University of Sheffield

William Briggs 12 Month Report

William Briggs

Supervisor: Dr Mark Stevenson

Panel: Professor Lucia Specia, Professor Richard Clayton

Department of Computer Science

October 14, 2018

Abstract

Medical data comes in large volumes, making it a challenge for systematic reviewers to process and find relevant information. Being able to apply automatic techniques to this field presents itself as a suitable application for natural language processing. As part of this report two areas were examined as potential candidates for future work. We critically evaluated existing methods to finding stopping points in ranked document collections as well as proposing some new methods. As part of the CLEF 2018 task for technology assisted reviews we built an IR system that uses limited information for retrieving relevant documents for a systematic review.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The problem of handling medical literature poses an interesting challenges for Natural Language Processing (NLP) researchers. The sheer volume of medical data makes it difficult for humans to process efficiently.

Evidence-based medicine has become an important aspect of health care and policy making. One key task is the creation of systematic reviews. Systematic reviews are transparent reviews that aim to pull together and critically analyse and summarise relevant literature to a topical question [8]. The process of creating a systematic review is rigorous and time consuming with varying degrees of complexity in-between steps [21]. This report will look at the existing work done using NLP as part of the systematic review process as well as the novel work done by myself so far.

We first review the stages involved in creating a systematic review 2.1. We break the steps down by looking at the PICO strategy [20]. By breaking the steps down, it becomes easier to examine potential candidates for applying NLP techniques to the process. Areas for research are then identified.

We then move on to look at stopping methods for systematic reviews. 2.3 Stopping methods are about finding a suitable stopping point given a list of ranked documents. Two existing stopping methods are examined; the target method and the knee method.

In the next section we identify some relevant research questions within the field 3. Three areas are focused on: Automated Decision Making of Relevant Studies 3.1, Information Extraction From Studies 3.2 and Stopping Methods

The work completed so far is then presented 4. We look at using curves to make predictions of finding a stopping point, including using a Gaussian process. We then present our work for the automatic query generation process by using systematic review protocols as a basis for inferring the query.

Finally we look at future work 5.

Chapter 2

Literature Survey

Systematic reviews have many different stages that propose themselves as a candidate for automation. This section is going to look at the techniques that have been applied for some of these stages in previous literature.

2.1 Steps of a Systematic Review

It is useful for us to break down the steps involved in creating a systematic review into subtasks. This way we can observe what techniques can be applied during the relevant sub tasks to improve the efficiency of the process. The following definitions are derived task simplifactions from the cochrane tutorial on systematic reviews: [19].

1. Question definition.
2. Relevant literature search.
3. Data Filtering.
4. Data Extraction.
5. Analysis and Data combination.

2.1.1 Question Definition

One of the best known techniques for formulating a systematic review question is known as the PICO strategy [20]. This technique focuses on exposing 4 pieces of information in the systematic review question: patient population, intervention or exposure, comparison or control and outcome.

Example: (credit goes to [20])

"Is animal-assisted therapy more effective than music therapy in managing aggressive behaviour in elderly people with dementia?"

P	elderly patients with dementia
I	animal-assisted therapy
C	music therapy
O	aggressive behaviour

A potential point of interest would be attempting to generate these questions automatically given some literature context.

2.1.2 Relevant literature search

After formulating a question, systematic reviews need to search for the relevant literature that surrounds this question.

Large medical database-such as Pubmed [1] contain relevant studies that can be used to create the review. These databases are typically very large and require concise queries to efficiently retrieve data.

Naturally this can be modelled as an information retrieval problem. We have a large number of documents and we wish to retrieve the most relevant ones. One task for the 2017 CLEF conference was to produce a ranking of the most relevant documents for topics [13]. Many techniques have been proposed for ranking of relevant documents, with varying degrees of performance [1] [15].

An important aspect of the relevant literature search step is the construction of the query. Query creators often apply filters (also known as hedges) to increase the effectiveness or/and the efficiency of the searching. Two key attributes for the query are the precision and the recall. By including synonymous phrases e.g: quality adjusted life or quality of well-being or disability adjusted life the sensitivity can be increased, but as expense of the precision. The creation of this query is a task that could potentially have some aspects of NLP applied to it.

2.1.3 Data Filtering

The data filter stage involves reducing the amount of documents returned by the initial query down to a smaller subset of relevant document. This is can also be referred to as the abstract screening phrase [13].

The length of this stage is highly dependant on how many documents were returned by the initial query, often in the excess of 5000 studies for a single query. In response to this, stopping criteria methods have been proposed that aim to optimize two key parameters; the

[1]https://www.ncbi.nlm.nih.gov/pubmed/

effort and the recall. That is to say we want to get as many relevant documents as possible, whilst looking at the fewest. Examples of approaches include the knee method [24] and the target method [5]. Other techniques could be applied and evaluated such as curve fitting, which is later examined in 2.3

2.1.4 Data Extraction

The data extraction phase involves pulling the relevant information from the filtered subset of studies. Examples of important information includes how many people took part in the study and what the results were.

Being able to extract the relevant information from studies presents itself as an information extraction problem. The task to automate the process of extracting relevant information would reduce time and complexities of manually reviewing studies [11].

2.2 Indexing and Querying Medline and Automated Query Generation

Medline is a large collection of medical literature and data from around the world. Typically each entity will contain a title and an abstract containing some information on the study. Whilest Medline as a whole is very easy to access [16], the large size and complexity of the data makes it difficult to retrieve the relevant information.

2.2.1 Automated Query Generation

Being able to automate query generation for literature searching would save systematic reviewers a significant amount of time. However, medical literature queries are typically complex and contain multiple levels of logical operators and synonymous term look ups. This makes the task of creating a query manually is a challenging task.

Rapid Automatic Keyword Extraction Algorithm

Rapid automatic keyword extraction (RAKE) is a keyword extraction algorithm [23]. This algorithm is used for taking the key pieces of information from text and is useful the domain of information extraction. This algorithm is of interest as it has potential usage within the field of query generation.

RAKE heavily relies on stop-words and punctuation separators as an indicator for the importance of a phrases and words. RAKE will iterate over sequences of words until a stop-word or separator is found, this phrase/word is then split and extracted. Frequency of occurrence (tf) and word co-occurrence matrices can then be used to reduce the key-word set down further.

RAKE can be further optimized by specifying minimum term frequency rates to capture more prominent terms.

2.3 Stopping Criteria

Stopping criteria is about finding the optimum point to stop reviewing a set of documents. This is important in a decision making process for maximizing efficiency. Consider having 100 relevant documents, where each document contains a binary value. If we looked at 1/3 of these documents and saw a trend of positive values, we could use this to infer the reliability of the remaining documents.

Two key methods have been proposed for finding stopping points so far, the target method [24] and the knee method. [5]. Both these methods are discussed below 2.3.3

2.3.1 Evaluation Metrics for Finding Stopping Points

In order to evaluate the suitability of stopping methods, we will use two evaluation metrics. The recall, which is the number of documents returned for a topic, and effort which is the number of documents that had to be examined.

$$Recall = \frac{|R|}{|D|} \tag{2.1}$$

Where R is the set of relevant documents found and D is the set of all relevant documents.

Similarly, effort is computed as:

$$Effort = \frac{|L|}{|D|} \tag{2.2}$$

Where L is the set of documents that were examined.

Naturally we could exclusively optimized each of the parameters by either returning everything in the document collection ($R = |D|$) or by just looking at a single document. ($L = 1$)

Therefore it becomes difficult for us to evaluate our stopping criteria as we need to consider both of these parameters adjacently.

