Test collections for web-scale datasets using Dynamic Sampling

by

Anmol Singh

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2022

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Dynamic Sampling is a non-uniform statistical sampling strategy based on S-CAL, a high-recall retrieval algorithm. It is used for the construction of statistical test collections for evaluating information retrieval systems. Dynamic Sampling has been shown to lead to comparable or better test collections compared to pooling methods, at a fraction of the assessment effort.

In this work, we adapt a high-recall retrieval system to run a Dynamic Sampling protocol for web-scale datasets. We use this to create relevance assessments for 30 topics from the TREC 2019 Medical Misinformation Track. We compare our relevance assessments to qrels created using two pooling based approaches. We also compare the official NIST qrels, which were based on ClueWeb12B (7% of the full dataset), to qrels based on the full ClueWeb12 dataset.

Our results suggest Dynamic Sampling yields a reasonably good test collection, with comparable or lower variance for most evaluation measures. For fixed depth measures like Precision@K, the NIST qrels based on ClueWeb12B appear to have higher bias with respect to the other qrels, suggesting that it might be better to use qrels based on the full collection when possible.

Acknowledgements

I would like to thank my advisor Dr. Gordon Cormack for his guidance and support throughout my program. This work would not have been possible without his patient efforts to help me understand this area and define the research problem.

I would like to thank Dr. Maura Grossman and Dr. Mark Smucker for being the readers of my thesis.

I would also like to thank Mustafa Abualsaud for teaching me all about HiCAL and Jean Wang for all our helpful discussions. Finally, I would like to thank all my friends in Waterloo who helped me feel at home even in the exceptional circumstances of this time.

Dedication

I dedicate this thesis to my parents and sister, whose support and love and presence in the last year has helped me finish this.

Table of Contents

List of Tables

Chapter 1

Introduction

Information retrieval systems are evaluated on the basis of how well they satisfy the information needs of a user. For a particular query, an information retrieval (IR) system returns a ranked list of documents, typically sorted in a decreasing order by a relevance score. In IR literature these ranked lists are referred to as system results. There are a number of evaluation measures to judge the quality of system results returned in response to a particular information need. Almost all of these measures are calculated based on relevance judgments for a set of documents, referred to as a test collection. High-quality test collections are important for evaluation measures to remain as representative of the quality of system results as possible.

The ideal test collection would have exhaustive, accurate relevance judgments for every document in the corpus. This is obviously not feasible for the vast majority of document collections, and hence most test collections are necessarily incomplete. The problem is particularly acute for modern web-scale document collections, where we can only hope to manually judge a tiny fraction of the documents.

If we can only judge a limited number of documents, the natural question that arises is how to choose the best judgement set for review. A uniformly random sample does not work because the number of relevant documents can be far lower than the total number of documents, dramatically so for web-scale collections. Over the years, variants of the pooling method have emerged as the most popular ways to select this judgement set, particularly for tasks at the annual TREC conference. The depth-k pooling method in particular can be considered the standard method of constructing test collections[6]. Pooling is discussed in more detail in Chapter 2, but we introduce the idea briefly here.

TREC tracks require participants to submit their system results for each task; these

are referred to as runs. The general idea of pooling is to select a judgement set from the collection formed by pooling together documents from all of the participating runs. This set, or a subset, is then sent to NIST assessors for relevance judgements. Notably, documents that are not assessed are assumed to be not relevant; this includes all documents not retrieved by any of the participating systems. This leads to a test collection that has enough relevant documents, and is reasonably good for the purpose of evaluating the participating systems.

A limitation of pooling is that by definition it only draws from the pool of runs to be evaluated. If all the submitted runs miss a class of relevant documents for some reason, then these will be entirely absent from the test collection. In recent work, Cormack and Grossman argue that pooling should be replaced by "Dynamic Sampling"[6] as the way to construct test collections. Dynamic Sampling is a non-uniform sampling strategy which can draw from the entire document collection. It involves reviewing documents using an active learning approach which was first developed for high-recall information retrieval.

High-recall retrieval is the problem of retrieving almost all relevant documents from a collection with minimal human review effort. In contrast to ad-hoc search, where usually the goal is to find any document that satisfies the information need, high-recall retrieval systems are designed to retrieve a much larger number of relevant documents. Two important applications that require high-recall are technology assisted review in legal contexts and systematic reviews of clinical trials in medical literature.

Past work on high-recall retrieval problems has led to the development of algorithms like CAL[3], AutoTAR[4], S-CAL[5] and systems like BMI[1] and HiCAL[1]. The Dynamic Sampling technique is an adaptation of the S-CAL algorithm to the domain of test collection construction. Past experiments have demonstrated that using Dynamic Sampling can lead to more accurate test collections with lower bias and comparable ability to rank system effectiveness[11].

Dynamic Sampling has only been tried for small and moderate sized document collections so far[6][11]. This thesis extends that line of investigation by creating a test collection for the web-scale ClueWeb12 dataset using Dynamic Sampling. In the next two sections, we provide a brief overview of our work and the outline of the rest of the thesis.

[1] https://cormack.uwaterloo.ca/trecvm/

1.1 Experiment Overview

We re-designed and extended some of the core components of an existing high-retrieval system - HiCAL[1] - to work for the web-scale ClueWeb12 collection. The main challenges here were wrangling the ClueWeb12 WARC data into a format suitable for a standard classification library, and creating a multi-threaded program to score the entire document collection as fast as possible.

The Dynamic Sampling strategy then had to be adapted to this novel setting. We conducted a small pilot study to help inform various hyper-parameter and design choices in the labelling protocol. The first step of our final protocol is to find 10 relevant seed documents for every topic using the ClueWeb12 search engine provided by the Lemur Project[2]. These seed documents, along with a pseudo-relevant document consisting of the seed query, were used as the positive examples to train an initial logistic regression classifier. The negatives examples were chosen by randomly picking 100 documents from the corpus. The assessor then uses HiCAL to review a fixed-sized batch of the top-scoring documents returned by the model. The model is then re-trained on these new judgments and it re-scores the collection and the process repeats again. The model draws from an diminishing subset of the document collection as we see more and more relevant documents. The system prompts the reviewer to stop labelling when it determines that the stopping criteria has been reached at the end of a batch review.

The author then spent about five weeks judging 12,693 documents for 30 topics from the TREC 2019 Medical Misinformation Track, drawing from the entire ClueWeb12 collection. For our experiment, we only focused on making relevance judgements and comparing them to the NIST relevance judgements. The Medical Misinformation Track also includes judgements for credibility and correctness, which are aspects we do not investigate. The labeling effort was discontinuous in nature due to the time it took to re-score the collection between each batch review. On an average, it took a few hours of effort spread across 1.5 days to label each topic.

We then compared the overlap between our test collection and the TREC official test collection. We also evaluated both collections on the basis of how they effectively they ranked 44 runs generated by different models in the Anserini toolkit.

Our results show that Dynamic Sampling is a feasible method for creating test collections for web-scale datasets, and does comparably well to pooling-based methods. Additionally, Dynamic Sampling led to rankings with lower bias and variance for shallow

[2]http://boston.lti.cs.cmu.edu/Services/clueweb12/lemur.cgi

fixed-depth measures like Precision@K and RBP@K, with $K \leq 30$. A second question we tried to answer was whether the TREC test collection, which was based on the B13 subset, is good enough for the purpose of ranking retrieval systems based on how they would perform on the full ClueWeb12 dataset. We observed that the bias for the B13 collection was higher for a number of evaluation measures, suggesting that it might be better to use Dynamic Sampling over the full collection instead.

1.2 Thesis Outline

The outline of the rest of this thesis is sketched out below.

In Chapter 2, we cover some background and related work. We discuss information retrieval evaluation measures, high-recall retrieval and construction of test collections.

In Chapter 3, we describe the design and implementation of our system. We also describe how our data was processed and set up, including the software tools built for these purposes.

In Chapter 4, we discuss the design of our study protocol and describe how our experiment was conducted.

In Chapter 5, we compare our test collection to the TREC official qrels. We also discuss them in the context of various evaluation measures.

In Chapter 6, we conclude by discussing the results of our study, including limitations and future work.

Chapter 2

Background and Related Work

2.1 Information retrieval evaluation

In this section we provide a brief overview of the most common ways to evaluate information retrieval systems. This will include both the measures and toolkits that we use in our work.

Evaluation metrics

Precision and recall are two fundamental effectiveness measures for binary classification problems. In the context of information retrieval, precision is the fraction of retrieved documents which are relevant, and recall is the fraction of relevant documents which are in the retrieved set. The set of retrieved documents for a particular query is called a system result. System results are assumed to be sorted in decreasing order of a relevance score, which is calculated for each document by the system based on how well it matches the query.

Evaluation measures try to capture a system's performance at the task of satisfying user information needs. A standard assumption is that users examine each document in a system result starting from the top till their information need is satisfied. A lot of measures include a rank cutoff K, representing the assumption that the user examines only the top K documents. Precision @ K is one of the most popular measures, measuring the fraction of the top K documents which are relevant. Average Precision (AP) considers the rank of each relevant document, and computes the average of Precision @ K values at each of those ranks. Mean Average Precision (MAP) is simply AP averaged over multiple queries.

AP and MAP can be calculated by the following formulae, where $P(k)$ is precision at the k^{th} document, and rel(k) is 1 if the k^{th} document is relevant and 0 otherwise.

$$\text{AP} = \frac{\sum_{k=1}^{n} P(k) \times \text{rel}(k)}{\text{number of relevant documents}}$$

$$\text{MAP} = \frac{\sum_{q=1}^{Q} \text{AP}(q)}{\text{Q}}$$

Precision @ K captures how well a system serves a user who looks at only the top K documents. The drawback is that it ignores the rest of the documents completely. MAP is a better summary measure of the performance of the system at different points, but it does not clearly correspond to plausible user behaviour. Rank-biased precision (RBP) [15] is a measure that tries to fix these shortcomings. It has a single parameter p that is the probability that the user will proceed to examine the next document in the system result. The parameter p represents the persistence of the user, and it is used to derive document weights based on the likelihood of the document being seen. Rank-biased precision is defined as the number of relevant documents, weighted by the probability that they are examined, divided by the expected total number of documents that the user examines.

$$\text{RBP} = (1-p) \cdot \sum_{i=1}^{d} \text{rel}(i) \cdot p^{i-1}$$

Discounted Cumulative Gain (DCG) [12] is another measure which weights highly ranked relevant documents more than low ranked ones. Each relevant document is given a weight of $1/\log_b(i+1)$ where b is a parameter (commonly $b = 2$) which controls magnitude of the weights, and i is the rank of the document. DCG is typically normalized by dividing all scores by the score for the ideal ranking, so that the resulting measure (called nDCG), has a value between 0 and 1. Calculating the ideal rankings requires knowledge of R, the total number of relevant documents, and this is a disadvantage of nDCG when compared to RBP.

$$\text{DCG}_\text{p} = \sum_{i=1}^{p} \frac{\text{rel}(i)}{\log_2(i+1)}$$

Other effectiveness measures that we consider are interpolated precision at different levels of recall and recall/precision values at various multiples of R.

Evaluation toolkits - trec_eval and DynEval

After we create our test collections and runs, we use a couple of standard tools to evaluate each run against each test collection for all the measures discussed above. For test collections created without sampling, we use trec_eval[1], used by the TREC community to evaluate ad-hoc retrieval runs. For test collections created using sampling, we use DynEval[2], which is an adaptation of trec_eval for statistical test collections.

2.2 Building test collections

For large datasets with millions of documents, it is not feasible to examine every document in the collection for relevance assessment. Even examining a small random subset might not work if the density of relevant documents is too low. Thus the central problem of building test collections is how to intelligently select a subset of the dataset for review such that we cover a reasonable number of relevant documents which are likely to be found by good retrieval systems. In this section we first cover the standard variants of pooling used by the TREC community. Then we briefly introduce active learning approaches to the high-recall retrieval problem, and the non-uniform sampling method derived from them as an alternative to pooling.

2.2.1 Pooling

The basic idea of pooling is to take the union of the results returned by different retrieval systems and then assess that set for relevance. Assuming that the retrieval systems are good enough, this ensures that enough relevant documents get included in the test collection.

The most commonly used variant of pooling is depth-k pooling, where the top scoring k documents from each retrieval system are added for assessment. Typically a value of 100 is used for k.

A way to reduce the assessment cost further is to employ a strategy to select only a subset of the depth-k pool for assessment. Variants of the pooling method differ in how they select documents from the pool. Some methods use statistical sampling to draw from the pool, and then statistically estimate the value of effectiveness measures that we would get if the full pool were assessed[2][20]. Active learning can also be applied to

[1] https://trec.nist.gov/trec_eval/
[2] https://cormack.uwaterloo.ca/sample/

selecting documents for assessment from this pool. Move-to-front pooling[10] and bandit methods[13] are a few examples of such strategies.

2.2.2 Dynamic Sampling

In high-recall retrieval, the goal is to find all or nearly all relevant documents in a collection while reviewing as few documents as possible. This is similar to the central problem of building test collections, and thus algorithms and techniques for high-recall retrieval can be adapted to the problem of sampling documents for test collections.

Active learning[17] is a general approach for training classifiers with fewer labeled examples. The basic idea is to let the classifier choose the examples to learn from. Continuous active learning [3][4] was one of the first protocols to use active learning to the high-recall retrieval problem, in the context of technology-assisted review (TAR). A limitation of CAL is that the required labeling effort is proportional to the number of relevant documents, which typically increases with the size of the collection. S-CAL[5] improves upon that by using sampling, leading to a protocol with $\mathcal{O}(\log N)$ labeling cost, where N is the number of unlabeled documents.

Dynamic Sampling[6] is S-CAL adapted to the problem of building test collections. Dynamic Sampling leads to test collections which are superior to ones built using static sampling strategies[6]. When compared to pooling, Dynamic Sampling has been shown to yield test collections which have a lower bias and a similar ability to rank system effectiveness, with much lesser assessment effort[11].

