Ollscoil *na* hÉireann, Gaillimh
National University of Ireland, Galway

The Evolution and Analysis of Term-Weighting Schemes in Information Retrieval

Ronan Cummins

Supervisor: Colm O'Riordan

May, 2008

A dissertation submitted to the National University of Ireland, Galway
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

Contents

Abstract

Information Retrieval is concerned with the return of relevant documents from a document collection given a user query. Term-weighting schemes assign weights to keywords (terms) based on how useful they are likely to be in identifying the topic of a document and are one of the most crucial aspects in relation to the performance of Information Retrieval systems. Much research has focused on developing both term-weighting schemes and theories to support them.

Genetic Programming is a biologically-inspired search algorithm useful for searching large complex search spaces. It uses a darwinian-inspired survival of the fittest approach to search for solutions of a suitable fitness. This thesis outlines experiments that use Genetic Programming to search for term-weighting schemes. A study of term-weighting schemes in the literature is undertaken and consequently, the function space is separated into three areas that represent three fundamental concepts in term-weighting.

Experiments using Genetic Programming to search these three function spaces show that term-weighting schemes that outperform state of the art term-weighting benchmarks can be found. These experiments also show that the new term-weighting schemes have general properties as they achieve high performance on unseen test data.

An analysis of the solution space of the term-weighting schemes shows that the evolved solutions exist in a different part of the space than the current benchmarks. These experiments show that the Genetic Programming approach consistently evolves solutions that return similar ranked lists in each of the three function spaces.

Furthermore, the best performing term-weighting schemes are formally

analysed and are shown to satisfy a number of axioms in Information Retrieval. A detailed analysis of the existing axioms is presented together with some amendments and additions to the existing axioms. This analysis aids in theoretically validating the term-weighting schemes evolved in the framework.

Finally, a secondary application of Genetic Programming to Information Retrieval is presented to show the potential for Genetic Programming in addressing other issues in Information Retrieval. This experiment shows that Genetic Programming can be used to combine further evidence in the retrieval process to enhance performance. This approach evolves schemes for use with two automatic query expansion techniques to increase retrieval effectiveness.

Acknowledgements

During my research I have received much help and support from people both within the Department of Information Technology and outside it. I would like to thank all of those people, to whom I am forever indebted. In particular, I would like to thank Colm O'Riordan without whose guidance, knowledge, encouragement and enthusiasm, this thesis would never have been possible. He has been of invaluable benefit throughout this journey.

I would like to thank all those who have given me feedback on this work. It has been greatly appreciated. I would also like to thank those with whom I shared the lab during the course of this work. They have been of invaluable benefit for discussions and collaboration. This work was carried out with the support of IRCSET (the Irish Research Council for Science, Engineering and Technology) under the Embark Initiative. I would like to thank them for their support.

Finally, the support from my friends and family has been continuous and unflinching. I cannot thank them enough.

Publications

Cummins R, O'Riordan C. (2004a) Determining general term-weighting schemes for the vector space model of information retrieval using genetic programming. In: McGinty L (ed) 15th Artificial Intelligence and Cognitive Science Conference (AICS 2004), Galway-Mayo Institute of Technology, Castlebar Campus, Ireland.

Cummins R, O'Riordan C. (2004b) Using genetic programming to evolve weighting schemes for the vector space model of information retrieval. In: Keijzer M (ed) Late Breaking Papers at the 2004 Genetic and Evolutionary Computation Conference, Seattle, Washington, USA.

Cummins R, O'Riordan C. (2005a) Developing local term-weighting schemes in information retrieval using genetic programming. Technical Report NUIG-IT-211205, National University of Ireland, Galway, Ireland.

Cummins R, O'Riordan C. (2005b) An evaluation of evolved term-weighting schemes in information retrieval. In: Herzog O, Schek HJ, Fuhr N, Chowdhury A, Teiken W (eds) CIKM 05: Proceedings of the 14th ACM international conference on Information and knowledge management, ACM press, Bremen, Germany, pp 305-306.

Cummins R, O'Riordan C. (2005c) Evolving co-occurrence based query expansion schemes in information retrieval using genetic programming. In:Creaney N (ed) The 16th Irish conference on Artificial Intelligence and Cognitive Science (AICS05), University of Ulster, School of Computing and Information

Engineering, University of Ulster, pp 137-146.

Cummins R, O'Riordan C. (2005d) Evolving general term-weighting schemes for information retrieval: Tests on larger collections. Artificial Intelligence Review 24(3-4):277-299.

Cummins R, O'Riordan C. (2006a) Evolving local and global weighting schemes in information retrieval. Information Retrieval 9(3):311-330.

Cummins R, O'Riordan C. (2006b) A framework for the study of evolved term-weighting schemes in information retrieval. In: Stein B, Kao O (eds) TIR-06 Text based Information Retrieval, Workshop. ECAI 2006, Riva del Garda, Italy.

Cummins R, O'Riordan C. (2006c) Term-weighting in information retrieval using genetic programming: A three stage process. In: The 17th European Conference on Artificial Intelligence, ECAI-2006, Riva del Garda, Italy.

Cummins R, O'Riordan C. (2007a) Evolved term-weighting schemes in information retrieval: an analysis of the solution space. Artificial Intelligence Review 26(1-2):35-47.

Cummins R, O'Riordan C. (2007b) An axiomatic comparison of learned term-weighting schemes in information retrieval. In: Delany SJ, Madden M (eds) 18th Irish Conference on Artificial Intelligence and Cognitive Science, Dublin Institute of Technology.

Cummins R, O'Riordan C. (2007c) An axiomatic study of learned term-weighting schemes. In: SIGIR07 workshop on learning to rank for information retrieval (LR4IR-2007), Amsterdam, Netherlands, pp 11-18.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Since the inception of the World Wide Web, there has been an huge increase in the amount of information available in electronic format. The time spent on-line searching for specific information has greatly increased and has led to frustration for many users. Searchers are often overwhelmed by the amount of material returned when searching for information on a particular topic. This problem of *information overload* has been identified as a serious problem in many facets of daily life. For both professional and leisure activities, the Web is increasingly becoming a more vital resource for acquiring information and knowledge on a vast array of topics. The ease at which information can be added in an unstructured format to the Web and other digital repositories is one of the reasons for the large increase in available information. Documents can be easily uploaded to sites with little or no modification. The relatively loose structure of simple Web pages and the reluctance of authors to conform to rigid structure when creating their documents has led to the popularity of document creation and sharing over the Web. Conversely however, it is this unstructured format and the large volume of information available that has led to problems with the retrieval of specific information.

Information Retrieval (IR) systems deal with natural language documents and queries and attempt to limit this overload by automatically returning

only those documents that are relevant to a user's need (query). Humans have the ability to correctly interpret ambiguous phrases and sentences in natural language. However, the automatic retrieval of natural language documents poses many problems. Polysemy (the same word having different meanings) and synonymy (multiple words having the same meaning) are two such phenomena that are problematic for the automatic retrieval of information. These phenomena can lead to ambiguity in natural language and therefore, automated systems have difficultly resolving such complexities. When a word has multiple meanings or senses (polysemy), it is difficult for an automated system to correctly choose which meaning is intended by the author. Conversely, when many words relate to the same concept (synonymy), it is difficult for such a system to map the different words to the same correct concept (meaning). It is these generalities and specialties, and indeed the various depths that resolving them can entail, that cause problems for IR systems.

Traditionally, IR was only a concern of information gatherers and catalogers such as librarians. However, with the advent of the World Wide Web, problems in the field of IR are increasingly seen as interesting real-world problems and have been brought to the forefront for many institutions and organisations. The main objective of traditional text-based IR is to return all and only relevant documents related to given query. IR is generally broader than modern Web Search; the aim of which is usually to satisfy a user's need as quickly as possibly (where users are typically not interested in *all* relevant documents). However, in many other professions, such as the legal and medical professions, searchers often need to find all relevant documents for a given information need. Current Web search engines are a product of continued research in traditional IR and remains a vibrant field today.

Many IR systems are based on the simplistic vector space model of retrieval (Salton et al, 1975). In this model, query terms (keywords) are weighted on how important they are likely to be in measuring information content and are then matched to terms in the documents. Documents with more occurrences of these weighted terms are scored higher. The scores for the terms in each document are aggregated giving a final score for the doc-

ument. Finally, a ranked list of documents is returned to the user with the document with the highest score ranked first. This approach, while simplistic, has the advantage of being effective, efficient and easy to implement. These types of approaches are often referred to as "bag of words" approaches due of the fact that the terms are treated independently of one another.

