University of Sheffield

Text Processing in Systematic Reviews for Cost Efficiency

William Briggs

Supervisor: Dr Mark Stevenson

Panel: Professor Lucia Specia, Professor Richard Clayton

Department of Computer Science

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. This report focuses on cost efficiency for systematic reviews by looking at how we can reduce the amount of data that needs to be looked at to make a decision. Existing approaches are examined and evaluated using a broad range of datasets. We developed and laid the foundations for new approaches within the field. We first looked at percentage cut-off methods by examining the similarity between documents in a set of rankings. We also looked at sampling methods by observing the general distribution of a dataset, building a model, and then making a prediction for the remainder of the data.

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 [9]. The process of creating a systematic review is rigorous and time consuming with varying degrees of complexity in-between steps [17]. Therefore the challenge of applying NLP to the systematic review process is being able to improve efficiency, but not compromise on rigour and reliability.

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 during the early stages of this PhD.

We first review the stages involved in creating a systematic review 2.1. We break the steps down by looking at the PICO strategy [16], standing for patient population, intervention or exposure, comparison or control and outcome. 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. 2.2 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. We focused the majority of the research on stopping criteria-techniques for maximizing cost effectiveness in a set of rankings. We also looked at information extraction from studies, which has potential use in further developing stopping methods.

The work completed so far is then presented 4. We first look at the oracle (best possible

results) for our dataset. We then move on to presenting a new method that uses the similarity between the documents in the rankings to find a stopping point. We move on to using sampling methods to infer the general distribution of data and make predictions as the remainder.

Finally we look at future work that will be undertaken for the next 2 years of the PhD. 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: [15].

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 [16]. 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 [16])

"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 selected for inclusion in a review. These databases are typically very large and require queries to retrieve data. These queries are structured, but not considered precise, as there is a strong emphasis on maximizing recall at the expense of precision.

Naturally this can be modelled as an information retrieval problem. We have a large number of documents and we wish to retrieve all of the relevant ones. Reviewers will aim to retrieve all of the relavent studies, this is because there is a emphasis achieving a binary result set of relevant studies, as opposed to rankings studies by relevance.

An important aspect of the relevant literature search step is the construction of the query. Query creators often apply filters (also known as hedges) [2] 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 recall can be increased, but at 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 done by idenfiying which retrieved studies match the inclusion criteria as defined by the PICO question. This is can also be referred to as the abstract screening phrase [11].

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

[2] `https://hiru.mcmaster.ca/hiru/HIRU_Hedges_home.aspx`

The length of this stage is highly dependent 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 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 [20] and the target method [7]. Other techniques could be applied and evaluated such as curve fitting, which is later examined in 2.2

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 [10].

2.2 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 indicator of being relevant or non-relevant. 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 [20] and the knee method. [7]. Both these methods are discussed below 2.2.3

2.2.1 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. The challenge of finding a stopping point is optimizing both of these parameters. We could look at everything in our rankings to obtain a perfect recall score, but at the consequence of making 100% effort.

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

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 [7]:

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

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 acceptabl*e if 70% of the relevant documents have been found [7]. As such, the reliability is an average over a search method. In Cormack and Grossman [7] a target is set of achieving 95% reliability. Setting a acceptability score above 70% will increase the amount of effort needed to reach the reliability threshold. In practise 70% recall is not considered a sufficient amount of documents to retrieve, but as existing work in this area uses this figure, it is easier for us to compare results.

2.2.2 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 [14] 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 [5] as they are able to simultaneously update parameters with a set of possible solutions. Multi-Objective Genetic Algorithm (MOGA) is an evolutionary algorithm that which performs Pareto optimization [13].

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.

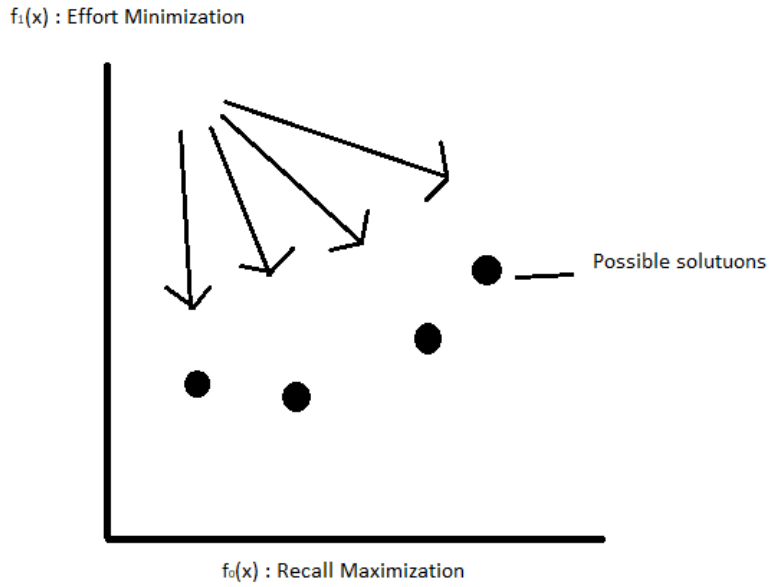

Figure 2.1: MOGA algorithm using various search directions. Image reproduced from [13]

2.2.3 Existing Stopping Methods

As discussed in 2.2, 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 2.2.1. It was first proposed by Cormack and Grossman [7]. 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:

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

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:

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

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.

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

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

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

This can be further simplified to:

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

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:

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

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

Knee Method

A different stopping kethod proposed by [20] 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.

