiev, G., Gobeill, J., Goeuriot, L., Gómez, P., Greenwood, M., Gschwandtner, M., Hanbury, A., Hajič, J., Hlaváčová, J., Holzer, M., Jones, G., Jordan, B., Jordan, M., Kaderk, K., Kainberger, F., Kelly, L., Kriewel, S., Kritz, M., Langs, G., Lawson, N., Markonis, D., Martinez, I., Momtchev, V., Masselot, A., Mazo, H., Müller, H., Pecina, P., Pentchev, K., Peychev, D., Pletneva, N., Pottecherc, D., Roberts, A., Ruch, P., Samwald, M., Schneller, P., Stefanov, V., Tinte, M. A., Urešová, Z., Vargas, A., and Vishnyakova, D. (2012). Khresmoi: Multimodal multilingual medical information search. In *Proceedings of the 24th International Conference of the European Federation for Medical Informatics*.

[Bodenreider, 2004] Bodenreider, O. (2004). The unified medical language system (UMLS): Integrating biomedical terminology.

[Clinchant and Gaussier, 2009] Clinchant, S. and Gaussier, É. (2009). Bridging language modeling and divergence from randomness models: A log-logistic model for IR. In *Advances in Information Retrieval Theory, Second International Conference on the Theory of Information Retrieval, ICTIR 2009, Cambridge, UK, September 10-12, 2009, Proceedings*, pages 54–65.

[Fox, 2011] Fox, S. (2011). Health topics: 80% of internet users look for health information online. Technical report, Pew Research Center.

[Goeuriot et al., 2014] Goeuriot, L., Kelly, L., Li, W., Palotti, J., Pecina, P., Zuccon, G., Hanbury, A., Jones, G., and Mueller, H. (2014). ShARe/CLEF eHealth evaluation lab 2014, task 3: User-centred health information retrieval. In *Working Notes for CLEF 2014 Conference*.

[Hull, 1993] Hull, D. (1993). Using statistical testing in the evaluation of retrieval experiments. In *Proceedings of the 16th Annual International ACM SIGIR*

*Conference on Research and Development in Information Retrieval*, SIGIR '93, pages 329–338, New York, NY, USA. ACM.

[Jones, 1972] Jones, K. S. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28:11–21.

[Koopman et al., 2012] Koopman, B., Zuccon, G., Nguyen, A. N., Vickers, D., Butt, L., and Bruza, P. (2012). Exploiting SNOMED CT concepts & relationships for clinical information retrieval: Australian e-health research centre and queensland university of technology at the TREC 2012 medical track. In *Proceedings of The Twenty-First Text REtrieval Conference, TREC 2012, Gaithersburg, Maryland, USA, November 6-9, 2012*.

[Luhn, 1957] Luhn, H. P. (1957). A statistical approach to mechanized encoding and searching of literary information. *IBM J. Res. Dev.*, 1(4):309–317.

[Macdonald et al., 2005] Macdonald, C., Plachouras, V., He, B., Lioma, C., and Ounis, I. (2005). University of Glasgow at WebCLEF 2005: Experiments in per-field normalisation and language specific stemming. In Peters, C., Gey, F. C., Gonzalo, J., Müller, H., Jones, G. J. F., Kluck, M., Magnini, B., and de Rijke, M., editors, *CLEF*, volume 4022 of *Lecture Notes in Computer Science*, pages 898–907. Springer.

[MacKay and Peto, 1994] MacKay, D. J. and Peto, L. C. B. (1994). A hierarchical dirichlet language model. *Natural Language Engineering*, 1:1–19.

[Manning et al., 2008] Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

[McCray, 2003] McCray, A. T. (2003). An upper-level ontology for the biomedical domain. *Comparative and Functional Genomics*, 4.1:80–84.

[National Library of Medicine (US), 2009] National Library of Medicine (US) (2009). UMLS®reference manual [internet]. `http://www.ncbi.nlm.nih.gov/books/NBK9676/`. Accessed: 2015-06-26.

[Ounis et al., 2006] Ounis, I., Amati, G., Plachouras, V., He, B., Macdonald, C., and Lioma, C. (2006). Terrier: A high performance and scalable information retrieval platform. In *Proceedings of ACM SIGIR'06 Workshop on Open Source Information Retrieval (OSIR 2006)*.

[Ounis et al., 2005] Ounis, I., Amati, G., V., P., He, B., Macdonald, C., and Johnson (2005). Terrier Information Retrieval Platform. In *Proceedings of the 27th European Conference on IR Research (ECIR 2005)*, volume 3408 of *Lecture Notes in Computer Science*, pages 517–519. Springer.