As mentioned, these "bag of words" approaches use some method to assign weights to terms which reflect the importance of these terms in determining the topic of the document. These methods of assigning weights are typically called *term-weighting schemes* or *term-weighting functions*. In the modern Web setting, they are sometimes referred to as *ranking functions*. Ultimately, it is the weights assigned to these terms that are crucial to the performance of these types of IR systems (Salton and Buckley, 1988). Thus, these term-weighting schemes are fundamental to many current search engines and the theories behind them are crucial in understanding the theoretical aspects in IR. IR is not only concerned with alleviating the problem of natural language retrieval but also of developing an underlying theory for retrieval. It is interesting, if not crucial, to understand the true nature of relevance in order to develop systems that accurately model this concept. Whether relevance is a unique concept or whether it has parallels in other areas of science is a fundamental question.

There are many sources of potential evidence that can be used to infer relevance given a user query. Implicit evidence is one such valuable source as the user does not consciously have to supply this information to the system. IR systems can gather this knowledge and augment the query automatically. Temporal aspects may be utilised for news related articles, while spatial or geographical evidence can be useful when looking for information on certain events in an area. However, a more fundamental problem in IR is the ambiguity of the language itself. A useful, if not necessary, step is to develop systems that can automatically disambiguate queries to improve performance. One such approach adds terms to the query which are useful in the retrieval process, but which come from knowledge in the language which is being searched. Automatic query expansion techniques attempt to alleviate the problem of vocabulary differences (or mismatch) in a language. In many of these auto-

matic query expansion techniques, samples of the language are analysed to determine similarities (synonymy) in terms. Queries can then be augmented and potentially useful information can be added. This thesis is concerned with using only evidence which can be automatically inferred from the documents and collection given an initial user query, in order to learn schemes which are useful in the retrieval of relevant documents.

1.2 Evolutionary Search and IR

Evolutionary computation techniques are stochastic artificial search methods that are inspired by natural biological systems. Genetic Programming (GP) (Koza, 1992) is one such approach. GP is inspired by Darwinian theory of natural selection, where individuals that have a higher fitness will survive longer and thus, will produce more offspring. These offspring will inherit characteristics similar to those of their parents and through successive generations useful characteristics will propagate.

GP can be viewed as an artificial way of selective breeding. Darwin (1859) summarised this as follows: "if variations useful to any organic being do occur, assuredly individuals thus characterized will have the best chance of being preserved in the struggle for life: and from the strong principle of inheritance they will tend to produce offspring similarly characterized". GP is particularly useful for searching large complex solution spaces as it makes few assumptions as to how good solutions are characterised. GP is often useful for 'real word' problems where an exhaustive search is computationally infeasible. Often in these types of problems, the interactions of variables in a potential solution can be very complex. As a consequence, GP is often used to automatically define functions whose variables combine and react in very complex ways. A useful feature of GP is that it produces a symbolic representation of a solution that can be further analysed. This can be advantageous when a general solution is required. In GP, solutions are created using some type of genetic material (the *genotype*). The solutions are placed in an environment and have a particular behaviour (the *phenotype*). Each solution can then be rated on how it performs in its environment (its *fitness*).

In many cases, creating a means for representing the genotype and defining a suitable fitness function can be a difficult problem in itself.

This thesis investigates a GP approach to search for useful term-weighting schemes in traditional text-based IR. Many theoretical approaches which develop term-weighting schemes have failed to outperform standard benchmarks and some fall short of these benchmarks. GP techniques have an inherent advantage over more traditional techniques of finding solutions, as solutions that prove useful (no matter how unappealing or unintuitive they appear to be) are retained in the population. As performance is the only factor in whether a solution, or part thereof, is retained in future generations, it is a different paradigm than many other heuristic methods. As GP is a non-deterministic algorithm, it may also be necessary to conduct multiple runs of the GP to produce useful solutions. By using this evolutionary learning approach, the space of possible term-weighting schemes can be searched in a guided manner.

1.3 Open Research Questions

It has been suggested that throughout the last decade the performance of ad-hoc term-weighting has plateaued (Voorhees and Harman, 2000). However, it is by no means definitive what form of term-weighting schemes consistently perform better than others, nor has it been suggested that there are no further advances to be made in the area of term-weighting. With the increase in computing power, there have been more and more attempts applying machine learning techniques to the domain of IR. Since the inception of GP in the early 1990s, it has been adopted by some researchers to help solve IR problems.

Previous approaches using GP in IR have shown that useful term-weighting functions can be found using GP (Oren, 2002b; Fan et al, 2004; Trotman, 2005). However, neither an in-depth analysis of the solutions produced nor an analysis of the search process itself has been conducted. For example, it is not known whether, given enough time (generations), each run of the GP would converge to a similar solution (e.g. some ideal solution). It is also not clear whether solutions produced using GP are fundamentally different from

the benchmarks on general data.

Recent research has attempted to formalise necessary aspects of theoretically valid term-weighting functions using axioms (Fang et al, 2004; Fang and Zhai, 2005). Many analytically developed term-weighting functions have been shown to adhere only to *some* of these axioms. It has been shown empirically that these axioms are indeed good estimators of term-weighting function correctness and that they are a useful tool in analysing these types of functions. However, it is not known whether an empirically based learning technique, such as GP, will find functions that adhere to these axioms. Furthermore, it is unknown whether any new characteristics of term-weighting functions, or further axioms, can be identified from the functions found using an evolutionary approach.

Another open question is whether such learning techniques can be utilised to improve current automatic query expansion techniques. There are many methods of expanding queries and re-weighting terms. The exact properties of the optimal expansion method is unknown. Furthermore, it is not known if any new properties or new function forms can be found to further increase performance.

1.4 Hypotheses and Contributions

This thesis examines an evolutionary learning approach to developing term-weighting schemes in IR. This work develops a framework to evolve a number of term-weighting schemes using an incremental process similar to some previous analytical approaches (Amati and Rijsbergen, 2002; Roussinov et al, 2005). This is in contrast to previous GP approaches to this problem (Oren, 2002b; Fan et al, 2004; Trotman, 2005) which learn entire term-weighting functions. The approach adopted in this work logically divides the term-weighting scheme into different functional parts and develops each part of the term-weighting scheme separately. Questions regarding the solution space and the phenotype of the resulting solutions, that have previously been ignored, are also addressed. Furthermore, the resulting term-weighting functions are analysed using an existing axiomatic framework. Previous ap-

proaches using GP to evolve term-weighting functions have not analysed the resulting functions and have only concentrated on performance. Finally, it is shown that GP can be utilised to help find novel functions in a query expansion framework. In particular, this thesis addresses the following research questions:

- Can term-weighting schemes that are comparable to, or better than, current benchmarks be found using GP adopting an incremental approach?
- As GP is a stochastic search algorithm different runs tend to produce different solutions. Do different runs of the GP converge to a similar type of solution or are all the best solutions from different runs vastly different from one another?
- Do the best term-weighting schemes adhere to known axioms to which all *good* term-weighting schemes should adhere? Can an empirical learning technique, like GP, find any new properties or characteristics of term-weighting functions?
- Can GP be useful in developing term-selection schemes for use in query expansion techniques by combining further evidence available in the collection about the language, in order to increase the performance of an IR system?

Hypotheses

Given these research questions, a number of hypotheses can be more formally stated and tested.

[*H1*] *The evolutionary process adopted (GP) can find term-weighting functions that outperform the best known benchmarks on unseen test collections using the incremental learning process outlined in this work.*

[*H2*] *The better solutions produced by the GP process are phenotypically closer to each other in the search space than existing benchmarks.*

[*H3*] *The evolved term-weighting schemes can be theoretically validated using an axiomatic approach to IR.*

[*H4*] *GP can be used to evolve schemes for automatic query expansion that improve performance over the best known benchmarks.*

In order to test these hypotheses, an IR system is developed that can compare term-weighting schemes on a set of documents and queries. This system must be flexible so that each part of the term-weighting scheme can be incrementally built upon. The system must be able to support a GP approach to learning term-weighting schemes. Furthermore, a number of distance metrics must be developed that can measure the difference in the positions of the relevant documents in the ranked lists produced by different schemes. These distance measures can tell us about the differences in the rankings produced by the solutions from the evolutionary process. The IR system must also be extended to incorporate automatic query expansion approaches. Using a somewhat similar approach, term selection schemes can then be used to expand queries with potentially useful terms and these functions can then be compared in the system. A GP approach is then incorporated so that these query expansion schemes can be learned automatically.

Contributions

This study makes some important contributions to the body of existing work in both GP and IR. This work presents experiments on the evolution of term-weighting schemes in text-based IR. This is the first examination using GP in an incremental process (where the term-weighting scheme is separated into three intuitive constituent parts). It shows that term-weighting schemes can be found that outperform state of the art benchmarks using such an approach. The incremental approach adopted is shown to be both theoretically sound and empirically reliable. The incremental process reduces the vast search space in a theoretically motivated manner and allows schemes to be learned which can be more easily analysed.