Chapter 3

Implementation

In this chapter, we go into the details of how our judgement system was designed and implemented. We first also describe the ClueWeb12 dataset, and how we pre-processed it to deal with various challenges posed by its enormous scale.

3.1 The ClueWeb12 dataset

We used the full ClueWeb12 dataset for our experiment. The ClueWeb12 dataset consists of 733,019,372 documents, stored in WARC files. The total number of WARC files is 33,447. The size of the compressed dataset is 5.54 TB and the full uncompressed data is 27.3 TB. This is by far the largest collection that Dynamic Sampling, and CAL approaches in general, have been tried on.

The ClueWeb12B collection was created as a representative, uniform 7% sample of the full dataset. It was created by taking every 14th document from each WARC file of ClueWeb12. It consists of 52,343,021 documents, stored in the same format in 33,447 WARC files. The compressed size of the dataset is 389 GB and uncompressed version is 1.95 TB.

The WARC file format is a standardized archival format used to store web crawls. In ClueWeb12, each web page that was crawled was first converted to a WARC record. A number of WARC records were grouped together into a single WARC file of about 1 GB, which was then compressed using gzip. The primary unique identifier for each record is the custom WARC-TREC-ID field in the WARC response header. This ID uniquely identifies

the location of the WARC file containing the record, and the sequence number of the record within that file.

3.2 Processing the data

We first distributed the full dataset roughly equally between 36 hard disks. This was done primarily so that the scoring could be parallelized as much as possible.

We derived several secondary files from the original WARC files to make the ClueWeb12 collection fit for use with HiCAL and also to deal with scale challenges. We created a modified version of the WARC files to make access to individual records faster. We also derived files containing TF-IDF feature representations to prepare the collection for scoring with models trained using the Sofia-ML library. We also created files corresponding to several additional subsets for every WARC file, to make it trivial to do downsampling. This also ensures that downsampling leads to reduced scoring time.

We describe each of these secondary files in detail in the following sub-sections, along with the software tools used to create them.

3.2.1 Individually compressed WARC records

In the standard ClueWeb12 collection, each compressed WARC file was created by concatenating WARC records and compressing them together. There is no way to locate individual records directly and decompress them in this format. The whole WARC file needs to be decompressed to seek an individual record inside it. To make this process more efficient we created a version of the WARC files in which all records are compressed individually using gzip and then concatenated together. For each such new WARC file, we also create an index of TREC ID to its offset in the WARC file. This gives us a way to seek individual WARC records in a compressed WARC file without decompressing and going through the entire file.

To do this we used the multi-purpose zchunk tool[1], which internally uses the zlib compression library. The zchunk tool was also used to create the TREC ID to WARC offset index file.

[1] zchunk was written by Dr. Gordon Cormack

3.2.2 Feature files for scoring

We used TF-IDF features as our document representations for training models and scoring documents using the Sofia-ML library. For each WARC file, we generate the concordance file containing the term frequency counts grouped by records. A DF file containing document frequencies for each term that appears at least once in the WARC file is also created. Both the concordance and the DF files were created using the zchunk tool. The DF files for each WARC file are then hierarchically combined for every sub-directory, every directory and every disk.

The DF file for the entire corpus contains 238.5 million terms, and is 3.5 GB in size. In a web-scale collection, we can expect almost all terms which occur only once to be meaningless character sequences. We pruned our DF file down to 104 million terms and 1.5 GB size by removing all such singleton terms.

The TF-IDF representations for each record in each file were then generated, using the pruned DF file and the term frequency counts in the concordance files. The TF-IDF files were further compacted by transforming them into a binary format, for faster scoring. We refer to these files as compact feature binaries in the rest of the chapter.

3.2.3 Downsampling the feature files

The original Dynamic Sampling algorithm scores the document collection in every iteration, and then selects a decreasing fraction of the B top scoring documents uniformly at random for judgement. Here B is a hyper-parameter that specifies the strata from which we sample at a decaying rate. Note that we score the entire collection even when we are sampling a decreasing fraction of the top scoring documents as the labelling progresses.

For the novel setting of a web-scale document collection, we would like to not only sample at a decaying rate but also reduce the scoring time as the algorithm progresses. We achieve this by simulating the decaying rate of sampling by scoring only a diminishing subset and returning all of the top K documents from it. If this subset is chosen uniformly at random, then this is equivalent to returning a diminishing fraction of the top B documents uniformly at random.

We created 11 diminishing subsets for the ClueWeb12 dataset, with separate files for each. Since we need these subsets only for the scoring step, we create these new files only for the compact feature binaries. Counting the full collection, this gives us 12 levels for the algorithm to progress through, with each level corresponding to a decreasing sampling rate.

The first level is the full dataset. Each subsequent level contains half the documents of the previous level. Since we intend to compare our judgements against the TREC 2019 Medical Misinformation qrels, we would prefer to judge more ClueWeb12B documents. In order to get more ClueWeb12B documents for review, we also give those documents special priority. We ensure that the full ClueWeb12B subset is present in every subset up to level 4. At that point we have reduced the subset size to 12.5% of the full dataset. To make the level 5 subset exactly half of the current size, we removed all non-ClueWeb12B documents and 1/8 of the ClueWeb12B documents. This gives us the level 5 subset, which is 6.25% of ClueWeb12, and contains only ClueWeb12B documents. After this point, we continue reducing the subset sizes by half as before. The total extra memory required by all the subset files is at-most the size of the full collection, and the smaller files ensure the scoring cost reduces by approximately half every time we downsample.

The naive method to create N subset files is to go through the full collection N times, picking every second document the first time, then picking every fourth document the second time and so on. But this requires lots of redundant reading of the records. Also, we need a way to ensure that every ClueWeb12B document i.e. every 14^{th} document is present in every subset up to level 4. Instead, we use a technique that requires us to read the collection only once and creates all subsets directly.

To create the subset files for all levels directly, we assigned a level number R to every record, which can be calculated based on just its sequence number in a WARC file. R denotes that this record is present in every subset up to and including level R. We modified the program that generates the compact binary representations to output the representation for each document D_i with level number R_i to the first R_i files.

The main challenge here was to ensure that the ClueWeb12B documents all get a level number 4 and above. We discuss the approach we used to generate the R values for our downsampling scheme below. We first describe the formula, then our modification to give ClueWeb12B documents priority.

Calculating the level number for each document in a WARC file

1. Let the documents in the WARC file be indexed starting from 1.
2. We want to include every 2^n-th document in the n-th subset.
3. Consider the document with index number D_i. If this is to be included in the n-th subset, then D_i must be divisible by 2^n. We can see that this also implies D_i is included in all the subsets 1 through $n-1$.

4. The maximum n such that a number D_i is divisible by 2^n is given by the count of zeroes at the end of the binary representation of D_i. Let this count be represented by Z_i.

5. This gives us a simple formula for R_i: $R_i = Z_i + 1$

Calculating the level number for each document in a WARC file, with priority to ClueWeb12B

1. We want to retain ClueWeb12B documents in our subsets for as long as possible.

2. The ClueWeb12B subset was created by taking every 14^{th} document of the full collection. The ClueWeb12B collection can be included till the fourth level of sampling i.e. 1/8 of the full collection. After that, we move to 1/16 of the collection and we need to drop some ClueWeb12B documents. We can do this by ensuring that every 14^{th} document gets an R number of at least 4.

3. We first assign each document an R number using the previous formula.

4. Consider all the index numbers in blocks of size 16. From the formula we can see that the R number of the 8^{th} position will be greater than 4 and 16^{th} position will be greater than 5.

5. These blocks can contain ClueWeb12B documents in exactly 7 ways, which repeat.

6. The first 6 blocks contain exactly one ClueWeb12B document.

    - We swap the R number of the ClueWeb12B document with the R number of the 16^{th} document, guaranteeing an $R \geq 5$.

7. The 7^{th} block contains two ClueWeb12B documents, one at the second position, one at the 16^{th} position.

    - We swap the R number of the ClueWeb12B document in the second position with the R number of the 8^{th} document, guaranteeing an $R \geq 4$.
    - The 16^{th} document already has an $R \geq 5$, so we don't need to do anything.

3.3 On-disk HiCAL

HiCAL is an open-source system for high recall retrieval. It consists of two core components of interest to us - the first is a web application for document assessment built using Django, a Python web framework. The second is CAL, a C++ based backend service implementing model training and scoring. HiCAL is best understood as a human-in-the-loop active learning system, with the frontend providing a user interface for a human judge, and the backend handling 1) training of a classifier based on relevance feedback from the judge 2) scoring of all documents and selection of the next judgement set. In this section we describe how we adapted each component of HiCAL to work for our novel setting of a web-scale document collection[2]. The main challenge here was to speed up the training and re-scoring steps as much as possible, in order to ensure a quick turnaround time for the human judge.

3.3.1 Lazy extraction of documents

Our document collection is stored as files containing individually compressed WARC records and files containing TF-IDF feature representations for the records in a single WARC file. Having access to the uncompressed documents is a necessity for sending them for judgment, and would also simplify the creation of a training set in our design. However, creating an uncompressed version of the entire collection would incur a significant storage cost of 27.3 TB. We observe that we will only ever need a tiny fraction of the documents for training and judgement. Thus we extract the full uncompressed documents in a lazy fashion - whenever we encounter a document ID in a training or judgement list that hasn't been de-compressed yet, we use zchunk to extract it from its compressed WARC file and store it at a unique path determined by the ID. This full path is also used as the unique identifier for documents internally, as this makes retrieval by the front-end trivial, obviating the need for an index lookup.

3.3.2 Adapting CAL training for the ClueWeb12 collection

The training in HiCAL is initialised with just the seed query as a positive example. In our experiment, we used an additional 10 seed positive documents gathered using an external search engine. We added a form to HiCAL frontend to input these documents as comma-separated TREC IDs, and also changed the training of the initial model to use them as

[2]For reference, the adapted version can be found at https://github.com/kshanmol/HiCAL-web-scale

positive examples. The input TREC IDs were also first mapped to our own unique internal IDs. Following standard CAL practice, a randomly chosen 100 documents were used as negatives.

HiCAL uses the Sofia-ML library to train a logistic regression classifier. In HiCAL, all document representations and classifier models are made available in-memory to allow for faster training. This is not feasible for our setting. We instead put the training set together in a new file and derive its TF-IDF features using the global DF file. The Sofia-ML library is invoked for training at the command line and the trained model is output to the corresponding session directory.

For our vocabulary size, the default trained model that Sofia-ML outputs is nearly 9 GB. It took 3.5 minutes for the training step to complete, most of which was spent in writing the model to disk. The model file contains the weight vector for all the terms in our vocabulary, most of which have a zero weight. We modified Sofia-ML to use a new compact representation for the model output, only including the feature index and the weight for terms with non-zero weight. This brought down the typical model size to under 500 KB, and reduced the training step time to 20 seconds.

3.3.3 Adapting CAL scoring for the ClueWeb12 collection

The feature representations to be scored for the full-collection cannot be stored in-memory in our setting. Instead they are stored in files distributed roughly over 36 hard disks as discussed in Section 3.2.2.

We use the multi-threaded myreadT program[3] to parallelize scoring as much as possible. It reads in the compact format trained model and creates 36 task queues, corresponding to the 36 disks containing the feature files. It also creates 1000 worker threads that do scoring, with each thread assigned to a particular task queue. The worker adds to its queue whenever it discovers a document scoring better than the 1000th biggest score so far. The main thread goes through these queues, inserting the records into the top 1000 overall using linear insertion.

Scoring using a single model takes about 9.5 minutes for the full collection (733 million documents). For scoring the subsets, the time taken reduces in proportion to the size of the subset file. So scoring the level 2 subset (365 million documents) takes about 5 minutes, the level 3 (180 million documents) takes about 2.5 minutes, and by the time we reach level 5, scoring takes under a minute.

[3]Written by Dr. Gordon Cormack

Chapter 4

Protocol Design and Experiment

In this chapter, we describe the design of the protocol that we used to create our test collection. We also describe the complete protocol and how we used it to label 30 topics from the TREC 2019 Medical Misinformation collection. To adapt the general technique of Dynamic Sampling to our setting, we had to make a number of hyper-parameter and design choices. We decided to first conduct a short pilot study by labelling a few topics from the TREC 2014 Web Track, which also utilises the ClueWeb12 collection. We first discuss the pilot and how it informed the design of our protocol.

4.1 Protocol design considerations

The primary aim of the pilot study was to develop a better feel for how our system works after being adapted for the web-scale setting. This would help us make informed design choices for our labeling protocol and experiment setup. We first list some of the open questions about the protocol we had before our pilot study.

- What should be the batch size of the documents sent for judgement? This defines how many new training examples are added when the model is retrained in the active learning loop. There have been several variants in CAL protocols, including a fixed size of 1 and exponentially increasing batch sizes. By default the HiCAL system retrains the model on every new judgement. But to make the system responsive for the judge, it continues serving the top scoring documents from the previous model and also caches any new judgements that arrived when the model was being retrained. For

our setting, such a level of responsiveness is not feasible because Dynamic Sampling requires us to retrain and rescore the collection after the judgement of each strata. For the full collection, it takes at least 10 minutes for the system to finish retraining and rescoring after it receives new judgements.

- During the course of judging documents, when should we perform downsampling on the collection? Downsampling corresponds to decreasing the sampling rate. Whenever we downsample, we essentially switch to a larger universe of documents to draw from. To see as many relevant documents as possible, we want to sample from as big a universe as possible. But if there are very few relevant documents for a topic, we might miss them if we downsample too quickly, and might not get an accurate estimate. Thus we want to keep downsampling, but only after we've seen enough relevant documents. A second related question here is - how quickly should we downsample? In our setting, for some topics, we might discover a huge amount of relevant documents. To go through them faster, it might make sense to sometimes do a double downsampling instead.

- At what point should we stop judging documents for a given topic? Previous attempts have used a fixed budget. We experiment with using a variable budget based on a stopping criteria. After enough exploration, we can expect to reach a point when the number of relevant documents in every batch starts going down. Our stopping criteria is based on the "knee method", where the knee is a cut-off point after which frequency of relevant documents has become significantly lower than the frequency of relevant document before it.

