Identifying Relevant Medical Research Papers: MS-UG-3

Student : Simon Drake

Supervisor : Dr Mark Stevenson

Module Code : COM3610

This report is submitted in partial fulfilment of the requirement for the degree BSc Computer Science by Simon Drake

Abstract

Screening studies for the inclusion in systematic reviews such that all relevant documents are found is a slow-moving and complex process. This task is growing steadily more difficult due to the large number of published studies and their increasing rate of publication. Text mining has been investigated as a tool to reduce a reviewer's workload. This study puts forward two objectives. Our first goal is to assess the workload reduction achievable when text mining is used to screen studies, while also maintaining a recall of at least 95%. Our second goal is to critically analyse how different machine-learning algorithms behave with different sampling methods and datasets. Our findings indicate that the Complement Naïve Bayes and the Support Vector Machine are the strongest algorithms to use for this task. Moreover the implementation of a voting system can reduce the workload by 26.6% while maintaining a recall of 98.2%.

Acknowledgements

I would like to express my very great appreciation to Dr Mark Stevenson for giving me the opportunity to work with him on this research project and whose suggestions and feedback were invaluable to its completion. A special thanks also to Amal Alharbi who together with Dr Mark Stevenson provided me with an initial dataset to work with.

Contents

List of Tables

Chapter 1: Introduction

1.1 Background

Screening for the inclusion of relevant studies in systematic reviews is a slow-moving and complex process. Systematic reviews bring together the findings from multiple studies in a reliable way and are often used to inform policy and practice [3]. The failure to include all relevant material introduces bias and can hinder the reliability of the results. This task is growing steadily more difficult due to the large number of published studies and their increasing rate of publication [4]. Moreover, these reviews need to be completed within well-defined timetables and budgets which can rarely scale to the size of the task. Thus, multiple researchers have looked into the use of automating (or semi-automating) this process in order to reduce the workload for human screeners.

1.1.1 The increasing momentum

Keeping up with information has always been a difficult task. This is true of all domains but especially in the medical field where an overwhelming number of people are pushing boundaries in the interest of improving the well-being of patients. This coupled with the accelerating growth of technological advancements makes for a powerful forward momentum. "An experienced reviewer can screen an average of two abstracts per minute. At this rate, a project with 5,000 abstracts requires 5 person days (40 hours) of uninterrupted work time. Abstracts for difficult topics may take several minutes each to evaluate, thus multiplying the total time needed to process them by several fold" [5]. From this statement, it is clear that an enormous amount of time and effort is required to screen one project when relying on human screeners alone. Additionally, McGowan and Sampson [6] note that some reviews can cost up to a quarter of a million dollars. A study published in 2010 by Bastian et al. [4] shows that in about 31 years we have gone from 14 reports of trials being published a day to 75 trials and 11 systematic reviews a day and "the plateau in growth has not yet been reached". The situation has most likely intensified since then. Finally according to Jaidee et al. [7] only about a third of the systematic reviews in the Cochrane library are being updated within recommended intervals of 2 years. We have gotten to the point where it is no longer feasible for human reviewers to perform all the steps in the screening stage manually. It is clear that automation or semi-automation of the screening process is vital to assemble all relevant studies timely and efficiently.

1.1.2 Progress has been made

Despite what may seem as an overwhelming problem is perhaps not without solutions. There has been a lot of progress in (semi-)automating the process of screening. In the next chapter we will dive further into the field highlighting studies in their quantitative natures. Notwithstanding it is worth looking at some of the qualitative studies that support the use of text mining for the screening stage here. O'Mara

Eves et al. conducted a systematic review on studies that explored the use of text-mining (TM) in the screening stage of systematic reviews. After a rigorous search they found 44 studies that were published between 2004[1] and 2014. The authors state that studies usually reported reductions in workload ranged between below 10% and up to more than 90% but tended to be between 40% and 50%. They also conclude, "Reviewers should engage with the computer science community to develop and evaluate methods and systems jointly" and "the use of TM to prioritise the order in which items are screened should be considered safe and ready for use in 'live' reviews"[8]. Furthermore a report by Shemilt et al. [9] explores the use of TM in the screening stage for scoping reviews. Scoping reviews are not the same as systematic reviews, they are conducted to explore and determine the boundaries of the evidence base used in the early stages of standard systematic reviews. Nonetheless for our purposes it worth looking at their findings: text mining performed ≈10 times better than conventional screening methods. On the other hand, TM methods are unlikely to be a complete replacement of current literature retrieval and information extraction methods. As Robin Paynter et al. [10] demonstrate, text mining is more likely to augment the existing use of techniques and tools as oppose to replace them.

There is some evidence to support Paynter et al.'s conclusion in that the combination of human and TM efforts may well be the final outcome. The SWIFT-Review is a software tool that uses TM and machine learning to prioritise a set of documents in order of relevance, a study conducted by Howard et al. found that: "Text-mining and machine learning software such as SWIFT-Review can be valuable tools to reduce the human screening burden and assist in problem formulation" [11]. Also, J. Thomas et al. [12] provide ways in which to combine human and TM efforts to make 'living' systematic reviews with limited time input from any one individual which could address the problem of not updating reviews on time highlighted by Jaidee et al. [7]. Ergo there is a promising landscape for human-TM methods.

1.1.3 Deeper into the problem

We are diving into a very specific problem here and by its very nature it has some native challenges: class imbalance, the importance of recall, hasty generalisation and the problem of reliability. Essentially this task involves sorting a set of documents into a relevant class and an irrelevant class. Most machine-learning algorithms assume that the class distribution is more or less even i.e. there is an equal number of instances in each class. This is not the case for our task. We are dealing with heavy class imbalance. Usually around 99% of the documents retrieved from a query are irrelevant. This poses a problem for many algorithms as their accuracy[2] can be very high (>98%) and still have not found any relevant documents. Therefore, the algorithms used are often ones that are either built for class imbalance or tailored to handle it. Evaluation metrics such as accuracy do not provide us with very informative statistics, instead the most common metric used is recall [8]. Recall allows us to

1. The first study to use text mining in the screening stage was in 2005 [1].
2. Accuracy is the portion of correct classifications an algorithm makes compared with the total number of classifications.

examine how many relevant documents have been found and is extremely important to avoid bias. Matwin et al. [13] mention that any “results below 95% recall are not acceptable for a user building a real systematic review”. Notwithstanding, it is not difficult to optimise: including all the documents in the test set would give 100% recall. Hence algorithms are often designed to optimise recall while maintaining “reasonable” precision. On the other hand this does introduce noise [14]. In abstraction a little noise is preferable to low recall, as this would minimise publication bias. As a further matter (semi-)supervised machine learning problems typically base their predictions on a training set. The training set “teaches” them what the respective classes should look like. Although what often occurs is that the training set does not truly represent the classes in all their properties and this can lead to “hasty generalisation”. This is another form of bias that needs to be avoided. Olorisade et al. investigated the reproducibility of studies in the field of using TM for citation screening [15]. They found that “it is currently difficult if not impossible to independently reproduce the results published in any of the studies investigated.”. This poses a direct conflict to the progress of this field. In order to deal with this problem they provide a check-list for researchers to refer back to in order to make their results reproducible and reliable. It is also worth noting at this point that the nature of medicine is high dimensional: the great number of symptoms and treatments makes for a vast lexicon. This is important because many machine learning algorithms deal with vectors of features and often the features are the individual words themselves. All of these issues form obstacles limiting the efficiency of TM to reduce workload and must be considered when dealing with the problem.

1.2 Our project

1.2.1 Study objectives

As mentioned in section 1.1.3 maintaining a high recall is fundamental to the successful completion of a systematic review. As such, the first goal of this study is to assess the workload reduction achievable when text mining is used to screen studies for the inclusion in systematic reviews, while also maintaining a recall of at least 95%. Our second goal is to critically analyse how different machine-learning algorithms, namely the Complement Naïve Bayes, Support Machine Vector, Adaboost, K-Nearest Neighbour and Neural Net behave with different sampling methods and datasets.

1.2.2 What comes next

In this dissertation we will go on to demonstrate that the use of text mining can be a powerful tool in reducing the workload for reviewers and can help enable them to complete reviews within deadlines despite the limitations described above. Firstly, we begin with an extensive review of previous work done showing the results of multiple techniques and algorithms. We review some studies that use different techniques to the ones we investigate in our experiments, however, we believe it is important to summarise the scope of this field and include possible alternatives and improvements to our designs. Following our review we state what our goals are in conducting this study and give a brief overview of the data and algorithms that we have used in our experiments. As a further matter we describe the metrics we used to evaluate them, the most important ones being recall, f-measure and workload reduction. Chapter 4 describes in more detail our methods with regards to the individual experiments as well as what we expect to see in the results. Moreover we provide an outline of our preprocessing and sampling methods. Chapter 5 focuses on displaying our results, discussing the findings and highlighting the goals we achieved. Furthermore we suggest possible future work. Finally, Chapter 6 brings together our work and highlights the main points.