This work presents a phenotypic analysis of the evolved schemes in each of the constituent search spaces. This has not been done before. It shows

that the GP finds solutions which are phenotypically close to one another. This confirms that the approach adopted finds solutions that are similar and that these evolved solutions are different to those of the benchmark solutions in terms of the ranking of relevant documents. This is also important in showing that the evolutionary guided process is in some way consistent and importantly shows that the new term-weighting schemes are different to the current benchmarks.

An axiomatic approach to IR is utilised to analyse the genotypes that result from the search process. This axiomatic approach, developed by Fang and Zhai (2005), has previously not been used to attempt to validate term-weighting functions produced from a learning approach. Importantly, this genotypic analysis validates the choice of learning paradigm and also aids in the development of another previously unknown axiom in IR. It is shown that the incremental approach adopted aids in the adherence to these axioms unlike previous GP approaches to this problem. A number of learned term-weighting approaches presented in the literature are analysed using the axioms. It is shown that the incrementally evolved term-weighting scheme adheres to more axioms than any other scheme. This is an important contribution as is the supplementation of the axiomatic approach by this work.

The three parts previously described follow a logical progression from fitness (performance) to phenotype (the ranking created by the schemes) to genotype (term-weighting function structure). While these three sections constitute the core of this work, a final section details further interesting work using GP to evolve functions for automatic query expansion. This final section presents work which uses a GP process to combine additional evidence to further enhance the search process. It shows that in some circumstances GP finds novel term-selection schemes for query expansion. This is the first time that term-selection schemes have been evolved in this way.

1.5 Thesis Overview

The motivation and contributions outlined in the preceding section are further discussed in the following chapters:

Chapter 2: Information Retrieval

This chapter discusses relevant background material in IR. The main stages and models in IR are discussed, as well as the motivation behind term-weighting in general. Term-weighting is shown to be a crucial aspect in IR and a general framework for term-weighting is outlined. Importantly, this framework is a generalisation that is consistent with most approaches to term-weighting reported in the literature.

Chapter 3: Evolutionary Computation

This chapter presents terminology, algorithmic detail and important concepts in the field of evolutionary computation and, in particular, the sub-field of GP. The advantages and disadvantages of this learning paradigm are discussed together with some theory regarding the evolutionary search process. The motivation for adopting a GP paradigm is briefly discussed.

Chapter 4: Related Work: Term-weighting Approaches

This chapter describes the current state of the art regarding term-weighting approaches in IR. Previous approaches applying evolutionary computation to IR are reviewed, together with specific work in IR that is referred to in this thesis. In particular, existing GP approaches to IR are discussed. The motivation for adopting a GP paradigm is discussed in more detail. An axiomatic approach to IR is also reviewed which introduces some important constraints to which all good term-weighting schemes should adhere. This formal specification of theoretically sound term-weighting schemes is useful for the analysis of learned term-weighting schemes later in this work. Some important contributions to document length normalisation are also briefly discussed.

Chapter 5: Design and Experimental Setup

This chapter describes the design of the system and the experimental setup. Important GP parameters are introduced and the terminal and function sets

for several of the experiments are outlined. The test collections and fitness function used are introduced before a brief description of the experiments is outlined.

Chapter 6: Evolving Global and Term-Frequency Schemes

This chapter presents several term-weighting functions which have been evolved in the framework. Term-weighting schemes for the first two parts of the term-weighting framework are outlined in this chapter. These functions are empirically validated on a number of test collections. Two of the three parts of the term-weighting scheme are developed in this chapter.

Chapter 7: Evolving Normalisation Schemes

This chapter develops the final part (normalisation) of the term-weighting scheme. Normalisation is a difficult problem and much research has focused solely on this. Therefore, it can often be seen as a separate problem, although it fits into the overall weighting framework. This chapter conducts an analysis of normalisation in order to determine aspects that affect normalisation in different retrieval settings. A number of evolved normalisation schemes are then presented and empirically validated.

Chapter 8: A Phenotypic Analysis of the Search Spaces

This chapter studies the evolution of the term-weighting schemes developed in the previous two chapters. Measures of the phenotypes of the evolved term-weighting schemes are introduced and are visually represented. This aids in the analysis of the evolved solutions in the search space.

Chapter 9: A Genotypic Analysis using Constraints

This chapter uses a number of constraints (axioms) in an attempt to theoretically motivate the term-weighting schemes presented in the preceding chapters. A new constraint is outlined and a number of learned term-weighting approaches are introduced and analysed to determine if they satisfy the constraints.

Chapter 10: GP for Automatic Query Expansion

This chapter details experiments that use GP to evolve schemes used for automatic query expansion. Two different approaches to query expansion are adopted and it is shown that GP can find schemes which are comparable to those currently used in modern approaches to query expansion.

Chapter 11: Conclusion

This chapter outlines the main contributions of this work and some interesting future directions are also discussed.

Chapter 2

Information Retrieval

This chapter introduces background material in traditional IR methods, models and evaluation. More specifically, section 2.1 introduces the various stages of a typical IR system and discusses some of the standard approaches adopted. Various models of IR are introduced in section 2.2 including the Boolean model, the vector space model and several probabilistic approaches. The focus of section 2.3 is a discussion of term-weighting approaches. This section (2.3) outlines a term-weighting framework that is used throughout this work and is deemed important in this regard. Automatic query expansion techniques are discussed in section 2.4, while IR evaluation metrics are discussed in section 2.5. The chapter concludes with a summary of the important points detailed within.

2.1 Stages of Retrieval

IR deals with the automatic search and retrieval of information according to specification by subject. In the context of information science, IR is often seen as a parallel counterpart to traditional data retrieval, but instead deals with retrieving information from semi-structured or unstructured information sets using natural language queries. However, natural language queries and documents lead to difficulties in retrieving accurate information for the user.

An IR system must make an effort to *interpret* the semantic content of

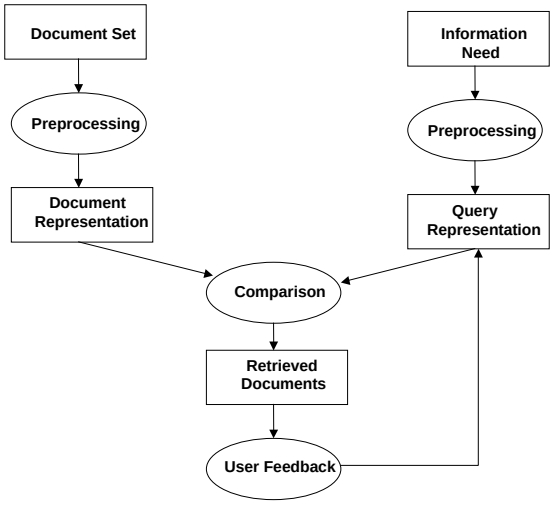

Figure 2.1: Components of a typical IR system

a document and rank the documents in relation to the user's query. Many algorithms and methods have emerged throughout the years to deal with problems at the various stages of a traditional IR system, some of which have been adopted as standard by many researchers. Figure 2.1 (Blair and Maron, 1985) outlines some of the stages of a traditional IR system. Some of these stages include pre-processing, document and query representation, comparison, evaluation and feedback. These stages are discussed next before the concepts behind specific models are outlined.

2.1.1 Preprocessing

The pre-processing stage of an IR system involves applying a set of well-known techniques to the query and documents in order to convert them to a more refined and concise format for the comparison stage. These techniques are employed in many well-known IR systems and common approaches include *stopword removal*, *stemming* and *thesaurus construction*.

Stopword removal is the removal of words that are too frequent in the

documents in the collection and consequently, are typically not good discriminators. Such words are frequently referred to as stop-words and are normally not considered as potential index terms. Articles, prepositions, and conjunctions are natural candidates for a list of stop-words. Elimination of stop-words reduces the size of the indexing structure considerably. It is typical to obtain compression in the size of the indexing structure of 40% or more, solely with the elimination of stop-words (Baeza-Yates and Ribeiro-Neto, 1999). A drawback of stop-word removal is that it may reduce recall for the system (i.e. the ability to return certain documents). Consider the common example of a user seeking documents containing the phrase "to be or not to be". Some stop-word removal techniques will often eliminate all of these terms and hence, a system will not be able to recognize documents which contain such a phrase.