4.2 Results of the pilot study

For our pilot study, we labeled 9 topics from the TREC 2014 Web Track. We chose 5 topics with low density (less than 30 relevant documents) and 4 topics with high density (more than 200 relevant documents) according to the NIST qrels. We labeled these with 3 different budgets - 200 documents for 3 topics, 300 documents for 4 topics and 400 documents for 2 topics. We picked a fixed batch size of 20 and slightly modified the downsampling criteria of Dynamic Sampling to require at least 10% of the most recent batch to be relevant. We discuss the rationale behind these choices in detail in the next section, in the context of the final protocol.

The topic-wise breakdown of the labeling effort is given in 4.1. Our effort (DS effort) was comparable to the NIST effort in terms of number of documents judged. We also

compute the inter-labeler agreement for the documents that were seen by both the author and the NIST assessors in two different ways. The standard way to measure agreement in the literature [19] is to calculate the overlap, which is the number of documents marked relevant by both, divided by the union of the documents marked relevant by either. This is also known as Jaccard similarity. The other score we calculate is the percentage agreement. This is simply the number of documents given the same label by both, divided by the number of documents seen by both. The overall overlap is 0.5 and the overall percentage agreement is 0.575. The overlap and agreement for topic 261 seemed notably lower. On investigating the documents, this turned out to be because the author went with a less strict interpretation of the topic query ("folk remedies for sore throat").

Topic	DS effort	NIST effort	Intersection	Overlap	Agreement
261	200	281	24	0	0.125
266	200	351	16	0.875	0.875
291	200	225	40	0.286	0.5
260	300	328	11	0.4	0.4545
271	300	266	2	0	0.5
272	300	367	8	0.5	0.5
290	300	259	30	0.55	0.7
282	400	334	22	0.727	0.727
285	400	284	28	0.714	0.714
Total	2600	2695	181	0.5	0.575

Table 4.1: Labeling effort (number of documents judged) and overlap by topic. The intersection is the number of documents judged by both. Overlap is measured as the fraction of documents marked relevant by both out of the documents marked relevant by either, and agreement is the simple percentage agreement of labels for the intersection set.

In 4.2, we compare the number of relevant documents found in the DS effort to that in the NIST effort. The second column, r_{DS}, is the number of documents seen and labeled relevant by the author. The third column, $\hat{R}_{DS}$, is the actual estimate of relevant documents given by Dynamic Sampling. Following Dynamic Sampling practice [5], the estimate was calculated by weighting each of the documents actually reviewed by the inverse of their inclusion probability. The last column is the number of documents marked relevant in the official NIST test collection.

Our overall estimate of total relevant documents is 3899, which is roughly 4 times the NIST number. This suggests that Dynamic Sampling is able to explore the space of relevant documents better with a similar effort.

Topic	r_{DS}	$\hat{R}_{DS}$	R_{NIST}
261	107	375	21
266	121	506	241
291	86	257	22
260	56	116	24
271	34	62	14
272	183	1042	227
290	52	111	30
282	192	1066	202
285	110	364	215
Total	941	3899	996

Table 4.2: Relevant documents found by the DS and NIST efforts. r_{DS} is the actual number labeled, and $\hat{R}_{DS}$ is the estimated number of documents after weighting the labeled documents with their inverse inclusion probability. Note - R_{NIST} is the best effort using depth-k pooling, but it is known by NIST that this is an underestimate.

4.3 Final experiment protocol

In this section we discuss the key design decisions we made for our protocol, and how the pilot study informed some of them. We end with a full description of the Dynamic Sampling protocol we arrived at in Algorithm 1.

4.3.1 Batch size

For our system, which can take between 5-10 minutes to refresh, we envision a user working on multiple topics simultaneously. After labeling a batch of documents for a particular topic, the backend starts the process of refreshing the model and scores. The user can move on to a batch of documents for another topic, and work on those while a fresh batch is being prepared for the first topic. This way the user can alternate between a few topics, keeping them occupied productively. In [11], the authors mention that it took them an average of 33 minutes to label 300 documents per topic. For our novel setting, we pick a batch size of 20, which would mean each batch can be expected to take about 2 minutes to label. A bigger batch size would keep the user occupied longer, but this comes at a cost of refreshing the model less frequently. We expect 20 to be a reasonable batch size with this trade-off in mind.

4.3.2 Downsampling

There is a lot of variance in the number of relevant documents for different topics. If there are very few relevant documents for a topic, then we want to see almost all of them. But if there are lots of relevant documents, we'd prefer to skip redundant ones in order to explore the corpus thoroughly and get a good statistical estimate. This involves downsampling so that we don't see too many documents from the same strata, which are likely to be similar. In Dynamic Sampling, whenever the number of documents assessed relevant (R) becomes greater than or equal to the decay threshold T, T is doubled, and this effectively halves the sampling rate.

We note that this criteria does not factor in the recent frequency of relevant documents. If the frequency of relevant documents drops a lot at any point, this might mean that we've found most of them already. In this case, we don't want to downsample on reaching T, so that we don't risk missing out on the few relevant documents that possibly remain. To factor this in, we add an additional criteria requiring the precision of the latest batch, termed marginal precision (MP), to be at least 0.1. This would mean that we will downsample whenever after reviewing a given batch of 20 documents, R becomes greater than or equal to T, and there were at least 2 relevant documents in that batch.

Double downsampling

As noted earlier, we conducted the pilot with the downsampling criteria discussed above. We noticed that in 3 topics where there were an abundant number of relevant documents, downsampling once did not seem to suffice. We also note that during the pilot we downsampled down to at most level 5. On this basis we added a rule for double downsampling. Whenever the downsampling criteria is met after reviewing at least 3 batches from a subset, we check the average MP for the last 3 batches. If it is high (greater than 0.5 i.e. more than 10 relevant documents), then we downsample twice instead of once. This new criteria would have been satisfied at some point for each of the 3 topics that motivated us to introduce it.

4.3.3 Stopping criteria

We are interested in trying a variable budget to counter the problem of variable relevance densities. We first set a minimum of 300 documents for each topic i.e. 15 batches of review. After 15 batches, we'd like to keep judging if the system keeps finding relevant documents

and to stop when the frequency of relevant documents goes down below a certain point. We use the following rule - after every batch, we stop if there exists a partition of the batches such that the average MP before the partition is more than 5 times the average MP after partition (with a minimum partition size of 5). This factors in the full history of the MPs of every batch. If the rate at which we were seeing relevant documents for some topic was only moderate to begin with, then this might prevent us from stopping too soon. When this rule was retroactively applied to our pilot labelling exercise, it asked to stop for two topics of very low density after 300 documents. It only asked to stop for one topic of higher density, after 400 documents. This suggests that our rule is reasonably effective at requiring stopping early only for topics with low density, and allows continued exploration of topics with higher density.

Algorithm 1 Dynamic Sampling Protocol

1: Find ten relevant seed documents for the topic using an interactive search platform.
2: The initial training set consist of a pseudo-relevant document containing the seed query and the 10 seed documents, all marked as "relevant".
3: Set the batch size B to 20.
4: Set the initial decay threshold T to hyperparameter N ($N = 25$).
5: Temporarily augment the training set by adding 100 random documents from the collection, labeled "not relevant".
6: Score all documents in the current subset using a model induced from the training set.
7: Remove the random documents added in step 5.
8: Select the highest scoring B documents not previously selected.
9: Render the relevance assessments for the B documents.
10: Add the assessed documents to the training set.
11: Calculate marginal precision $mp = r/B$, where r is the number of relevant documents found in this batch.
12: Check if we have reached the stopping criteria at this point. If yes, then we stop the process. Else continue to the next step.
13: If the total number of assessed relevant documents $R \geq T$ and $mp \geq 0.1$, double T and downsample.
14: If average mp of last 3 batches from this subset is 0.5 or more, downsample once again.
15: Repeat 5 through 14 until the stopping criteria in 12 is met.

4.4 Labeling effort and timeline

After the protocol was finalized, the author spent about five weeks of effort judging documents for 30 topics from the TREC 2019 Medical Misinformation Track. We found that alternating between labeling two topics worked reasonably well. It took about a few hours of effort spread across a day or two to finish judging a pair of topics, including time taken to find the 10 seed documents.

It took anywhere between 15 minutes to an hour and a half to find the 10 relevant seed documents for a single topic. The labeling process was less continuous than we had hoped - we often had to wait for the system to finish refreshing both the models. This behaviour was partly expected, but the system also took longer when it had to work on refreshing two models simultaneously. When two models were refreshing on the full collection simultaneously, it took around 12 minutes for each. When it was just one model, it took 9.5 minutes.

We labeled a total of 12693 documents for the 30 topics, including the seed documents found using interactive search. This means we labeled an average of 423.1 documents for every topic. This is a comparable effort to the NIST test collection, which have a total of 13669 judgements for the same 30 topics.

Table 4.3 shows the topic-wise breakdown of the documents we labeled using Dynamic Sampling (DS effort) and the documents NIST labeled (NIST effort). We also compute the overlap and percentage agreement for the documents that were seen by both the author and the NIST assessors. The overall overlap is 0.469, which is in the ballpark of the overlap seen in the pilot study, and also comparable to the overlap observed between different NIST assessors [19].

Note that here we only show our labeling effort and agreement for the documents that we judged manually. This mostly just serves as a basic sanity check for the label sets. Since we use Dynamic Sampling, our effort is effectively higher than this, and our statistical estimates of the number of relevant documents are also higher. Those will be discussed in comparison with the NIST test collection in the next chapter.

Topic	DS effort	NIST effort	Intersection	Overlap	Agreement
1	470	425	58	0.167	0.828
2	443	639	33	0.143	0.455
3	390	499	7	0.75	0.857
4	330	475	80	0.576	0.688
5	610	527	75	0.61	0.787
6	310	379	90	0.636	0.778
7	390	523	23	0.5	0.957
8	490	348	45	0.513	0.578
9	310	423	68	0.435	0.809
10	370	362	27	0.083	0.593
11	470	611	48	0.487	0.583
12	390	409	107	0.521	0.579
13	310	468	10	0.286	0.5
15	490	498	108	0.605	0.861
16	610	425	88	0.279	0.5
17	590	478	24	0.375	0.792
18	390	476	41	0.333	0.854
19	410	448	50	0.55	0.64
20	330	571	34	0.615	0.853
21	310	488	8	0.5	0.75
22	370	496	55	0.143	0.564
23	450	413	79	0.49	0.684
24	390	519	5	0	0.4
25	350	485	27	0.455	0.778
26	370	452	23	0.364	0.696
27	350	358	26	0.167	0.808
28	310	326	76	0.786	0.882
29	510	328	41	0.185	0.463
30	590	392	115	0.426	0.661
31	590	428	57	0.5	0.842
Total	12693	13669	1528	0.469	0.707

Table 4.3: Labeling effort (number of documents judged) and overlap by topic. The intersection is the number of documents judged by both. Our overall overlap with the NIST assessors is 0.469, which is in the range previously observed in literature [19].

Chapter 5

Results and Discussion

In this chapter, we compare the relevance assessments we created using our Dynamic Sampling protocol with the relevance assessments created by NIST assessors using pooling for the TREC 2019 Medical Misinformation Track[1]. We note again that we're comparing the NIST and Dynamic Sampling judgements on just relevance and not the overall track task, which includes aspects like credibility and correctness. We also note that the NIST did graded relevance judgements, which we convert to binary relevance judgements (i.e. relevant or not relevant) for comparisons with the Dynamic Sampling judgements, which are binary.

In TREC parlance, the files containing relevance assessments are called "qrels". We use the terms "qrels" and "relevance assessments" interchangeably in the rest of the thesis. The document corpus, the set of topics and the corresponding qrels together form a test collection. We note again that the NIST qrels were created by drawing documents for judgement from the smaller ClueWeb12B subset, and the Dynamic Sampling qrels were created by drawing from the full collection. In the first section, we examine and compare the space of documents explored by the two approaches. In the second section, we look into how well the qrels do at the task of evaluating IR system results.

5.1 Comparing the qrels

The qrels can be compared in two ways - how many relevant documents were found in each, and how much they agree on what kinds of documents are relevant.

[1] All relevance assessments can be found at https://github.com/kshanmol/2019-med-misinfo-qrels

5.1.1 Relevant documents discovered

The number of documents which were judged relevant in the Dynamic Sampling (DS) qrels is 2862. However, since Dynamic Sampling draws documents from diminishing subsets of the full corpus, the estimated total is higher. Recall that the k^{th} subset was formed by taking a uniform random sample consisting of half of the $(k-1)^{th}$ subset, starting with the full ClueWeb12 corpus. From this we can derive a simple weighting scheme to estimate the total number of documents based on inverse inclusion probabilities described by Pavlu [16]. If a document D_i was drawn from the k^{th} subset, its inclusion probability is $2^{-(k-1)}$ and it is given a weight of 2^{k-1}, starting with $k = 1$ for the full collection. Following this, the estimated total number of relevant documents according to the DS qrels comes out to be 13018.

The number of relevant documents in the NIST qrels is 2265. These qrels were drawn from ClueWeb12B, which is a 1/14 uniform random sample of the full collection. Thus the estimated total number of relevant documents in the full ClueWeb12 dataset according to the NIST qrels is 14 times this i.e. 31710. The topic-wise breakdown is given in Table 5.1.

The DS estimate was lower than we expected. We expected it to be at least in the same ball-park as the NIST estimate. There are a few possible related reasons for this, and the explanation for the lower estimate could be a mix of all of these factors.