1.3 Relationship with my degree

During the course of my degree we have been exposed to concepts and implementations of machine-learning tools in multiple modules. “Machines and Intelligence” in year 1 provided us with a first glance at the scope and depths of the machine learning field. Once we had basic skills in programming and vector mathematics we studied “Data Driven Computing” where we programmed a Principal Component Analysis system. Similar concepts are used in the SVM use in our tests. Finally “Text Processing” in year 3 was the biggest contributor to this project. We learnt how tokenisation and stemming worked which are vital to efficient information retrieval. Moreover we studied Naïve Bayes classification, the core algorithm which is extended to the Complement Naïve Bayes we use in this study. On the whole many modules provided the foundations needed for this project, other skills needed to be acquired and refined such as efficient literature reviewing, machine learning concepts and programming techniques.

Chapter 2: Literature review || Previous work || State of the evidence base

According to Jonnalagadda and Petitti [1] the first occurrence of screening papers using TM for the inclusion in systematic reviews was in 2005. Since then and over the course of the past 14 years, numerous experiments have been conducted to better understand the extent to which TM can aid the screening process of abstract and full text documents. Seldom are there cases where a reduction in workload is not reported. However given the diversity of the approaches and the lack of overlap between evaluations it is impossible to conclude whether one approach is better than another in terms of performance [8]. As this is a very specific machine learning task there are some ways in which these studies overlap. They all use very specific algorithms adapted to the task at hand. Many have been previously studied and very often the same algorithms are used repeatedly as they have proven to perform well. They all define specific ways to represent the corpora, each document is represented using features, these features are either extracted straight from the document or processed further into other representations. Some use semi-supervised machine learning algorithms in order to reduce the need for human labelled training data, this is referred to as active learning. Furthermore some focus and specialise on certain aspects of the task listed in section 1.1.3. This chapter is structured in such a way that puts this overlapping into contrast. We are going to start by outlining the purposes behind (semi-)automating the task of screening documents in terms of reducing the workload involved in a similar way to O'Mara-Eves et al. [8]. Then, we are going to break down the evidence base such that the specific focuses of the individual studies are grouped together and presented thematically.

2.1 Purposes

A number of strategies have been undertaken with regards to reducing workload. One such strategy has been to decrease the number of items to be screened. This approach has been widely adopted due to the sheer quantity of data that database queries can return, especially for broader research questions. It also directly addresses the issue at hand in a direct way. We can define two sub-approaches here; one involves making explicit in/out decision of instances, used by Razavi et al. [16] and Frunza et al. [14]. The other was used by Shemilt et al. [9]. This approach involves returning a ranked list of studies in order of relevance on which instances under a certain threshold are discarded. One powerful advantage to the latter of the sub-approaches is that a reviewer can get a strong grasp of the inclusion criteria in the early stages of the screening process. O'Mara-Eves et al. highlight that: good initial identification of relevant citations lead to an overall increase in screening rate [8]. In most cases, when deciding on the evidence base for the inclusion in a systematic review many reviewers take part. "Common systematic review practices stipulate that two reviewers are used at the screening phases of a systematic review to review each abstract of the documents retrieved after a simple query-based search" [14]. This is supported by the fact that a single screener can inadvertently introduce bias into the study selection process either because of their understanding of the inclusion criteria or through their interpretation of the titles and abstracts [8]. This brings us to our second main strategy: the use of text mining as a second screener, this technique was used by Bekhuis and Fushman [17]. A machine learning algorithm effectively acts as a “double check” confirming the screening of the previous reviewer(s) or bringing it into question. Finally increasing the rate of screening has also been investigated by Felizardo et al.[18]. The idea here is to provide the screener with visual aids to help them come to a decision faster by providing visual connections between documents using terminology or other contextual information.

2.2 Algorithms

Several studies have researched the performance of specific machine learning algorithms. A systematic review conducted by Arji et al. gave an extensive overview of the performance of a range of algorithms [19]. They highlight the strengths and weaknesses of the algorithms and noted that future methods will most likely contain hybrid methods that combine strengths and wedge out weaknesses. For our purposes we will consider their evaluation on Decision Trees (DT), Support Vector Machines (SVM), Bayesian Networks (BN), Random Forests (RF) and Neural Nets (NN)[3]. Their assessment is as follows: DTs are easy to understand but tend to be biased towards the majority class for imbalanced datasets. SVMs have been used most frequently. They have strong generalisation capabilities when trained on a small sample of the data, however, they implement black box computations. BNs are resistant to noise but their classifications are often unreliable. RFs perform well with high dimensional and highly correlated data. NNs model complex relationships well yet their models are also black boxes. This review outlines many strengths and weakness of these algorithms and can provide insight into which ones should be chosen given a dataset.

The following studies provide strong quantitative evidence on the performance their algorithms. Adeva et al. compared 4 algorithms: Naïve Bayes, K-Nearest Neighbour (KNN), SVM and Roccio [20]. They found that the best results were given by the NB and the SVM, particularly the NB returned the lowest number of false negatives and the SVM performed just as well using just titles than titles and abstracts. Whereas Matwin et al. focused solely on a factorised version of the Complement Naïve Bayes (FCNB) [13], providing strong evidence that it can outperform voting perceptron based classification systems. Their average workload reduction was 33.5%, 15.0% higher than the voting perceptron. They also make the argument that Bayesian classifiers are easier to understand for users in the medical field in comparison to SVMs mainly due their parameters making them a finer choice. Finally we look at the work done by Bekhuis and Fushman [17] who compared a KNN, NB, Complement Naïve Bayes (CNB) and EvoSVM[4] algorithm. They found that the EvoSVM performed best. It is also worth noting that although Frunza et al. [14] did not provide data to justify their decision they chose to use a CNB as it was said to outperform DTs, SVMs, instance-based learning and boosting. Overall it seems as though SVMs may be harder to understand due to their parameters but perform very well, and together

3. they also compare association rule and k-means clustering algorithms however that falls out of the scope of this report.
4. Description of evoSVM is given in [2].

with the CNB they are the better choices for screening studies. However we can see conflicting arguments as to which algorithm is better.

2.3 Features

As aforementioned, in order to perform machine learning each document is represented using features. The standard features used in this task are bag of words (BoW) and TF-IDF. As these are benchmark features we will not highlight studies that have only used these representations. Firstly we will examine a systematic review summarising some key points, then look at feature selection methods finishing with feature representation methods.

The systematic review carried out by Mujtaba et al. [21] did not look at the task of screening studies for inclusion in systematic reviews but rather clinical text classification. This review of 72 primary studies published between January 2013 and January 2018 is extremely comprehensive and we feel is important to look at due to the overlap in domains. Their findings are as follows: The most common features used are BoW, n-gram, bag of phrases (BoP) and bag of concepts (BoC). However, these representations are weak compared to others firstly because word order and grammar are disregarded. On top of that, they do not capture word inversion and subset matching. Secondly, they do not consider word-level synonymy and polysemy. To address the first limitation BoC could provide better classification and GoW can address the second limitation. Moreover BoP, BoC and GoW produce huge feature sets, this would be addressed by using information gain or chi-square feature selection schemes which can help in reducing dimensions.

There has been extensive work done on how to represent documents, here we highlight a few from this set: Kontanatsios et al. used 2 feature representation techniques in their experiments, the first was the standard BoW representation the second was spectral embedding based on BoW which “approximates similarity between BoW on their relative location in a lower dimensional space” [22]. Algorithms using this representation performed better in almost all cases. Unified Medical Language System (UMLS) concepts have been identified and extracted from corpora in other cases ([5] [14] [23]). These terms tend to boost classification performance. Razavi et al. used 2nd order co-occurrence vectors [16]. This type of feature representation takes into account the proximity between words to extract semantic value. They based it on the following assumptions: you shall know a word by the company it keeps, meaning of words are determined by their distributional patterns and words that occur in similar context will have similar meanings. This representation technique proved quite successful in comparison with the BoW approach, most specifically in regards to avoiding false negatives. Kouznetsov et al. also used 2nd order co-occurrence reporting positive results [24]. Finally Miwa et al. used latent dirichlet allocation (LDA), which “is a Bayesian generative model that represents documents as a mixture of hidden topic distributions, each of which is represented by a floating value”. Using LDA improved the performance of their classifier compared to their baseline methods.

Some studies decided to devote their research into picking the most informative features: Chi-squared is a method which computes the Chi-square statistic for a class. This statistic measures how much two categorical variables are related to each other. It was used by Frunza et al. and Kouznetsov et al. ([14] [24]). Information gain is another statistic used to select features, one can think of information gain as loss of entropy or disorder in data. This measure was used by Frunza et al. and Freund et al. ([14] [25]). Bekhuis and Fushman also used information gain to optimise the classifier's parameters [17]. Finally Bi-Normal Separation is the last feature selection technique we will mention. It "measures the separation between the threshold occurrences of a feature in one of the two classes" [14] this was the 3rd and last feature selection method examined by Frunza et al.

Unrelated to unique feature representations or selection methods but a worthy contender in this section we find Adeva et al. who focused on the number of features and found that a smaller number of features can produce better results [20].