In response to this we can make use of two more evaluation metrics that were proposed by Cormack and Grossman [5]:

$$reliability = P[acceptable(S) == 1] \tag{2.3}$$

reliability is computed over all searches and is read as the probability of the acceptability being 1. Where acceptability is calculated as:

$$acceptability(S) = \begin{cases} 1, & recall(S) >= 0.7. \\ 0, & recall(S) < 0.7. \end{cases} \tag{2.4}$$

A stopping point is deemed to be acceptable if 70% of the relevant documents have been found. As such, the reliability is an average over a search method. In Cormack and Grossman [5] a target is set of achieving 95% reliability.

2.3.2 Formulating Stopping Problem

In some cases, reviewers might be content in missing a proportion of relevant documents if a certain level of recall could still be guaranteed. This problem can be defined as a multi-objective optimization problem. We want to simultaneously optimize two competing objectives: maximization of recall recall and minimization of effort.

We can use linear scalarization [18] as a simple way of optimizing both of these objectives. We define our problem as follows

$$\theta_0 f_0(x) - \theta_1 f_1(x) \rightarrow max \tag{2.5}$$

Where $f_0(x)$ is the recall function, θ_0 is a weight to associated the importance of the recall and $f_1(x)$ is the effort function, θ_1 is a weight to associated the importance of the effort. We use a minus symbol to indicate minimizing the effort function.

Another way of optimizing is to use evolutionary algorithms. Evolutionary algorithms have been show to perform better in multi-objective optimization [3] as they are able to simultaneously with a set of possible solutions. Multi-Objective Genetic Algorithm (MOGA) is an evolutionary algorithm that which performs Pareto optimization [17].

MOGA works by combining a weighted sum of multiple scalarization functions into a single scalar fitness function. Randomization is used to select different optimization directions.

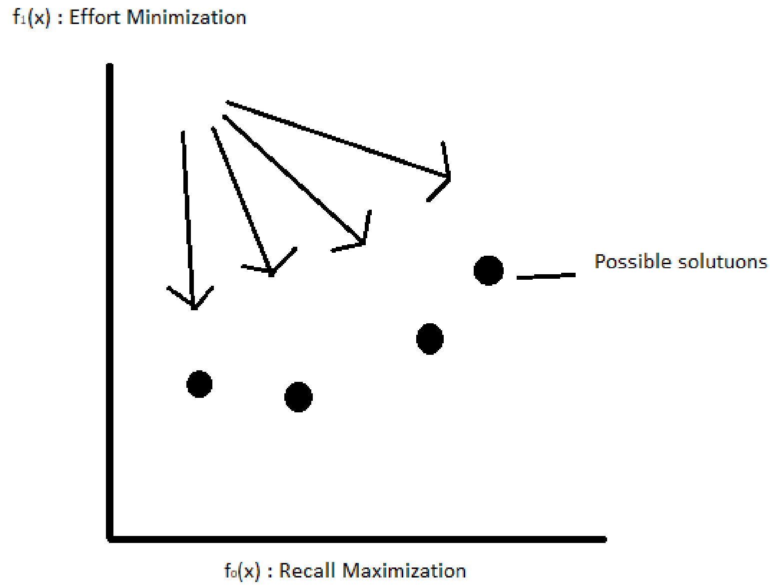

Figure 2.1: MOGA algorithm using various search directions. Image reproduced from [17]

2.3.3 Existing Stopping Methods

As discussed in 2.3, existing methods for finding stopping points in ranked documents have been proposed.

Target

The target method is an approach that can guarantee a certain a certain level of reliability 2.3.1. It was first proposed by Cormack and Grossman [5]. This method randomly samples returned documents until a target number T of relevant document are found.

The target T denotes how many documents we should randomly select from our initial query. A larger value of T will increase the effort required as we are more likely to select a document towards the end of the query set. Documents are looked at until the target point T has been reached.

We first compute a random target set of relevant documents. We then calculate the last document in the target set and mark that as our target point:

$$\underset{last}{d} = \underset{d \in T}{argmaxrelrank}(d) \tag{2.6}$$

It must hold that d is in the target set. *relrank* determines whether or not a document is relevant.

Figure 2.2: Visualisation of target method last relevant document selection. C is number of documents in collection.

Increasing our target set size is likely to increase the probability the last document being towards the end of the document collection.

We can calculate the recall of the point by looking at the relevance rank of the last document:

$$recall = \frac{relrank(d_{last})}{R} \tag{2.7}$$

Where R is the number of relevant documents.

For our method to be deemed reliable we must achieve 70% recall with a 95% average over all topics.

$$P(\frac{relrank(d_{last})}{R} \geqslant 0.7) \geqslant 0.95 \tag{2.8}$$

Assuming we have a large number of relevant documents R we need to determine cut-off c

$$P(\frac{R - relrank(d_{last})}{R} > c) = 0.05 \tag{2.9}$$

This can be further simplified to:

$$P(R - relrank(d_{last}) > cR) = 0.05 \tag{2.10}$$

Which translates to the probability of the remaining relevant documents being higher than the cut-off point should be 0.05.

For this to hold, cR documents must be absent from T. This occurs with the probability:

$$\left(1 - \frac{10}{R}\right)^{cR} = 0.05 \tag{2.11}$$

Which can become:

$$c = \frac{log(0.05)}{Rlog(1 - \frac{10}{R})} \tag{2.12}$$

In cases where R has more than 10 relevant documents it follows:

$$c < \lim_{R \to \infty} \frac{log(0.05)}{Rlog(1 - \frac{10}{R})} = 0.299573 < 0.3 \tag{2.13}$$

Finally we have:

$$R \leq 10 \cup P(\frac{relrank(d_{last})}{R} \geq 0.7) \geq 0.95 \tag{2.14}$$

Overall, while the target method is shown to acquire 95% reliability, the effort needed is often significantly highly, often requiring us to look at huge volume of documents.

Knee Method

A different stopping kethod proposed by [24] is known as the knee method. This approach uses a curve to generate a 'knee', which is then used for predicting a stopping point. This approach is likely to be highly dependant on the quality of the initial rankings. This is because we need a curve that reaches a peak quickly, before flattening out.

We use a vertical line panning the length of the ranking set and use it to calculate the distance from the ranking at each point. The point with the maximum distance is chosen as a suitable stopping point.

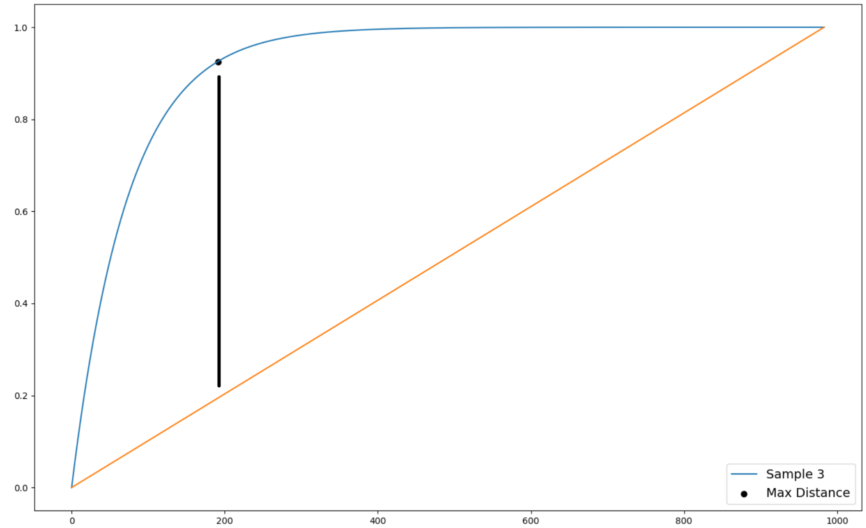

Figure 2.3: Example of using knee method to find a stopping point. Image inspired from [24]

We can see in the above example the method has predicted we look at around 200 of the 1000 documents to achieve a suitable stopping point.

This method also imposes an additional constraint for rankings of a large volume.