Stemming refers to the transformation of a word (term) to its base or root form. Often, a user specifies a word in a query but only a variant of this word occurs in a relevant document. Plurals and past tense suffixes are examples of syntactical variations that prevent a perfect match between a query term and a respective document term. Stemming algorithms reduce similar terms to a common root form by identifying morphological derivations. An example of a stem is the term 'inform', which is the stem for the variants 'informed', 'informing', 'informs' and 'information'. Like stop-word removal, stemming also reduces the size of the indexing structure as the number of distinct index terms is reduced. Although there are a few suffix removal algorithms, one of the simplest and most widely used is Porter's (1980) stemming algorithm. Despite being simpler than other more sophisticated algorithms, Porter's algorithm yields comparable results (Baeza-Yates and Ribeiro-Neto, 1999). It should be noted that semantic information can be lost by stemming, but in general, it does not damage and often improves the performance of IR systems (Baeza-Yates and Ribeiro-Neto, 1999), while at the same time decreasing the size of the index structure.

The main purposes of *thesaurus construction* is to provide a standard vocabulary for indexing and searching, to assist users with locating terms for proper query formulation and to provide classified hierarchies that allows

the broadening and narrowing of the current query according to the needs of the user. Thesauri can be constructed via manual or automatic approaches and are often used in query expansion techniques (outlined later in this chapter). Manual creation of a thesaurus requires the knowledge of the language, while automatic creation is based on calculating measures relating to the co-occurrence of terms throughout the document collection. The main advantage of thesauri is the potential of achieving retrieval of information based on concepts rather than on words.

2.1.2 Document and Query Representation

This stage of the IR process usually involves the adoption of a specific model of IR. There are many models of IR which adopt various assumptions and employ varying techniques and theories from different areas of science. Some classical models of IR will be discussed later in this section. Typically, the specific model adopted utilises established transforms within the model to map the document and query into conceptual objects that can then be related in that model of relevance. While this may seem rather abstract, it will become clearer when some specific models have been introduced. It is also beneficial to view this stage at an abstract level so that fewer assumptions are made as to how relevance is actually modelled.

2.1.3 Comparison

All models of IR employ some type of distance, similarity, probability or comparison function that is usually inherent within the model itself. At this stage the query and document have already been transformed into structures or objects under the operations available in the model. All that is required now is to use a suitable function to provide a numerical measure of relatedness (estimated relevance) between each document and the query (in their transformed form). The numerical measure produced is constrained only by the model itself. The documents which are more related to the query are then returned to the user.

2.1.4 Evaluation

Now that a set of documents has been *deemed* relevant by the system and returned in response to a user's query, it is important that a method of systematically evaluating the relative or absolute performance of such a system is available. Relevance assessments for specific queries are generated by experts in the specific subject (topic) of the query. These relevance judgments are determined by a human subject and thus, it is the goal of the IR system to best simulate the human notion of relevance. In traditional text-based IR, relevance is assumed to be a binary objective judgment (i.e. a document is either deemed relevant to a query or not). The notion of relevance, and in particular the aforementioned assumption, is one with which people new to the field of IR have some difficultly. While the binary assumption of relevance used to test most IR models and systems is by no means a perfect or indeed correct view, it is an assumption that is useful when an empirical measurement is required. This assumption has remained over the years for many aspects of evaluation. A number of studies have indicated that relevance does indeed follow regular patterns and is not as subjective as one may initially assume (Saracevic, 1997). While initially appearing subjective, relevance can often be clouded by the ambiguity of natural language and the noise introduced to the IR process by a number of other underlying assumptions.

2.1.5 Feedback

Feedback in IR is concerned with improving the retrieval process by typically enabling the user to supply exemplars of relevant documents or other fragments of useful information following an initial retrieval run of the IR system. The extra information gathered about the user's intended information goal is fed back into the system. Some feedback is explicit and users are requested to indicate certain relevant or non-relevant documents. Highly descriptive terms or features are then extracted from these documents and aggregated with the initial information request (query). This expanded query is resubmitted to the system in the hope that the extra information will provide

further evidence in discovering the ideal set of relevant documents. However, it has been noted that users rarely explicitly provide this type of information or at least are reluctant to avail of such techniques when they are available (Ruthven and Lalmas, 2003). Therefore, implicit (or automatic) feedback techniques have become a useful alternative in such circumstances. Implicit feedback usually involves the automatic selection of assumed relevant documents from an initial retrieval run. These assumed relevant documents are then treated in a similar way to the explicit approach.

For some models of retrieval these feedback techniques are implicit, and thus such models need little or no adaption to incorporate feedback. For other models, incorporating feedback is a rather ad hoc process as the model may not have an underlying theory, or indeed known operations or transforms, for feedback to be incorporated.

2.2 Models of Retrieval

2.2.1 Boolean Model

The Boolean model of IR (Rijsbergen, 1979) is a classical model which uses traditional Boolean logic and set theory to identify documents that match a Boolean type query. It is based on the simplistic concept that if a document contains a term or a set of terms that satisfy a query, then that document is deemed relevant. The standard logical operators 'AND', 'OR' and 'NOT' can be used to construct queries that in turn identify a set of documents. This model is common and easy to implement but it suffers from obvious drawbacks. There is no concept of a partial match, as a document is either contained in the returned set (deemed relevant) or not (deemed irrelevant). Another disadvantage of this approach is that it places a burden on the user to attain the knowledge to construct good Boolean queries. Often this model retrieves too many or too few documents to be of use to the average user.

The extended Boolean model (Salton et al, 1983) attempts to overcome some of the limitations of the traditional Boolean model by introducing partial membership to the set of relevant documents. This is achieved by util-

ising fuzzy (or non-crisp) set theory to determine the membership of each document to the set of relevant documents (determined by the query). In essence, this introduces the concept of a weighting on each of the query terms ranging between 0 and 1, while the classic Boolean model can be viewed as only allowing two distinct values. It is apparent that the weighting on these terms ultimately determines the degree of membership to the set of relevant documents and thus, the overall performance of such a model. This partial matching also introduces the idea of a ranking over the set of possibly relevant documents, rather than strict membership. This model of retrieval can be viewed as a natural progression from the classic Boolean model.

2.2.2 Vector Space Model

The vector space model (Salton et al, 1975) is one of the most well known models of IR. In this model, the documents and query are are viewed as vectors in a multidimensional term space. Each dimension is assumed to be orthogonal (i.e. terms are viewed as occurring independently). Using a Euclidean distance function (usually cosine similarity), the distance between each document and the query is measured. Each document is then ranked on how close it is to the query (the closest document being ranked first). This model (or variant thereof) is one of the underlying models in use in many of the search engines of today. It is appealing due to its efficiency, ease of implementation and the fact that it achieves good performance compared to other more sophisticated models of retrieval. However, it should be noted that each dimension does not count equally toward the final score a document is given. Again, weighting plays an important role in the ranking of the documents. Dimensions are assigned different weights initially (corresponding to the importance of that term), while the degree to which the document extends in the dimension is determined by occurrences of terms within the document itself. While the model is appealing, in that it provides a method for utilising the properties (terms) in the documents in a existing framework for similarity, its performance ultimately depends on the weighting of the terms (Salton and Buckley, 1988). However, one drawback is that terms are

treated as mutually independent (term-independence). This assumption ignores the structure of the document and as a result semantic information is lost.

Latent semantic indexing (LSI) (Deerwester et al, 1990) can be considered a vector based approach that attempts to overcome the term-independence assumption. The entire collection of documents can initially be viewed as a term-document matrix. Vector approaches to IR are usually characterised by high dimensionality due to the number of terms in an entire collection. LSI reduces the high dimensionality of the aforementioned term-document matrix using singular-value decomposition. This reduction can uncover previous unknown (or latent) dependencies between terms. Thus, this reduction of the dimensionality is analogous to uncovering concepts in the collection. The dimensionally-reduced document vectors are compared against the similarly reduced query vector in an orthogonal vector space as before using some type of matching function (the cosine function is common). This technique can succeed in returning relevant documents that contain none of the query terms and has shown improved results when compared to the vector space model. One difficulty lies in choosing the size of the reduced space to which the documents and queries must be mapped. If the size chosen is too large, then the system is a vector space equivalent. Conversely, if the size is too small, it results in very coarse-grained retrieval. A major disadvantage is that for very large collections, the extremely high dimensionality of the data can affect scalability as memory consumption and computation time increase dramatically.

2.2.3 Probabilistic Model

The probabilistic model of retrieval is based on the probability ranking principle (PPR) (Robertson, 1977) which states that optimal performance is achieved when documents are ranked by their probability of relevance. This, in turn, is estimated based on the distribution of terms in relevant and non-relevant documents. Thus, these models are interested in providing an optimal ranking over the document set. The binary independence model (BIR)

(Robertson and Sparck Jones, 1976) proves that an optimal weighting for terms, based on some underlying assumptions, can be achieved. Again, the terms are assumed to be independent of each other. Initially, this may seem like a rather bold assumption (as in the vector space model), for in reality it seems that authors of documents carefully choose terms based on the existing content within the document. However, it is an assumption that persists in many models of retrieval due to its simplicity and due to the difficulty and cost associated in gathering details of the interdependence and co-occurrence of terms. Another assumption made is that relevance is only based on the presence and absence of query terms in documents. These assumptions provide for a limited but workable model and provide an optimal weighting strategy under such assumptions. Importantly, knowing the optimal weighting strategy when given details about the distribution of terms in the relevant and non-relevant document sets can aid the development of weighting schemes in more difficult circumstances. Indeed, determining the optimal weighting strategy becomes more difficult when no details about the distribution of terms in the relevant document set are known a priori. Therefore, information on the probable relevance distribution must be estimated. Again, this is achieved using measures of term occurrences in the documents and collection as a whole. This estimation is typically achieved using a term-weighting scheme.