2.4 Active learning

One of the most challenging aspects of designing supervised machine learning algorithms is providing enough training data that adequately models unseen instances. Building a decent training set requires a great amount of expert hours. Thus, a lot of research has gone into semi-supervised learning: using algorithms to augment the training set dismissing the need for human-labelled data, this is referred to as "active learning". We often talk about "label propagation" or "label spreading" when referring to these semi-supervised algorithms. The idea is to start from a small training set and using that to label unseen instances closest to the cluster of training data we do have or those closest to the decision boundary. The distinction between the two are known as "certainty-based" or "uncertainty-based" labelling respectively. Kontonatsios et al. adopt the "cluster assumption" [26] to investigate both certainty-based and uncertainty-based label propagation. They highlight that active learning has poor performance in the first few iterations as there is a low number of human labelled instances which constrains the hypothesis space[5]. Their findings show that utility improves in the early stages compared to baseline learning methods for both label propagation approaches. Moreover Liu et al. argue that supervised methods hold little promise for creating new reviews and are better adapted to updating current reviews due to the amount of labelled data already available [23]. They used self-training with label spreading to identify the most certain unlabelled instances then trained an SVM on the augmented training set and used it to classify the remaining unseen instances. They found that active learning with supervised SVM gave better recall and precision than other methods tested even with a small amount of seed data. What's more, we have Wallace et al. [5] who used a custom active-learning algorithm (Patient Active Learning) built for imbalanced datasets to drastically reduced the manual screening of abstracts by up to 50% without missing any relevant documents. PAL works by first requesting labels randomly until a good picture of the minority class has been found, it then performs undersampling. These results provide strong evidence to the potential of semi-automation.

5. The set of all possible decision boundaries.

They do point out that during testing and development their algorithm was optimised for the datasets they were testing on, still the results were promising over 10 independent runs. Finally, Miwa et al. [27] employed a weighting method in order to alleviate the data imbalance problem. They would weigh the positive instances by the ratio of relevant to irrelevant documents in the training set. This was done for both standard labelling methods and they found that the weighting method with certainty labelling is very promising. Overall it is difficult to draw conclusions as to which are the best methods for active learning, it would seem that certainty and uncertainty-based labelling are both strong candidates for active learning however using more evolved versions or weighting methods could prove even better.

A couple of studies combined their algorithms with human knowledge to investigate the effectiveness of hybrid methods. For instance, Zhou et al [28] used an extended version of the label propagation algorithm (LPA) to specifically deal with the low amount of initial labelled data. Using some prior knowledge about the datasets in graphical representations their algorithm can perform better than the standard LPA. Also, Wallace et al. [29], used a domain expert and medical decision theory associated relative costs to false positives and false negatives based on the ratio of recall to accuracy. The findings indicated that using domain knowledge may help in avoiding the missing cluster effect and that the external input also helps to avoid hasty generalisation. Moreover uncertainty based learning may be more effective in terms of recall. These two examples show that it may be more effective to combine human efforts with TM efforts rather than having them replaced by TM efforts. This reinforces Paynter et al.'s [10] conclusion.

2.5 Dealing with imbalance

Oversampling and undersampling has been shown to help with the problem of class imbalance. Undersampling has proven to be better with large domains such as the one used in this study [30]. There are two sub-approaches to undersampling: one is referred to as random undersampling. This is done by randomly picking instances from the majority class and discarding them to balance the classes. This approach was taken by Miwa et al. [27] and Wallace et al. [5] in their active-learning experiments. The second approach is known as aggressive undersampling, the idea here is that the classifier is first trained and then irrelevant instances that are closest to the hyperplane are thrown out. This is meant to displace the hyperplane closer to the positive instances. However Miwa et al. note that "this method has the potential drawback that it cannot produce a reasonable classifier since the method does not use the instances close to the separating hyper-plane." [27]. Both of these studies reported strong results, Wallace et al. mention that PAL with aggressive undersampling performs the best [5].

2.6 The importance of recall

The importance of identifying *all* relevant studies is emphasised repeatedly in various reports. One way of dealing with this has been the implementation of a voting system or committee to help ensure as much as possible that all the studies are found. Multiple such examples who all differ in subtle ways. For instance, Kouznetsov et al. [24] used a 5-classifier voting scheme to reduce the number of studies that needed to be screened. They used a CNB, Discriminative Multinomial Naïve Bayes (DMNB), DT and 2 Adaboosts reporting a recall of 91.6%. Miwa et al. [27] combined the classifications of an ensemble of classifiers by multiplying their predictions together. They note that the voting scheme performed better than baseline methods. Furthermore, Wallace et al. [29] only incorporated domain expert decisions if their committee of 2 had different classifications. What's more, Frunza et al. [31] seem to be the only ones who investigated how the different number of votes would effect the results. They state that a 4-vote scheme provided the best recall but the 2-vote scheme gave the best F-measure. Finally, Cohen et al. [32] use a voting-perceptron algorithm. The subtle difference is that this algorithm actually uses the same classifier, only it saves each prediction vector and uses them as "voters". Also worth noting is that Cohen et al. [32] and Matwin et al. [13] expressed their goals in terms of 95% recall and used as their main evaluation measure work saved over sampling at 95% recall. Matwin et al [13] mention that any "results below 95% recall are not acceptable for a user building a real systematic review". The results of these experiments are a little conflicting however they note that in general the voting-perceptron algorithm can be used as a confidence measure. From the evidence base we can infer that voting schemes are not always better when we are looking at f-measure and can sometimes be outperformed by the CNB. However they are a good solution to achieving high recall and reducing bias.

2.7 Overfitting and cross-validations

Overfitting is a situation whereby a classifier has modelled a set of data points far too closely and will use this overly fitted model on unseen data points. A model like this would get perfect scores on identical cases to the ones trained on but very poor scores for other instances. To get around this problem data is split into training sets and test sets so the classifier performance can be evaluated on unseen instances. In order to rely on these evaluations the tests are cross validated so that the classifier is tested on all the data in partitions. We can find examples of this in many studies such as Bekhuis et al. [17] who optimised their classifiers parameters by performing 10-fold cross-validations, this allowed them to find the optimal parameters without having an overfitted model. Adeva et al.[20] also used 10-fold cross-validation to rely on their results. However Thomas G. Dietterick [33] argues that 10-fold validation can overestimate performance. These findings influenced Matwin at al. [13] and Cohen et al. [32] in their decision to use 5x2 cross-validation which is performed by randomly splitting the data into halves and using each half as a test set and training set. This method is repeated 5 times. It seems that both methods are improvements to not using any cross-validation but if an overestimation should be avoided then 5x2 would be a better method.

2.8 Summary

As we have seen there has been an abundance of research in this field. In all the cases we notice researchers report positive results sometimes with some constraints. We have seen that reducing the number of items to screen is the most common approach as it directly reduces the workload for a researcher. Moreover, there exist many algorithms to choose from, each of them with strengths and weaknesses but the SVM and the CNB are considered to be the strongest candidates. There seems to be a plethora of ways to represent the document. We can select which features to represent the document with statistical methods such as the chi-squared statistic or information gain as well as represent them in different forms by using features such as UMLS features or 2^{nd} order co-occurence. Furthermore active learning has been investigated to reduce the need for a large initial training set, these techniques have proven to reduce a screener's burden. Finally we looked at some ways to deal with the challenges of this particular problem. In order to deal with the class imbalance one can perform undersampling on the training set, aggressive undersampling might be the best method compared to basic methods. Also, the implementation of a voting system or a committee has shown to increase recall while maintaining strong f-measure. Some studies ([32] [13]) only expressed their results in terms of 95% recall mentioning that anything lower was not acceptable in real-world circumstances. Moreover the use of cross-validation should be used in order to make results reliable, 5x2 cross-validation is a better method than 10-fold cross-validation as it does not over estimate a classifier's performance. In the following sections we investigate for ourselves the efficiency of a range of algorithms on multiple data sets. We pay close attention to the efficiency of the CNB and the SVM in comparison with each other. Moreover we investigate the use of a voting system and see how it compares to any one strong classifier and whether certain algorithms can complement each other. As pointed out by Thomas G. Dietterick [33] 5x2 cross-validation gives more realistic results and so this is what we will use together with an extension of this cross-validation technique. Finally we will investigate the effect of undersampling in comparison with our other methods.

Chapter 3: Requirements and analysis

As we have seen in Chapter 2 there has been a plethora of algorithms and techniques used to investigate the use of TM to screen studies. We will focus on a subset of these, specifically, we are going to optimise and test 5 algorithms individually and then look at their combinations in a committee under different sampling methods. As outlined in Chapter 2 Cohen et al. and Matwin et al. both measured their experiments by a workload reduction at no less than 95% recall. Drawing on this approach our main goal in these experiments is to assess the workload reduction achievable while keeping recall above 95%. Our second goal is to inspect how the individual algorithms behave with different sampling methods and datasets. In order to develop a fully fledged system capable of providing reliable results we need a pre-labelled dataset on which we can test it, strong implementations of each algorithms and a way to evaluate them. This chapter focuses on these aspects in some depth.

3.1 Dataset

The data we will use was collected by Kanoulas et al. [34] for a task whereby universities participated in submitting different ranking algorithms which were tested on a range of studies retrieved for Diagnostic Test Accuracy reviews which according to Leeflang et al. [35] are to be within the most challenging. These reviews where conducted in the past and submitted by Cochrane researchers. They are publicly available and can be found in the Cochrane Library[6]. Each review in the collection is represented by a “topic file” which consists of the Topic ID, the title of the review, the Boolean query and the set of PubMed Document Identifiers (PMIDs) returned by running the query in MEDLINE. We will perform our main tests[7] on the topics described in Table 3.1 using our sampling methods defined in Chapter 4. The reason we did not chose more topics is two-fold. First of all we are cross validating over the topics increasing reliability. Secondly, and linked to the first reason, due to our sampling and validating methods our experiments took around 20 hours to complete (including optimisation stages) which was all we could accommodate in terms of time.