It was found that the knee method is a better approach for finding a stopping point than the Target method [5]. The recall was always found to be better and the reliability was found to be the same or higher for 6 out 8 test collections.

2.4 Summary

We first examined the steps of a systematic review 2.1 and looked at potential areas to introduce NLP techniques.

Next we looked at how we can handle large scale databases like Medline 2.2. We also introduced the key-word extraction algorithm RAKE as a candidate for extracting relevant information from medical documents 2.2.1.

We then began to examine stopping criteria 2.3. We looked at the intuition behind looking for stopping points in ranked sets of documents. We defined our stopping problem as a multi-objective optimization problem 2.3.2. We looked into two existing approaches, the target method and the knee method

Chapter 3

Research Questions

In this chapter we will present some possible research areas for systematic reviews.

3.1 Automated Decision Making of Relevant Studies

Once relevant studies have been identified by their abstracts, reviewers are required to process the studies and extract useful information for the systematic review. Many of these studies will not be useful and can be discarded, but not without the cost of the reviewer having to look at the content.

A useful piece of research would be to determine if a study is relevant to the systematic review question.

There are two potential routes that could be taken for the investigating this task. The first approach would be to use existing information from the systematic review (ie the protocol) and determine if the study contains the information. A second approach would be to a semi-supervised learning method, having the reviewer look at a subset of the studies, so we can then build a classifier for the remaining studies. Both of these approaches can be applied to the abstract screening and the data filtering stage of the systematic review. For the abstract screening we would be trying to predict if a study is relevant, by looking at the abstract. For the data filtering stage we would be looking at the actual text of the study (typically a pdf file)

There are many challenges for this research topic. Not all studies are publicly available and are often protected by publishing licences. This makes it difficult to gather data. Another challenge is having to deal with such a broad range of data, as well as the pdf format studies.

3.2 Information Extraction From Studies

Much of the existing work in information extraction for systematic reviews is done using sentance-level information extraction [21] [12]. Often approached using classification methods to determine if a sentence does or does not contain relevant information [10].

3.2.1 PICO Extraction

A method for extracting PICO sentances was proposed [10]. This approach uses a Naive Bayes classifier against 3 of the 4 PICO elements. Training data was acquired using a bootstrapping technique, looking at structured abstracts that are contain key-word headings. The macro-averaged F-Measure was found to be 84%.

Further work [9] by the same author looked at whether the first sentences of in the structured abstracts were enough to train a classifier. This approach looks at 3 of the 4 PICO elements. The averaged macro average F-Measure was found to be 71%.

This existing work assumes extraction of PICO elements at abstract level. No work was found [12] in the extraction of PICO elements from the full texts. One interesting area of research, would be looking at learning a classifier from the abstracts and applying it to the full texts.

3.3 Stopping Methods

We examined existing working on finding on stopping methods in our literature survey 2.3. We can investigate further methods finding stopping points in systematic review rankings.

We can look at applying machine learning algorithms for plotting a regression based system of relevant documents. Techniques that could be interesting to experiment with include a generic curve fitting and a Gaussian Process (GP).

Chapter 4

Novel Work

In this section the work completed so far will be presented. Two main areas of the systematic review process has been focused on: stopping criteria and indexing/querying PubMed.

4.0.1 CLEF 2017 Runs

For the CLEF 2017 Technologically Assisted Reviews in Empirical Medicine [13], participants were expected to submit runs for ranking documents. Participants were given complex boolean queries that could be used for extracting relevant information to rank the documents. These runs were later released for public access [1]

Run	Description	Reference
Sheffield-run-2	Sheffield-2 used tfidf similarity along with standard pre-processing	[1]
Waterloo A-rank-cost	Waterloo used a baseline model implementation from the TREC Total Recall Track	[6]
Waterloo B-rank-cost	-	[6]
auth run-1	AUTH used a learning-to-rank approach and used both batch and active learning	[2]
auth run 2	-	[2]
ntu run-1	Used convolutional neural networks (CNN)	[14]
ucl full-text	Used a deep learning model architecture	[26]

Table 4.1: The 6 runs sampled from the CLEF 2017 task.

We have taken 6 ranking sets of which are of different quality. The Waterloo and AUTH ranks are the best rankings followed by Sheffield. The UCL and NTU submissions feature poorer quality rankings.

CLEF 2017 runs will follow a format similar to the example below. The most important information being the topic id and the document id.

```
CD010775 NF 19307324 1 0.27152011529138564 Test-Data-Sheffield-run-2
```

[1]https://github.com/CLEF-TAR/tar/tree/master/2017-TAR/participant-runs

Results can be evaluate using qrel files, supplied as part of the CLEF 2017 data. These files contain relevant documents for each topic, and are formatted as follows:

```
CD008803 0 21467181 0
CD008803 0 20872357 1
CD008803 0 23837966 0
```

Where the 1 or 0 on the right side indicates if the document is relevant for a study.

4.1 Baseline Approaches to Stopping

We can now establish some baseline approaches for finding stopping points. Approaches are heavily dependant on the initial rankings of the document collection, and naturally assume more relevant documents feature towards the start of the collection.

4.1.1 Oracle Scores

Looking at the oracle scores for each set of rankings will tell us the best we can possibly for do this task. We will assume we are satisfied with 70% recall. These scores are calculated as the minimum amount of effort that could be made to achieve 70% recall. Results are taken as averages over all topics.

Submission	recall	reliability	effort
Test_Data_Sheffield-run-2	0.7	1.0	0.11
Waterloo A-rank-cost	0.7	1.0	0.07
Waterloo B-rank-cost	0.7	1.0	0.06
auth run-1	0.7	1.0	0.08
auth run-2	0.7	1.0	0.09
ntu run-1	0.7	1.0	0.4
ucl full-text	0.7	1.0	0.67

Table 4.2: Lowest effort possible to find 70% of relevant documents.

These scores highlight the importance of the ranking methods for finding a stopping point. The best performer, Waterloo B-rank-cost needs just 6% effort to hit 100% reliability. The worst performer, ucl full-text requires 67% effort for the same level of reliability.

4.1.2 Percentage cut-off method

As a first approach we could simply take a cut of the document collection and evaluate how many relevant documents we have retrieved. This likely to be very dependant on the initial rankings of the document collection.

% of Documents	10%			25%			50%			75%			90%		
Run	Recall	Reliability	Effort	-	-	-	-	-	-	-	-	-	-	-	-
Sheffield-run-2	0.49	0.16	0.10	0.74	0.66	0.25	0.91	0.93	0.50	0.98	1.00	0.75	0.99	1.00	0.90
Waterloo A-rank-cost	0.80	0.6	0.10	0.91	0.93	0.25	0.98	1.00	0.50	0.99	1.00	0.75	0.99	1.00	0.90
Waterloo B-rank-cost	0.73	0.63	0.10	0.90	0.93	0.25	0.98	1.00	0.50	0.99	1.00	0.75	0.99	1.00	0.90
auth run-1	0.72	0.63	0.10	0.90	0.93	0.25	0.97	1.00	0.50	0.99	1.00	0.75	0.99	1.00	0.90
auth run 2	0.68	0.60	0.10	0.88	0.9	0.25	0.97	1.00	0.50	0.99	1.00	0.75	0.99	1.00	0.90
ntu run-1	0.19	0.00	0.10	0.39	0.06	0.25	0.62	0.43	0.50	0.83	0.83	0.75	0.91	0.93	0.90
ucl full-text	0.08	0.00	0.10	0.22	0.00	0.25	0.46	0.03	0.50	0.70	0.70	0.75	0.87	0.93	0.90

Table 4.3: Comparison of results between rankings when looking at a percentage of the ranked documents.

All scores are averaged over the entire ranking set.

Sheffield has an average recall of 0.49 by the time 10% of the documents have been observed. Waterloo has achieved 80% recall by this point. After looking through 25% of the rankings Waterloo and AUTH have achieved around 90% recall of documents.