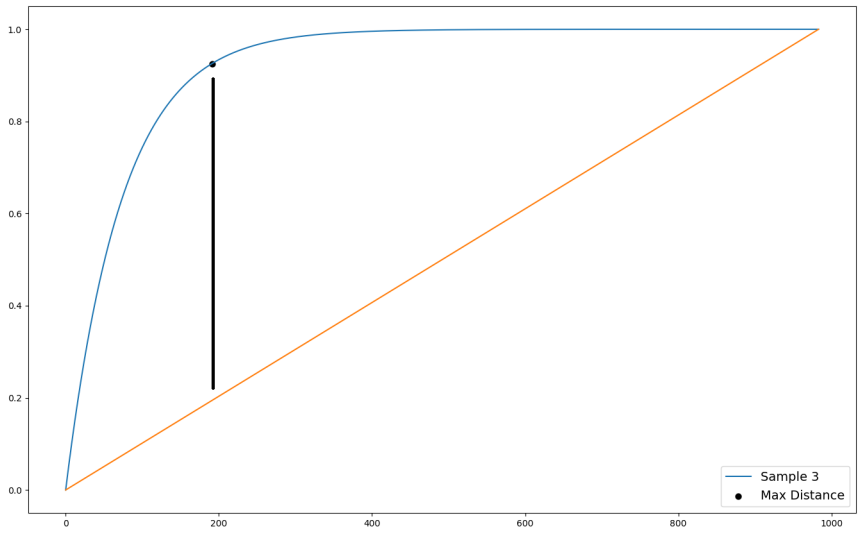

Figure 2.3: Example of using knee method to find a stopping point. Image inspired from [20]

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 [7]. 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.

2.3 Summary

We first examined the steps of a systematic review 2.1 and looked at potential areas to introduce NLP techniques. We broke down the stages of a systematic review such that we can examine what techniques we could apply at each stage.

We then began to examine stopping criteria 2.2. 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 2.2.2. We looked into two existing approaches, the target method and the knee method.

Chapter 3

Research Questions

In this chapter we will present research questions for finding stopping points in systematic reviews.

1 - Development of stopping methods that can be applied to a wider range of document rankings

The target and knee methods discussed previously 2.2.3 2.2.3 are assumed to have weights that are sensitive to the ranking set. We want to apply these methods to our own ranking set using the same weight that was established for these methods.

We then want to look at methods that could be applied to a range of different ranking sets and still perform with the same level of efficiency.

2 - Establishing unsupervised approaches to stopping

As previous methods have a reliance on a weighting or some type of prior assumptions about the ranking collection they fall under supervised approaches to stopping.

Potentially we can use the ranking set itself to develop stopping method that use the information given to us from the ranking algorithm (e.g cosine similarity score). We can then descend down the rankings using the score to establish a stopping point.

Chapter 4

Novel Work

In this section the work completed so far will be presented.

We first look at the CLEF 2017 runs, these are the datasets we will use for applying and evaluating our stopping methods. We establish a baseline result (Oracle); this tells use the best performance we could possibly get for each run. We then move on to evaluate two new methods that we developed to stopping, a percentage cut-off and a similarity score cut-off approach.

Further work looks at a different approach to stopping using partition sampling. We use a portion of the document collection to observe the distribution of the data and then predict the remaining portion. We first apply a curve to our data and then move on to use a Poisson Process.

Finally we look at an idea for building upon the dataset we are using by extracting more information from a study's full text. We present some initial findings as to how possible it is to retrieve these texts through automated techniques.

4.1 CLEF 2017 Runs

For the CLEF 2017 Technologically Assisted Reviews in Empirical Medicine [11], 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]

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

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

Run	Description	Reference
Sheffield-run-2	Sheffield-2 used tfidf similarity along with standard pre-processing	[2]
Waterloo A-rank-cost	Waterloo used a baseline model implementation from the TREC Total Recall Track	[8]
Waterloo B-rank-cost	-	[8]
auth run-1	AUTH used a learning-to-rank approach and used both batch and active learning	[3]
auth run 2	-	[3]
ntu run-1	Used convolutional neural networks (CNN)	[12]
ucl full-text	Used a deep learning model architecture	[22]