2.2.4 Language Model

The language model (Ponte and Croft, 1998; Zhai and Lafferty, 2001) can be viewed as part of the probabilistic family of IR models as it utilises probabilities regarding term occurrences. However, it is derived from speech recognition techniques that try to predict the most likely word sequence given a sample spoken phrase. In IR, this model views documents as language models and estimates the probability that a specific language model will generate the given query. However, as not all of the language models (documents) contain all possible terms, smoothing techniques are used to provide a non-zero probability that a specific language model (document) contains

a specific query term. The probabilities for each language model generating the specified query are used to rank the documents for the user. This model has shown comparable performance with the more traditional probabilistic model of retrieval but has the added advantage of providing a nonparametric theoretically sound basis for retrieval.

2.3 Term-Weighting

The frequency of term occurrences was first proposed as a measurement of the usefulness of a term by Luhn (1958). Luhn devised a counting method to select significant words in a corpus. His technique ranked the words in order of frequency and employed two cut-off points. It can be noticed that when words are ranked in such a way they exhibit Zipfian characteristics (Zipf, 1949). An upper cut-off point rejected top-ranked terms, terms that are too frequent to be of use in distinguishing documents. A lower cut-off point rejected very low ranked terms, as these terms are very infrequent and do not contribute significantly to the content of the article. Thus, the technique measures what he called the *resolving power* and promoted terms which were neither too rare nor too frequent.

Figure 2.2 shows the terms that are promoted using Luhn's idea of resolving power. This method was rather ad hoc as the cut-off points were arbitrarily chosen. However, Luhn's rather simple idea to use the frequency of occurrence was essentially the beginning of many of the ideas surrounding term-weighting schemes that are employed today. Later, work into term-weighting approaches (Yu et al, 1982; Greiff, 1998) validated much of Luhn's work on his notion of *resolving power*.

It should now be apparent that nearly all, if not all, models of retrieval depend on some sort of scheme (or strategy) for assigning different weights to terms to determine how useful they are likely to be in determining the relevance of a document to a specific query. The particulars of the term-weighting schemes in each of the models that were discussed have not been expanded upon. This section will deal with term-weighting explicitly and can in most parts, if not in its entirety, be applied to all the models of

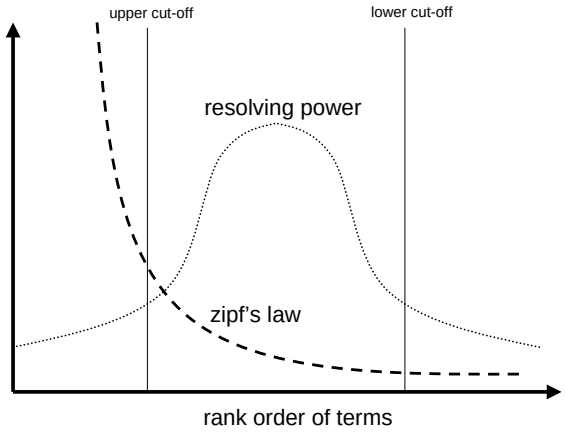

Figure 2.2: Resolving power of significant terms

retrieval that involve term-weighting. While the underlying theories may be very different for various models, the realisation of many of the models may be mathematically very similar. While models play an important part in the further development of the theory of IR, it is their performance that often decides the successful uptake of the particular model.

As it has been identified that term-weighting plays a vital role in the importance of IR systems, many attempts to improve the performance of such systems have focused on term-weighting in a standard intuitive framework using a "bag of words" approach (i.e. assuming term-independence). Many attempts (especially in the field of machine learning) have intentionally neglected adopting a specific model or theory of IR a priori and dealt solely with developing useful term-weighting schemes based on a few underlying assumptions. This, however, is not to say that term-weighting schemes produced from such approaches have no underlying theory (as clearly they must), nor is it true to say that nothing can be inferred about the true nature of IR from these schemes. Indeed, useful schemes produced from learning approaches could improve the theoretical understanding of retrieval. They

Table 2.1: Measures/Features used in term-weighting

Key	Description of Measure
Q	a query
D	a document
C	a document collection
N	the number of documents in the collection
V	the size of vocabulary of the collection
t	a term in the corpus
tf_t^D	the number of occurrences of term t in document D
tf_t^Q	the number of occurrences of term t in query Q
df_t	the number of documents in which term t occurs
cf_t	the number of occurrences of term t in the collection
dl	the length of a document (repeated words)
dl_{avg}	the average document length in the collection
ql	the length of a query (repeated words)
T	the total number of words in the collection

could ultimately lead to improvements and insights that may be applied to other models. Unfortunately, a lack of theoretical analysis can often dissuade researchers from adopting useful weighting schemes produced from these newer learning approaches.

2.3.1 Standard "Bag of Words" Framework

The "bag of words" approach to retrieval assumes that terms occur independently and that no relevance judgments are supplied to the system for a specific query. Term-weighting schemes in these frameworks are developed using measures of the query-terms. For the following discussion about the composition of term-weighting schemes, it is necessary to introduce some of these measures. Table 2.1 shows some of the measures that are typically employed in term-weighting schemes. Although the notation in the literature is often quite varied, the style adopted here is in line with much research, is mathematically consistent and is in a useful form for the discussions and analysis in later sections of this work.

At this point, it is neccessary to introduce a standard framework for term-weighting schemes. The function outlined (2.1) can be thought of as a generalisation of a family of term-weighting schemes. While it does not represent the complete space of entire term-weighting schemes (which is boundless), it does incorporate most, if not all, term-weighting schemes reported in the literature. The score ($S()$) of a document D in relation to a query Q can be calculated as follows:

$$S(Q, D) = \sum_{t \in Q \cap D} (ntf(D) \cdot gw_t(C) \cdot qw_t(Q)) \tag{2.1}$$

In this framework, there is a basic term-discrimination element (or global component), a normalised term-frequency element (or within-document component) and a query term element (or within-query component). The term-discrimination element ($gw_t(C)$) aims to determine how useful a search term is, by using characteristics of the term in the collection C as a whole. Typically, terms that occur in fewer documents are given a higher weight as they tend to be better descriptors of that document. Most term-weighting approaches include some type of term-discrimination element either directly or indirectly to promote terms that are likely to be better able to identify certain documents.

The normalised term-frequency ($ntf(D)$) aims to provide *two* effects on each specific document D using within-document measures. Its first aim is to promote documents which have a higher occurence of query terms. This is achieved using a term-frequency influence component. It is intuitive that a document with more occurrences of query terms should be ranked higher than a document with fewer occurrences. However, not all documents are of similar length and thus, the term-frequency is normalised in some way to avoid over weighting longer documents simply because they contain more of these terms. A document that is longer may simply have a broader topic and should not be promoted over shorters documents which may be more concise and preferable to the user. Basically, the concept of normalisation is a measure of the concentration of query terms in a document. Documents with a greater concentration of query terms should be promoted ahead of

documents with a lower concentration. As yet there has not been a discussion about how these effects can be achieved.

The remaining component of the generalised framework (equation (2.1)) describes the weight assigned to the terms appearing in the actual query ($qw_t(Q)$). This component is typically a simple description and it has been shown in many studies that using the actual query term-frequency for such a component does not lead to any decrease in performance compared to a more complex form for this component. This is typically because queries are quite short and supply quite a limited amount of information about the frequencies and characteristics of the terms themselves. Most of this information is available from the much larger documents and collection as a whole. The $BM25$ equation (which is defined in full in the next section) uses a query term weighting of $\frac{(k_3+1)\cdot tf_t^Q}{k_3+tf_t^Q}$ which performs very well when k_3 is 1000 (Walker et al, 1997). This is close to a linear weighting in terms of ranking when k_3 is such a large value. For short queries this value can often become redundant as query terms tend to only appear once and therefore are assigned the same query term weight.