The limitation of this approach is the effort required looking through documents is still high. We want to lower this effort as much as possible, whilst only having to observe relevant documents.

4.1.3 Similarity score cut-off method

A similarity score method will assume each document in the rankings has a score associated with it. Consider the ranking format described in 4.0.1. The 5th column describes a similarity between the document and the query. Similarity scores gradually decline as we descend down the rankings. For this experiment we only use the Sheffield rankings.

The similarity score can be used derive a stopping point. This method works by looking at documents D_i and D_{i+1} and determining if the difference between the similarity scores has become too large. This method will work on the basis that documents that are no longer relevant will have a sudden drop in score such that we can identify this as our stopping point.

$$Difference(D_i, D_{i+1}) > C \tag{4.1}$$

Where difference returns a score of how close document D_i and D_{i+1} are together and C is a cut-off constant. We can expand this to an example:

$$(1 - (0.73/0.75)) * 100 > 0.015 \tag{4.2}$$

Here we are saying if the two documents' scores are above 1.5% then we should stop looking down the rankings.

$dif(D_i, D_{i+1})$	recall	reliability	effort
0.01%	0.025	0.000	0.0023
0.05%	0.120	0.100	0.100
1%	0.359	0.333	0.333
2%	0.880	0.860	0.860
5%	1.000	1.000	1.0000

Table 4.4: Similarity cut-off comparison for stopping for Test_Data_Sheffield-run-2. Using cosine similarity scores.

These results are highly sensitive to the similarity score and show it's difficult to use this score as an affective measure for stopping. We found similarity scores rarely have sudden drops in values, making it difficult to use this method to identify a stopping point.

As an alternative approach, we can look at just the top document D_1, and compare it to the succeeding documents D_{1+i} in the rankings. We can formulate this as follows:

$$Difference(D_1, D_{1+i}) > C \tag{4.3}$$

$dif(D_1, D_{1+i})$	recall	reliability	effort
10%	0.048	0.000	0.004
20%	0.063	0.000	0.007
30%	0.113	0.000	0.152
40%	0.191	0.030	0.029
50%	0.319	0.060	0.063
60%	0.460	0.200	0.113
70%	0.638	0.433	0.210
80%	0.841	0.800	0.387
90%	0.979	1.000	0.679
100%	1.000	1.000	1.000
85.5%	**0.934**	**1.000**	**0.538**

Table 4.5: Similarity cut-off comparison for stopping for Test_Data_Sheffield-run-2. Using cosine similarity scores. Compares first document to succeeding documents

We found results to be much better when using the first document to look for a stopping point. 85.5% difference in first and subsequent documents was found to be the point for hitting 1.0 reliability. This comes at the expense of making just over 50% effort.

4.2 Sample Method to Stopping

The limitations of the methods presented in 4.1 is that they they still require looking through a large volume of documents.

The approaches in this section will use sampling methods. These approaches assumes we have sensible ranking algorithm for returning documents for a query. They work by generating a sample set, which acts as our model for predicting a stopping point.

The first step is to generate a sample set. We used an interval method for generating our set, i.e select every Nth document. The intuition being that the distribution of relevant document in the sample set, should be similar to that of the real set of relevant documents. This makes it suitable to use as a model.

4.2.1 Curve Fitting

Our first approach is to fit a non-linear curve against a sample set. We used a sample size of 3.

$$F(x) = n - a \exp^{-kx} \tag{4.4}$$

Where a, k and n are learnt weights and x is an associated return rate for a document. We generate the curve using the non-linear least squares algorithm [25].

Curve Predictions

The number of topics was reduced down from 30 to 23 using a cut-off parameter. This reduces some of topics that contain fewer relevant documents and do not generate suitable prediction curves. We used a cut-off parameter of 0.5%. This can be read as the number of relevant documents in a document collection must be atleast 0.5%

We also include a confidence interval evaluation for lower bounded range of a 3σ confidence interval. The key advantages of using a curve as method of evaluating stopping criteria is being able to make use of this confidence interval in a real systematic review. In the context of a systematic reviewers at the data filtering stage, we could specify that the system is 95% certain that 70% of relevant documents have been found. At which point the reviewer can decide if its worth continuing to look at documents.

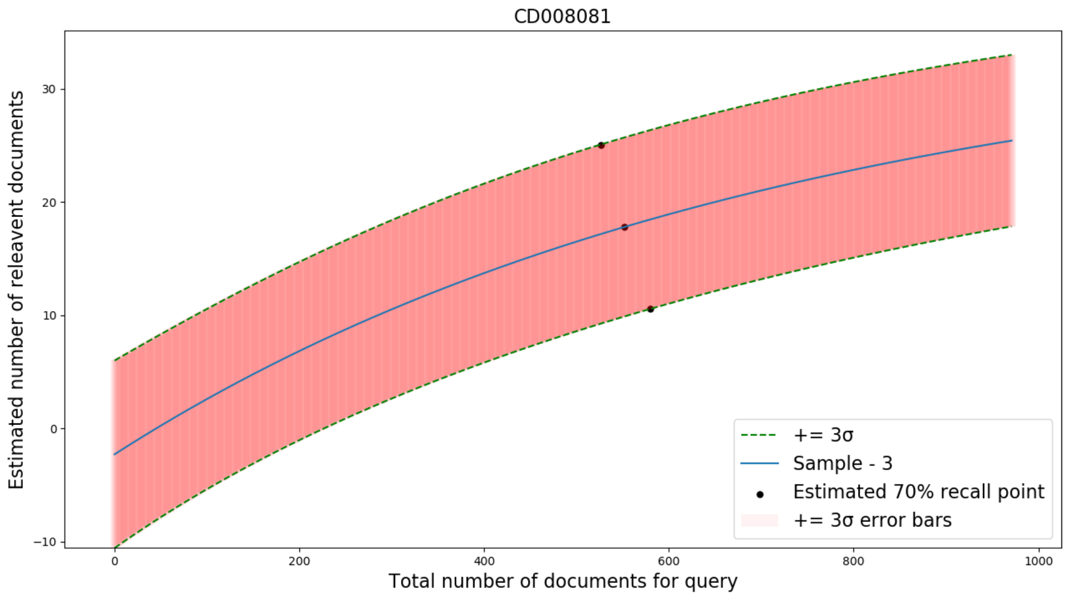

Figure 4.1: Example of a prediction curve for topic CD008081. Confidence bars are included over 3σ. Estimated point of hitting 70% denoted by black point.

Submission	recall-lower	reliability-lower	effort-lower	topics sampled
Test_Data_Sheffield-run-2	0.69 0.74,	0.52 0.60	0.48 0.51	23
Waterloo A-rank-cost	0.71 0.73,	0.47 0.47	0.43 0.44	23
Waterloo B-rank-cost	0.71 0.75,	0.52 0.82	0.41 0.43	23
auth run-1	0.72 0.74,	0.52 0.60	0.41 0.42	23
auth run-2	0.70 0.72,	0.52 0.60	0.42 0.43	23
ntu run-1	0.76 0.74,	0.56 0.52	0.72 0.70	23
ucl full-text	0.86 0.94,	0.82 0.86	0.91 0.95	23

Table 4.6: Evaluation of curve fitting for different CLEF 2017 runs. lower = lower-bound confidence interval. Sample size = 3. Results are taken as averages over all topics for search method. with 0.5% cut-off

We have deliberately compared two of the better participant rankings (Waterloo and auth) and two of the lower performers (ntu and ucl). We can see the quality of the initial rankings significantly influences the performance of our stopping criteria. This suggests there is a important relationship between using a curve to predict a stopping point and how good the initial ranking of documents is.

Some of the ranking methods struggle to produce curves and when combined with a cut-of parameter produce become not worth considering in our evaluation. In this situation we simply returned everything for the given topic, resulting in 100% recall at the expense of 100% effort.