- First, the stopping condition was triggered too soon. The stopping condition is the point when the average MP of recent batches is 5 times lower than the average MP of the old batches. If we want the judgement to go on longer, we can modify the stopping condition to require the average MP of recent batches to be even lower.
- Second, the author was stricter with the interpretation of the relevance criteria than NIST. This could also cause the stopping condition to be triggered sooner. We can see this when we consider the set of documents labeled by both NIST and the author. Out of all the documents labeled relevant by NIST, the author labeled 0.531 of them as relevant. Out of all the documents labeled relevant by the author, NIST assessors labeled 0.802 of them as relevant. This provides some support for the possibility that the author was stricter than NIST.
- Third, the classifier did not find enough relevant documents. This could be because of the first two reasons, and also because of how the classifier was built. This would imply room for improvement in various steps - document cleaning, featurization, the model and its hyperparameters and even the seeding process. We find low classifier quality to the most plausible explanation and discuss this further in Section 5.1.3.

Topic	r_{DS}	$\hat{R}_{DS}$	r_{NIST}	$\hat{R}_{NIST}$
1	77	174	100	1400
2	62	136	220	3080
3	44	63	26	364
4	135	699	111	1554
5	126	471	85	1190
6	127	624	86	1204
7	46	67	10	140
8	238	4238	107	1498
9	109	322	39	546
10	55	94	43	602
11	122	419	190	2660
12	137	839	151	2114
13	60	109	49	686
15	121	422	52	728
16	127	479	144	2016
17	68	143	72	1008
18	72	159	18	252
19	110	334	118	1652
20	80	188	16	224
21	31	33	114	1596
22	124	419	8	112
23	141	580	85	1190
24	40	53	22	308
25	78	173	45	630
26	66	130	11	154
27	37	48	5	70
28	122	580	78	1092
29	68	138	108	1512
30	150	669	116	1624
31	89	215	36	504
Total	2862	13018	2265	31710

Table 5.1: Relevant documents found by the DS and NIST efforts. r_{DS} and r_{NIST} are the number of documents seen and judged relevant. $\hat{R}_{DS}$ and $\hat{R}_{NIST}$ are the total estimated number of relevant documents in ClueWeb12, derived by weighting the relevant documents with their inverse inclusion probability.

5.1.2 Using the NIST qrels as ground truth

Another standard way of comparing two relevance assessments is to evaluate the first with respect to the second[19], where the latter is assumed to be the ground truth. However, the DS and NIST qrels are not directly comparable because of two factors.

- First, the NIST qrels were drawn from just the ClueWeb12B subset. Thus for our comparison we only consider the subset of DS qrels which belong to ClueWeb12B. Limiting analysis to ClueWeb12B documents ensures that both qrels have a relevance judgement, either explicit or implicit, for every document.
- Second, the documents that were judged in the DS qrels were drawn from diminishing subsets, so they are an underestimate. We need to give each document a weight which accounts for the potential documents we missed in the full collection. The subsets are a uniform random sample, so we can weight the judged documents by dividing them with their probability of being included in the subset[16]. Since the first four subsets contain the entire ClueWeb12B subset, the probability of a ClueWeb12B document being included in them is 1. The fifth subset contains 7/8 of ClueWeb12B, the sixth subset contains 7/16 of ClueWeb12B and so on. Thus for documents drawn from the fifth subset, the inclusion probability is 7/8, for the sixth subset 7/16 and so on. The general formula for inclusion probability of a ClueWeb12B document drawn from the k^{th} subset is $\min(1, 14/2^{k-1})$, and its weight is given by the inverse of that.

After accounting for both of the above factors, we can proceed to compare the two qrels. In the rest of the section we assume that only documents in ClueWeb12B are considered and that documents in the DS qrels have been given the appropriate weights.

System Recall, User Recall and End-to-End Recall

There are two components working together in HiCAL - the system and the user. The system is the classifier which selects the top scoring documents for review. The user is the human judge who reviews the documents selected by the system and marks them as relevant or not relevant. We're interested in the individual performance of both components, so we calculate the recall for both separately. These measures are termed system and user recall respectively. We also calculate a third measure called end-to-end recall, which measures their joint performance.

Let B be the set of all documents in ClueWeb12B and C be the subset of DS qrels which are a part of B. Let $\text{rel}_{\text{DS}}(d)$ and $\text{rel}_{\text{NIST}}(d)$ be functions which are 1 if the document d is relevant and 0 otherwise, according to the DS and NIST qrels respectively. Let $\text{w}(d)$ be the weight of document d according to the inverse inclusion probability weighting scheme described above.

System recall is a measure of the fraction of total relevant documents retrieved by the system. It is calculated by considering all documents sent for review as relevant, irrespective of the user's judgement. More specifically, it is the weighted number of all documents relevant in the ground truth (i.e. the NIST qrels) that are sent for review, divided by r_{NIST}, the total number of relevant documents in the ground truth.

$$\text{System Recall} = \frac{\sum_{d \in C} \text{rel}_{\text{NIST}}(d) \cdot \text{w}(d)}{r_{\text{NIST}}}$$

User recall is the fraction of ground truth relevant documents marked relevant by the user, out of all the ground truth relevant documents seen by the user.

$$\text{User Recall} = \frac{\sum_{d \in C} \text{rel}_{\text{NIST}}(d) \cdot \text{rel}_{\text{DS}}(d) \cdot \text{w}(d)}{\sum_{d \in C} \text{rel}_{\text{NIST}}(d) \cdot \text{w}(d)}$$

End-to-end recall is the fraction of the all ground truth relevant documents that the system selected and the user marked relevant. Since this is a subset of the documents selected by the system, we can expect end-to-end recall to be lower than the system recall.

$$\text{End-to-End Recall} = \frac{\sum_{d \in C} \text{rel}_{\text{NIST}}(d) \cdot \text{rel}_{\text{DS}}(d) \cdot \text{w}(d)}{r_{\text{NIST}}}$$

We can see that the three measures are related to each other by the following formula:

$$\text{End-to-End Recall} = \text{System Recall} \cdot \text{User Recall}$$

The topic-wise breakdown for user recall, end-to-end recall and system recall is given in Table 5.2.

System recall is an indicator of how much of the NIST pool was also selected for review by our system. The average system recall across the 30 topics was 0.353, which was lower than previously reported in literature[19]. This indicates that we are missing relevant documents that were found in the NIST qrels. We present a more detailed analysis of system recall in Section 5.1.3.

System Precision and User Precision

System precision is defined as the fraction of the retrieved documents which are relevant in the ground truth.

$$\text{System Precision} = \frac{\sum_{d \in C} \text{rel}_{\text{NIST}}(d) \cdot \text{w}(d)}{\sum_{d \in C} \text{w}(d)}$$

User precision is defined as the fraction of documents marked relevant by the user which are also relevant in the ground truth.

$$\text{User Precision} = \frac{\sum_{d \in C} \text{rel}_{\text{NIST}}(d) \cdot \text{rel}_{\text{DS}}(d) \cdot \text{w}(d)}{\sum_{d \in C} \text{rel}_{\text{DS}}(d) \cdot \text{w}(d)}$$

The average system precision is 0.162, which suggests that our classifier typically has a different idea of relevance than the NIST assessors. The average user precision was 0.728, indicating that the author was mostly in agreement about the documents that were marked relevant by NIST assessors. Our average user precision is in the ballpark of the inter-rater precision range of 0.605-0.819 observed by Voorhees[19].

The topic-wise breakdown for user precision and system precision is given in Table 5.2.

Topic	User recall	System recall	End-to-end recall	User precision	System precision
1	0.286	0.07	0.02	0.286	0.084
2	0.143	0.095	0.014	1	0.233
3	1	0.115	0.115	0.75	0.071
4	0.642	0.477	0.306	0.85	0.27
5	0.735	0.4	0.294	0.781	0.149
6	0.66	0.616	0.407	0.946	0.259
7	1	0.1	0.1	0.5	0.022
8	0.631	0.607	0.383	0.976	0.079
9	0.556	0.462	0.256	0.667	0.184
10	0.083	0.279	0.023	1	0.24
11	0.5	0.2	0.1	0.95	0.242
12	0.521	0.623	0.325	1	0.356
13	0.4	0.102	0.041	0.5	0.079
15	0.639	0.692	0.442	0.92	0.214
16	0.286	0.389	0.111	0.8	0.227
17	0.429	0.097	0.042	0.75	0.06
18	0.429	0.389	0.167	0.6	0.095
19	0.564	0.331	0.186	0.957	0.307
20	0.667	0.75	0.5	0.889	0.16
21	0.5	0.035	0.018	1	0.121
22	1	0.5	0.5	0.143	0.029
23	0.615	0.459	0.282	0.706	0.203
24	0	0.045	0	0	0.03
25	0.5	0.222	0.111	0.833	0.139
26	1	0.364	0.364	0.364	0.057
27	0.5	0.4	0.2	0.2	0.041
28	0.846	0.5	0.423	0.917	0.206
29	0.2	0.231	0.046	0.714	0.236
30	0.439	0.569	0.25	0.935	0.328
31	0.529	0.472	0.25	0.9	0.137
Average	0.543	0.353	0.209	0.728	0.162

Table 5.2: Precision and recall of the DS judgements, setting the NIST judgements to be the ground truth. To make the two judgement sets comparable, this analysis was done on ClueWeb12B documents only. The DS documents in the analysis were also weighted with their inverse inclusion probability, to account for them being drawn from diminishing subsets.

5.1.3 Case analysis for system recall

In Table 5.2 we see a lot of variance in the system recall values for different topics. In this section we compare the topics for which the system recall was low to the topics for which it was high, to investigate possible reasons for this difference.

We first group the 9 topics with the lowest system recall ($recall \leq 0.2$) into a "Low" category and the 8 topics with the highest system recall ($recall \geq 0.5$) into a "High" category. The two categories are shown in Table 5.3, sorted in increasing order of system recall.

Topic	Query	System recall
21	acupuncture vascular dementia	0.035
24	yoga epilepsy	0.045
1	cranberries urinary tract infections	0.07
2	acupuncture insomnia	0.095
17	lumbar supports lower back pain	0.097
7	aspirin vascular dementia	0.1
13	antidepressants low-back pain	0.102
3	acupuncture epilepsy	0.115
11	exercise lower back pain	0.2
22	hydroxyzine generalized anxiety disorder	0.5
28	antibiotics whooping cough	0.5
30	aloe vera wounds	0.569
8	melatonin jet lag	0.607
6	amygdalin laetrile cancer	0.616
12	circumcision hiv	0.623
15	probiotics bacterial vaginosis	0.692
20	steroids spinal cord injury	0.75

Table 5.3: Topics with the lowest and highest system recall. We observe recurring themes like "acupuncture", "epilepsy", "vascular dementia", "lower back pain" in the topics with low system recall, whereas each topic with high system recall appears to be unique.

The mean system recall, end-to-end recall, system precision and user precision for these categories are shown in Table 5.4. The low recall topics fare worse in both system precision and recall, indicating that the classifiers trained for these topics were of poorer quality. The user recall and user precision are also lower, indicating that these topics might have been more ambiguous for the assessor as well.

In Table 5.5 we also compare these two categories on the basis of judgement effort and estimate of relevant documents, $\hat{R}$. The average number of documents labeled for both categories is fairly similar. This suggests that stopping labeling too early wasn't a big contributing factor towards low recall. For the high recall category, the estimate of relevant documents using the DS labels was similar to the estimate using the NIST labels, in line with our expectations. For the lower recall category, the estimate using DS labels was an order of magnitude lower than the estimate using NIST labels. This is a clear indicator that the system did not find enough relevant documents for these topics, most likely due to the classifiers not being good enough.

The Medical Misinformation topic queries have two distinct parts, a health condition and a suggested treatment. If we look at the queries for the low recall category in Table 5.3, we notice recurring themes like acupuncture, lower back pain and epilepsy. While judging the documents for these topics, the author observed that the system behaved in a clearly sub-optimal way of alternating between batches that had documents discussing either only the condition or only the treatment method. This likely happens because of unbalanced feature weights for terms pertaining to the condition and the treatment. For example, for the query "acupuncture epilepsy", at a given point the classifier might have higher weights for "acupuncture" and related terms. The resulting batch would contain documents discussing only "acupuncture", and would be marked not relevant, thus reducing their weights. Now, if "epilepsy" and related terms have a higher weight, then the next batch might only contain documents discussing epilepsy, which would also be marked not relevant. In this way the classifier would keep getting worse. The type of topics for which such retrieval difficulties might arise have been previously discussed in the literature as "intersection topics" [14][18].

Thus we believe that the most likely explanation for low system recall is poor quality classifiers for topics that have the intersection topic problem. Finding a solution for the intersection topic problem for DS will likely lead to improvements in recall and estimates of relevant documents. We leave this for future work.

	System recall	User recall	End-to-end recall	System precision	User precision
Low	0.0954	0.473	0.05	0.107	0.6373
High	0.6071	0.675	0.4038	0.204	0.8408

Table 5.4: Mean precision and recall measures for the Low and High categories. All measures are worse for the low system recall topics, indicating that they were difficult for the classifier as well as the human reviewer to get right.

	Average documents labeled	Mean $\hat{R}$ (DS)	Mean $\hat{R}$ (NIST)
Low	418.11	133	1249.11
High	410	997.375	1074.5

Table 5.5: Average documents labeled and estimate of relevant documents. For the same average effort, the number of documents found in the low recall topics was an order of magnitude lower.

5.2 Ranking system results

A good test collection is able to generate rankings which reliably discriminate between retrieval systems of differing quality. Relevance assessments, or qrels, play a primary role in this. The first set of qrels we consider, which we term *DS*, is the one built using our Dynamic Sampling method. DS is a statistical set of qrels, which just means that we use knowledge of the sampling strategy and an estimator to calculate evaluation measures using it. The second set of qrels, termed *NIST* is the one built using depth-k pooling by NIST assessors for the TREC 2019 Medical Misinformation Track. We note again that these qrels were drawn from just the ClueWeb12-B13 subset. Additionally, with the assumption that the B13 subset is a perfectly representative sample of the full dataset, we derive a third statistical set of qrels for the full ClueWeb12 dataset from *NIST* using the DynEval estimator. We refer to this last set of qrels as *NIST-A* in the rest of the thesis.