[Ounis et al., 2007] Ounis, I., Lioma, C., Macdonald, C., and Plachouras, V. (2007). Research directions in Terrier. *Novatica/UPGRADE Special Issue on Web Information Access, Ricardo Baeza-Yates et al. (Eds), Invited Paper*.

[Palotti et al., 2015] Palotti, J., Zuccon, G., Goeuriot, L., Kelly, L., Hanbury, A., Lupu, G. J. J. M., and Pecina, P. (2015). CLEF eHealth evaluation lab 2015, task 2: Retrieving information about medical symptoms. In *CLEF 2015 Online Working Notes*. CEUR-WS.

[Saleh et al., 2015] Saleh, S., Bibyna, F., and Pecina, P. (2015). CUNI at the CLEF eHealth 2015 task 2. In *CLEF 2015 Labs and Workshops, Notebook Papers*.

[Saleh and Pecina, 2014] Saleh, S. and Pecina, P. (2014). CUNI at the ShARe/-CLEF eHealth evaluation lab 2014. In *Working Notes for CLEF 2014 Conference*, pages 226–235.

[Salton et al., 1975] Salton, G., Wong, A., and Yang, C. S. (1975). A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620.

[Shenwei et al., 2014] Shenwei, W., Nie, J., Liu, X., and Liu, X. (2014). An investigation of the effectiveness of concept-based approach in medical information retrieval. In *Working Notes for CLEF 2014 Conference*, pages 236–247.

[Srinivasan, 1996] Srinivasan, P. (1996). Query expansion and MEDLINE. *Inf. Process. Manage.*, 32(4):431–443.

[Stanton et al., 2014] Stanton, I., Ieong, S., and Mishra, N. (2014). Circumlocution in diagnostic medical queries. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '14, pages 133–142, New York, NY, USA. ACM.

[Wilcoxon, 1945] Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83.

[Zhai and Lafferty, 2004] Zhai, C. and Lafferty, J. (2004). A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214.

[Zuccon et al., 2015] Zuccon, G., Koopman, B., and Palotti, J. (2015). Diagnose this if you can: On the effectiveness of search engines in finding medical self-diagnosis information. In *Advances in Information Retrieval (ECIR 2015)*, pages 562–567. Springer.

# List of Tables


1.1 Concept, Term, String and Atom Identifiers . . . . . 7

2.1 Statistics of the query sets . . . . . 20

3.1 Number of occurrence of selected relations in UMLS. . . . . 31

4.1 Performance of systems using original query terms on training set 37
4.2 Performance of query expansion system using synonyms with idf as their terms selection criteria (`dir`) . . . . . 38
4.3 Performance of query expansion system using synonyms with idf as their terms selection criteria (`lgd`) . . . . . 38
4.4 Performance of query expansion system using synonyms with idf as their terms selection criteria (`pl2f`) . . . . . 39
4.5 Summary of best systems on query expansion using synonyms and idf as their terms selection criteria . . . . . 39
4.6 Performance of query expansion system using synonyms with preferred names as their terms selection criteria (`dir`) . . . . . 40
4.7 Performance of query expansion system using synonyms with preferred names as their terms selection criteria (`lgd`) . . . . . 40
4.8 Performance of query expansion system using synonyms with preferred names as their terms selection criteria (`pl2f`) . . . . . 41
4.9 Summary of best systems on query expansion using synonyms and preferred names as their terms selection criteria . . . . . 42
4.10 ID of relations inclusion and exclusion. . . . . 43
4.11 Performance on system using Dirichlet prior model with query expansion using non-synonymous related concepts with different relations. . . . . 43
4.12 Performance on system using LGD weighting model with query expansion using non-synonymous related concepts with different relations. . . . . 44
4.13 Performance on system using PL2F model with query expansion using non-synonymous related concepts with different relations. . 45
4.14 Number of unexpanded queries. . . . . 45
4.15 Summary of selected systems with query expansion using non-synonymous related concepts, using different models. . . . . 46
4.16 Performance of system with blind relevance feedback using idf as their terms selection criteria (`dir`) . . . . . 46
4.17 Performance of system with blind relevance feedback using idf as their terms selection criteria (`lgd`) . . . . . 47
4.18 Performance of system with blind relevance feedback using idf as their terms selection criteria (`pl2f`) . . . . . 47
4.19 Summary of best systems with blind relevance feedback using idf as their terms selection criteria . . . . . 48
4.20 Performance of field weighting on system with query expansion using synonyms using idf as their terms selection criteria (`dir`) . . 48