4.2.2 Gaussian Process Fitting

As an alternate approach to fitting a simple curve, we can apply a Gaussian Process (GP) [7]. We will apply a constant kernel plus a squared-exponential kernel. GP was implemented using scikit learn classes. [2]

Submission	recall-lower	reliability-lower	effort-lower	topics sampled
Test_Data_Sheffield-run-2	0.73 0.73,	0.73 0.73	0.50 0.50	23
Waterloo A-rank-cost	0.70 0.70,	0.40 0.40	0.42 0.42	23
Waterloo B-rank-cost	0.73 0.73,	0.71 0.71	0.41 0.41	23
auth run-1	0.74 0.75,	0.56 0.60	0.42 0.42	23
auth run-2	0.72 0.73,	0.52 0.52	0.42 0.42	23
ntu run-1	0.67 0.67,	0.40 0.40	0.64 0.64	23
ucl full-text	0.62 0.62,	0.52 0.56	0.82 0.82	23

Table 4.7: Comparison of different of sample method using gp for different CLEF 2017 runs. lower = lower-bound confidence interval. Sample size = 3. Results are taken as averages over all topics for search method. with 0.5% cut-off

We found the GP is not truly representing the distribution of the data.

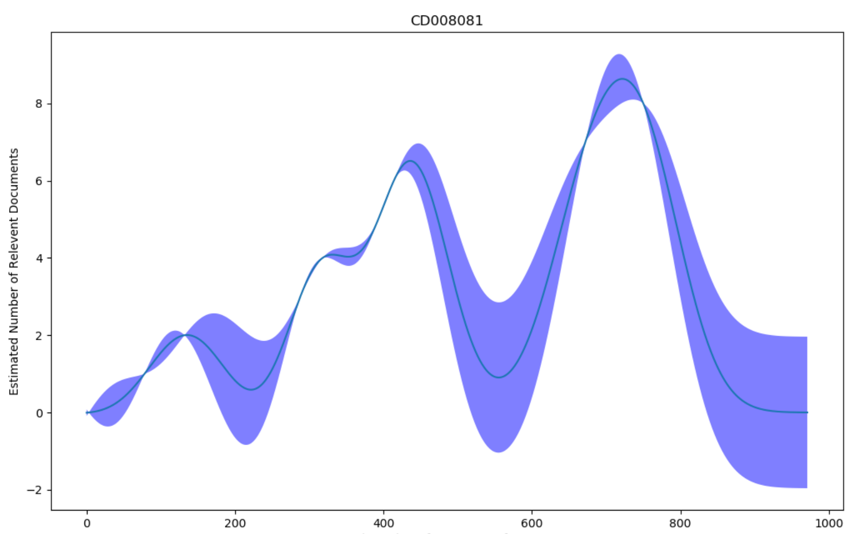

Figure 4.2: Visualisation of using a confidence interval for predicting a stopping point using a gp.

[2] http://scikit-learn.org/stable/modules/gaussian_process.html

4.2.3 Conclusion on Curve fitting and GP

We implemented two methods for predicting stopping points in ranked medical studies. Our first approach used a general curve to estimate the point in which 70% recall is likely to have been hit. Our second method used a Gaussian Process in the same way. We used a sampling method to generate our curves to makes predictions about the remaining studies.

4.2.4 Comparing Methods

Method	Target	recall	reliability	effort
Knee Method	-	0.888	0.866	0.640
Target Method	10	0.952	0.960	0.652
Sheffield-run2-curve	-	0.75	0.640	0.510
Sheffield-run2-cutoff(85.5%)	-	0.934	1.000	0.553

Table 4.8: Comparing target method, knee, cut-off and curve fitting along with confidence interval. Using Sheffield-run-2

We can evaluate how well this method does against an existing set of relevance rankings. We will use the Sheffield run data from the CLEF 2017 task [13]. As the target method allows us to specify our level of reliability, we needed a target T of 10 to hit 95% reliability. We can see that the reliability of our method does not come close to the Target method. Our curve, however has a much lower effort. The cut-off method described in 4.1.2 beats both the curve-fitting and the target methods.

Overall we believe that the Knee and Target methods are too sensitive to the ranking algorithm being used. On our Sheffield-run2 rankings we can see the performance for both these methods is significantly lower than that reported in previous work [5]

4.3 Poisson Process for stopping points

A Poisson Process can be used to model points in time in which events occur. In this situation we wish to observe the rate in which a relevant document occurs in a collection of documents.

We can observe the relevant document distribution for each topic by plotting relevant document occurrences across the whole rankings.

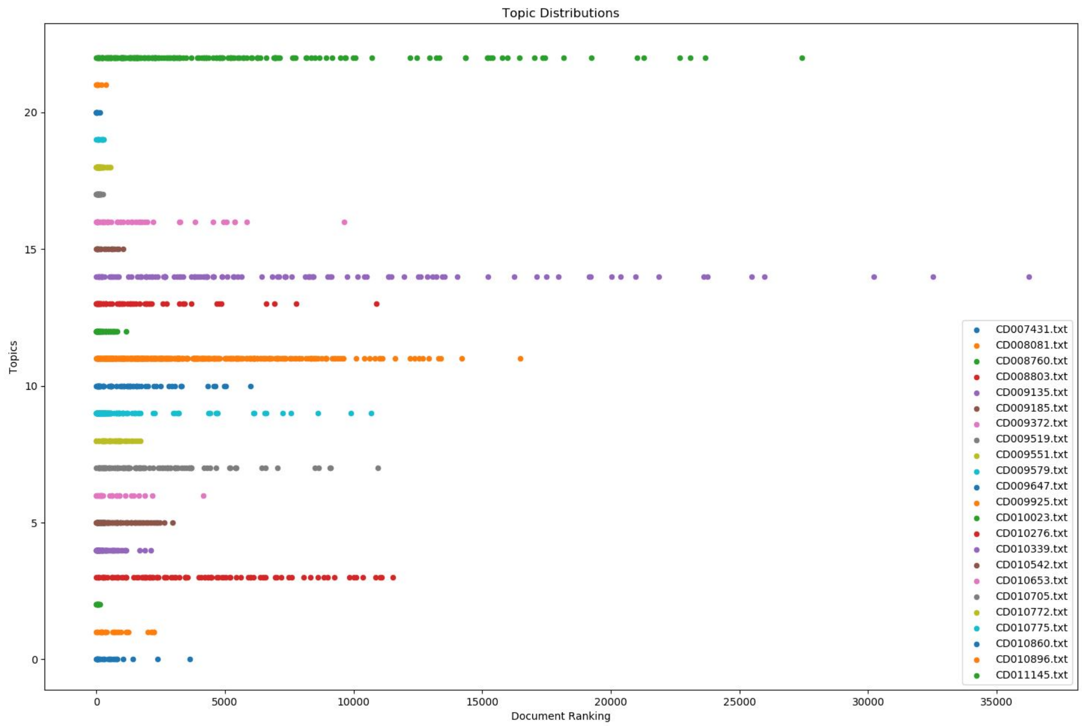

Figure 4.3: Relevant document distribution over Sheffield dataset

The quality of the initial rankings will determine how many relevant documents occur towards the start of the distribution. Naturally the number of relevant documents decrease as we proceed down the rankings.

The Poisson distribution is a way for us to model the occurrences of relevant documents in a fixed time-frame, in our situation, the number of documents returned by the query. To estimate the overall rate of which relevant documents occur, we can observe how many relevant documents occur within a threshold.

$$\lambda = \frac{r_i}{|D|} \tag{4.5}$$

Where r_i is the number of relevant documents in a sample set and $|D|$ is the number of documents in the sample set.