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

poorer quality rankings. We tried to take runs that had a varying level of quality in the 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
```

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. Therefore the quality of the datasets can be observed by looking at how many of the relevant documents feature towards the top of the rankings.

4.2 Baseline Approaches to Stopping

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

4.2.1 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 used the 70% recall benchmark.

These scores are calculated as the minimum amount of effort that could be made to achieve 70% recall. We first loop over each topic and update a count each time a relevant document is found. We then examine the recall each time the count is updated and determine if the

threshold score has been reached (70%). If 70% has been reached we stop iterating over the rankings and calculate the effort made up to this point.

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.

4.2.2 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.

4.2.3 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 4.1. The 5th column describes a similarity between the document and the query. Similarity scores gradually decline as we descend down the rankings.

For the Sheffield set of rankings, the similarity score comes from the cosine similarity between a vectorized query and document.

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 is difficult to use this score as an effective 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.

4.3 Sample Methods to Stopping

The limitations of the methods presented in 4.2 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. Therefore the quality of these methods will depend on the sparseness of the document rankings, making it challenging to establish a single stand-out method.

The first step for these methods 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 complete set of relevant documents. This makes it suitable to use as a model.

4.3.1 Curve Fitting

Our first approach is to fit a non-linear curve against a sample set. We used a sample size of 3. We opted for an exponential curve as the number of relevant documents found will increase as more documents are looked, but eventually level out as relevant documents become less frequent.

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

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 [21].

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 does not generate suitable prediction curves. We used a cut-off parameter of 0.5%. This can be read as the proportion of relevant documents in a document collection must be atleast 0.5%, which was found to be a suitable amount to give us a good volume of topics.

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.

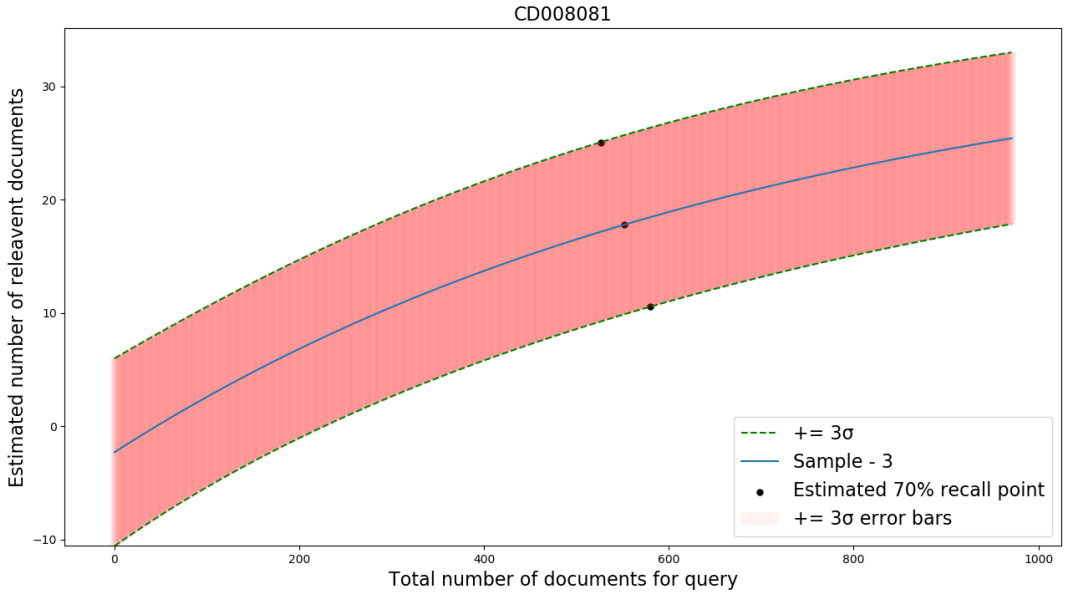

Figure 4.1: Example of a prediction curve for topic CD008081. Confidence bars are included over 3σ. Estimated point of hitting 70% denoted by black point.

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 datasets to produce curves due to the sparsity of relevant documents. In situations where this occurred, we returned everything for the given topic, resulting in 100%

recall at the expense of 100% effort.

4.4 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.

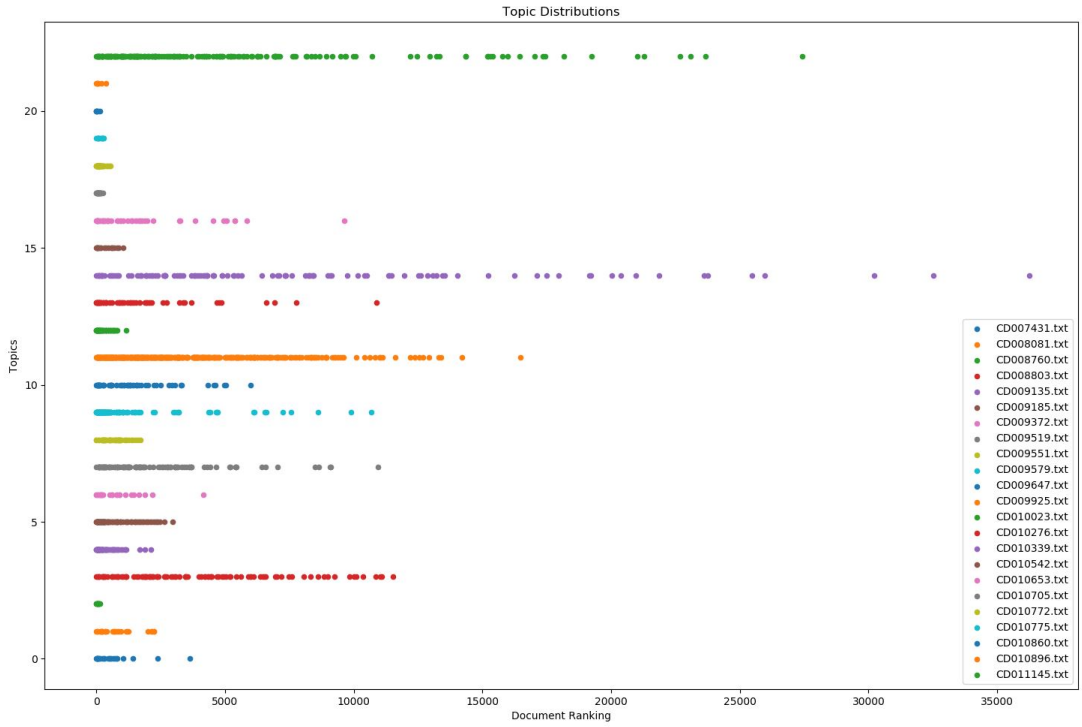

Figure 4.2: Relevant document distribution over Sheffield dataset

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:

$$P = 1 - e^{0.7n} \tag{4.7}$$

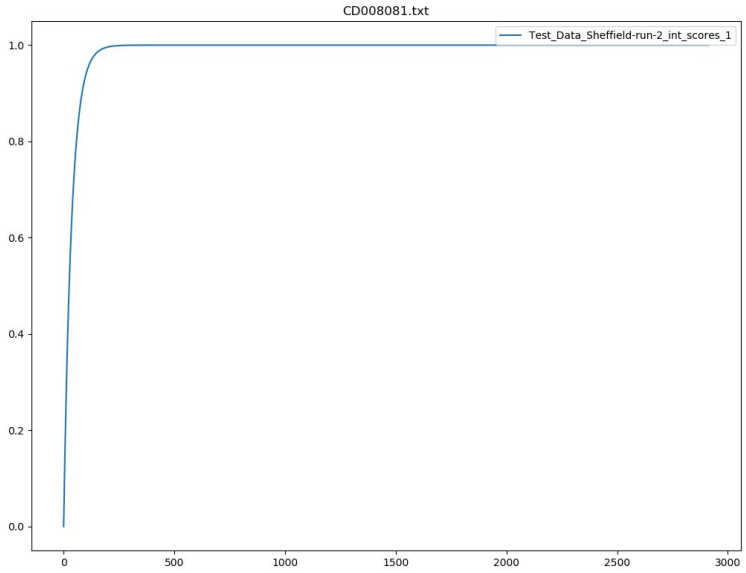

Figure 4.3: Probability of seeing atleast one relevant document by sampling 10% of documents for topic CD008081

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.

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

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.

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

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.

$$s = \sum_{0,i<0.95}^{|n|} \frac{(\lambda i)^r}{stirling(r!)} e^{-\lambda i} \tag{4.10}$$