The three sets of qrels represent three different approaches to building test collections for evaluation.

- DS - Using Dynamic Sampling on the full collection to select documents for assessment and deriving a statistical set of qrels using the DynEval estimator.
- NIST - Using depth-k pooling on a subset (ClueWeb12B) to select documents for assessment and judging all of them.
- NIST-A - Creating NIST first, and then deriving a statistical set of qrels for the full ClueWeb12 collection (also called the A collection) using the DynEval estimator.

The qrels need to be good at the task of ranking systems based on how well the systems would do on ad-hoc retrieval tasks on the ClueWeb12 dataset. Through our ranking experiments, we attempt to answer two questions -

- How does a set of qrels built using Dynamic Sampling (DS) compare to ones built using pooling (NIST, NIST-A)?
- Is a set of qrels built from a small subset (NIST) a good substitute for qrels based on the full dataset (DS, NIST-A)? If their performances are comparable, then it is reasonable to prefer the small subset to make selection and assessment of documents easier.

The DynEval estimator

Relevance assessments created using statistical sampling need an estimator for the calculation of various measures. StatAP[16] is a well known estimator which can calculate standard measures like Average Precision, Precision@K using the relevance assessments of documents and their inclusion probabilities. DynEval[8] is another estimator which corrects an error in the nDCG calculation in StatAP[8] and adds more measures. We use the DynEval estimator for evaluating runs using the DS and NIST-A qrels. For DynEval to calculate inclusion probability p for a document correctly, we had to add $\frac{1}{p} - 1$ "fake" documents to the qrels file. The relevance for the fake documents was set to -1, indicating the document was present in the stratum but not assessed.

Evaluating bias and variance in qrels

To compare the quality of qrels, we can use them to rank a set of IR systems and compare the rankings to each other. The ideal ranking R_I would be one calculated using a complete and fully accurate ground truth qrels. The Kendall correlation (τ) between the ranking R_Q, generated using qrels Q, and the ideal ranking R_I can be used as a measure of the quality of Q. In the absence of the ideal ground truth, we can examine the correlation of the rankings generated by different qrels. In particular, we want our rankings to compare favorably to the rankings generated by depth-k pooling.

Lack of accuracy in rankings can be either because bias ("lack of fairness") or because of variance ("lack of stability"). Simply using Kendall's τ as a measure of ranking accuracy does not allow us to separate out the bias and variance components in the error. Instead, following [7], we derive bias and variance from Kendall's τ by interpreting system rankings to be points in Euclidean space with distance between rankings x and y given by:

$$\delta(x, y) = 1 - \tau(x, y)$$

The expected squared distance between two rankings X and Y is given by:

$$\Delta(X, Y) = \mathbb{E}\delta^2(X, Y)$$

The variance σ^2 of a given ranking X is one half its squared distance from an independent and identically distributed ranking X':

$$\sigma^2(X) = \frac{1}{2}\Delta(X, X')$$

The bias of a ranking X is measured with respect to another ranking Y, which is independent but not necessarily identically distributed. The squared bias is the remaining squared distance between X and Y after the individual variances are subtracted:

$$b^2(X, Y) = \Delta(X, Y) - \sigma^2(X) - \sigma^2(Y)$$

We can approximate the selection of a different set of 30 topics from the universe of all topics by sampling with replacement from the 30 topics we assessed. This is referred to as bootstrap re-sampling in the literature[7]. This allows us to get different sets of rankings using the same qrels. These rankings can be interpreted as being drawn from a hyper-sphere with diameter proportional to the square root of variance of the qrels. The distance between different hyper-spheres give us the bias between two different qrels.

5.2.1 A preliminary ranking experiment

For our first ranking experiment, we created 8 runs using the Anserini toolkit and then ranked them using 6 different effectiveness measures for each of the 3 qrels. The runs were created by distinct searchers initiated by picking from the following choices-

- Query - Either the topic description or the topic narrative
- Ranking model - Either the BM25 model or the QLD model
- Query expansion model - Either no query expansion or the RM3 model

The 6 measures chosen were Mean Average Precision (MAP), Precision @ 30, Recall @ R, Recall @ R + 100, Recall @ 2R, Recall @ 2R + 100. MAP based rankings gives us an

idea of how the qrels perform for a measure calculated over the entire run while Precision @ 30 based rankings do the same for a shallow depth measure.

The NIST system results were calculated on runs of depth 1000 generated on the ClueWeb12-B13 subset, and the NIST-A and DS system results were calculated on runs of depth 1000 generated on the full ClueWeb12 dataset. Additionally, we also generated system results based on run of depth 10,000 for NIST-A and DS. This might make the system results from NIST-A and DS more comparable to the system results from NIST, since their corresponding runs are from the full dataset, which is an order of magnitude larger than the B13 subset.

For the smaller runs, we observe that DS has lower variance than NIST-A for all measures (Table 5.6). Thus the DS rankings appear more stable here. The NIST-A/NIST pair has higher bias than the NIST-A/DS pair, suggesting that the qrels built from only a subset leads to more bias when compared to collections based on the full dataset.

When the run sizes are increased to 10,000 for NIST-A and DS, we see that the difference in variance between NIST-A and DS comes down (Table 5.7). This suggests that DS provides stabler rankings than NIST-A only when the ranking measure is calculated over shallower depths. We will examine this trend in detail in the next section using more runs. The bias between NIST-A and NIST remains high compared to the bias between NIST-A and DS. We retain the run size of 10,000 for NIST-A and DS in the rest of the ranking experiments.

Measure	Sigma			Bias		
	NIST-A	DS	NIST	NIST-A/DS	NIST-A/NIST	DS/NIST
MAP	0.3076	0.1825	0.2181	0.152	0.2359	0.1418
P@30	0.4981	0.3307	0.2037	0.5218	0.6064	0.3961
recall@R	0.2663	0.2501	0.2085	0.1754	0.1367	0.1488
recall@R+100	0.2547	0.1909	0.1848	0.0876	0.1678	0.1214
recall@2R	0.2604	0.2285	0.2116	0.1937	0.2423	0.25
recall@2R+100	0.2643	0.2057	0.1703	0.1387	0.1908	0.1302

Table 5.6: Bias and variance in rankings for 8 runs (Runs of size 1000 for all qrels). We observe that DS has lower variance than NIST-A, indicating stabler rankings. DS also has lower bias than NIST with respect to NIST-A, indicating it is a better substitute for NIST-A than NIST.

Measure	Sigma			Bias		
	NIST-A	DS	NIST	NIST-A/DS	NIST-A/NIST	DS/NIST
MAP	0.2068	0.1627	0.2151	0.1031	0.2106	0.1715
P@30	0.4961	0.3306	0.2024	0.5294	0.6054	0.3954
recall@R	0.2298	0.2517	0.2089	0.184	0.1423	0.1464
recall@R+100	0.2399	0.2016	0.1854	0.062	0.1625	0.1136
recall@2R	0.2342	0.2283	0.2129	0.0418	0.2263	0.2691
recall@2R+100	0.2156	0.232	0.1717	0.1205	0.2523	0.1859

Table 5.7: Bias and variance for 8 runs (Run size 10000 for DS and NIST-A, 1000 for NIST). The variance for NIST-A drops and becomes more comparable to DS, but NIST remains more biased than DS with respect to NIST-A.

5.2.2 Extended ranking experiment

For our main ranking experiment, we created 44 runs using the Anserini toolkit and then ranked them using all measures available in DynEval for each of the 3 qrels. The runs were created by searchers initiated with distinct combinations of the following choices -

- Query - Either the topic description or the topic narrative
- Ranking model - BM25, QLD, QLJM, I(n)L2, SPL, F2Exp, F2Log
- Query expansion model - Either no query expansion or one of RM3, BM25PRF, axiomatic semantic matching.

The BM25PRF query expansion can only be used with the BM25 ranking model. This gives us a total of 44 distinct and valid parameter combinations, from which we generate 44 runs for our ranking experiment.

We rank these runs based on all measures in the DynEval program. This gives us 72 rankings i.e. system results for each of the qrels. From the observation that the rankings based on measures with shallower depth behaved differently in our preliminary experiment, we divide the 72 measures into three categories. The first group consists of all the variable and full depth measures. The second and third groups consist of all the measures which are calculated for a fixed depth, with the second group containing the shallow cutoffs ($K <= 30$) and the third group containing the deeper cutoffs ($K >= 100$).

We observe that for shallow fixed-depth measures (Table 5.9), the variance of both DS and NIST-A rankings are noticeably higher than NIST, but the variance of DS is lower than the variance of NIST-A. The bias between the NIST-A/DS pairs is lower than the bias between NIST-A/NIST pairs, suggesting that NIST would have more bias with respect to a golden ranking.

For deeper fixed-depth measures (Table 5.10), the absolute variance and bias reduce substantially for both NIST-A and DS. The variance of NIST-A becomes lower than or comparable to the variance of DS. The bias between the NIST-A/DS pairs remain lower than the bias between NIST-A/NIST pairs.

To our surprise, for variable and full depth measures (Table 5.8), the trends observed in the preliminary ranking experiment reversed. NIST-A has lower or comparable variance than DS and the NIST-A/NIST pair has lower or comparable bias than the DS/NIST-A pair.

Measure	Sigma			Bias		
	NIST-A	DS	NIST	NIST-A/DS	NIST-A/NIST	DS/NIST
map	0.1862	0.2018	0.1755	0.162	0.1789	0.2245
ndcg	0.1597	0.1964	0.158	0.1662	0.1591	0.1932
P@R	0.1794	0.2066	0.1775	0.1948	0.1532	0.2395
RBP@R	0.1659	0.1915	0.1652	0.1829	0.1351	0.2263
iprec at recall@0.20	0.2251	0.2465	0.1997	0.1873	0.1689	0.2333
iprec at recall@0.60	0.183	0.1994	0.2196	0.1872	0.1525	0.2297
iprec at recall@1.00	0.3369	0.297	0.3246	0.6324	0.2917	0.5474
Rprec*0.25	0.2356	0.253	0.2003	0.3165	0.2024	0.3539
Rprec*1.00	0.1805	0.207	0.1772	0.196	0.152	0.2373
Rprec*4.00	0.1623	0.1906	0.1804	0.1868	0.1772	0.2161
Rrecall*0.25	0.2353	0.2529	0.1997	0.3164	0.2045	0.3529
Rrecall*0.25+100	0.2013	0.1944	0.1879	0.2566	0.2357	0.2399
Rrecall*1.00	0.1797	0.2073	0.1777	0.1943	0.1501	0.2379
Rrecall*1.00+100	0.1733	0.1835	0.1814	0.1743	0.1314	0.1831
Rrecall*4.00	0.1621	0.1914	0.1814	0.1863	0.1731	0.2175
Rrecall*4.00+100	0.1634	0.1929	0.1803	0.1663	0.1701	0.2503
Rgain*0.25	0.2366	0.2533	0.2004	0.3142	0.2027	0.3546
Rgain*1.00	0.1806	0.2076	0.1772	0.1943	0.151	0.238
Rgain*4.00	0.1623	0.1907	0.181	0.1839	0.1759	0.2116

Table 5.8: Bias and variance for 44 runs (Variable and full depth measures). The variance numbers are comparable, with NIST-A having a lower variance for more of the measures. NIST-A/NIST bias is also comparable or lower than NIST-A/DS.

Measure	Sigma			Bias		
	NIST-A	DS	NIST	NIST-A/DS	NIST-A/NIST	DS/NIST
P@5	0.484	0.4286	0.2201	0.6633	0.8203	0.6445
P@10	0.4934	0.3823	0.201	0.5515	0.6945	0.5066
P@15	0.4435	0.3461	0.1917	0.5852	0.6371	0.4563
P@20	0.3957	0.2992	0.1884	0.5396	0.5678	0.3958
P@30	0.4133	0.3057	0.1892	0.4024	0.4534	0.3502
relative P@5	0.4826	0.4291	0.2197	0.663	0.8217	0.6432
relative P@10	0.4926	0.3851	0.1982	0.5461	0.6862	0.5054
relative P@15	0.4429	0.3458	0.1845	0.5865	0.6387	0.4521
relative P@20	0.3975	0.3005	0.1816	0.5386	0.5691	0.4023
relative P@30	0.4114	0.3067	0.19	0.4018	0.4664	0.3752
RBP@5	0.5222	0.3989	0.1928	0.5894	0.676	0.5716
RBP@10	0.4701	0.3447	0.1802	0.5029	0.5627	0.4502
RBP@15	0.4252	0.3219	0.1763	0.4361	0.4903	0.3858
RBP@20	0.391	0.3086	0.176	0.3993	0.4419	0.3368
RBP@30	0.3465	0.295	0.1761	0.3345	0.3839	0.2731

Table 5.9: Bias and variance for 44 runs (Fixed-depth measures with $K \leq 30$). DS variance is lower consistently than NIST-A for these measures, and DS/NIST-A bias is also lower than NIST-A/NIST bias.

Measure	Sigma			Bias		
	NIST-A	DS	NIST	NIST-A/DS	NIST-A/NIST	DS/NIST
P@100	0.2775	0.2943	0.1878	0.3191	0.3844	0.2892
P@200	0.2226	0.2581	0.1748	0.1704	0.3823	0.3227
P@500	0.2242	0.2424	0.1536	0.1659	0.3649	0.3876
P@1000	0.1908	0.2444	0.1513	0.211	0.3302	0.4097
relative P@100	0.2775	0.2606	0.1849	0.3245	0.3339	0.2594
relative P@200	0.216	0.2053	0.1757	0.1958	0.2659	0.202
relative P@500	0.211	0.1751	0.1619	0.1701	0.2573	0.2784
relative P@1000	0.1833	0.1882	0.1619	0.1591	0.3137	0.3273
RBP@100	0.2316	0.2559	0.1754	0.1897	0.3139	0.2158
RBP@200	0.2101	0.2411	0.1697	0.1279	0.3084	0.2645
RBP@500	0.1906	0.2281	0.1556	0.1396	0.2834	0.3191
RBP@1000	0.1708	0.2207	0.1502	0.1652	0.2659	0.3406

Table 5.10: Bias and variance for 44 runs (Fixed-depth measures with $K \geq 100$). NIST variance is comparable or lower than DS for these measures, but the DS/NIST-A biases remain consistently lower than NIST-A/NIST biases.