Topic	Number of documents	% relevant at the abstract level
CD009519	5971	1.74%
CD011984	8192	5.54%
CD011975	8201	7.55%

Table 3.1: Characteristics our datasets.

6. http://www.cochranelibrary.com/
7. We do not optimise our algorithms on this data.

3.2 Feature representation and algorithms

Our classification task involves analysing a document represented as feature vectors and deciding which class it belongs to (relevant or irrelevant). These feature vectors can take many forms as we have seen in Chapter 2. However one of the most common representations is tf-idf, this is the one we will use. Moreover we will look at the functionality of our algorithms, namely: the Complement Naïve Bayes (CNB), Support Vector Machines (SVM), K-Nearest Neighbour (KNN), Adaboost and Neural Nets (NN)[8].

Tf-idf term weighting

A common assumption is that words that occur many times in a document will give useful information about the overall content of that document. However this is not always the case, some words can occur often but carry no real meaningful value (e.g. "and", "the"). These words will appear often in every document. Tf-idf term weighting compensates for this fact using the idea is that a term which occurs often in the document but is rare in the overall document collection will carry a high weight. The formula for tf-idf is:

$$tf\text{·}idf(t,d)=tf(t,d)\cdot idf(t)$$

where

$$idf(t)=\log\frac{n+1}{1+df(t)}+1$$

and $tf(t,d)$ is the number of times the term appears in the document, $df(t)$ is the number of documents the term appears in and n is the total number of documents in the document set.

Complement Naïve Bayes

The Complement Naïve Bayes classifier we will use is the one outlined by Rennie et al. in a 2003 study [36]. This algorithm is particularly suited for imbalanced data sets. The weight of a term, i, is calculated by summing the tf-idf values for that term over all the documents d that are not in the same class and adding a smoothing parameter, a_i. This value is then divided by that same computation for all the terms in the feature vector plus the sum of smoothing parameters for each term:

8. All of these algorithms are implemented by scikit-learn's libraries and we are explaining them using their outline.

$$w_{ci} = \log \frac{a_i + \sum_{j:y_j \neq c} d_{ij}}{a + \sum_{j:y_i \neq c} \sum_k d_{kj}}$$

The result is then normalised and that gives the weight for that term with respect to a class. This is done for all the terms. The document is given the class that is the poorest complement match:

$$c = \underset{c}{argmin} \sum_i t_i w_{ci}$$

Support Vector Machine

The Support Vector Machine approaches a classification task as follows : each document is represented as a vector of tf-idf values and each vector has a position in N-dimensional space (where N is the number of terms in the vector). The SVM finds the hypothesis with the lowest error, i.e. the decision boundary/hyperplane with the largest margin between the any instance. Each instance is classified based on its position relative to the hyperplane.

Specifically: given training vectors $x_i \in \mathbb{R}^p, i = 1, \ldots, n,$ in two classes, and a vector $y \in \{1, -1\}^n$, SVC solves the following primal problem:

$$\min_{w,b,\zeta} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \zeta_i$$

subject to $y_i (w^T \varnothing(x_i) + b) \geqslant 1 - \zeta_i, \zeta_i \geqslant 0, i = 1, \ldots, n$ [9]

K-Nearest Neighbour

To understand how the K-nearest neighbour works we can visualise the same N-dimentional space as described for the SVM. Each vector is represented by a position which will be classified based on the label of its neighbours; its k -nearest neighbours.

Euclidean distance is used to calculate the distance between positions :

$$dist(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

9 https://scikit-learn.org/stable/modules/svm.html#svm-kernels

where n is the number of training instances.

Adaboost

The idea behind the Adaboost algorithm is to use a set of poor performing classifiers (such as shallow decision trees) to form one strong classifier. It does this by iteratively choosing subsets of the training data based on previous predictions then weighs the classifiers based on their accuracy. The predictions of each of the weak learners are summed to produce a final classification. Specifically the Adaboost algorithms works as follows [37]:

1. Initialise the weights for the training instances:

$$w_i = \frac{1}{n} for\, i = 1,2.\dots, n$$

where n is the number of training samples

2. For $m = 1$ to M

a) Fit a classifier $T^{(m)}(x)$ to the training data using weights w_i .

b) Compute the error associated with the classifier:

$$err^{(m)} = \frac{\sum_{i=1}^{n} w_i \prod (c_i \neq T^{(m)}(x_i))}{\sum_{i=1}^{n} w_i}$$

c) Smooth the error:

$$a^{(m)} = \log \frac{1 - err^{(m)}}{err^{(m)}} + \log(K-1)$$

Where K is the number of classes.

d) Update the weights for $i = 1, \dots, n.$:

$$w_i \leftarrow w_i \cdot \exp(a^{(m)} \cdot \prod (c_i \neq T^{(m)}(x_i)))$$

e) Re-normalise w_i .

3. Output

$$C(x)=\underset{k}{argmax}\sum_{m=1}^{M} a^{(m)}\cdot\prod\left(T^{(m)}(x)=k\right)$$

<u>Neural Nets</u>

Neural networks have a number of layers where one neuron in a layer is the sum of the products of each neuron in the previous layer and their respective weights. The first layer is given the input in our case each neuron in the first layer will represent the tf-idf value for one term. Each neuron in the output layer represents the classes. An input instance will be given the class associated with the output neuron that has the highest value. Specifically:

$$argmax(output)$$

where

$$output=\sum_j w_{ij} o_j$$

where $output$ is the output vector consisting of all the activation values for each outer neuron.

When the network is trained the weights of the network are updated according to a loss function: the cost of a misclassification is propagated backwards and all the weights are updated to get closer to a correct classification.

3.3 Evaluation metrics

In order to evaluate a classifier's performance on a screening task such as this we compare it's classification to some gold standard where each document is labelled as relevant or irrelevant. There are 4 types of classifications for this task in terms of evaluation:

True Positive TP : Relevant document classified as relevant.
False Positive FP : Irrelevant document classified as relevant.
True Negative TN : Irrelevant document classified as irrelevant.
False Negative FN : Relevant document classified as irrelevant.

These four values are employed to make up the following metrics which we will use to evaluate our classifications:

Recall

Ratio of truly relevant documents classified to all relevant documents.

$$Recall = \frac{TP}{TP + FN}$$

Precision

Ratio of truly relevant documents classified to all documents classified as relevant.

$$Precision = \frac{TP}{TP + FP}$$

F-measure

A metric which combines recall and precision in such a way that we can weight one or the other.

$$Fß = \frac{(ß^2 + 1)TP}{(ß^2 + 1)TP + FP + ß^2 FN}$$

Where ß is the relative importance given to recall and precision, if ß = 1 then recall and precision are given equal weights, if ß > 1 then recall is favoured, for 0 < ß < 1 precision is favoured. As explained in previous sections if relevant documents are misclassified then this introduces bias in the review. As a consequence the optimisation criteria we have chosen is F5 (ß=5) which strongly favours recall but still puts some weight on precision.

Studies Scanned Out

Studies scanned out represents the percentage of studies that reviewers do not need to screen because they have been screened out by the classifier.

$$SO = \frac{TN + FN}{TP + FP + TN + FN}$$

Time

Time taken to screen in minutes.

$$time = t$$

Workload reduction

We will be looking at the results of classifiers which are trained on some percentage of the data and tested on the rest. This scenario is supposed to reflect what could happen if a team of human reviews label an initial portion of the documents retrieved from a query and then use TM methods to screen the rest. Thus our final evaluation metric will be the total amount of workload reduction relative to the full set of documents retrieved in a Boolean query:

$$Workload\, reduction = \frac{SO * percentage\, of\ studies\, screened\, using\, TM}{100}$$

Chapter 4: Implementation

Our tests are 3-fold with a pre-optimisation stage. Before each of the following experiments we optimise 5 different machine learning algorithms on data from previous reviews under a range of parameters, namely, CNB, SVM, Adaboost, KNN and NN. Our tests consist of two phase runs with different sampling methods. First we test our algorithms individually and then together as a voting system investigating the level of recall attainable, how different combinations behave and comparing the voting system with individual runs. These two phases are repeated with different sampling methods in order to investigate how the training set effects the result: first we use 50% of the data, then we use 65% and finally we randomly undersample the training set. This chapter outlines the specific methods we have chosen for our experiments and briefly talks about how they are implemented.

4.1 Sampling and validation

We implement and compare three different sampling methods. As discovered by Thomas G. Dietterich [33] 10-fold cross-validation can overestimate the performance of classifiers. Cohen et al. [32] found that a 5x2 cross-validation is more realistic. As detailed previously, this technique involves splitting the data randomly in half. One half is used as the training set and the other the test set. This process is repeated 5 times and results are averaged. We use 5x2 cross-validation as our baseline sampling method. However as this only provides half the data to train classifiers on, we implement an adapted version of 5x2 which uses 65% of the training data to train and 35% to train on. We refer to this second technique as “5x65/35 cross-validation”. Finally, we cross-validate 5 random undersampling runs (henceforth referred to as “5x2 undersampled cross-validation”), which involves taking out, at random, as many instances from the majority class as necessary in order for the number of instances in each class to be equal. Effectively it forces the classes to be balanced. We anticipate the undersampling will decrease precision as it pushes the decision boundary further away from the minority class however F-measure should increase. The implementation of 5x2 cross-validation is done by scikit-learn’s library for our optimisation phase. However we design our own implementation[10] for subsequent sampling methods and experiments for efficiency reasons.