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. Therefore this method would only be suitable if we had a random relevance rate.

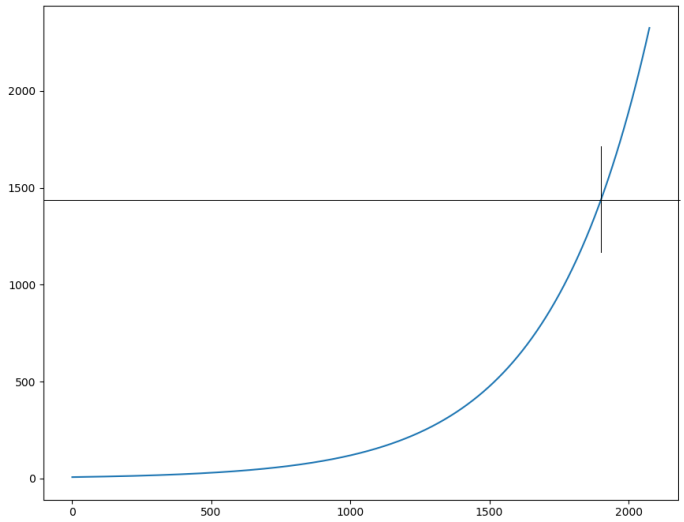

Figure 4.4: Homogeneous Poisson process overestimating the rate of documents, resulting a prediction curve that would expect us to look at 1500 of the 2000 documents to reach 70% recall

4.4.1 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 [1].

Non-Homogeneous Poisson Definition

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

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

As we are only interested in knowing the total number of relevant documents, we assume that we are integrating from 0 to the total number of documents.

$$\Lambda(0,n)=\int_0^n ae^{kx}dx=\left[\frac{a}{k}e^{kx}\right]_0^n=\frac{a}{k}(e^{kn}-1) \tag{4.15}$$

So

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

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.

Implementation Non-Homogeneous Poisson Process

Window Sampling It is useful for us to estimate the rate at 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 4.7: Using window sampling with a window size of 2 either side.

The challenge is being able to find an optimum sample size. A sample size that is too small will not provide enough information on the distribution of the surrounding documents. A sample size that is too large is a risk, as the if the documents rankings are sparse the rate estimations would not be as useful.

Poisson Process Steps We created a Python-based implementation using scipy for the curve modelling. Our steps can be broken down as follows:

1. Load a run final into memory. For this task we used the Sheffield run data from CLEF 2017.
2. Create a frequency based distribution of the data for relevant documents, e.g 1, 1, 2, 3, 3, 3, 3, 4
3. Normalize the data for positions in which relevant documents occur e.g x = 1, 2, 3, 4, 5 y = 2, 12, 17, 34, 61
4. Use normalized data along with window sampling 4.4.1 to create a probabilistic relevant document frequency in a given window.
5. Take a sample percentage from the distribution (start at 1%). Use this percentage to fit an exponential curve (ae^{kx}) using non-linear least squares [2]. This provides values for a and k which can then be substituted into equation 4.16 to give the distribution of the total number of relevant documents.
6. Use weights from curve fit and Non-Homogeneous Poisson process to estimate the number of relevant documents across the entire collection. For values from $r : 0 \mapsto n$ we will get a probability of r relevant documents. Assuming 95% of documents is acceptable, we can sum over the probability mass until this value is reached.

[2] `https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html`

Finding an Optimum Window Size To find a general window size for sampling we will observe the sum of the mean squared error over all topics for sampled data from 10% to 40%. We will consider sample window sizes in the range of 1 to 100. This will tell us how well our exponential curve is fitting to the data for each window sample size.

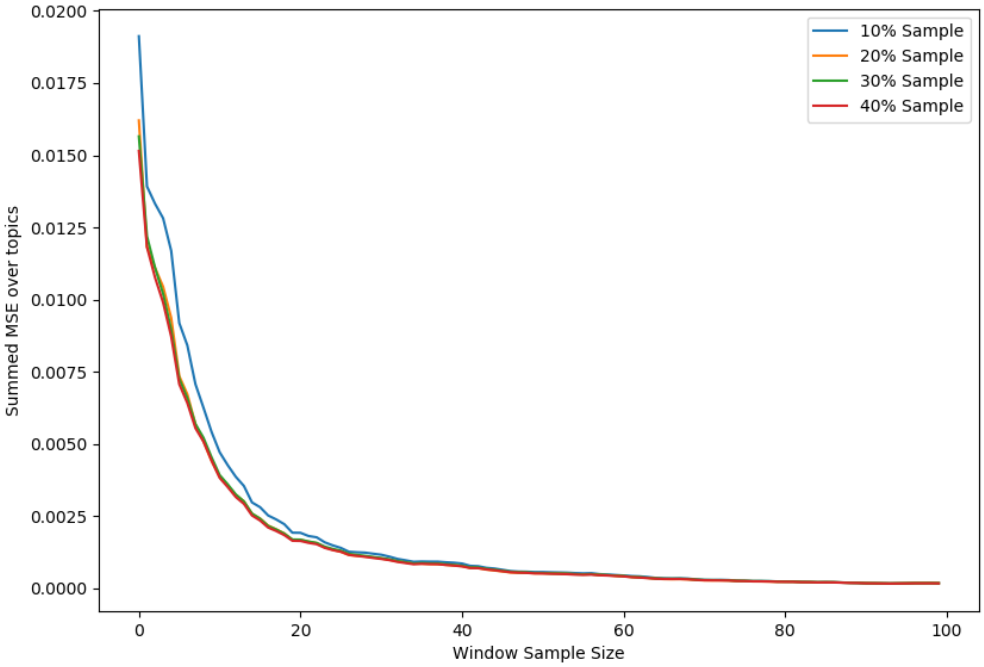

Figure 4.5: Comparison of different sample portions against sample window size

We can see a fairly steep drop decline between using a sample window of 1 to 20. As we reach 50, the curve has become almost flat. Therefore, we will use a sample window of 50 for the remainder of this work.

Results for Non-Homogeneous Poisson Process Using the above definition and sample window we created a non-homogeneous Poisson process and generated some results. We still consider different sample portions for generating our initial curve.

% of Documents	10%			20%			40%			60%			80%			90%		
Run	Recall	Reliability	Effort	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Sheffield-run-2	0.92	0.86	0.76	0.97	1.0	0.85	0.99	1.0	0.96	1.0	1.0	0.99	1.0	1.0	0.99	1.0	1.0	0.99
Waterloo A-rank-cost	0.89	0.86	0.72	0.96	1.0	0.86	0.99	1.0	0.92	0.99	1.0	0.96	0.99	1.0	0.97	0.99	1.0	0.98
Waterloo B-rank-cost	0.90	0.82	0.77	0.92	1.0	0.87	0.98	1.0	0.92	0.99	1.0	0.96	0.99	1.0	0.98	0.99	1.0	0.98
auth run-1	0.93	0.91	0.78	0.97	0.95	0.93	0.99	1.0	0.95	1.0	1.0	0.99	0.99	1.0	0.99	1.0	1.0	0.99