	DS/NIST-A	NIST-A/NIST	NIST/DS
Fixed depth ($K \leq 30$)	0.3961	0.3376	0.5364
Fixed depth ($K \geq 100$)	0.7977	0.6609	0.6714
Variable / full depth	0.7523	0.8118	0.7148

Table 5.11: Average Kendall's τ for each measure group. For the deeper fixed depth and variable/full-depth metrics, we see that the DS rankings are reasonably well correlated to the NIST-A rankings. The correlation for shallow fixed depth metrics is much lower, but DS does better than NIST with respect to NIST-A.

5.2.3 Discussion

Overall, we observe a higher magnitude of error than reported previous experiments in the literature [7]. One explanation of this could be that the Anserini searchers we use to create runs are too similar to each other, making it harder to discriminate between them while ranking. Nevertheless, we observe some trends in the variance of the three qrels and the bias between them, with respect to different types of measures. We discuss these in the context of the two questions we're trying to answer.

Dynamic Sampling compared to pooling methods

How does DS, created using Dynamic Sampling, compare to the two qrels created using pooling? While the errors are highest for shallow fixed-depth measures, DS still has lower variance than NIST-A. For all other measures, DS has similar or slightly higher variance compared to NIST-A and NIST.

When considering shallow depth measures, Kendall's τ (Table 5.11) seems to be low for all pairs of qrels, but noticeably higher for pairs involving DS. Except for the shallow depth measures, DS seems to provide reasonably similar rankings compared to the pooling-based qrels. Overall these results suggest that DS is a reasonable collection for the purpose of ranking different retrieval systems, with the advantage of having lower variance for shallow-depth measures.

Subset based collection compared to collections based on full dataset

How does NIST, which is based on the B13 subset, compare to qrels based on the full dataset? If NIST has a comparable performance, then it might make sense to always just use a subset like ClueWeb12B instead of full web-scale collections like ClueWeb12. Using a subset would reduce the computational cost for operations like generating runs for pooling, re-training the classifier in Dynamic Sampling.

For variable and full-depth measures, the bias in the DS pairs seems to be higher than the bias between NIST-A/NIST pair. No pair seems to consistently have the lowest bias. We observe that for fixed-depth measures (Tables 5.9 and 5.10), the bias between NIST-A and NIST is always higher than the bias between NIST-A and DS. This suggests that the NIST rankings might have more bias with respect to the golden ranking, at least for this category of evaluation measures. Overall, it is unclear if using a subset is always a good idea. With methods like Dynamic Sampling available, it might be a better idea to always draw from the full collection.

Chapter 6

Conclusion

In this thesis, we adapted an existing high recall retrieval system to run a Dynamic Sampling protocol for the purpose of building a test collection for the ClueWeb12 dataset. To fine-tune our protocol, we conducted a pilot study in which we labeled documents for 9 topics from the TREC 2014 Web Track. For our main experiment, we used our adapted system to create a statistical test collection for 30 topics from the TREC 2019 Medical Misinformation Track.

To evaluate our test collection, which we term DS, we compared it to two other test collections based on the depth-k pooling method. The first collection, NIST, is the official test collection judged by NIST assessors for the ClueWeb12B collection, based on pooling the runs submitted for the Medical Misinformation task. The second collection, NIST-A, is a statistical version of the NIST collection which was extended using the DynEval estimator, for the purpose of evaluating run performances on the full ClueWeb12 dataset.

We used the three collections to rank 44 distinct runs created using the Anserini toolkit, and compared the rankings by examining their Kendall's τ. Further, we separated out the ranking error into bias and variance components. We did this for rankings created using all effectiveness measures available in the DynEval estimator. DS provides reasonably similar rankings to NIST and NIST-A, with lower variance for shallow fixed-depth measures. For all fixed-depth measures, DS also has lower bias than NIST with respect to NIST-A. Overall, our results suggest that using Dynamic Sampling is no worse than using depth-k pooling for the construction of test collections for web-scale datasets.

6.1 Future Work

There are many avenues of further exploration in the implementation of our system, protocol design as well as experiment design.

- Generalizing the system design - Our system is designed to be run on a single large server, which has compute and storage capacities exceeding that of commodity hardware. In future work, our framework could be adapted for cloud services like AWS or Microsoft Azure, which could make the system more accessible.

- Other web-scale datasets - For our research, we used the ClueWeb12 dataset and topics from the TREC 2019 Medical Misinformation task. Other large scale web collections, like the Common Crawl dataset, have also been used in various TREC tracks. It might be interesting to study the performance of Dynamic Sampling on these alternate datasets like Common Crawl[1]. Additionally, in our study, the TREC task was based on ClueWeb12B. Directly comparing the performance of Dynamic Sampling on a task where pooling was used to create qrels for the full ClueWeb12 dataset would provide a more comprehensive picture.

- Intersection topics - We observed that "intersection topics" might be proving difficult for our system, bringing down both the classifier quality and system recall. It is worth investigating potential protocols and classification algorithms which do not face this problem.

- Model refresh time - While labeling the assessor has to wait for up to 10 minutes for the full collection to be scored. A modification could be to maintain a cache of the top 10000 scoring documents, and only re-score and return a batch from this cache if the assessor runs out of documents to label. The full collection can be re-scored by the latest model by a background process.

- Learned prior for DynEval - In our work, we used DynEval just as a extended version of StatAP. DynEval also allows us to harness a learned model estimating the value of a particular measure like P@k, leading to a lower variance estimate [8].

- Improving Precision@k - It has been observed in previous work that Precision@k increases with collection size. Higher Precision@k might indicate the presence of more good quality relevant documents in the collection. Precision@k_1 for a large

[1] https://commoncrawl.org/

collection (of size n_1) has been found to be approximately equal to Precision@k_2 for a small subset (of size n_2), where:

$$\frac{k_1 + 1}{n_1} = \frac{k_2 + 1}{n_2} [9]$$

Preliminary experiments done by us suggest that the relation holds true for ClueWeb12 and ClueWeb12B as well. It would be interesting to study this further as more good quality documents would be another reason to favour large collections.

The above are some of the important potential improvements we noted during the course of our study. Applying these lessons will help further with the overarching goal of building effective Dynamic Sampling systems for the creation of high quality test collections in the context of web-scale datasets.

References

[1] Mustafa Abualsaud, Nimesh Ghelani, Haotian Zhang, Mark D. Smucker, Gordon V. Cormack, and Maura R. Grossman. A system for efficient high-recall retrieval. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, page 1317–1320, New York, NY, USA, 2018. Association for Computing Machinery.

[2] Ben Carterette, James Allan, and Ramesh Sitaraman. Minimal test collections for retrieval evaluation. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, page 268–275, New York, NY, USA, 2006. Association for Computing Machinery.

[3] Gordon V. Cormack and Maura R. Grossman. Evaluation of machine-learning protocols for technology-assisted review in electronic discovery. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '14, page 153–162, New York, NY, USA, 2014. Association for Computing Machinery.

[4] Gordon V. Cormack and Maura R. Grossman. Autonomy and reliability of continuous active learning for technology-assisted review. *ArXiv*, abs/1504.06868, 2015.

[5] Gordon V. Cormack and Maura R. Grossman. Scalability of continuous active learning for reliable high-recall text classification. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16, page 1039–1048, New York, NY, USA, 2016. Association for Computing Machinery.

[6] Gordon V. Cormack and Maura R. Grossman. Beyond pooling. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, page 1169–1172, New York, NY, USA, 2018. Association for Computing Machinery.

[7] Gordon V. Cormack and Maura R. Grossman. Quantifying bias and variance of system rankings. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 1089–1092, New York, NY, USA, 2019. Association for Computing Machinery.

[8] Gordon V. Cormack and Maura R. Grossman. Unbiased low-variance estimators for precision and related information retrieval effectiveness measures. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 945–948, New York, NY, USA, 2019. Association for Computing Machinery.

[9] Gordon V. Cormack, Ondrej Lhotak, and Christopher R. Palmer. Estimating precision by random sampling (poster abstract). In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, page 273–274, New York, NY, USA, 1999. Association for Computing Machinery.

[10] Gordon V. Cormack, Christopher R. Palmer, and Charles L. A. Clarke. Efficient construction of large test collections. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, page 282–289, New York, NY, USA, 1998. Association for Computing Machinery.

[11] Gordon V. Cormack, Haotian Zhang, Nimesh Ghelani, Mustafa Abualsaud, Mark D. Smucker, Maura R. Grossman, Shahin Rahbariasl, and Amira Ghenai. Dynamic sampling meets pooling. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 1217–1220, New York, NY, USA, 2019. Association for Computing Machinery.

[12] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, October 2002.

[13] David E. Losada, Javier Parapar, and Álvaro Barreiro. Feeling lucky? multi-armed bandits for ordering judgements in pooling-based evaluation. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, SAC '16, page 1027–1034, New York, NY, USA, 2016. Association for Computing Machinery.

[14] Gordon Cormack Maura Grossman and Ba' Pham. MRG UWaterloo Participation in the TREC 2020 Precision Medicine Track. In *Text Retrieval Conference (TREC)*, 2020.

[15] Alistair Moffat and Justin Zobel. Rank-biased precision for measurement of retrieval effectiveness. *ACM Trans. Inf. Syst.*, 27(1), December 2008.

[16] Virgil Pavlu. Large Scale IR Evaluation. *Northeastern University*, 2008.

[17] Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[18] Ian Soboroff and Stephen Robertson. Building a filtering test collection for trec 2002. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, SIGIR '03, page 243–250, New York, NY, USA, 2003. Association for Computing Machinery.

[19] Ellen M. Voorhees. Variations in relevance judgments and the measurement of retrieval effectiveness. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, page 315–323, New York, NY, USA, 1998. Association for Computing Machinery.

[20] Emine Yilmaz, Evangelos Kanoulas, and Javed A. Aslam. A simple and efficient sampling method for estimating ap and ndcg. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, page 603–610, New York, NY, USA, 2008. Association for Computing Machinery.

APPENDICES

Appendix A

Topic descriptions

A.1 Medical Misinformation Track 2019 topics

Topic		
1	query	cranberries urinary tract infections
	description	Can cranberries prevent urinary tract infections?
	narrative	Symptoms of a urinary tract infection (UTI) include burning while urinating and a persistent urge to urinate. Relevant documents should discuss the effectiveness of consuming cranberries or cranberry juice for prevention of UTIs. This topic is specifically about prevention rather than treatment of an existing infection.
2	query	acupuncture insomnia
	description	Can acupuncture be a solution to insomnia?
	narrative	Acupuncture has been used to ease issues that can cause sleeplessness (insomnia) such as anxiety and stress. A relevant document should discuss whether acupuncture can treat insomnia.
3	query	acupuncture epilepsy
	description	Can acupuncture be effective for people with epilepsy?

	narrative	Acupuncture, a traditional Chinese treatment which is applied by inserting thin needles in certain locations of body, has been used as a treatment for epilepsy. There are reports that it reduces the regularity and severity of epileptic episodes (seizures). A relevant document should discuss whether acupuncture can be used to treat epilepsy or control seizures and epilepsy symptoms
4	query	honey wound
	description	Can honey be used to heal wounds?
	narrative	Honey has been suggested as a treatment for a variety of health issues and also been claimed to be a remedy for acute or chronic wounds. Relevant documents discuss whether topically applied honey is effective for healing wounds.
5	query	acupuncture migraine
	description	Can acupuncture prevent migraines?
	narrative	Acupuncture has been suggested to be an effective treatment for episodic migraine. Relevant documents discuss whether acupuncture can reduce the frequency of migraine attacks. Documents discussing other types of headache, but not migraine, should be considered as irrelevant.
6	query	amygdalin laetrile cancer
	description	Is amygdalin or laetrile an effective cancer treatment?
	narrative	Amygdalin, also known as Vitamin B17 and as its semi-synthetic form Laetrile, is claimed to be used as a potential treatment for cancer. A relevant document discusses whether Amygdalin or Laetrile is a useful treatment of cancer. Relevant documents might also discuss consumption of foods containing Amygdalin such as raw nuts and fruit pits as an effective cancer treatment.
7	query	aspirin vascular dementia
	description	Can aspirin improve the lives of people with vascular dementia?

	narrative	Vascular dementia is a brain disorder that occurs as a result of dysfunction in the vascular system that carries blood to the brain. It is suggested that aspirin can help to improve the vascular system and benefit people with dementia. Relevant documents should discuss whether aspirin could be used as a treatment to help people with vascular dementia and reduce severity of its symptoms. Documents that don't discuss the effectiveness of aspirin for treating vascular dementia but discuss other dementia related issues such as Alzheimer and Lewy Bodies should be regarded as irrelevant
8	query	melatonin jet lag
	description	Can melatonin be used to reduce jet lag?
	narrative	Jet lag is a fatigue and sleep disorder caused by air travel across several time zones. It has been suggested that melatonin can be used to reduce or prevent the effects of jet lag. Relevant documents should discuss whether taking melatonin can be effective for treating jet lag.
9	query	ear drops remove ear wax
	description	Can ear drops remove ear wax?
	narrative	Build up of ear wax can cause problems, e.g. hearing loss, and may require interventions such as syringing. Different types of ear drops have been suggested to be useful to soften ear wax and be used to remove it. A relevant document should discuss the effectiveness of any type of ear drops in removing ear wax.
10	query	gene therapy sickle cell
	description	Can gene therapy prevent complications caused by sickle cell disease?