4.2 Preprocessing

Amal Alharbi and Dr Mark Stevenson participated in the CLEF eHealth 2017 [38] task 2. To prepare their data they downloaded the documents for each topics represented by the PMIDs and extracted the text of the title, abstract and MeSH terms. The MeSH terms are sometimes indistinguishable from English words and so they were pre-processed to remove whitespace and punctuation as well as prefixed with the string “Mesh”. For example “Medical History Taking” would become

10. We make use of Python’s built in random library.

"MeshMedicalHistoryTaking". We use the data extracted for this task as well as the preprocessed mesh terms.

For our preprocessing we use a combination of scikit learn's library and our own functions. Each document undergoes the following steps: The Title, Abstract and MeSH terms are read in from text files and concatenated into one string. Our tokeniser strips the string of punctuation and transforms it to lower case. It then removes any single letters, digits or stop list words[11]. Finally each word is stemmed using a Lancaster Stemmer[12]. Our tokens are then processed to tf-idf values using scikit-learn's feature extraction library (from which we have called our tokeniser). The final representation of each document in the topic consists of a vector of tf-idf values for all the words in the training set.

4.3 Methods

4.3.1 Optimising the algorithms.

The CNB, SVM, Adaboost, KNN and NN we use all take different parameters which affect their performance. These parameters will often change in relation to the nature of the dataset. In order to avoid overfitting for one specific domain we optimise these algorithms on a range of topics from past reviews. We wanted to make our experiments highlight the possibilities of using text mining methods in real life situations. Hence, our optimisation approach is one that can be replicated with data from previous reviews which is available in the Cochrane library. The parameters we need to optimise are either chosen from a set of options (such as the kernel for the SVM) or number values (such as the smoothing parameter for the CNB). Thus for parameters chosen from a set we simply chose the best performing algorithm with regards to F5. For parameters which are number values we perform multiple optimisation runs converging the values to a best candidate[13]. We expect that the performance of the Adaboost and Neural Net to improve as the number of estimators (weak learners) and the number of hidden layers increase. Increasing these values will increase the computational load as such we set a time threshold to 2 hours (which since we are using 5-cross-validations only represents 24 minutes for one run). This would still be much faster than human screening: if a human takes 30 seconds to screen an abstract screening half the data we use here would take about 31 hours[14]. Table 4.1 shows the parameters we need to optimise as well as their type.

Algorithm	**Parameter**	**Value**
CNB	Smoothing parameter	Float
SVM	Kernel	Linear, RBF

11. The stop list we use is available at www.ncbi.nlm.nih.gov/books/NBK3827/table/pubmedhelp.T.stopwords/
12. The stemmer is imported from the NLTK library.
13. We have implemented our optimisation stage with the help of scikitlearn's GridSearchCV library.
14. Average number of studies in our topics*30/3600

	Penalty parameter for error	Float
KNN	Number of neighbours (K)	Integer
NN	Number of hidden layers	Integer
	Solver	Ibfgs, adam
Adaboost	N_estimators	Integer
	Algorithm	SAMME, SAMME.R

Table 4.1: Parameters to optimise for each algorithm

3.3.2 Two Phase Test

This two phase test is performed 3 times, one for each sampling methods: 5x2 cross-validation, 5x65/35 cross-validation and 5x2 undersampled cross-validation.

Individual analyses

Once we optimised our algorithms to achieve the best F5 we assess their performance on a different set of topics (imitating the real life situation where reviewers test their optimised algorithms on the current review). We will evaluate them on a number of metrics, namely; Recall, Precision, F5, SO and time taken. At this stage we do not expect the Adaboost, KNN or NN to perform as well as the CNB and SVM but we hoped that they would contribute to the next phase. Once the classification is performed we serialise the prediction vectors[15] and test classes for use in the next stage.

Voting

Each of our algorithms have very different approaches to making a classification as seen in section 3.2. Therefore there is a good chance some can detect positive instances that others can't and vice versa. Even if they have the same recall and precision the true positives may be different documents. Moreover a voting system can help to avoid hasty generalisation as it uses multiple decision boundaries. Thus by implementing a voting system we hope to unite the detections of true positives increasing the overall recall (and F5). We do expect precision to drop as more false positives will be included however our ambition is to increase recall while still providing a strong F5 and a good workload reduction. In order to directly compare the voting system with individual algorithms we look at exactly the same data splits as used for the individual testing. The voting system takes the prediction for multiple classifiers on the same data and adds all the positive classifications together. An instance is classed as positive if it has one or more votes.

15. The vector which states for each instance whether the classifier classed them as positive or negative.

Chapter 5: Results and Discussion

We begin this chapter by discussing the progress of our optimisation stage outlined in Chapter 4. Following that we analyse the results of our experiments bringing back some literature we reviewed in Chapter 2. Finally we conclude by suggesting possible further work for this study.

5.1 Optimisation Phase

We did not encounter any issues implementing our optimisation stage. We performed between 1 and 3 optimisation runs on our algorithms. The best parameters are shown in Table 5.1. Much to our surprise F5 did not continually increase with the Adaboost's number of estimators and the Neural Net's number of hidden layers. In fact as we passed a certain limit their performance got worse. This was helpful as it allowed us to stay well under our time budget and converge to a strong value.

Algorithm	Parameter	Value
CNB	Smoothing parameter	0.0075
SVM	Kernel	Linear
	Penalty parameter for error	0.07
K-NN	Number of neighbours (K)	3[16]
NN	Number of hidden layers	300
	Solver	Ibfgs
Adaboost	N_estimators	400
	Algorithm	SAMME.R

Table 5.1: Final parameter values for our algorithms.

5.2 Two Phase Test

Table 5.2 displays the performance of each algorithm under the specified metrics for the predefined topics on 5x2 cross-validation sampling. As expected the CNB and the SVM were the strongest algorithms. They outperformed the rest by at least 35.3% in terms of F5 and 42.3% in terms of recall. The SVM out performed CNB by having 4.9% better recall and only 0.5% lower precision, resulting in a higher F5 at 76.9%. It is worth noting that the CNB forms its predictions in 0.3 seconds whereas the SVM takes 4 minutes. As expected our other 3 algorithms executed poorly as they all had F5 values lower than 38% with the Adaboost performing the least well on 28.8% F5 and taking an average of 16 minutes for its runs. Interestingly, all except the CNB had better recall and F5 for topics with worse

16. We will now refer to the K-NN as the 3-NN.

class imbalance. This may be due to the fact that we optimised our algorithms on poorly balanced data to begin with.

Algorithm	Topic	Recall	Precision	F5	SO	Time (minutes)
CNB	CD009519	77.6%	24.2%	71.1%	94.7%	0.0
	CD011984	80.0%	22.5%	72.8%	80.4%	0.0
	CD011975	81.9%	26.2%	75.7%	76.5%	0.0
	Average	**79.8%**	**24.3%**	**73.2%**	**83.9%**	**0.0**[17]
SVM	CD009519	86.8%	22.2%	77.7%	93.5%	1.7
	CD011984	84.2%	22.7%	76.2%	79.5%	4.8
	CD011975	83.1%	26.6%	76.8%	76.6%	5.1
	Average	**84.7%**	**23.8%**	**76.9%**	**83.2%**	**3.9**
Adaboost	CD009519	35.1%	53.4%	35.5%	98.9%	11.1
	CD011984	24.0%	36.6%	24.3%	96.4%	18.4
	CD011975	26.3%	36.8%	26.6%	94.6%	15.9
	Average	**28.5%**	**42.3%**	**28.8%**	**96.6%**	**15.9**
Neural Net	CD009519	45.5%	64.2%	45.8%	98.8%	0.4
	CD011984	31.5%	42.8%	31.8%	95.9%	0.8
	CD011975	32.0%	40.3%	32.2%	94.0%	0.9
	Average	**36.3%**	**49.1%**	**36.6%**	**96.2%**	**0.7**
3-NN	CD009519	47.9%	45.6%	47.7%	98.3%	2.7
	CD011984	32.5%	39.9%	32.7%	95.5%	5.8
	CD011975	33.0%	37.9%	33.2%	93.5%	5.8
	Average	**37.5%**	**41.1%**	**37.9%**	**95.8%**	**4.8**

Table 5.2: Results from individual testing using 5x2 cross-validation.

As we can see in Table 5.3 the SVM and CNB remain predominant in our 5x65/35 cross-validation tests and the SVM is still the strongest. Surprisingly providing an extra 15% of the data for the classifiers to be trained on does not increase performance by a significant margin. All of the algorithms have better recall but also a larger loss in precision. In fact we only saw an increase in F5 for the CNB and the Adaboost. The other algorithms have lower F5 measures. The CNB is the only algorithm to have had a significant improvement: an increase of 3.2% recall and 2.3% F5. In addition, we can notice that once again the class imbalance is not directly related to the number of relevant documents retrieved.