auth run-2	0.91	0.95	0.70	0.96	0.95	0.87	0.99	1.0	0.93	1.0	1.0	0.99	1.0	1.0	0.99	1.0	1.0	0.99

Table 4.8: Comparison of results using a non-homogeneous Poisson process and different sample sizes

Results for Non-Homogeneous Poisson Process using Dynamic Sample Size The limitation of using a set sample percentage is that it is not suitable for all topics. In some situations we may reach a desired level of recall sooner by sampling few documents. For this reason we alter our method slightly to increase the sample over an iteration until we estimate that we have reached our desired level of recall. We add two more steps to our process:

7 Determine if estimated number of relevant documents across the entire collection has reached the desired level of recall.

8 Repeated steps 5-7 until condition 7 is satisfied.

Run	Recall	Reliability	Effort	Topics Ran
Sheffield-run-2	0.90	0.95	0.67	14/23
Waterloo A-rank-cost	0.90	0.95	0.64	20/23
Waterloo B-rank-cost	0.92	1.0	0.69	19/23
auth run-1	0.92	0.95	0.59	21/23
auth run-2	0.91	0.95	0.60	20/23

Table 4.9: Using an iterative approach to a non homogeneous Poisson process. Scores are macro averages over all topics

This imposes a further challenge in that a non-fitting curve may result in an estimation never reaching a deserved level of recall. After applying some data manipulation to the sampled data we were able to get the majority of the topics running using a non homogeneous Poisson process. The maximum possible topics that could be ran is 23. In situations where the topic could not generate a curve/the Poisson process was not able to predict a stopping point we applied an effort score to the topic (1.0).

Overall, we still found this approach to be highly dependant on the quality of the ranking collection. This stemmed from the rate function not complying with an exponential curve when using sampled data that is not sufficiently ordered. This can be observed by considering a situation where the probabilistic sampled data fluctuates as we descend down the rankings. We would always expect succeeding values to be less than or equal too the previous value.

4.5 Comparing Methods

Comparing our new methods to existing methods we can evaluate how well our methods are doing. We will use the Sheffield run data from the CLEF 2017 task [11].

Method	Target	Recall	Reliability	Effort
Knee Method	-	0.88	0.86	0.64
Target Method	10	0.95	0.96	0.65
Sheffield-run2-curve	-	0.75	0.64	0.51
Sheffield-run2-cutoff(85.5%)	-	0.93	1.0	0.53
Sheffield-run2-poisson	-	0.90	1.0	0.67
Sheffield-run2-poisson + cutoff(85.5%)	-	0.89	0.95	0.54

Table 4.10: Comparing target method, knee, cut-off and curve fitting along with confidence interval. Using Sheffield-run-2

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 the cut off method alone does quite well, however the limitation of this approach is that we had to learn the best cut-off point (85.5%). The Poisson process does well, however the effort is still high, due to the number of topics that could not be ran.

Finally, we include an ensemble solution of both the Poisson process and the cut-off method. For topics that failed during the Poisson process stage we apply the cut-off method. For the Sheffield rankings this meant 14 topics went through the Poisson process and the remaining 9 went through the cut-off method.

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 [7]. The cut-off method, whilst very simple does well in comparison to the other methods, but is limited by having to find the optimum cut-off rate. The Poisson process shows promise, but needs very tweaking and optimization to be further improved.

4.6 Automatic Full Text Retrieval

A common theme throughout all of the methods presented above is that we actually using very little information to find a stopping point. What would be useful is to use the abstracts and full texts to as a way of determining relevant and non-relevant documents. Abstracts call easily be retrieved by mapping the study ids in the rankings to a PubMed query. Full texts are much more challenging to retrieve due to restrictions in licences and general availability. This