Term-weighting schemes are sometimes described using two triples (Salton and Buckley, 1988; Zobel and Moffat, 1998). One triple describes the weight assigned to the terms in the document, while the second triple describes the weight assigned to the terms in the query. Each triple contains a term-discrimination element, a term-frequency element and a form of normalisation (as with the framework outlined here). However, with the advent of TREC data (Harman, 1993) it has been noted that the triple describing the weight assigned to terms in the query can be reduced to a simple linear within-query term-frequency (Jones et al, 2000; Singhal, 2001; Fang and Zhai, 2005). The framework outlined (equation 2.1) is also consistent with this view of a term-weighting scheme and can be further reduced to the following form with very little loss of expression:

$$S(Q,D) = \sum_{t \in Q \cap D} (ntf(D) \cdot gw_t(C) \cdot tf_t^Q) \tag{2.2}$$

where tf_t^Q becomes the entire query weighting triple and $ntf(D) \cdot gw_t(C)$

describes the document weighting triple. It is worth remembering that the $ntf(D)$ component provides *two* within-document effects. This discussion simply shows that both forms are very similar (the triple description being a slightly more general framework) and is included for completeness.

2.3.2 Benchmark Term-weighting Schemes

BM25

The $BM25$ scheme (Robertson et al, 1995) is, at present, one of the best known and most successful term-weighting scheme in IR. It is derived from the probablistic model of retrieval and is also the most widely used benchmark against which to compare new term-weighting schemes. Each document D is scored against a query Q in the following manner:

$$BM25(Q,D) = \sum_{t \in Q \cap D} \left(\frac{tf_t^D}{tf_t^D + k_1 \cdot ((1-b) + b \cdot \frac{dl}{dl_{avg}})} \cdot w_1 \cdot tf_t^Q\right) \tag{2.3}$$

where k_1 and b are tuning parameters that affect retrieval performance. k_1 is called the term-frequency influence parameter as it controls how much influence an extra occurrence of a term will receive. Its operating range is from 1.0 to 2.0. b is the normalisation parameter and controls the normalisation aspect of the function. Possible values for b range between 0.0 and 1.0. Setting $b = 0$ will ignore the document length in weighting while a higher value for b will more heavily penalise longer documents. The query term weighting used in this framework (tf_t^Q) is slightly different to the original (Robertson et al, 1995) but has been used successfully in many studies. w_1 is the term-discrimination element of this function and is calculated as follows:

$$w_1 = log(\frac{N - df_t + 0.5}{df_t + 0.5}) \tag{2.4}$$

This is a form of the classic *inverse document frequency* (idf) function (Sparck Jones, 1972) which has been the basis for many term-weighting and feature extraction techniques over the years. Despite being a very simple formula,

idf has stood the test of time and is the basis for many benchmark term-weighting schemes. The *idf* factor simply assigns a higher weight to terms that occur in less documents as these terms are ultimately better descriptors of that document. It can be noted that *idf* is inconsistent with Luhn's idea of resolving power for very infrequent terms (terms with a low rank) as infrequent terms recieve the highest weight under *idf*.

Pivoted Document Length Normalisation

Another successful matching function is the pivoted document length normalisation scheme (Singhal et al, 1996). The score of a document in this scheme is calculated as follows:

$$PIV(Q,D) = \sum_{t \in q \cap d} \left(\frac{1 + log(1 + log(tf_t^D))}{(1-s) + s \cdot \frac{dl}{dl_{avg}}} \cdot w_2 \cdot tf_t^Q\right) \tag{2.5}$$

where s is the normalisation parameter referred to as the slope and has a default value of 0.2. w_2 is the *idf* function as found in the pivoted normalisation scheme and is identified as follows:

$$w_2 = log\left(\frac{N+1}{df_t}\right) \tag{2.6}$$

It can be seen that both of these weighting functions consist of a term-discrimination part (*idf*) and a type of normalised term-frequency (i.e. they can both be represented by a single triple).

2.4 Automatic Query Expansion Techniques

The problem of term mismatch in IR arises from the fact that different people use different terminology when referring to the same concepts. Synonyms are particularly difficult for IR systems as the query keywords may not match the keywords of a relevant document. For example, when searching for information about fixing a "leaking tap", many relevant documents may be written by American authors who use the word "faucet" instead of "tap". This would typically eliminate a whole set of documents. Automatic query

expansion techniques try to overcome this problem by adding extra keywords to the query in order to automatically modify the query representation and improve the performance of the query. Local and global query expansion techniques are two approaches that have been developed which attempt to alleviate this term-mismatch problem. While both of these automatic expansion techniques are introduced here, they can be viewed quite differently in terms of where they fit in the retrieval process. Local query expansion can be thought of as an automated feedback process, while global query expansion can be viewed as a part of the preprocessing process. This will become clearer in the following sections.

2.4.1 Local Query Expansion

Local query expansion (also known as blind or pseudo relevance feedback) uses terms from the top P ranked documents of an initial retrieval run to add to the original query. The original query is submitted to the system and a ranked list is returned. This approach to expansion initially assumes that the top few documents of this ranked list are relevant. These documents provide a pool of expansion terms from which to select. Useful terms are then selected from this pool based on characteristics of the terms in the documents, in the pool and in the collection as a whole. The selected terms are added to the original query and are then weighted using some term-weighting approach. The newly formulated query is then resubmitted to the system. This form of expansion performs well when several documents in the top few documents of the initial run are actually relevant, as many of the added terms are related to the query topic. However, when there are relatively few or no relevant documents in the top P documents returned, the quality of the added terms are poor as they are not related to the query topic. This type of expansion is referred to as local query expansion as the pool of possible expansion terms comes from a small number of related documents. This form of expansion is computationally feasible even for large document collections.

As mentioned, the terms to add to the query are chosen based on their frequency characteristics. As the top P documents are assumed relevant, the

Robertson/Sparck-Jones weight (Robertson and Walker, 1999) developed for the probabilistic model of IR is often used to determine the weight for these terms. This weight (w_{rsj}) is calculated as follows:

$$w_{rsj} = log(\frac{(pdf_t + 0.5)/(P - pdf_t + 0.5)}{(df_t - pdf_t + 0.5)/(N - df_t - P + pdf_t + 0.5)}) \tag{2.7}$$

where P is the number of pseudo-relevant documents, N is the number of documents in the collection, df_t is the document frequency of term t and pdf_t is number of pseudo-relevant documents that term t occurs in. A simple but effective term-selection scheme (Robertson and Walker, 1999) used in practice is as follows:

$$TSV_t = pdf_t \cdot w_{rsj} \tag{2.8}$$

which simply chooses terms with a high w_{rsj} weight that appear in many of the top P documents. A number of terms ($|E|$) are then chosen based on this TSV_t score and these are added to the query. The weight applied to these expanded terms is usually the w_{rsj} weight. The number of terms ($|E|$) and number of top ranked documents (P) deemed relevant are usually fixed (Billerbeck and Zobel, 2003). In recent years, local query expansion has been shown to increase the performance of certain types of queries (Mitra et al, 1998).

2.4.2 Global Query Expansion

Global query expansion (also known as thesaurus or co-occurrence based expansion) uses terms from the corpus as a whole to add to the original query. This model of expansion assumes that terms that co-occur in many documents are semantically related (i.e. have some synonymous relationship). Thus, by analysing the entire collection of documents for term-term relationships or co-occurrences, an automatic domain-specific thesaurus can be constructed. This automatic approach to thesaurus construction has an advantage over a manual thesaurus as it can associate terms that may be synonymous only within a certain domain or context. Using a manual the-

saurus for individual term synonyms also ignores the context of the query as terms are treated independently.

Global query expansion has been less successful than its automatic local counterpart on larger TREC style test collections over the past decade (Harman, 1993). It is also computationally expensive to create term-term associations between all terms in the corpus as modern large document test collections can contain hundreds of thousands of terms. Many approaches only use a subset of the terms in the entire collection and then determine co-occurrence relationships. To develop term-term relationships, conceptually the role of documents and terms are interchanged in the retrieval model. In essence, documents become the features of the term. Thus, two terms that appear in the same document are indexed by a similar feature and are deemed to have some type of synonymous relationship. Many formulas have been proposed to measure the association between two terms using co-occurrence data. Term-term relationships are often measured using the cosine similarity measure which is determined as follows:

$$cos(t_1, t_2) = \frac{df_{t1,t2}}{\sqrt{df_{t1} \cdot df_{t2}}} \tag{2.9}$$

where t_1 and t_2 are two terms, $df_{t1,t2}$ is the number of documents in which both t_1 and t_2 occur. df_{t1} is the number of documents in which df_{t1} occurs. There are many variations of such formulas which aim to accurately find the best synonyms for a term. As choosing expansion terms in isolation can often ignore the concept of the query, it is often beneficial to measure possible expansion terms against each term in the query and aggregate the results. Expansion terms can then be chosen on how close they are to the entire query and not just individual terms. This approach (Qiu and Frei, 1993) has shown to be of benefit on certain smaller collections.