Supposing we sample 10% of the 1000 documents, of which 7 relevant documents occur:

$$\lambda = \frac{7}{100} = 0.07 \tag{4.6}$$

We can use the rate parameter to estimate the probability of there being atleast one relevant documents, after observing n documents:

$$P = 1 - e^{-Rn} \tag{4.7}$$

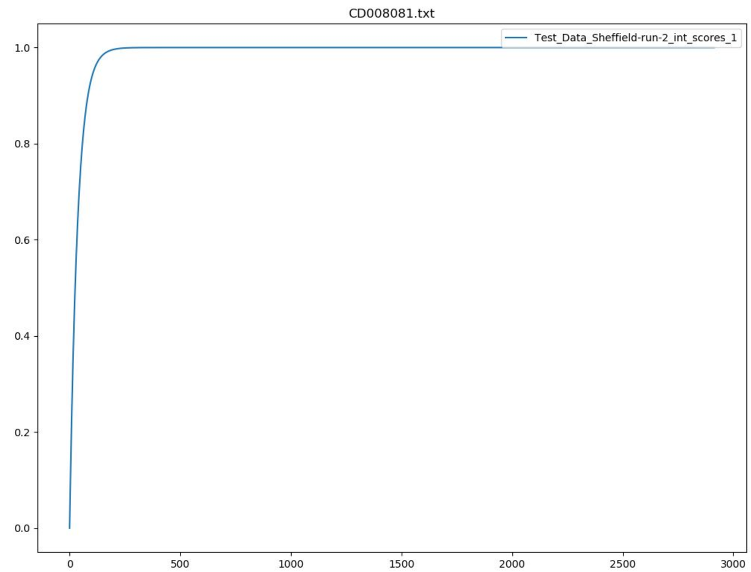

Figure 4.4: Probability of seeing atleast one relevant document by sampling 10% of documents for topic CD008081

The plot shows as we look at more documents, we are increasingly likely to have seen one relevant document. While this is useful to know, we can not use this as a method for predicting a suitable stopping point.

A Homogeneous Poisson Process can be used to model the occurrences of relevant document and then used to predict the probability of there bring r relevant documents after n documents have been observed.

$$P(n = r) = \frac{(\lambda n)^r}{r!} e^{-\lambda n} \tag{4.8}$$

Due to the high likelihood that $r!$ will be exceeding large, we can use a stirling approximation, maintaining a similar value as to what we would have obtained computing the factorial.

$$r \approx \sqrt{2\pi n} \left(\frac{r}{e}\right)^r \tag{4.9}$$

By summing over the probability mass from for values of n between 1 and the size of the document collection we can estimate at what point 95% reliability (stopping point s) is reached.

$$s = \sum_{i \leq 0.95}^{|n|} \frac{(\lambda i)^r}{stirling(r!)} e^{-\lambda i} \tag{4.10}$$

The limitation of using an Homogeneous Poisson Process is that the rate parameter is constant throughout distribution. This means if we looked at 10% of the rankings the rate of relevant documents would be assumed to be constant for the remainder of the document collection.

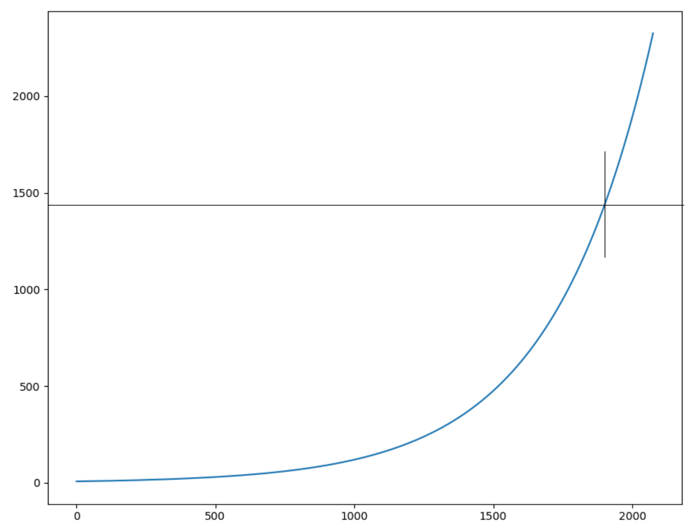

Figure 4.5: Homogeneous Poisson Process overestimating the rate of documents, resulting a prediction curve that would expect us to look at 1500 of the 2000 documents to reach 70% recall

4.3.1 Non-Homogeneous Poisson Process

As the rate parameter is varies throughout the distribution of documents, we can use a Non-Homogeneous Poisson Process. A Non-Homogeneous Poisson process is similar to an ordinary Poisson process, except that the average rate of arrivals is allowed to vary with time.

Window Sampling

It is useful for us to estimate the rate in which relevant documents occur. By iterating over each document in a returned set of documents, and evaluating the relevant documents in a given window, we can estimate this rate parameter.

Document Rank	Relevant	RelScore
1	Y	1/3=0.33
2	Y	3/4=0.75
3	N	4/5=0.80
4	Y	3/5 =0.60
5	Y	2/5=0.40
6	N	2/5=0.40
7	N	1/4=0.25
8	N	0/3=0.00

Table 4.9: Using window sampling with a window size of 2 either side.

By using the number of relevant documents that occur within a given window, we are able to get a declining rate distribution.

Non-Homogeneous Poisson Defination

We can use a non-homogenous poission process to estimate the number of relevant documents in a given interval by integrating the rate function $\lambda(x)$ across the interval.

We first define our intervals as a and b and integrate the values between them with respect to x

$$\int_a^b \lambda(x)d(x) = \Lambda(a,b) \tag{4.11}$$

Therefore the probability of there being r relevant documents in the interval (a, b)

$$P(N(a,b) = r) = \frac{[\Lambda(a,b)]^r}{r!}e^{-\Lambda(a,b)} \tag{4.12}$$

We can make the assumption that the rate in which relevant documents appear is an exponential function.

$$\lambda(x) = ae^{kx} \tag{4.13}$$

Therefore

$$\int_a^b \lambda(x)d(x) = \int_a^b ae^{kx}dx = \frac{a}{k}e^{kx} \tag{4.14}$$

As we are only interested in knowing the total number of relevant documents, we can assume $a = 0$ and $b = n$ where n is the total number of documents.

$$\Lambda(0,n) = \int_0^n ae^{kx}dx = \frac{a}{k}e^{kx} = \frac{a}{k}(e^{kn} - 1) \tag{4.15}$$

So

$$P(N(0,n) = r) = \frac{(\Lambda(0,n))^r}{r!}e^{-\Lambda(0,n)} = \frac{\left(\frac{a}{k}(e^{kn} - 1)\right)^r}{r!}e^{-\left(\frac{a}{k}(e^{kn}-1)\right)} \tag{4.16}$$

Therefore given values for a and k which we can learn from fitting an exponential curve, we can predict the probability of there being r relevant documents within the entire set of n documents

4.4 Indexing and Querying Medline with Limited Information

CLEF 2018 [4] presents an appropriate sub-task for using a limited amount of information to retrieve relevant documents. Normally, reviewers are require to construct complex Boolean queries to retrieve data from Medline. The objective of CLEF 2018 Sub-Task 1: No Boolean Search [4] is to search effectively and efficiently bypassing the construction of the Boolean query.

4.4.1 Acquiring Key Information from A Systematic Review Protocol

A systematic review protocol is created before the systematic review process is started. A systematic review protocol describes the rationale, hypothesis, and planned methods of the review. The Pubmed query is created manually with the help of the protocol. Here we are looking to generate a suitable query/relevant information from the protocol to then automatically query Pubmed.