final section provides some initial work on retrieving full texts automatically and evaluating the retrieval rate.

We developed an experiment that uses the qrel file from the CLEF 2017 task. The qrel contains a list a studies for each Cochrane systematic review, as-well as a flag indicating whether or not it was used in the finished review. We will first read this qrel file into memory and send a request to the following resources in attempt to retrieve the pdf:

- Springer Link
- Humana Press
- Blackwell Synergy
- Wiley
- Science Direct
- Choose Science Direct
- Ingenta Connect
- Cell Press
- jbc
- Nature
- Nature Reviews
- Pubmed Central
- PNAS
- Cold Spring Harbour

Type	Total Number in qrel[50 P/R Max]	PDFs retrieved	Rate
Relevant Documents	607	213	0.35
Non-Relevant Documents	1500	436	0.29
Both	2107	2107	0.30

Table 4.11: Return rate of PDF studies for 30 systematic reviews using content-level qrel file

As the the number of Non-Relevant studies was exceedingly high (Over 120000 for all reviews), we limited the number for each study to 50.

We are able to acquire 30% of the pdfs through automated techniques. Our return rate relevant documents is slightly higher, this is beneficial as the information is much more useful.

4.7 Chapter Summary

This section presented some new approaches to finding stopping points for the data filtering stage of the systematic review process. We found approaches that use sampling to have strong potential, but challenging to develop due to having to find an optimum sample size and varying quality in data rankings. We found two methods that only need to transverse down the document ranking set and do not require sampling to actually provide good results. We also looked at how we can acquire new information from the rankings by obtaining full texts and/or abstracts.

Chapter 5

Future Work

The future work for this PhD will look at different techniques to finding stopping points. This will look at adapting stopping methods so that they do not **depend on ranked document collections**. This makes absolute sense as a study is never considered to be more relevant than any other. Instead, they are considered to be relevant or non relevant.

5.1 Rank-based methods to Stopping

An 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 4.2.3 we examined a technique that uses the top document as the pseudo document for comparing subsequent documents. To improve the accuracy of the pseudo point we could try using different intervals in the rankings and taking average over a set of points:

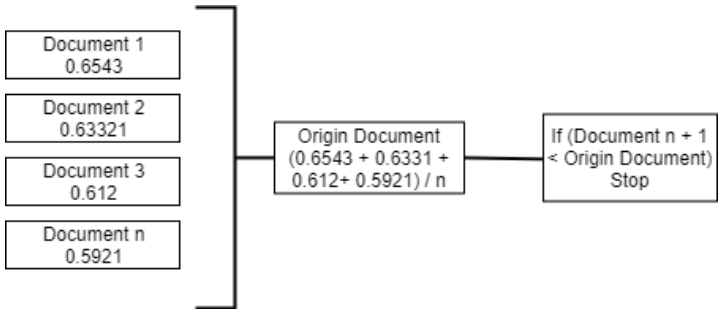

Figure 5.1: Calculating Origin Average

The % difference between document $n+1$ and the pseudo 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 pseudo document using the document abstract text:

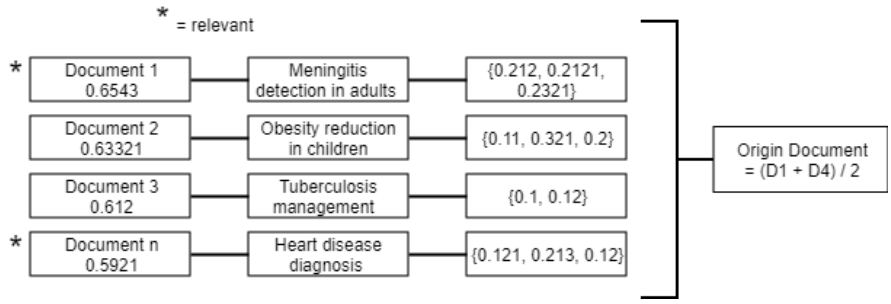

Figure 5.2: Generating Origin vector using document abstract

We have further developed the method 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.

This method can be tweaked and optimized by standard by applying standard text processing/NLP techniques:

- Using Word2Vec to represent abstract features
- Language modelling abstract features, trying out different NGram sizes
- Pre-processing of a abstracts

Finally we can use the work in 4.6 as part of this method by extracting further useful information from the full texts. This information could be used to expand the content of the abstract as-well act as a separate source for further information for finding a stopping point.

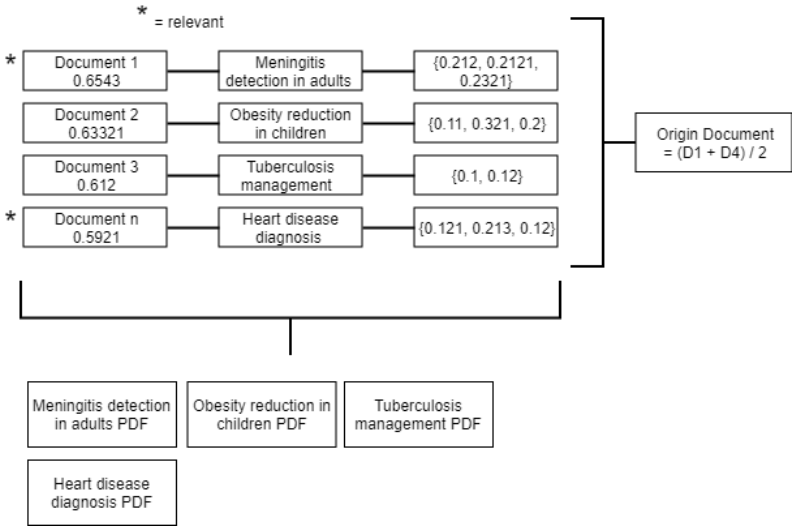

Figure 5.3: Generating Origin vector using document abstract and text

Another method to stopping that could be applied is implementing a classifier. We would still look to use the abstracts as training data for relevant and non-relevant documents. This method would assume we take a sample set of documents as our training data. This training data would then be used to build a classifier to determine if the rest of the documents are relevant or non relevant. We can infer the expected number of relevant documents from the sample and make the assumptions about how many we would need to find to attain a reliability score of 95%.