17. . The CNB took an average of 0.005 minutes (0.3 seconds). Rounded to 1 decimal place is 0.0.

Algorithm	Topic	Recall	Precision	F5	SO	Time (minutes)
CNB	CD009519	80.6%	23.4%	73.5%	94.5%	0.0
	CD011984	86.4%	22.2%	77.7%	79.0%	0.0
	CD011975	82.0%	25.2%	75.4%	76.3%	0.0
	Average	**83.0%**	**23.6%**	**75.5%**	**83.3%**	**0.0**
SVM	CD009519	85.4%	20.8%	75.9%	93.4%	2.4
	CD011984	87.5%	21.8%	78.4%	78.3%	6.8
	CD011975	82.5%	26.3%	76.1%	77.2%	7.4
	Average	**85.1%**	**23.0%**	**76.8%**	**83.0%**	**5.5**
Adaboost	CD009519	40.3%	60.5%	40.7%	98.9%	16.9
	CD011984	28.1%	40.1%	28.4%	96.2%	27.3
	CD011975	29.2%	36.8%	29.5%	94.2%	27.0
	Average	**32.5%**	**45.8%**	**32.9%**	**96.4%**	**23.7**
Neural Net	CD009519	45.8%	68.8%	46.4%	99.0%	0.4
	CD011984	30.0%	39.2%	30.3%	95.9%	1.2
	CD011975	34.8%	39.8%	35.0%	93.7%	1.4
	Average	**36.9%**	**49.3%**	**37.2%**	**96.2%**	**1.0**
3-NN	CD009519	48.7%	40.0%	48.1%	98.1%	3.1
	CD011984	33.9%	37.8%	35.0%	95.0%	5.6
	CD011975	33.4%	38.1%	33.5%	93.6%	5.6
	Average	**39.0%**	**38.6%**	**38.9%**	**95.6%**	**4.8**

Table 5.3: Results from individual testing using 5x65/35 cross-validation.

The results from our undersampling phase have some astronomical differences compared to the last two phases. Our best algorithm in terms of recall is the 3-NN with 93.3% a baffling 8.2% higher than the SVM in our second phase. It also maintained a reasonable F5 to achieve this level of recall. CNB also has a strong recall with 92.4% and a stronger F5 than the 3-NN with 73.8%. Our best performing algorithm in terms of F5 is the Neural Net with 75.5%, however, with 87.3% recall this statistic is largely due to its precision.

Algorithm	Topic	Recall	Precision	F5	SO	Time (minutes)
CNB	CD009519	95.4%	6.6%	62.5%	76.5%	0.0
	CD011984	92.1%	17.2%	78.9%	71.5%	0.0
	CD011975	89.6%	22.1%	80.1%	69.5%	0.0
	Average	**92.4%**	**15.3%**	**73.8%**	**72.5%**	**0.0**
SVM	CD009519	79.7%	27.7%	74.2%	95.4%	0.0

	CD011984	73.8%	24.8%	68.5%	84.1%	0.1
	CD011975	69.8%	31.7%	66.7%	83.4%	0.2
	Average	**74.4%**	**28.1%**	**69.8%**	**87.6%**	**0.1**
Adaboost	CD009519	93.6%	11.2%	72.6%	86.4%	1.0
	CD011984	80.3%	17.1%	70.3%	74.9%	1.5
	CD011975	81.0%	21.6%	73.2%	71.8%	1.7
	Average	**85.0%**	**16.6%**	**72.0%**	**77.7%**	**1.4**
Neural Net	CD009519	95.4%	13.1%	76.0%	88.0%	0.0
	CD011984	83.5%	19.6%	74.2%	77.2%	0.0
	CD011975	82.9%	25.4%	76.2%	75.5%	0.0
	Average	**87.3%**	**19.4%**	**75.5%**	**80.2%**	**0.0**
3-NN	CD009519	97.1%	3.9%	50.4%	59.8%	0.0
	CD011984	93.2%	12.6%	74.7%	60.6%	0.2
	CD011975	89.6%	17.3%	77.2%	61.0%	0.3
	Average	**93.3%**	**11.3%**	**67.4%**	**60.5%**	**0.2**

Table 5.4: Results from individual testing using 5x2 undersampled cross-validation.

These results conclude our individual experiments for each of these algorithms. Our findings reinforce what was summarised in Chapter 2: the CNB and SVM are the strongest candidates in using TM for the screening of studies ([13] [14] [17] [20]). The CNB proves to perform well under all sampling methods and is also extremely fast. Moreover, it may benefit from a augmented training set. The SVM outperformed the CNB in the first two phases but not when undersampling. As these findings form conflicting reports as to which one is better we cannot validate any results on which one is the strongest. However we can recommend the SVM when there has been no undersampling and the CNB otherwise. The Adaboost, 3-NN and Neural Net which all performed poorly with the first 2 sampling methods are much stronger when the training set is undersampled showing that class imbalance can seriously hinder their efficiency. In fact, the KNN may be a surprising alternative to the SVM when undersampling. Finally we can see that class imbalance within a particular topic is not directly related to the performance of the algorithms, some performed best where the imbalance was most extreme. This may be due to the way we optimised the algorithms or due to the nature of the datasets. Taking our best performing algorithm in terms of recall we have achieved a workload reduction of 41.6% at a recall of 84.7% with our 5x2 cross-validation sampling method, a reduction in workload of 29.1% at a recall of 85.1% using 65% of the data and 46.7% at 93.3% recall using undersampling. However, none of these algorithms on their own meet the minimum of 95% recall needed to be acceptable by human reviewers. As a consequence further improvements are needed to achieve our main goal. On the other hand we have managed to critically analyse how these individual algorithms behave with different sampling methods and datasets achieving our second goal.

<u>Voting System</u>

Table 5.5 shows the results of our voting system for 5x2 cross-validations. Using all 5 algorithms we achieve an average of 90.5% recall and 79.7% F5, meaning we have managed to increase recall by 5.8% and F5 by 2.8% compared to our best performing individual algorithm (SVM) using this sampling method. The performance of the voting system in terms of F5 drops slightly as we remove votes however precision does increase. We also observe that the CNB and SVM alone complement each other by 5.1% recall and 2.5% F5. Moreover despite the fact that the Adaboost and NN perform poorly individually their united F-measure is 45.7% which is 9.1% higher than the NN on its own (the best of the two) and a strong increase relative to their individual scores. This demonstrates they find positive instances that their counterpart does not find without compromising precision.

Algorithms	Recall	Precision	F5	SO
5 Voters				
CNB, AdaBoost, NN, 3NN, SVM	**90.5%**	20.4%	**79.7%**	79.2%
4 Voters				
CNB, AdaBoost, NN, 3NN	84.1%	23.5%	76.4%	82.7%
CNB, AdaBoost, NN, SVM	90.1%	20.7%	79.5%	79.5%
CNB, AdaBoost, 3NN, SVM	**90.4%**	20.4%	79.6%	79.3%
CNB, NN, 3NN, SVM	**90.4%**	20.6%	**79.7%**	79.5%
AdaBoost, NN, 3NN, SVM	85.8%	23.0%	77.4%	82.4%
3 Voters				
CNB, Adaboost, NN	82.7%	24.1%	75.4%	83.2%
CNB, Adaboost, 3NN	83.7%	23.5%	76.1%	82.7%
CNB, Adaboost, SVM	90.0%	20.7%	79.4%	79.6%
CNB, NN, 3NN	82.9%	23.7%	75.5%	83.0%
CNB, NN, SVM	90.0%	20.8%	79.5%	79.8%
CNB, 3NN, SVM	**90.2%**	20.6%	**79.6%**	79.5%
Adaboost, NN, 3NN	55.6%	37.3%	54.5%	92.6%
Adaboost, NN, SVM	85.1%	23.5%	77.1%	82.8%
Adaboost, 3NN, SVM	85.6%	23.1%	77.3%	82.4%
NN, 3NN, SVM	85.5%	23.3%	77.3%	82.7%
2 Voters				
CNB, Adaboost	81.5%	24.0%	74.4%	83.4%
CNB, NN	81.1%	24.3%	74.3%	83.7%
CNB, 3NN	82.4%	23.7%	75.1%	83.2%
CNB, SVM	**89.8%**	20.9%	**79.4%**	79.8%

Adaboost, NN	45.9%	42.8%	45.7%	94.5%
Adaboost, 3NN	49.2%	37.9%	48.6%	93.9%
Adaboost, SVM	84.9%	23.5%	76.9%	82.9%
NN, 3NN	49.7%	39.9%	49.2%	93.9%
NN, SVM	84.9%	23.8%	77.1%	83.1%
3NN, SVM	85.4%	23.4%	77.2%	82.7%

Table 5.5: Averages over all topics for our voting system applied on 5x2 cross-validation runs.