2.5 Evaluation

As of this point, it has not been shown how the different models of IR are compared. It is neccessary to be able to empirically measure the effectiveness

of an IR system in order to compare them. Document test collections and evaluations metrics that are integral to the evaluation aspect to most IR systems are now introduced.

2.5.1 Document Test Collections

An IR system is measured by how well it satisfies users' queries. Document test collections are used to test and evaluate the performance of IR systems. They are comprised of three parts: documents, queries (or topics) and relevance judgements. A document test collection with queries and predetermined human relevance assessments are used to evaluate the performance of a specific system. The accuracy of the returned rank list of documents can then be measured and analysed. The TREC (Harman, 1993) collections afford researchers a large body of documents and queries with which to evaluate their approaches. Results on TREC data provides substantial evidence to support claims about specific IR approaches. TREC also has various tracks, which focus on different facets of the IR problem. An ad hoc retrieval track, a blog retrieval track, a question answering track and a genomics track are examples of such. However, the preprocessing and construction of a system capable of testing TREC collections requires the commitment of substantial time and resources, which leads some researchers to experiment initially with smaller collections.

2.5.2 Evaluation Metrics

The two main measures of retrieval effectiveness are *precision* and *recall*. Precision is a measure of the accuracy of the returned set of documents, while recall is a measure of the coverage of the system. For example, if a system only returns two documents, of which only one is relevant, the precision of that system would be 50% ($\frac{1}{2}$). However, if that collection has a total of 10 relevant documents, its recall would be 10% ($\frac{1}{10}$). An IR system strives for high precision and high recall.

$$precision = \frac{returned \cap relevant}{returned} \tag{2.10}$$

$$recall = \frac{returned \cap relevant}{relevant} \tag{2.11}$$

where *returned* is the set of documents returned by the system for a specific query and *relevant* is the set of relevant documents in the collection for that specific query.

Mean Average Precision

Usually, the measures of precision and recall are combined, as a user is typically interested in high precision and high recall. Average precision (AP) is a combination of precision and recall applied to a ranked-list setting and is calculated as follows:

$$AP = \frac{1}{R}\sum_{r=1}^{N} P(d_r) \cdot Rel(d_r) \tag{2.12}$$

$$R = \sum_{r=1}^{N} Rel(d_r) \tag{2.13}$$

where N is the number of documents in the collection, d_r is the document at rank r, $P(d_r)$ is the precision at rank r, R is the number of relevant documents in the collection for the query and $Rel(d_r)$ is the binary relevance judgement of the document at that rank r. To test the effectiveness of an IR system, each system is usually presented with a number of queries (or topics). The average precision is calculated for each of these queries and the mean of the average precision for the queries is reported. This metric is known as *mean average precision* (or MAP). This is a standard measure in IR retrieval effectiveness. It is a stable measure (Buckley and Voorhees, 2000) which means that when it reports a sizeable difference in retrieval effectiveness, it usually indicates that a difference does indeed exist. The importance of using statistical tests for retieval evaluations has been shown. Student's *t-test* is quite a reliable test in these circumstances (Sanderson and Zobel, 2005). It has also been indicated that as a general rule of thumb an absolute difference in MAP between two systems on 50 topics (queries) of

5% tends to be significant (Sanderson and Zobel, 2005).

F-Measure

In different contexts, users may be more interested in precision than recall (or indeed vice versa). In a typical web setting, a user may simply want one relevant document on a particular topic. In this case, the user is more interested in precision and does not typically need all relevant documents. In a medical or legal setting, recall becomes important as practitioners may wish to find all relevant documents for a particular information need. The standard *F-measure* (Rijsbergen, 1979; Losee, 2000) combines precison and recall while a more general *F-measure* allows this preference to be altered.

$$F_\alpha = (1 + \alpha) \cdot \frac{precision \cdot recall}{\alpha \cdot precision + recall} \tag{2.14}$$

where α is a measure of the preference for recall. The higher this value (α), the higher the preference for recall. The standard F-measure is recovered when α is set to 1.

Precision-Recall Curves

In practical circumstances, when a user issues a query, the list of documents is ranked in decreasing order of relevance. In this situation, the recall and precision measures vary as the user proceeds with the examination of the answer set. Thus, a more detailed evaluation requires plotting a precision-recall (PR) curve. Recall is placed on the horizontal axis and precision on the vertical axis. Essentially, these curves are plotted using points extrapolated at the appearance of each correct document (i.e. the horizontal value is increasing with the percentage of relevant documents found). For example, if a document ranked 25^{th} is examined and found to be the 5^{th} relevant document seen out of a known relevant set of 50 documents, the point at 10% recall, 20% precision can plotted. Furthermore, it is standard procedure to graph PR curves at 11 standard levels of recall (0% to 100% in increments of 10%). PR curves are useful because they allow the observation of the quality of the overall answer set.

Ranked List Correlation Metrics

Distance measures between ranked lists are currently used in IR to calculate the degree to which to ranked lists are correlated. Spearman's rank correlation and Kendall's tau correlation are two common correlations that measure the difference between ranked sets of data. Both Spearman's rank correlation and Kendall's tau use all of the ranked data in a pair of ranked lists to determine the closeness of the ranked lists. These measures are typically used to measure how different the output from two systems actually are (i.e. if two systems rank documents differently). It is possible that two different term-weighting functions (or indeed two vastly different models) could produce similar ranked lists in practice and thus have a similar performance. It is also possible that two *different* rankings could have the same performance (i.e. they may return different relevant documents).

2.6 Summary

The main contribution of this chapter is the motivation for the development of term-weighting schemes and the outline of the three term-weighting components. Term-weighting is shown to be a crucial aspect to many models of retrieval. Term-weighting can directly help with insights into how relevance should be modelled. A "bag of words" approach to retrieval is introduced which typically uses two common assumptions, and often forgoes an explicit theoretical model of retrieval, in order to focus on term-weighting and overall performance.

Importantly, a general term-weighting framework is introduced in which term-weighting schemes can be described by three effects (a triple), which weight the terms appearing in the document. Essentially, this framework divides the function space into three parts which can be searched in turn. Section 2.4 discusses automatic query expansion and the chapter concludes with a discussion on IR systems evaluation.

Chapter 3

Evolutionary Computation

In this chapter, evolutionary computation techniques are introduced in section 3.1. The evolutionary approach adopted in this work is briefly motivated before a discussion of GP is presented in section 3.2. The basic GP algorithm is introduced before the constitution of potential solutions is outlined using terminal and function sets. The mapping from genotype to phenotype to fitness is explained and this is mapped to terminology in the IR domain for the problem of term-weighting. Some other interesting characteristics of GP are explained before the summary (section 3.3) concludes this chapter.

3.1 Evolutionary Algorithms

Evolutionary computation techniques traditionally can be broken down into four main algorithms namely, evolutionary strategies (ES), evolutionary programming (EP), genetic algorithms (GA) and genetic programming (GP) (Whitley, 2001). All four algorithms are biologically-inspired stochastic optimisation techniques. While evolutionary strategies and evolutionary programming primarily use mutation to achieve different solutions, genetic algorithms and genetic programming make use of recombination (crossover) to develop new solutions.

ES and GAs typically adopt a fixed vector representation of a solution. The sets of values in these encoded vectors are modified using different tech-

niques until a suitably fit representation is found. These algorithms are useful when a static representation of a solution is required. However, to develop a term-weighting scheme, it is not the actual weights of the document vector that should be optimised (as the actual optimal weights for documents in one collection may not be optimal for another collection), it is the method of determining the weights in the document vector that needs to be optimised. EP is a technique that is quite similar to ES and was initially used to evolve finite state automata (Fogel et al, 1966). These types of methods can be viewed as parameter optimisation methods (Whitley, 2001) and depending on the encoding can be used to optimise functions of various types.

GP, however, is a not a parameter optimisation method, but a method of automatic programming which can be used to automatically define functions or actual programs (Whitley, 2001). These functions or programs are evaluated in a system and thus GP is a more dynamic expressive approach (i.e. they need to be executed before the fitness is known). GP is typically more expressive than the other evolutionary approaches and can lead to more general solutions as they aim to optimise at more abstract level. However, much of this depends on the actual problem domain itself and the landscape of the search space. EP mainly uses mutation to modify some representation of a solution and is used to search local areas of the search space in parallel. GP however can be used to create programs or functions of any type (possibly non-continuous) depending on the encoding of the possible solutions. Due to the flexibility of GP and its expressiveness, it is an obvious choice for developing functions when aiming to make as few assumptions as possible as to the constitution of such functions.

Some previous approaches applying evolutionary computation to IR are reviewed in the next chapter. The remainder of this chapter will discuss the GP algorithm. Much of the terminology and techniques adopted by GP can be found in the other evolutionary paradigms.