5.2 Forest Plot Approaches to Stopping

Trying to look at different papers that answer the same question is difficult, especially when these papers come to different conclusions [4]. A forest plot takes all of the relevant information asking the same question, extracts the relevant statistical data and displays it on a single axis.

Each study within the forest plot will carry a level of importance for how much information it contains. Studies that contain more participants will carry a greater weighting in the decision making. The size of the rectangle in the forest plot indicates with importance of the study.

Studies also carry upper and lower bounded confidence intervals of 95%. Studies with a lower range between confidence intervals are considered more reliable sources of information. Confidence intervals in the forest plot with longer lines with have less influence on the pooled result (diamond).

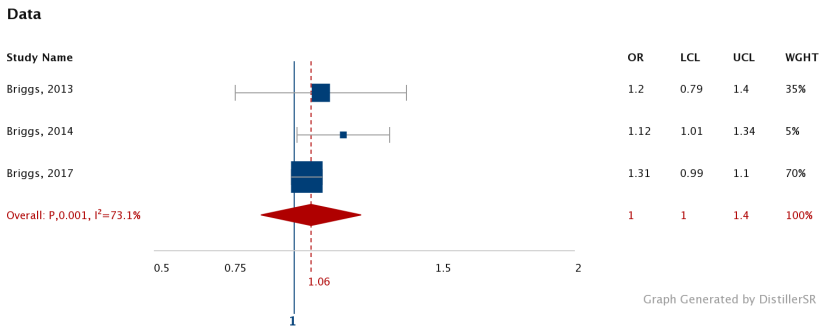

Figure 5.4: Example Forest plot

A stopping method could be developed that uses the meta data from these forest plots to make a decision. Once the pooled result does not change significantly after looking at more studies it could be assumed a decision can be made. Therefore the approach could be assembled as follows:

1. Start looking at studies and extracting meta data
2. Add meta data to forest plot
3. Update pooled result
4. Determine weight of change between old and new pooled result
5. Repeat steps 1 to 4 and stop when result stops changing

The challenge would be finding appropriate time to conclude that the pooled result has stopped changing. The task of extracting the meta data could also be modelled as a separate information extraction problem.

5.3 Gnatt Chart

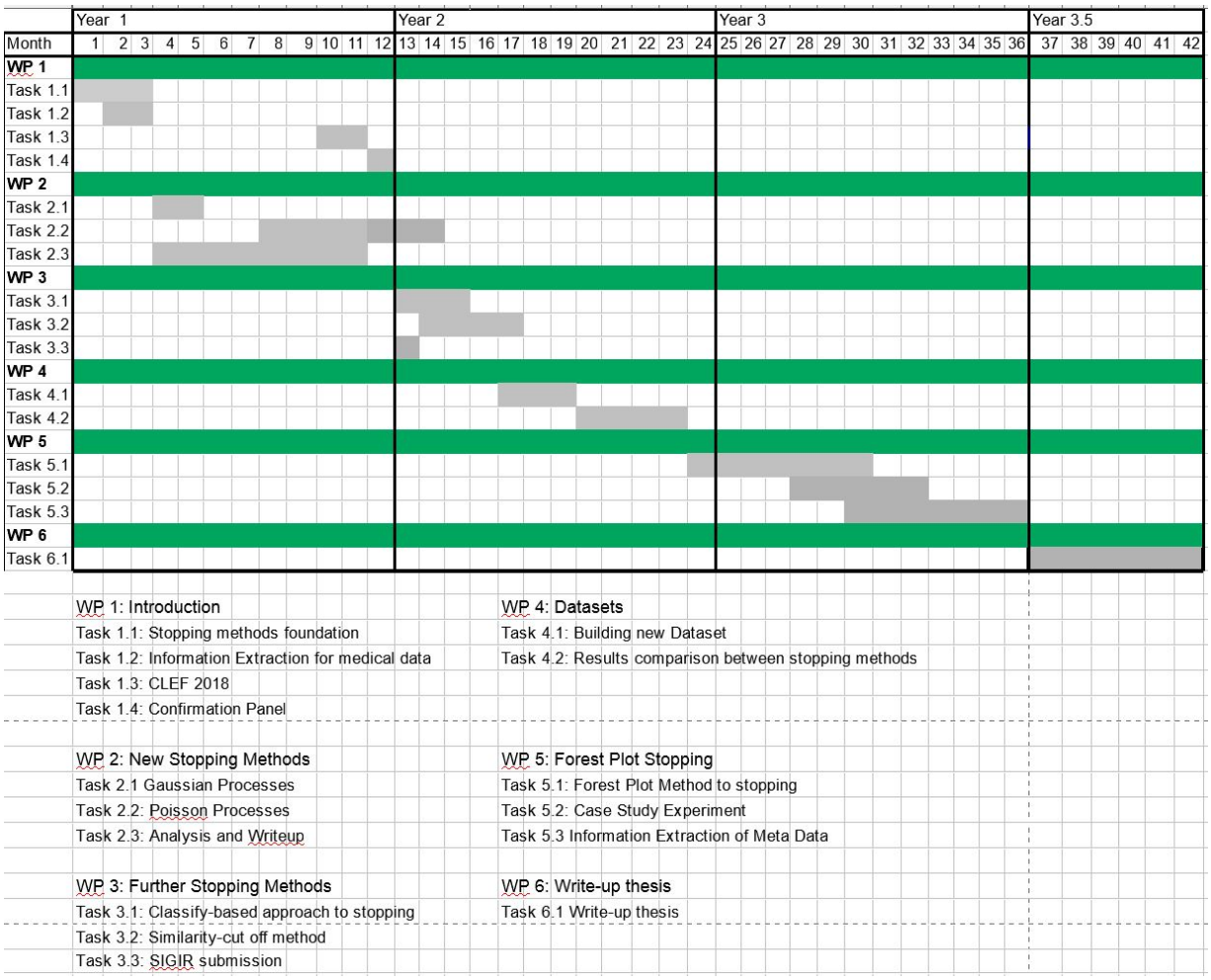

Figure 5.5: Gnatt Chart

Initial work focused on background reading around stopping points as well as exploring other ideas as for applying text processing within systematic reviews. Existing approaches to stopping were looked at and evaluated using a new dataset. We submitted some work for the 2018 CLEF conference and attended and presented the work.

We applied a curve fit approach and a Guassian process approach to stopping. Finding the curve showed some promise, but was not flexible in allowing us to specify a desired level of recall. We discovered a GP was not suitable due us being able to infer the distribution of our data already. An idea that stemmed from using a GP was a Poisson process. It was discovered a non homogeneous Poisson process could be suitable to our problem due to its focus on having variable rate. We applied the NHPP and found results to show promise, but limited due to the variance in quality between topics in the dataset. These results were then

written up and compared to previous stopping methods.

Future work will expand on the stopping methods. We will apply a classifier based approach to stopping and look at how additional information can be used strengthen the quality of the dataset rankings. We would first like to map the ranking sets back to the abstracts and use this information predict a stopping point. We would also like to expand this further by using information from the full texts for the same purpose. We would like to further refine the work on the Poisson process approach and submit to SIGIR 2019.

We would also like to expand the datasets so that we have a more robust and general way of testing. The information presented in the current data rankings is variable. We would like these to be consistent. We could also build our own dataset by using the cochrane data and mapping the information back to PubMed.

Another approach that we will apply is using the forest plots as a way for finding a stopping point. We will first attempt to extract this forest plot information from the sample reviews. This approach would work by descending down the rankings until a relevant document has been found. We would update our forest plot with this information and continue to descend down the rankings. The forest plot would keep getting updated until we can make a decision that a stopping point has been found. As as further incentive behind this idea we would also like to apply this approach within a real systematic review process and examine the effectiveness of the strategy. The process of extracting the meta data for updating the forest plot could also be included as an information extraction problem.