We used RAKE [23] to extraction key-words from a protocol. The minimum word occurrence count is set to 1, as the protocols are typically small. We used a Pubmed stop list as the phrase splitting parameter. Example shown below:

> Topic: CD008122
> Title: Rapid diagnostic tests for diagnosing uncomplicated P. falciparum malaria in endemic countries
> Objective: To assess the diagnostic accuracy of RDTs for detecting clinical P. falciparum malaria (symptoms suggestive of malaria plus P. falciparum parasitaemia detectable by microscopy) in persons living in malaria endemic areas who present to ambulatory healthcare facilities with symptoms of malaria, and to identify which types and brands of commercial test best detect clinical P. falciparum malaria.

> endemic countries objective|ambulatory healthcare facilities|rapid diagnostic tests|falciparum parasitaemia detectable|malaria endemic areas|diagnostic accuracy|falciparum malaria

The | symbol represents a separation between a phrase. The protcols are pre-processed as follows: Reference removal, lowercase, words less than N length removed, pubmed stoplist. We decided to not perform any stemming/additional manipulation at this stage, due to uncertainty of query format.

The key-word-query receives some final pre-processing prior to being loaded into our information retrieval (IR) system. We used a Lancaster stemmer to reduce words down to a base form. The result is as follows:

> endem country object amb healthc facil rapid diagnost test falcipar parasitaem detect malar endem area diagnost acc falcipar malar

4.4.2 Indexing Pubmed

Pubmed was downloaded from the online resource [3]. We processed the xml files and retrieved the information for each study - title, id, abstract. To reduce the size, we store each record into a local database, containing only the relevant information for each study.

We used Apache Lucene [4] to generate an index for the Pubmed local database. The abstract and title were concatenated together. Pre-processing was done using the same format as the query: Pubmed stoplist [5], Lancaster stemmer and lower-casing.

[3] https://www.ncbi.nlm.nih.gov/home/download/
[4] https://lucene.apache.org/
[5] https://www.ncbi.nlm.nih.gov/books/NBK3827/table/pubmedhelp.T.stopwords/

4.4.3 Runs

- **sheffield-Boolean** The Sheffield Boolean runs uses words that occur the most in the document and the query as a basis for ranking. Documents that contain more query terms will feature higher in the overall rankings. We used the Apache Lucene Boolean similarity class for our implementation.[6]

- **sheffield-tfidf** The Sheffield tfidf run uses a cosine simularity measure to compare the simularity between the query and the pubmed article. Documents and queries are represented as tfidf vectors. We used the Apache Lucene tfidf similarity class for our implementation.[7]

- **sheffield-bm25** This run uses the bm25 similarity measure [22]. We used the Apache Lucene bm25 similarity class for our implementation.[8]

4.4.4 Results

Results were generated using the eval script from the CLEF 2017/2018 task [13]. We calculated the top N results over the CLEF 2017 training set. We include a random baseline to provide a comparison between results.

Run	recall	ap	lastrel	wss100	wss95	normarea	N
Random-baseline	0.005	0.002	126.7	0.00	0.00	0.024	-
Train-Data-Sheffield-bm25-Run1-objective-only	0.538	0.034	3039.051	0.101	0.108	0.431	5000
Train-Data-Sheffield-tfidf-Run1-objective-only	0.354	0.007	2633.718	0.021	0.023	0.247	5000
Train-Data-Sheffield-boolean-Run1-objective-only	0.313	0.034	3039.051	0.101	0.108	0.431	5000
Train-Data-Sheffield-bm25-Run1-objective-only	0.680	0.034	12310.231	0.169	0.172	0.592	25000
Train-Data-Sheffield-tfidf-Run1-objective-only	0.601	0.007	14883.744	0.13	0.136	0.455	25000
Train-Data-Sheffield-boolean-Run1-objective-only	0.471	0.007	12974.205	0.03	0.029	0.381	25000

Table 4.10: Results for IR Pubmed system. Comparison for both 5000 and 25000 thresholds

As we increase the number of documents we return, the recall naturally increases. When we return 25000 documents for each topic, we are able to obtain a total recall rate of over 58%. However, the precision (ap, average precision) is very low, suggesting a significant amount of the documents are not useful. BM25 was found to be the best ranking method, followed by tfidf and boolean.

Improvements could certainly be made to this system:

- MeSH headings would be useful in expanding the range of the query to capture synonymous terms.

[6]https://lucene.apache.org/core/7_0_1/core/org/apache/lucene/search/similarities/BooleanSimilarity.html
[7]https://lucene.apache.org/core/7_0_1/core/org/apache/lucene/search/similarities/TFIDFSimilarity.html
[8]https://lucene.apache.org/core/7_0_1/core/org/apache/lucene/search/similarities/BM25Similarity.html

- Tokenization could be optimized to capture phrases of different sizes.
- Introducing a cost or stopping point to remove the amount of non-relevant documents. We can see for the 25000 documents set of results the last relevant document was around the 20000 point, meaning we could drop the last 5000 from our result set.

4.4.5 Pubmed automatic query Conclusion

We built an IR system using Apache Lucerne and compared three separate ranking methods. We found bm25 ranking gave the best results overall.

We found we were able to achieve fair results with a little optimization techniques to the index and query data.

We compared the performance of our system across different return thresholds, naturally finding as we increase the returned number of documents we get a higher recall. This comes at the expense of reduced precision.

We suggested further improvement to our system, such as including a phrase model for more robust features for both index and query.

Chapter 5

Future Work

The future work for this PhD will look at different techniques to finding stopping points. Specifically we will focus on text processing approaches to stopping.

Any area that could be further expanded on is the use of using the similarity score in the initial rankings as a basis for comparing documents. In novel work section 4.1.3 we examined a technique that uses the top document as the origin point for comparing subsequent documents. To improve the accuracy of the origin point we could try using different points and taking average over a set of points:

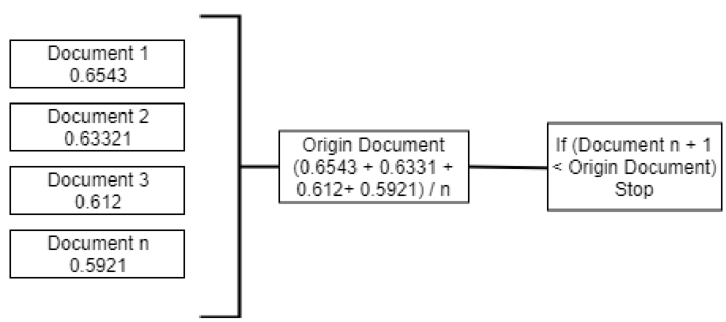

Figure 5.1: Calculating Origin Average

The % difference between document $n + 1$ and the origin document would still be used to derive the stopping point.

This approach has the advantage of taking into consideration variability in the quality of the rankings. It also has the advantage of being entirely unsupervised and does not require us to sample the initial document collection.

A further development of this method is to create an origin document using the document text content itself:

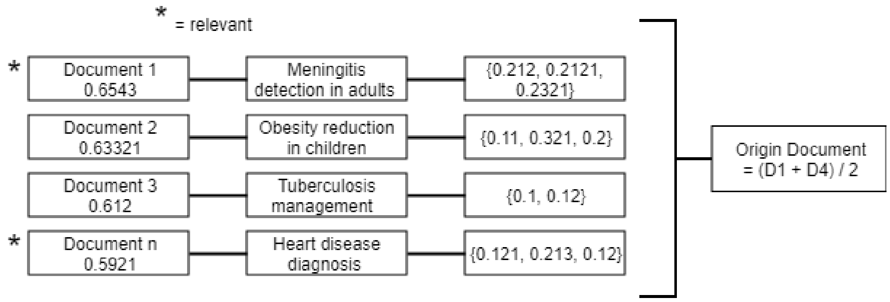

Figure 5.2: Generating Origin vector using document content

Here we have a further development in that we are using the top n documents to calculate an average document vector by using the content of the document. We can then use vector similarity comparisons such as euclidean distance and cosine similarity to compare the origin document vector to further documents down the rankings.