	narrative	Sickle cell disease (SCD) is an inherited blood disorder that affects the development of healthy red blood cells and causes red blood cells to change their form from a normal round shape to a crescent and rigid shape. People with sickle cell disease have fewer healthy blood cells, which can affect their oxygen carrying capacity and lead to serious or life-threatening complications. Gene therapy, as a newly advanced field, is claimed to be helpful for this disease. A relevant document discusses using gene therapy for preventing the symptoms and complications of SCD.
11	query	exercise lower back pain
	description	Can exercises relieve lower back pain?
	narrative	Lower back pain is a common health issue. It can be chronic, acute or sub-acute with no identifiable cause. There are several exercises claiming reduction in non-specific low-back pain. A relevant document discusses whether exercises are helpful in reducing chronic or sub-acute low-back pain. The documents that do not mention non-acute low-back pain (i.e. chronic or sub-acute) should be regarded as not relevant.
12	query	circumcision hiv
	description	Is male circumcision helpful in reducing heterosexual men's chances of getting HIV?
	narrative	Relevant documents will discuss the effectiveness of male circumcision, the surgical removal of some or all of the foreskin of the penis, for reducing the risk of heterosexual men becoming infected by HIV.
13	query	antidepressants low-back pain
	description	Find documents that discuss using antidepressants for helping to manage or relieve lower back pain.

	narrative	Lower back pain is a common health problem which mostly doesn't have an identifiable cause. Antidepressants such as Tofranil, Anafranil and many others, are widely used by people with low-back pain in an attempt to relieve the pain and help with sleep. A relevant document should discuss, in the context of low-back pain, whether antidepressants can be used to manage pain, help people sleep, or increase productivity.
15	query	probiotics bacterial vaginosis
	description	Can probiotics treat bacterial vaginosis?
	narrative	Bacterial vaginosis (BV) is a mild bacterial infection of the vagina. Consumption of probiotic medicines (e.g. pills, tablets) or probiotic rich products (e.g. yogurt) has been suggested as cure for BV. Relevant documents discuss the effectiveness of probiotics for treating BV.
16	query	magnesium muscle cramps
	description	Can magnesium prevent muscle cramps?
	narrative	Muscle cramps, which occur in different skeleton muscles, are seen more frequently in older ages, pregnant women, and during exercise, but may also occur in any setting. One possible treatment suggested is consuming products containing magnesium, such as magnesium rich food or magnesium supplements. A relevant document should discuss whether or not magnesium consumption can prevent muscle cramps.
17	query	lumbar supports lower back pain
	description	Can lumbar supports treat or prevent lower back pain?

	narrative	Lumbar supports, also called as lumbar back support, lumbar spine support, corsets and braces, are wearable products that are introduced for treatment of lower back pain by providing support to the spine. Relevant documents discuss the effectiveness of lumbar supports in preventing new occurrences of or treating existing low-back pain. Documents that don't discuss the effectiveness of lumbar supports for low-back pain but for other spine issues, e.g. neck pain, should be regarded as irrelevant.
18	query	electrical stimulation male urinary incontinence
	description	Can electrical stimulation devices treat male urinary incontinence?
	narrative	Electric stimulation using non-implanted devices is suggested as a treatment for male urinary incontinence (urine leakage). A relevant document should discuss whether electrical stimulation with such devices can prevent urine leakage, or treat urinary incontinence in men.
19	query	honey children cough
	description	Can honey offer relief from cough symptoms in children?
	narrative	Relevant documents should discuss the effectiveness of using honey for relieving cough symptoms in children.
20	query	steroids spinal cord injury
	description	Can steroids be used as a treatment for spinal cord injury?
	narrative	Relevant documents should discuss whether or not the steroid, methylprednisolone, is helpful for acute spinal cord injury.
21	query	acupuncture vascular dementia
	description	Can acupuncture treat vascular dementia?

	narrative	Dementia is a set of symptoms associated with loss of memory or thinking skills. Vascular dementia (VaD) is a subclass of dementia that occurs when the circulation system fails to carry enough blood to the brain. Among other interventions having side effects, acupuncture is said to be an effective treatment of VaD. A relevant document discusses whether or not acupuncture is an effective treatment for VaD. Documents discussing other types of dementia (e.g. Alzheimer, Lewy Bodies) should be regarded as irrelevant.
22	query	hydroxyzine generalized anxiety disorder
	description	How effective is hydroxyzine (also known as Atarax) for treating generalized anxiety disorder?
	narrative	Anxiety disorder is a common class of psychiatric disorders covering generalized anxiety disorder (GAD), panic attack, and phobia related disorders such as Acrophobia. Relevant documents should discuss the effectiveness of hydroxyzine (also sold under different brand names such as Atarax) for treating GAD or controlling GAD related anxiety, but may or may not mention other types of anxiety disorders.
23	query	insulin gestational diabetes
	description	Is insulin an effective treatment for gestational diabetes?
	narrative	Gestational diabetes (also known as gestational diabetes mellitus or GDM) is a subclass of diabetes that occurs during pregnancy and usually goes away after birth. However, GDM may have harmful consequences to the baby during birth and one solution suggested is insulin injections or anti-diabetic medicines. A relevant document discusses whether insulin or other anti-diabetic pharmacological medicines can be used to treat GDM. Documents not discussing GDM, but other types of diabetes (i.e. Type I and Type II), should be regarded as irrelevant.
24	query	yoga epilepsy

	description	Can yoga control epilepsy?
	narrative	It has been suggested that some types of yoga involving postural and breathing exercises can help to control epileptic episodes and treat epilepsy. Relevant documents discuss whether yoga is effective at controlling epilepsy and helps to reduce the frequency or duration of seizures.
25	query	fish oil ulcerative colitis
	description	Can fish oil be used to maintain the remission of ulcerative colitis?
	narrative	Ulcerative Colitis (UC) is an inflammatory disease affecting the rectum and colon and can eventually cause internal wounds. It has been suggested that fish oil can be used for people with UC to manage its symptoms. Relevant documents should discuss whether fish oil can be an effective solution to managing symptoms of UC.
26	query	vaccine common cold
	description	Can vaccines prevent the common cold?
	narrative	The common cold is a viral infection with common symptoms such as sneezing, runny nose, sore throat etc. and is different than the flu (influenza). Relevant documents should discuss whether vaccines can prevent the common cold. Documents not discussing the effectiveness of vaccines on the common cold, e.g. they discuss influenza, should be regarded as not relevant.
27	query	antibiotics children wet cough
	description	Can antibiotics be used as a treatment for wet cough (productive or chesty cough) in children?

	narrative	Wet cough, also known as a productive cough or chesty cough, is a cough that produces mucus or phlegm. In children, when a wet cough is persistent (e.g. for four weeks) it is considered a symptom of bacterial or viral infection that needs attention. Relevant documents discuss whether or not antibiotics can heal wet cough. Documents not discussing the effectiveness of antibiotics for persistent wet cough but for issues such as whooping cough, or ones not mentioning its effect on children should be regarded as irrelevant.
28	query	antibiotics whooping cough
	description	Can antibiotics be used as a treatment for whooping cough (pertussis)?
	narrative	Pertussis (whooping cough) is a bacterial infection that causes episodes of acute cough and can lead to death. Relevant documents discuss the effectiveness of antibiotics to treat pertussis.
29	query	antibiotics children pneumonia
	description	Can antibiotics be use to treat community acquired pneumonia in children?
	narrative	Antibiotics have been suggested as a treatment for children diagnosed with community acquired pneumonia, which is a different health issue than hospital acquired pneumonia. A relevant document should discuss this claim, whether or not antibiotics can heal children with pneumonia acquired outside hospitals. Documents discussing only hospital acquired pneumonia (nosocomial pneumonia) should be regarded as irrelevant.
30	query	aloe vera wounds
	description	Can you apply aloe vera to treat wounds?
	narrative	Aloe vera is a cactus-like plant that has been used in cosmetic and skin care products. It has been suggested that aloe vera is useful as a wound remedy. Relevant documents discuss whether aloe vera, topically applied as a gel or cream, can heal wounds.
31	query	exercise hot flashes night sweats menopause
	description	Are exercises helpful in reducing hot flashes and night sweats in menopausal women?

narrative	Exercising has been suggested to ease vasomotor menopausal symptoms such as hot flashes/flushes and night sweats. Relevant documents discuss whether or not exercise can help to reduce hot flashes and night sweats in menopausal women.

Table A.1: Medical Misinformation Track 2019 topics judged during the experiment

A.2 Web Track 2014 topics (Pilot study)

Topic		
260	query	the american revolutionary
	description	Find a list of the major battles of the American Revolution.
261	query	folk remedies sore throat
	description	What folk remedies are there for soothing a sore throat?
266	query	symptoms of heart attack
	description	What are the symptoms of a heart attack in both men and women?
271	query	halloween activities for middle school
	description	What activities are good for middle-school-aged children to celebrate Halloween?
272	query	dreams interpretation
	description	Find data on how to generally interpret dreams.
282	query	nasa interplanetary missions
	description	What interplanetary missions has NASA implemented or has planned?
285	query	magnesium rich food
	description	Which foods are rich in magnesium?
290	query	norway spruce
	description	How do you identify a Norway Spruce?
291	query	sangre de cristo mountains
	description	What are some cities/destinations within the Sangre de Cristo mountains region?

Table A.2: Web Track 2014 topics judged during the pilot study

Appendix B

Full metric tables

B.1 Rank Correlations

	DS/NIST-A	NIST-A/NIST	NIST/DS
map	0.8478	0.797	0.759
ndcg	0.8076	0.8478	0.778
P@R	0.7865	0.833	0.7548
relative P@R	0.7865	0.833	0.7548
RBP@R	0.8182	0.8562	0.7632
iprec at recall@0.00	0.1755	0.4123	0.1501
iprec at recall@0.10	0.6892	0.8076	0.6533
iprec at recall@0.20	0.8076	0.8203	0.7463
iprec at recall@0.30	0.7548	0.7907	0.7696
iprec at recall@0.40	0.7949	0.814	0.7949
iprec at recall@0.50	0.8372	0.8309	0.7696
iprec at recall@0.60	0.8076	0.8393	0.7696
iprec at recall@0.70	0.8372	0.8499	0.7801
iprec at recall@0.80	0.8203	0.8372	0.7674
iprec at recall@0.90	0.685	0.7717	0.7104
iprec at recall@1.00	0.2558	0.6448	0.3488
Rprec	0.7865	0.833	0.7548
Rprec*0.25	0.6512	0.7949	0.611
Rprec*0.50	0.7696	0.8753	0.7378
Rprec*1.00	0.7865	0.833	0.7548

	DS/NIST-A	NIST-A/NIST	NIST/DS
Rprec*2.00	0.8161	0.8076	0.7759
Rprec*4.00	0.8013	0.8182	0.759
Rprec*8.00	0.7949	0.8372	0.6998
Rrecall*0.25	0.6512	0.7949	0.611
Rrecall*0.25+100	0.74	0.7632	0.7653
Rrecall*0.50	0.7696	0.8753	0.7378
Rrecall*0.50+100	0.7632	0.7865	0.7822
Rrecall*1.00	0.7865	0.833	0.7548
Rrecall*1.00+100	0.8161	0.8626	0.797
Rrecall*2.00	0.8161	0.8076	0.7759
Rrecall*2.00+100	0.8288	0.8393	0.8034
Rrecall*4.00	0.7992	0.8203	0.7632
Rrecall*4.00+100	0.833	0.8055	0.7188
Rrecall*8.00	0.7949	0.8351	0.6977
Rrecall*8.00+100	0.7865	0.8436	0.6934
Rgain*0.25	0.6512	0.7949	0.611
Rgain*0.50	0.7696	0.8753	0.7378
Rgain*1.00	0.7865	0.833	0.7548
Rgain*2.00	0.8161	0.8076	0.7759
Rgain*4.00	0.8013	0.8182	0.7632
Rgain*8.00	0.7949	0.8351	0.6977
set P	0.7252	0.8245	0.6131
set recall	0.7357	0.8351	0.7653
set relative P	0.7357	0.8351	0.7653
set F	0.7336	0.8203	0.6173
P@5	0.1649	0.0275	0.3002
P@10	0.3383	0.2072	0.463
P@15	0.315	0.2939	0.5349
P@20	0.4228	0.3805	0.6195
P@30	0.5412	0.5053	0.6512
relative P@5	0.1649	0.0254	0.2981
relative P@10	0.3383	0.2199	0.4545
relative P@15	0.315	0.2854	0.5307
relative P@20	0.4228	0.3805	0.6237
relative P@30	0.5412	0.4863	0.6195
RBP@5	0.2283	0.2241	0.37

	DS/NIST-A	NIST-A/NIST	NIST/DS
RBP@10	0.4123	0.3869	0.5391
RBP@15	0.5264	0.5032	0.6385
RBP@20	0.5729	0.5307	0.6871
RBP@30	0.6364	0.6068	0.7167
P@100	0.6765	0.5877	0.6913
P@200	0.8266	0.5941	0.6364
P@500	0.8182	0.6047	0.5793
P@1000	0.7928	0.6216	0.5455
relative P@100	0.6512	0.6596	0.7505
relative P@200	0.7949	0.7463	0.8118
relative P@500	0.8457	0.7294	0.7019
relative P@1000	0.8182	0.6681	0.63
RBP@100	0.797	0.666	0.7548
RBP@200	0.871	0.666	0.7061
RBP@500	0.8457	0.6998	0.63
RBP@1000	0.8351	0.6871	0.6195

Table B.1: Kendall's τ for 44 runs - all metrics

B.2 Variance and Bias

Measure	Sigma			Bias		
	NIST-A	DS	NIST	NIST-A/DS	NIST-A/NIST	DS/NIST
map	0.1862	0.2018	0.1755	0.162	0.1789	0.2245
ndcg	0.1597	0.1964	0.158	0.1662	0.1591	0.1932
P@R	0.1794	0.2066	0.1775	0.1948	0.1532	0.2395
relative P@R	0.1806	0.2066	0.1768	0.1918	0.148	0.239
RBP@R	0.1659	0.1915	0.1652	0.1829	0.1351	0.2263
iprec at recall@0.00	0.4785	0.403	0.2559	0.6089	0.5024	0.7153
iprec at recall@0.10	0.23	0.2657	0.1805	0.2966	0.1821	0.3365
iprec at recall@0.20	0.2251	0.2465	0.1997	0.1873	0.1689	0.2333
iprec at recall@0.30	0.2178	0.2357	0.2061	0.199	0.1713	0.2088
iprec at recall@0.40	0.2028	0.2246	0.2123	0.2104	0.1736	0.1982
iprec at recall@0.50	0.1964	0.2103	0.2213	0.1822	0.1495	0.2212
iprec at recall@0.60	0.183	0.1994	0.2196	0.1872	0.1525	0.2297
iprec at recall@0.70	0.1826	0.1923	0.2329	0.1771	0.1659	0.2121