The final six months have been reserved for writing up, as this PhD is funded for 3.5 years.

Chapter 6

DDP

- Attended Healtex - UK HEALTHCARE TEXT ANALYTICS CONFERENCE.
- Lab demonstrator for module COM4519 Cloud Computing. (2017)
- Undertook module HAR6169 Study Design and Systematic Review Methods.
- Marked assignments for COM3110 Text Processing. (2017)
- 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.
- Attended CLEF 2018.
- Gave talk at CLEF 2018 on using limited information for querying PubMed.
- Lab demonstrator for Text Processing, Java Programming and Python Programming modules (2018)

Bibliography

[1] 6. Non-homogeneous Poisson Processes, howpublished = `http://www.randomservices.org/random/poisson/nonhomogeneous.html`, note = Accessed: 2018-11-01.

[2] 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).

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

[4] Cantley, N. Tutorial: How to read a forest plot, howpublished = `https://www.students4bestevidence.net/tutorial-read-forest-plot/`, note = Accessed: 2018-11-01.

[5] 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.

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

[7] 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.

[8] 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.

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

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

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

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

[13] 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–.

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

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

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

[17] 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.

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

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

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

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

[22] 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.

Appendices

Appendix A

A.1 Indexing and Querying Medline with Limited Information

CLEF 2018 [6] 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 [6] is to search effectively and efficiently bypassing the construction of the Boolean query.

A.1.1 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 [19] to extraction key-words from a protocol. The minimum word occurrence count is set to 1, as the protocols summaries 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

A.1.2 Indexing Pubmed

Pubmed was downloaded from the online resource [1]. 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 [2] 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 [3], Lancaster stemmer and lower-casing.

A.1.3 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.[4]
- **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.[5]

[1] https://www.ncbi.nlm.nih.gov/home/download/
[2] https://lucene.apache.org/
[3] https://www.ncbi.nlm.nih.gov/books/NBK3827/table/pubmedhelp.T.stopwords/
[4] https://lucene.apache.org/core/7_0_1/core/org/apache/lucene/search/similarities/BooleanSimilarity.html
[5] https://lucene.apache.org/core/7_0_1/core/org/apache/lucene/search/similarities/TFIDFSimilarity.html

- **sheffield-bm25** This run uses the bm25 similarity measure [18]. We used the Apache Lucene bm25 similarity class for our implementation.[6]

A.1.4 Results

Results were generated using the eval script from the CLEF 2017/2018 task [11]. 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 A.1: 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.
- 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.

A.1.5 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.

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

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.