When training on 65% of the data, recall increases to 90.7% but F5 falls to 79.2% suggesting that the ratio of false positives to true positives is more severe when there is less test data. Table 5.2 and Table 5.5 show conclusively that, using these sampling methods, the union of any of these algorithms results in an increase in recall and at least equal F5 compared to any of the individual algorithms. They also show that the union of any combination of algorithm is not as strong in terms of recall as their combination with an additional voter. This shows that each of these algorithms can complement each other even by a few true positives. For any number of voters the best F5 is achieved when both CNB and SVM are voting. The second tier of good results occur when either the CNB or SVM are voting. Poor results occur when neither are voting. When 2 are voting the second best is achieved with SVM and 3NN. When 3 are voting CNB, SVM and 3NN are the best performers, followed by CNB, SVM and NN. When 4 are voting CNB, NN, 3NN and SVM are the best followed closely by CNB, 3NN, Adaboost and SVM. Therefore, we can deduce that the strength of the combinations of any algorithms is directly related to how well they perform individually. Thus, there is no combination such that their combined scores are augmented enough to outperform alternative combinations of stronger individual algorithms.

Algorithms	Recall	Precision	F5	SO
5 Voters				
CNB, AdaBoost, NN, 3NN, SVM	**90.7%**	19.5%	**79.2%**	78.9%
4 Voters				
CNB, AdaBoost, NN, 3NN	85.8%	22.3%	77.2%	82.0%
CNB, AdaBoost, NN, SVM	89.9%	19.8%	78.8%	79.3%
CNB, AdaBoost, 3NN, SVM	90.5%	19.5%	79.0%	79.0%
CNB, NN, 3NN, SVM	**90.6%**	19.6%	**79.2%**	79.1%
AdaBoost, NN, 3NN, SVM	87.0%	22.3%	77.9%	82.2%
3 Voters				
CNB, Adaboost, NN	84.5%	23.2%	76.6%	82.7%
CNB, Adaboost, 3NN	85.5%	22.4%	77.0%	82.2%
CNB, Adaboost, SVM	89.8%	19.8%	78.7%	79.3%

CNB, NN, 3NN	85.1%	22.6%	76.7%	82.4%
CNB, NN, SVM	89.8%	19.9%	78.8%	79.4%
CNB, 3NN, SVM	**90.4%**	19.6%	**79.0%**	79.1%
Adaboost, NN, 3NN	57.8%	35.6%	56.3%	92.3%
Adaboost, NN, SVM	85.8%	22.7%	77.3%	82.6%
Adaboost, 3NN, SVM	86.7%	22.3%	77.7%	82.3%
NN, 3NN, SVM	86.7%	22.4%	77.8%	82.3%
2 Voters				
CNB, Adaboost	84.1%	23.3%	76.3%	82.9%
CNB, NN	83.6%	23.5%	76.0%	83.1%
CNB, 3NN	84.7%	22.6%	76.4%	82.5%
CNB, SVM	**89.7%**	19.9%	**78.7%**	79.5%
Adaboost, NN	47.6%	43.6%	47.3%	94.5%
Adaboost, 3NN	51.2%	36.4%	50.2%	93.5%
Adaboost, SVM	85.4%	22.8%	77.0%	82.8%
NN, 3NN	51.3%	37.3%	50.5%	93.6%
NN, SVM	85.5%	22.9%	77.1%	82.8%
3NN, SVM	86.4%	22.4%	77.5%	82.5%

Table 5.6: Averages over the topics for our voting system applied on 5-65/35 cross-validation runs

Table 5.7 shows the results of each combination of algorithms when the data is undersampled. Using 5 voters we have achieved an average recall of 98.2%, 7.5% higher than in the previous phase. However F5 decreased by 11.1% showing that many more false positives were included. We also notice that there is no visible contribution by SVM between 4 and 5 votes. In fact our best result is achieved with the CNB, NN, 3-NN and Adaboost where we managed to reach 99.2% recall on the topic CD009519 whereby the minority class only makes up 1.74% of the total amount of documents. These results are the same with or without the SVM. Moreover we can see that any combination of 4 voters attains at least 97.5% recall and a workload reduction of 26.6%.

Algorithms	Recall	Precision	F5	SO
5 Voters				
CNB, AdaBoost, NN, 3NN, SVM	**98.2%**	9.9%	**68.1%**	53.2%
4 Voters				
CNB, AdaBoost, NN, 3NN	**98.2%**	9.9%	68.1%	53.2%
CNB, AdaBoost, NN, SVM	96.8%	12.7%	**73.6%**	65.5%
CNB, AdaBoost, 3NN, SVM	98.1%	9.9%	68.1%	53.3%
CNB, NN, 3NN, SVM	97.8%	10.5%	68.7%	55.6%

AdaBoost, NN, 3NN, SVM	97.5%	10.3%	68.6%	55.5%
3 Voters				
CNB, Adaboost, NN	96.7%	12.7%	73.6%	65.6%
CNB, Adaboost, 3NN	**98.1%**	9.9%	68.0%	53.4%
CNB, Adaboost, SVM	95.9%	13.0%	73.5%	66.4%
CNB, NN, 3NN	97.7%	10.5%	68.7%	55.7%
CNB, NN, SVM	95.6%	14.3%	74.9%	69.8%
CNB, 3NN, SVM	97.2%	10.6%	68.5%	55.6%
Adaboost, NN, 3NN	97.3%	10.3%	68.5%	55.6%
Adaboost, NN, SVM	93.9%	15.1%	**76.5%**	72.6%
Adaboost, 3NN, SVM	97.2%	10.4%	68.6%	56.0%
NN, 3NN, SVM	96.7%	11.1%	69.2%	58.5%
2 Voters				
CNB, Adaboost	95.8%	13.0%	73.4%	66.6%
CNB, NN	95.4%	14.4%	74.8%	70.1%
CNB, 3NN	**97.1%**	10.6%	68.5%	56.1%
CNB, SVM	93.4%	15.1%	74.3%	71.8%
Adaboost, NN	92.6%	15.1%	**75.5%**	73.1%
Adaboost, 3NN	96.7%	10.4%	68.3%	56.3%
Adaboost, SVM	89.8%	16.4%	75.4%	75.9%
NN, 3NN	96.1%	11.1%	68.9%	58.8%
NN, SVM	90.3%	19.1%	77.7%	79.0%
3NN, SVM	95.2%	11.2%	68.6%	59.7%

Table 5.7: Averages over the topics for our voting system applied on 5x2 undersampled cross-validation runs

This data shows the potential of a voting system. Using undersampling and the combination of the CNB, 3NN, NN and Adaboost we have achieved an average of 98.2% recall and a workload reduction of 26.6%. Our top score is 99.2% recall which was the same as Frunza et al.'s [31] highest recall and better than Razavi et al's [16] using their methods and datasets. In this instance, with 54.7% of the studies scanned out we achieved a workload reduction of 27.4% at 99.2% recall. To put things into perspective if an experienced reviewer takes 30 seconds to screen a study, by reducing the workload by 27.4% we can effectively save 13.7 hours of screening time at 99.2% recall. As Wallace et al. [5] mentioned, complicated abstract may take much longer to screen multiplying the time needed several fold (and the time saved using this method). Therefore, this statistic is our absolute lower bound of possible time saved. Moreover if we set the threshold at 95% we can see that the CNB and NN scanned out 70.1% of the studies with 95.4% recall resulting in a workload reduction of 35.5%. However, we would argue that due to the nature of this task and the risk of including bias only the highest level of recall should be considered as applicable in real world scenarios. These results conclude our experiments and fulfil our primary goal by achieving strong workload reduction percentages with >95% recall.

Limitations

Unfortunately, we could not check all the boxes provided by Olorisade et al. [15] for enabling the reproducibility of text mining studies. This was mainly due to the fact that we could not provide links to the raw and cleaned data set because of our limited storage resources. However, by specifying our methods as well as providing the code we ran to acquire our results and the specific topics we used, our results are reproducible, taking into account the stochastic nature of our sampling methods. Moreover, our results are largely a consequence of our optimisation techniques and the nature of the data. In order to validate them further we would need to test the same optimisation techniques on different data.

5.3 Further Research

As we conclude this Chapter, it is important to note some improvements that could be made to the system we have implemented. Changes could be made to the feature representation and selections schemes, to the algorithms used, to our sampling methods or to the state of the initial and final data. As a feature representation scheme we chose TF-IDF, further possible work could be to use alternative representations such as UMLS or 2nd order co-occurrence. Moreover we chose all the words in the document minus those in a stop list to be features in our vectors. A more rigorous feature selection scheme such as the use of the CHI2 statistic may improve results. Furthermore in order to assess the effectiveness of algorithms compared to the ones we used we could use alternative machine-learning algorithms on this dataset using our methods. This could also provide us with an extra "voter" possibly increasing recall even further. Miwa et al. [27] used aggressive undersampling which is a more developed form of undersampling compared to random undersampling. As such we could try to repeat our experiments with aggressive undersampling. What's more, the voting system we implemented could also be extended to return a ranked list of positive instances whereby studies with more votes appear first. This would most likely increase the rate of screening as the top section of the list would be easily labelled. Finally, the implementation of a label propagation algorithm, such as the one used by Kontonatsios et al. [22], could reduce the reviewer's workload further by eliminating the need of such a large initial training set.