Measure	Sigma			Bias		
	NIST-A	DS	NIST	NIST-A/DS	NIST-A/NIST	DS/NIST
iprec at recall@0.80	0.1741	0.2083	0.2028	0.1997	0.1721	0.2593
iprec at recall@0.90	0.1838	0.1762	0.2144	0.2984	0.2104	0.2725
iprec at recall@1.00	0.3369	0.297	0.3246	0.6324	0.2917	0.5474
Rprec	0.1795	0.2072	0.1768	0.1951	0.1525	0.2388
Rprec*0.25	0.2356	0.253	0.2003	0.3165	0.2024	0.3539
Rprec*0.50	0.1967	0.222	0.1755	0.2257	0.1312	0.2531
Rprec*1.00	0.1805	0.207	0.1772	0.196	0.152	0.2373
Rprec*2.00	0.1612	0.1942	0.1758	0.1697	0.1707	0.226
Rprec*4.00	0.1623	0.1906	0.1804	0.1868	0.1772	0.2161
Rprec*8.00	0.1685	0.205	0.1773	0.1901	0.1536	0.2659
Rrecall*0.25	0.2353	0.2529	0.1997	0.3164	0.2045	0.3529
Rrecall*0.25+100	0.2013	0.1944	0.1879	0.2566	0.2357	0.2399
Rrecall*0.50	0.1985	0.2225	0.1754	0.2242	0.1287	0.2535
Rrecall*0.50+100	0.1825	0.1829	0.1849	0.2199	0.1818	0.2135
Rrecall*1.00	0.1797	0.2073	0.1777	0.1943	0.1501	0.2379
Rrecall*1.00+100	0.1733	0.1835	0.1814	0.1743	0.1314	0.1831
Rrecall*2.00	0.1619	0.1935	0.1755	0.1712	0.1707	0.2272
Rrecall*2.00+100	0.1635	0.1865	0.1796	0.1713	0.1625	0.1973
Rrecall*4.00	0.1621	0.1914	0.1814	0.1863	0.1731	0.2175
Rrecall*4.00+100	0.1634	0.1929	0.1803	0.1663	0.1701	0.2503
Rrecall*8.00	0.1688	0.2069	0.1759	0.1875	0.1544	0.2638
Rrecall*8.00+100	0.1663	0.2083	0.1686	0.196	0.1494	0.2666
Rgain*0.25	0.2366	0.2533	0.2004	0.3142	0.2027	0.3546
Rgain*0.50	0.197	0.2233	0.1766	0.2235	0.1292	0.2559
Rgain*1.00	0.1806	0.2076	0.1772	0.1943	0.151	0.238
Rgain*2.00	0.1618	0.193	0.1752	0.1701	0.1718	0.2249
Rgain*4.00	0.1623	0.1907	0.181	0.1839	0.1759	0.2116
Rgain*8.00	0.1679	0.2049	0.1765	0.1878	0.1547	0.2659
set P	0.1561	0.2931	0.1521	0.2052	0.1731	0.3072
set recall	0.1652	0.2181	0.1628	0.2314	0.1514	0.2214
set relative P	0.1649	0.2189	0.1618	0.2291	0.1494	0.2226
set F	0.1548	0.2831	0.1503	0.2032	0.1703	0.305
P@5	0.484	0.4286	0.2201	0.6633	0.8203	0.6445
P@10	0.4934	0.3823	0.201	0.5515	0.6945	0.5066
P@15	0.4435	0.3461	0.1917	0.5852	0.6371	0.4563

Measure	Sigma			Bias		
	NIST-A	DS	NIST	NIST-A/DS	NIST-A/NIST	DS/NIST
P@20	0.3957	0.2992	0.1884	0.5396	0.5678	0.3958
P@30	0.4133	0.3057	0.1892	0.4024	0.4534	0.3502
relative P@5	0.4826	0.4291	0.2197	0.663	0.8217	0.6432
relative P@10	0.4926	0.3851	0.1982	0.5461	0.6862	0.5054
relative P@15	0.4429	0.3458	0.1845	0.5865	0.6387	0.4521
relative P@20	0.3975	0.3005	0.1816	0.5386	0.5691	0.4023
relative P@30	0.4114	0.3067	0.19	0.4018	0.4664	0.3752
RBP@5	0.5222	0.3989	0.1928	0.5894	0.676	0.5716
RBP@10	0.4701	0.3447	0.1802	0.5029	0.5627	0.4502
RBP@15	0.4252	0.3219	0.1763	0.4361	0.4903	0.3858
RBP@20	0.391	0.3086	0.176	0.3993	0.4419	0.3368
RBP@30	0.3465	0.295	0.1761	0.3345	0.3839	0.2731
P@100	0.2775	0.2943	0.1878	0.3191	0.3844	0.2892
P@200	0.2226	0.2581	0.1748	0.1704	0.3823	0.3227
P@500	0.2242	0.2424	0.1536	0.1659	0.3649	0.3876
P@1000	0.1908	0.2444	0.1513	0.211	0.3302	0.4097
relative P@100	0.2775	0.2606	0.1849	0.3245	0.3339	0.2594
relative P@200	0.216	0.2053	0.1757	0.1958	0.2659	0.202
relative P@500	0.211	0.1751	0.1619	0.1701	0.2573	0.2784
relative P@1000	0.1833	0.1882	0.1619	0.1591	0.3137	0.3273
RBP@100	0.2316	0.2559	0.1754	0.1897	0.3139	0.2158
RBP@200	0.2101	0.2411	0.1697	0.1279	0.3084	0.2645
RBP@500	0.1906	0.2281	0.1556	0.1396	0.2834	0.3191
RBP@1000	0.1708	0.2207	0.1502	0.1652	0.2659	0.3406

Table B.2: Bias and variance for 44 runs - all metrics

B.3 Average Differences

Measure	Average Value			Mean Difference (%age)		
	NIST-A	DS	NIST	DS vs NIST-A	NIST-A vs NIST	NIST vs DS
map	0.1874	0.2013	0.2073	7.42%	-9.62%	3.00%
ndcg	0.5422	0.5827	0.5068	7.47%	6.99%	-13.03%

Measure	Average Value			Mean Difference (%age)		
	NIST-A	DS	NIST	DS vs NIST-A	NIST-A vs NIST	NIST vs DS
P@R	0.2414	0.239	0.2422	-0.97%	-0.34%	1.32%
relative P@R	0.2414	0.239	0.2422	-0.97%	-0.34%	1.32%
RBP@R	0.204	0.2003	0.2027	-1.81%	0.64%	1.19%
iprec at recall@0.00	1.3272	0.8453	0.6378	-36.31%	108.11%	-24.55%
iprec at recall@0.10	0.4912	0.4449	0.4453	-9.43%	10.33%	0.08%
iprec at recall@0.20	0.3678	0.3441	0.3478	-6.45%	5.73%	1.10%
iprec at recall@0.30	0.2957	0.2837	0.2837	-4.05%	4.21%	0.01%
iprec at recall@0.40	0.2472	0.2308	0.2361	-6.63%	4.73%	2.26%
iprec at recall@0.50	0.2015	0.1888	0.1934	-6.31%	4.18%	2.45%
iprec at recall@0.60	0.1539	0.1518	0.1515	-1.31%	1.56%	-0.22%
iprec at recall@0.70	0.1092	0.1171	0.1089	7.29%	0.20%	-6.98%
iprec at recall@0.80	0.0706	0.0862	0.0725	22.05%	-2.63%	-15.86%
iprec at recall@0.90	0.0303	0.0492	0.0347	62.30%	-12.79%	-29.35%
iprec at recall@1.00	0.0036	0.0104	0.0049	188.88%	-25.64%	-53.45%
Rprec	0.2414	0.239	0.2422	-0.97%	-0.34%	1.32%
Rprec*0.25	0.3524	0.3612	0.3576	2.49%	-1.46%	-0.98%
Rprec*0.50	0.3086	0.3029	0.3102	-1.84%	-0.53%	2.42%
Rprec*1.00	0.2414	0.239	0.2422	-0.97%	-0.34%	1.32%
Rprec*2.00	0.1787	0.1741	0.1739	-2.58%	2.77%	-0.12%
Rprec*4.00	0.1214	0.1146	0.1179	-5.58%	3.00%	2.82%
Rprec*8.00	0.0732	0.0707	0.073	-3.33%	0.17%	3.27%
Rrecall*0.25	0.0881	0.0903	0.0894	2.50%	-1.46%	-0.99%
Rrecall*0.25+100	0.1333	0.1841	0.3631	38.15%	-63.30%	97.22%
Rrecall*0.50	0.1543	0.1514	0.1551	-1.84%	-0.53%	2.42%
Rrecall*0.50+100	0.1887	0.2281	0.3859	20.86%	-51.09%	69.17%
Rrecall*1.00	0.2414	0.239	0.2422	-0.97%	-0.34%	1.32%
Rrecall*1.00+100	0.2682	0.292	0.4249	8.86%	-36.88%	45.55%
Rrecall*2.00	0.3575	0.3483	0.3478	-2.58%	2.77%	-0.12%
Rrecall*2.00+100	0.3751	0.3842	0.4825	2.43%	-22.27%	25.59%
Rrecall*4.00	0.4856	0.4585	0.4715	-5.57%	3.00%	2.82%
Rrecall*4.00+100	0.4957	0.4788	0.5556	-3.40%	-10.79%	16.04%
Rrecall*8.00	0.5852	0.5657	0.5842	-3.33%	0.17%	3.27%
Rrecall*8.00+100	0.5885	0.5777	0.6269	-1.83%	-6.13%	8.51%
Rgain*0.25	0.0881	0.0903	0.0894	2.50%	-1.47%	-0.99%
Rgain*0.50	0.1543	0.1514	0.1551	-1.83%	-0.53%	2.41%
Rgain*1.00	0.2413	0.239	0.2422	-0.96%	-0.34%	1.31%

Measure	Average Value			Mean Difference (%age)		
	NIST-A	DS	NIST	DS vs NIST-A	NIST-A vs NIST	NIST vs DS
Rgain*2.00	0.3574	0.3483	0.3478	-2.56%	2.76%	-0.13%
Rgain*4.00	0.4855	0.4585	0.4714	-5.56%	3.00%	2.80%
Rgain*8.00	0.5851	0.5657	0.5841	-3.31%	0.17%	3.25%
set P	0.0673	0.0341	0.0508	-49.41%	32.63%	49.05%
set recall	0.6719	0.7846	0.7049	16.77%	-4.68%	-10.16%
set relative P	0.6719	0.7846	0.7049	16.77%	-4.68%	-10.16%
set F	0.1175	0.0599	0.0919	-48.98%	27.88%	53.27%
P@5	0.4433	0.4779	0.4379	7.79%	1.25%	-8.37%
P@10	0.4497	0.4423	0.3973	-1.65%	13.17%	-10.16%
P@15	0.4759	0.4212	0.3712	-11.48%	28.21%	-11.88%
P@20	0.4677	0.4071	0.3507	-12.96%	33.36%	-13.85%
P@30	0.4547	0.3868	0.3205	-14.93%	41.88%	-17.15%
relative P@5	0.4433	0.4779	0.4379	7.79%	1.24%	-8.37%
relative P@10	0.4497	0.4423	0.4003	-1.65%	12.33%	-9.48%
relative P@15	0.4759	0.4212	0.3815	-11.48%	24.72%	-9.42%
relative P@20	0.4677	0.4071	0.3713	-12.96%	25.98%	-8.80%
relative P@30	0.4547	0.3868	0.3612	-14.93%	25.87%	-6.61%
RBP@5	0.4621	0.4712	0.41	1.98%	12.69%	-12.98%
RBP@10	0.4582	0.4256	0.3668	-7.11%	24.91%	-13.81%
RBP@15	0.4533	0.4014	0.337	-11.44%	34.48%	-16.03%
RBP@20	0.4474	0.3842	0.3142	-14.13%	42.42%	-18.24%
RBP@30	0.4361	0.3592	0.2799	-17.64%	55.79%	-22.07%
P@100	0.413	0.3107	0.2098	-24.78%	96.84%	-32.46%
P@200	0.3738	0.2593	0.149	-30.61%	150.90%	-42.56%
P@500	0.3041	0.191	0.0844	-37.21%	260.26%	-55.79%
P@1000	0.241	0.1394	0.0508	-42.15%	374.54%	-63.58%
relative P@100	0.4143	0.3332	0.36	-19.58%	15.09%	8.04%
relative P@200	0.3852	0.3166	0.4516	-17.82%	-14.70%	42.66%
relative P@500	0.3566	0.3476	0.6048	-2.50%	-41.05%	73.97%
relative P@1000	0.3514	0.4414	0.7049	25.60%	-50.14%	59.70%
RBP@100	0.3823	0.2775	0.176	-27.41%	117.20%	-36.58%
RBP@200	0.3364	0.2271	0.1235	-32.49%	172.29%	-45.60%
RBP@500	0.263	0.1612	0.0695	-38.71%	278.59%	-56.90%
RBP@1000	0.2045	0.1164	0.041	-43.06%	398.35%	-64.76%

Measure	Average Value			Mean Difference (%age)		
	NIST-A	DS	NIST	DS vs NIST-A	NIST-A vs NIST	NIST vs DS

Table B.3: Average values and differences for 44 runs - all metrics