5.1 Gnatt Chart

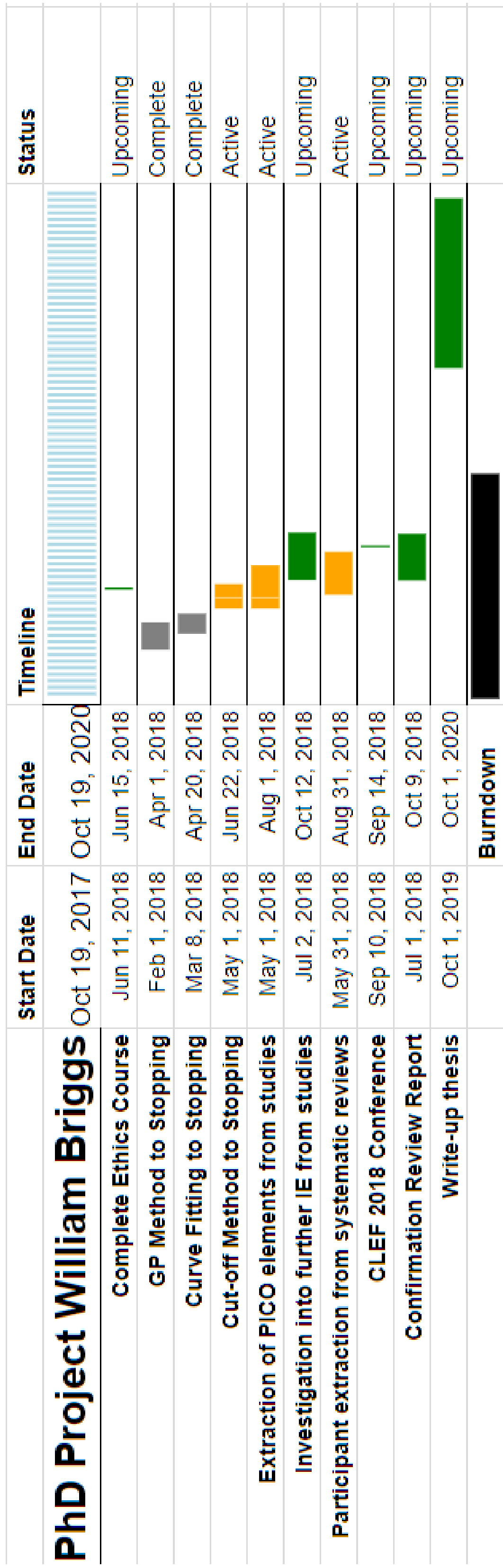

Figure 5.3: Gnatt Chart

Chapter 6

DDP

- Attended Healtex - UK HEALTHCARE TEXT ANALYTICS CONFERENCE
- Lab demonstrator for module COM4519 Cloud Computing
- Undertook module HAR6169 Study Design and Systematic Review Methods
- Marked assignments for COM3110 Text Processing
- Enrolled on FCE6100 Professional Behaviour and Ethical Conduct
- Completed TRAINING NEEDS ANALYSIS (TNA) form.
- Used Learning Management System (LMS) to attend 6 teacher training courses.
- Gave introduction talk to NLP group.
- Contributed to 2018 CLEF lab.
- Became member of Text Processing for Health Technology Assessment (TePHTA).
- Published paper for 2018 CLEF conference.

Bibliography

[1] Alharbi, A., and Stevenson, M. Ranking abstracts to identify relevant evidence for systematic reviews: The university of sheffield's approach to clef ehealth 2017 task 2. *CLEF 2017* (2017).

[2] Anagnostou, A., Lagopoulos, A., Tsoumakas, G., and Vlahavas, I. A cost-effective hybrid ltr approach for document ranking. *Working Notes of CLEF* (2017).

[3] Chiandussi, G., Codegone, M., Ferrero, S., and Varesio, F. E. Comparison of multi-objective optimization methodologies for engineering applications. *Comput. Math. Appl. 63*, 5 (Mar. 2012), 912–942.

[4] CLEF. Clef. https://sites.google.com/view/clef-ehealth-2018/task-2- technologically assisted-reviews-in-empirical-medicine.

[5] Cormack, G. V., and Grossman, M. R. Engineering quality and reliability in technology-assisted review. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval* (2016), ACM, pp. 75–84.

[6] Cormack, G. V., and Grossman, M. R. Technology-assisted review in empirical medicine: Waterloo participation in clef ehealth 2017. *Working Notes of CLEF* (2017), 11–14.

[7] Ebden, M., et al. Gaussian processes for regression: A quick introduction. *The Website of Robotics Research Group in Department on Engineering Science, University of Oxford* (2008).

[8] Gough, D., Oliver, S., and Thomas, J. *An Introduction to Systematic Reviews.* Sage, London, 2012.

[9] Huang, K.-C., Chiang, I.-J., Xiao, F., Liao, C.-C., Liu, C. C.-H., and Wong, J.-M. Pico element detection in medical text without metadata: Are first sentences enough? *Journal of Biomedical Informatics 46*, 5 (2013), 940 – 946.

[10] Huang, K.-C., Liu, C., Yang, S.-S., Xiao, F., Wong, J.-M., Liao, C.-C., and Chiang, I.-J. Classification of pico elements by text features systematically extracted from pubmed abstracts. 279–283.

[11] Jonnalagadda, S. R., Goyal, P., and Huffman, M. D. Automating data extraction in systematic reviews: a systematic review. *Systematic reviews 4*, 1 (2015), 78.

[12] Jonnalagadda, S. R., Goyal, P., and Huffman, M. D. Automating data extraction in systematic reviews: a systematic review. *Systematic Reviews 4*, 1 (Jun 2015), 78.

[13] Kanoulas, E., Li1, D., Azzopardi, L., and Spijker, R. Clef 2017 technologically assisted reviews in empirical medicine overview.

[14] Lee, G. Medical document classification for systematic reviews using convolutional neural networks. *Working Notes of CLEF* (2017), 11–14.

[15] Lee, G. E. A study of convolutional neural networks for clinical document classification in systematic reviews: Sysreview at clef ehealth 2017. *CLEF 2017* (2017).

[16] medline. medline. https://www.nlm.nih.gov/bsd/pmresources.html.

[17] Murata, T., and Ishibuchi, H. Moga: multi-objective genetic algorithms. In *Proceedings of 1995 IEEE International Conference on Evolutionary Computation* (Nov 1995), vol. 1, pp. 289–.

[18] Noghin, V. D. Linear scalarization in multi-criterion optimization. *Scientific and Technical Information Processing 42*, 6 (Dec 2015), 463–469.

[19] Nunn, J. cochranes. `http://cccrg.cochrane.org/animated-storyboard-what-are-systematic-reviews`.

[20] of Tasmania, U. pico. `https://utas.libguides.com/SystematicReviews/FormulateQuestion`.

[21] O'Mara-Eves, A., Thomas, J., McNaught, J., Miwa, M., and Ananiadou, S. Using text mining for study identification in systematic reviews: a systematic review of current approaches. *Systematic Reviews 4*, 1 (Jan 2015), 5.

[22] Robertson, S., Walker, S., Jones, S., Hancock-Beaulieu, M., and Gatford, M. Okapi at trec-3. pp. 109–126.

[23] Rose, S., Engel, D., Cramer, N., and Cowley, W. Automatic keyword extraction from individual documents. *Text Mining: Applications and Theory* (2010), 1–20.

[24] Satopaa, V., Albrecht, J., Irwin, D., and Raghavan, B. Finding a "kneedle" in a haystack: Detecting knee points in system behavior.

[25] scipy. leastsqures. `https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.least_squares.html#scipy.optimize.least_squares`.

[26] Singh, G., Marshall, I., Thomas, J., and Wallace, B. Identifying diagnostic test accuracy publications using a deep model. *Working Notes of CLEF* (2017), 11–14.