Chapter 6: Conclusion

In this study we have investigated the use of text mining in the screening of studies for the inclusion in systematic reviews. We had two goals. Firstly to assess the workload reduction achievable while also maintaining a recall of at least 95%. Our second goal was to critically analyse how different machine-learning algorithms, namely the Complement Naïve Bayes, Support Machine Vector, Adaboost, K-Nearest Neighbour and Neural Net behave with different sampling methods and datasets. The CNB and SVM appear to be the best algorithms to use when there is no undersampling while the others are unsuitable. In fact the SVM performed best with 84.7% recall and 76.9% F5. On the other hand, the CNB performs its predictions extremely quickly. Providing an augmented training set for the classifiers to train on may only benefit the CNB by a considerable amount. Other forms of optimisation should be considered. Interestingly the class balance in the dataset is not directly related to the efficiency of the classifiers, in fact most proved to perform better on the most imbalanced topic. This suggests that optimisation techniques and the nature of the data have a strong role to play in text mining tasks. When we undersampled the training set we saw a dramatic increase in effectiveness for all algorithms except the SVM. In fact the KNN may be a suitable alternative to the SVM using this sampling method. The fact that the SVM performed poorly was likely due to our optimisation method. Using a different combination of parameters we expect the SVM could be stronger. We noticed a strong increase in recall and F-measure once we implemented our voting system with 90.5% recall and 79.7% F5 using our first sampling method. By implementing our voting system using the first two sampling methods, we found that the union of any of these algorithms results in an increase in recall and at least equal F5 compared to any of the individual algorithms. They also show that the union of any combination of algorithm is not as strong in terms of recall as their combination with an additional voter. Moreover the strength of the combinations of any algorithms is directly related to how well they perform individually. We noticed that as we removed voters precision increased but F5 dropped. Finally, the use of a voting system on undersampled data seems to be a very strong candidate in reducing a reviewer's workload at >95% recall. When we used the CNB, Adaboost, NN and KNN in combination using undersampled data we achieved an average of 98.2% recall and our best score was 99.2% recall with a workload reduction of 27.4%. This amounted to a lower bound of 13.7 reviewing hours saved but could possibly be a lot higher. We are confident that with further developments these techniques could be augmented to achieve an even higher workload reduction at very strong levels of recall. Therefore this study may have helped to provide the foundations for a text mining system to be used for future systematic reviews.

References

[1] S. Jonnalagadda, “NIH Public Access,” vol. 6, no. 0, pp. 5–17, 2014.

[2] I. Mierswa, “Evolutionary Learning with Kernels :,” pp. 1553–1560, 2006.

[3] D. Gough and D. Elbourne, “Systematic Research Synthesis to Inform Policy , Practice and Democratic Debate,” pp. 225–236, 2002.

[4] H. Bastian, P. Glasziou, and I. Chalmers, “Seventy-Five Trials and Eleven Systematic Reviews a Day : How Will We Ever Keep Up ?,” vol. 7, no. 9, 2010.

[5] B. C. Wallace, T. A. Trikalinos, J. Lau, C. Brodley, and C. H. Schmid, “Semi-automated screening of biomedical citations for systematic reviews,” *BMC Bioinformatics*, vol. 11, 2010.

[6] J. McGowan and M. Sampson, “Systematic reviews need systematic searchers,” vol. 93, no. January, 2005.

[7] W. Jaidee, D. Moher, and M. Laopaiboon, “Time to Update and Quantitative Changes in the Results of Cochrane Pregnancy and Childbirth Reviews,” vol. 5, no. 7, pp. 1–7, 2010.

[8] A. O’Mara-Eves, “Using text mining for study identification in systematic reviews: a systematic review of current approaches,” *ACM SIGGRAPH 2016 Posters - SIGGRAPH ’16*, pp. 1–2, 2015.

[9] I. Shemilt *et al.*, “Pinpointing needles in giant haystacks : use of text mining to reduce impractical screening workload in extremely large scoping reviews,” no. June, 2013.

[10] E. Erinoff, J. Lege-matsuura, S. Potter, R. Paynter, and L. L. Ba, “Commentary on EPC methods : an exploration of the use of text-mining software in systematic reviews,” vol. 84, pp. 33–36, 2017.

[11] B. E. Howard *et al.*, “SWIFT-Review : a text-mining workbench for systematic review,” December, 2016.

[12] J. Thomas *et al.*, “Living systematic reviews : 2 . Combining human and machine effort,” vol. 91, 2017.

[13] S. Matwin *et al.*, “A new algorithm for reducing the workload of experts in performing systematic reviews,” pp. 446–453, 2010.

[14] O. Frunza, D. Inkpen, and S. Matwin, “Building systematic reviews using automatic text classification techniques,”. August, pp. 301–311, 2010.

[15] B. K. Olorisade, P. Brereton, and P. Andras, “Reproducibility of studies on text mining for citation screening in systematic reviews : Evaluation and checklist,” *J. Biomed. Inform.*, vol. 73, pp. 1–13, 2017.

[16] A. H. Razavi, S. Matwin, D. Inkpen, and A. Kouznetsov, “Parameterized contrast in second order soft co-occurrences: A novel text representation technique in text mining and knowledge extraction,” *ICDM Work. 2009 - IEEE Int. Conf. Data Min.*, pp. 471–476, 2009.

[17] T. Bekhuis and D. Demner-fushman, “Artificial Intelligence in Medicine Screening nonrandomized studies for medical systematic reviews : A comparative study of classifiers,” *Artif. Intell. Med.*, vol. 55, pp. 197–207, 2012.

[18] K. R. Felizardo, G. F. Andery, F. V Paulovich, R. Minghim, and J. C. Maldonado, “A visual analysis approach to validate the selection review of primary studies in systematic reviews,” *Inf. Softw. Technol.*, vol. 54, no. 10, pp. 1079–1091, 2012.

[19] G. Arji, R. Safdari, H. Rezaeizadeh, and A. Abbassian, “A systematic literature review and classification of knowledge discovery in traditional medicine - Computer Methods and Programs in Biomedicine,” *Comput. Methods Programs Biomed.*, vol. 168, pp. 39–57, 2019.

[20] J. J. García Adeva, J. M. Pikatza Atxa, M. Ubeda Carrillo, and E. Ansuategi Zengotitabengoa, “Automatic text classification to support systematic reviews in medicine,” *Expert Syst. Appl.*, vol. 41, no. 4 PART 1, pp. 1498–1508, 2014.

[21] G. Mujtaba *et al.*, “Clinical text classification research trends : Systematic literature review and open issues,” vol. 116, pp. 494–520, 2019.

[22] G. Kontonatsios *et al.*, “A semi-supervised approach using label propagation to support citation screening,” *J. Biomed. Inform.*, vol. 72, pp. 67–76, 2017.

[23] J. Liu, P. Timsina, and O. El-gayar, “A comparative analysis of semi-supervised learning : The case of article selection for medical systematic reviews,” pp. 195–207, 2016.

[24] M. Sehatkar, A. Kouznetsov, L. Seaward, and P. O. Blenis, “Classifiers and Collective Ranking Techniques,” pp. 1–4.

[25] F. Park, B. Laboratories, L. Technologies, M. Hill, and Y. Freund, “Selective Sampling Using the Query by Committee,” , pp. 133–168, 1997.

[26] O. Chapelle, “Cluster Kernels for Semi-Supervised Learning.”

[27] M. Miwa, J. Thomas, A. O’Mara-Eves, and S. Ananiadou, “Reducing systematic review workload through certainty-based screening,” *J. Biomed. Inform.*, vol. 51, pp. 242–253, 2014.

[28] K. Zhou, A. Martin, Q. Pan, and Z. Liu, “International Journal of Approximate Reasoning SELP : Semi-supervised evidential label propagation algorithm for graph data clustering ” *Int. J. Approx. Reason.*, vol. 92, pp. 139–154, 2018.

[29] B. C. Wallace, K. Small, C. E. Brodley, and T. A. Trikalinos, “Active learning for biomedical citation screening,”, pp. 173-181, 2010.

[30] N. Japkowicz, “Learning from Imbalanced Data Sets : A Comparison of Various Strategies,” pp. 10–15, 2000.

[31] O. Frunza, D. Inkpen, S. Matwin, W. Klement, and P. O’Blenis, “Exploiting the systematic review protocol for classification of medical abstracts,” *Artif. Intell. Med.*, vol. 51, no. 1, pp. 17–25, 2011.

[32] A. M. Cohen, W. R. Hersh, K. Peterson, and Po-Yin Yen, “Reducing Workload in Systematic Review Preparation Using Automated Citation Classification,” pp. 206–219.

[33] T. Dietterich, “Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms.”, 1997.

[34] E. Kanoulas, D. Li, L. Azzopardi, and R. Spijker, "CLEF 2018 Technologically Assisted Reviews in Empirical Medicine Overview," 2018.

[35] M. M. G. Leeflang, J. J. Deeks, Y. Takwoingi, and P. Macaskill, "Cochrane diagnostic test accuracy reviews," *Syst. Rev.*, vol. 2, no. 1, pp 1-6, 2013.

[36] J. D. M. Rennie, L. Shih, J. Teevan, and D. R. Karger, "Tackling the Poor Assumptions of Naive Bayes Text Classifiers,", 2003.

[37] M. Adaboost, J. Zhu, H. Zou, S. Rosset, and T. Hastie, "Multi-class AdaBoost," vol. 2, pp. 349–360, 2009.

[38] A. Alharbi and M. Stevenson, "Ranking abstracts to identify relevant evidence for systematic reviews: The University of Sheffield's approach to CLEF eHealth 2017 Task 2: Working notes for CLEF 2017,", 2017.