4.21 Performance of field weighting on system with query expansion using synonyms using idf as their terms selection criteria (`lgd`) . . 49
4.22 Performance of field weighting on system with query expansion using synonyms using idf as their terms selection criteria (`pl2f`) . 50
4.23 Summary of best systems with field weighting on system with query expansion using synonyms using idf as their terms selection criteria . . . . . 50
4.24 Performance of field weighting on system with query expansion using synonyms using preferred names as their terms selection criteria (`dir`) . . . . . 51
4.25 Performance of field weighting on system with query expansion using synonyms using preferred names as their terms selection criteria (`lgd`) . . . . . 51
4.26 Performance of field weighting on system with query expansion using synonyms using preferred names as their terms selection criteria (`pl2f`) . . . . . 52
4.27 Summary of best systems with field weighting on system with query expansion using synonyms using preferred names as their terms selection criteria . . . . . 52
4.28 Performance of field weighting on system with query expansion using non-synonymous concepts, using Dirichlet prior model . . . 53
4.29 Performance of system using blind relevance feedback with field weighting, using LGD weighting model. . . . . 53
4.30 Performance of system using blind relevance feedback with field weighting, using PL2F model. . . . . 54
4.31 Summary of best systems with field weighting on systems with query expansion using non-synonymous terms, with differen models and weight. . . . . 54
4.32 Performance of system using blind relevance feedback with field weighting, using Dirichlet prior model. . . . . 54
4.33 Performance of system using blind relevance feedback with field weighting, using LGD weighting model. . . . . 55
4.34 Performance of system using blind relevance feedback with field weighting, using PL2F model. . . . . 56
4.35 Summary of best systems using blind relevance feedback with field weighting . . . . . 56
4.36 Performance of systems where body part terms are weighted differently (`dir`) . . . . . 57
4.37 Performance of systems where body part terms are weighted differently (`lgd`) . . . . . 57
4.38 Performance of systems where body part terms are weighted differently (`pl2f`) . . . . . 58
4.39 Summary of best systems where body part terms are weighted differently . . . . . 58
4.40 Performance of linear interpolation of various systems. . . . . 59
4.41 Summary of best linearly interpolated systems. . . . . 60
4.42 Parameters for the best sytems for linear interpolation. . . . . 60
4.43 Summary of performance of selected systems on training set. . . . 61

5.1 Performance of selected systems on test set `test_14`. . . . . 63
5.2 Performance of selected systems on test set `test_15`. . . . . 65

# List of Figures


1.1 Example of relations in the MRREL file. . . . . 7
1.2 "Biologic Function" Hierarchy . . . . . 9
1.3 affects Hierarchy . . . . . 10
1.4 A part of relations between semantic types in the Semantic Network 10
1.5 Illustration of the application of Rocchio's algoritm for relevance feedback . . . . . 13

2.1 A snippet of a raw document file. . . . . 20
2.2 Example of a discharge summary. . . . . 21
2.3 Example of a query for 2014 shared task. . . . . 21
2.4 Example of a query of 2015 shared task. . . . . 22
2.5 Eexample of an annotated document. . . . . 23

3.1 Overview of indexing process in Terrier. . . . . 24
3.2 Example of a query file in TREC format. . . . . 26
3.3 Example of a query file in Single Line format. . . . . 26
3.4 Overview of Retrieval process in Terrier. . . . . 27
3.5 Example of entries in the MRCONSO file. . . . . 28
3.6 Example of entries in concept_dict. . . . . 28
3.7 Example of queries expanded by terms with the highest idf score. 29
3.8 Example of queries expanded by preferred terms. . . . . 30
3.9 Example of queries expanded by related concepts. . . . . 32
3.10 Example of queries expanded by blind relevance feedback. . . . . 33
3.11 Example of weighted queries. . . . . 34
3.12 The "Anatomical Structure" hierarchy . . . . . 35
3.13 An example of queries with term weighted based on their semantic type. . . . . 36

# List of Abbreviations

| | |
|---|---|
| CLEF | Conference and Lab of the Evaluation Forum |
| CUI | Concept Unique ID |
| DFR | Divergence From Randomness |
| df | document frequency |
| idf | inverse document frequency |
| IR | Information Retrieval |
| LGD | Log-logistic DFR |
| MAP | Mean Average Precision |
| NDCG | Normalized Discounted Cumulative Gain |
| P@5 | Precision at 5 |
| P@10 | Precision at 10 |
| PL2F | Per-Field Normalization model |
| tf | term frequency |
| TREC | Text REtrieval Conference |
| UMLS | Unified Medical Language System |
| UNJ@10 | Unjudged documents at 10 |