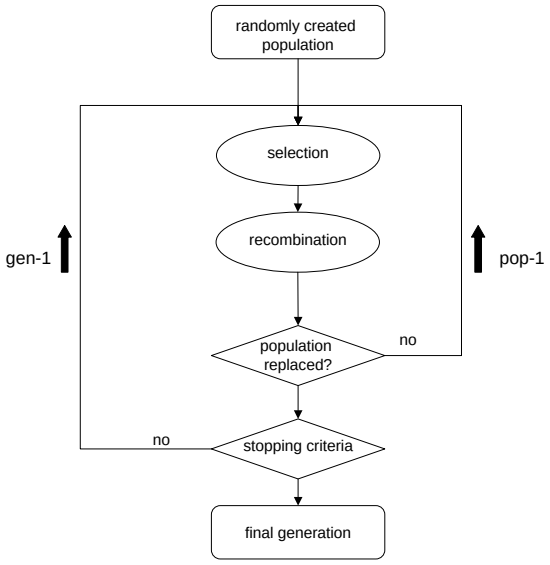

Figure 3.1: Basic Flow of a GP

3.2 Genetic Programming

John Koza was one of the primary founders of the field of GP in the early 1990s (Koza, 1992). Inspired by the successes of Holland and Goldberg in traditional GAs, the GP area has grown to help solve problems in a wide variety of areas. GP is inspired by the Darwinian theory of natural selection, where individuals that have a higher fitness will survive longer and thus will produce more offspring. These offspring will inherit characteristics similar to their parents and through successive generations the useful characteristics will strive. In this section the basic GP algorithm will be introduced together with GP terminology, some of which has been adopted from the field of biology.

3.2.1 Algorithmic Detail

GP is a heuristic stochastic search algorithm, inspired by natural selection (Darwin, 1859), and is useful for navigating large complex search spaces. Figure 3.1 shows the typical stages involved in a GP. Initially, a population of solutions is created randomly (although some approaches seed the initial population with known solutions). The solutions are encoded as trees and can be thought of as the genotypes of the individuals (i.e. the genetic makeup of a solution). Each tree (genotype) contains nodes which are either functions (operators) or terminals (operands). The values on the nodes of each tree are referred to as *alleles*. Each solution is rated based on how it performs in its environment. This is achieved using a *fitness function*. Having assigned the fitness values, *selection* can occur. Individuals are selected for reproduction based on their fitness value. There are various different methods used to select individuals but all are based in some way on the fitness of the individual. As a result, fitter solutions will be selected more often.

Once selection has occurred, *recombination* can start. Recombination (or genetic reproduction) can occur in variety of ways. The most common form is one-point *crossover*, where two different individuals (parents) are selected and two new individuals (children) are created by combining the genotypes of both parents. Another method in which a new solution can be obtained is *mutation*. Mutation involves the random change of the allele of a gene which can result in the random change of a subtree to create a new individual. Selection and recombination occurs until the population is replaced by newly created individuals. Once the recombination process is complete (i.e. a new generation of solutions has been created), each individual's fitness in the new generation is evaluated and the selection process starts over again. The process usually ends when a predefined number of generations is complete, until convergence of the population is achieved or once an individual is found with an acceptable fitness.

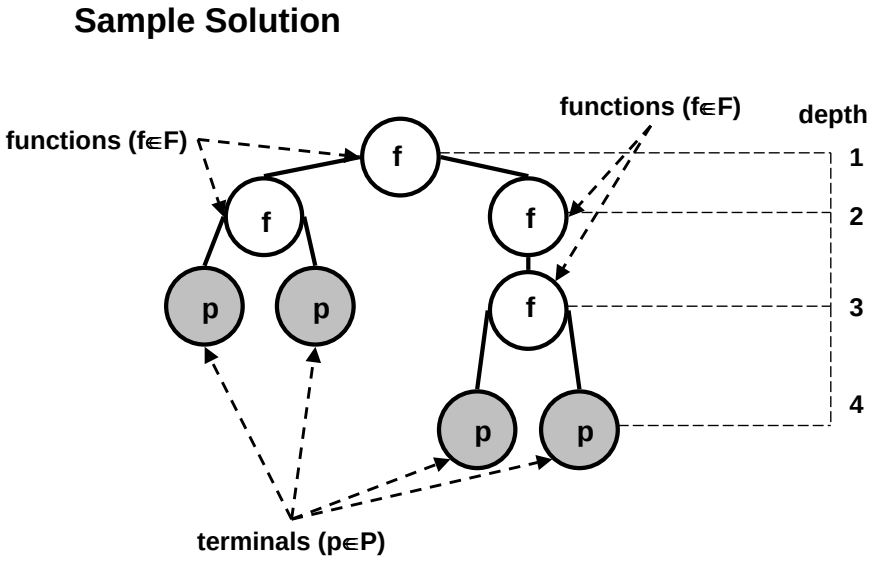

Figure 3.2: Representation of a solution

3.2.2 Functions and Terminals

The function set (F) and terminal set (P) are the sets of operators and operands that can makeup a possible solution. These can be viewed as the individual's genes which in turn make up a possible solution. Solutions are modelled as trees with terminals as leaf nodes and functions as internal nodes as shown in Figure 3.2. The choice of functions and terminals is an important part in the development of a GP. The size and composition of the set of functions and terminals influences the size of the search space and also has implications on possible bias in the space. If one chooses a large function and terminal set, the size of the search space may be so large that it takes the GP a long time to converge, if at all. It may also be the case that many of the functions and terminals are useless and are eliminated from the population quite quickly. The depth of a solution is also indicated in Figure 3.2. In some GP approaches the maximum depth of a solution is limited, as an unlimited depth creates a vast search space. Other approaches

penalise longer trees (which is reflected in the fitness function) so that shorter solutions are promoted.

In order to make as few assumptions as possible as to how good solutions are characterised, the choice of functions and terminals should be as primitive as possible. By this, it is meant that terminals should be as atomic as possible and that functions should not be overly complex. By including predefined solutions in the population or by encoding parts of known solutions as terminals, the search space is biased to-wards these types of solutions. This can result in convergence to a local optimal (i.e. a locally fit but globally suboptimal solution). This may be an acceptable outcome if one wishes to explore a region of the space in which known good solutions lie. However, in general, premature convergence is seen as detrimental and something to be avoided when one wishes to explore a search space with as few assumptions as possible.

3.2.3 Selection, Crossover and Mutation

There are many strategies for selecting parent solutions for recombination. All of these strategies are based in some way on the fitness of an individual. Tournament selection is one of the most common selection methods used. In tournament selection, a number of solutions are chosen at random from the population and these solutions compete with each other. The fittest solution is then chosen as a parent. This process is repeated to attain another parent. The number of solutions chosen to compete in the tournament is called the *tournament size* and this can be increased or decreased to increase or decrease the speed of convergence.

An example of one-point crossover is shown in Figure 3.3. For this example, the functions used are standard arithmetic operations, while the terminals used are taken from the domain of IR as detailed in the previous chapter. One-point crossover is the norm in GP. A single point is chosen in both parent trees and the subtrees are swapped at this point. This creates two new individuals for the next generation. Crossover acts as a global search technique. This means that the newly created solutions are not necessarily close

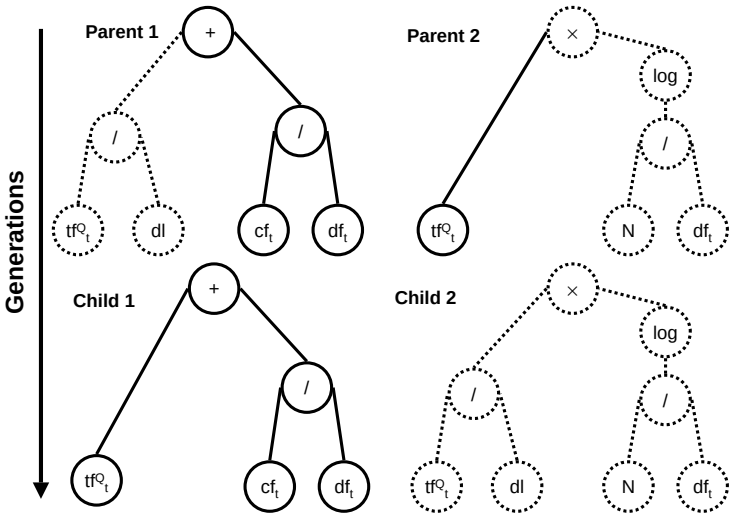

Figure 3.3: Example of crossover in GP

to either of their parents in the solution space. It may be worth remembering that in the first generation the solutions are randomly distributed throughout the search space. Crossover is the main method utilised in searching the space of potential solutions. Consequently rates of crossover are usually high (e.g. 90% of new individuals may be created by crossover).

Other methods of creating new solutions, like mutation, can occur. An example of this is shown in Figure 3.4. Mutation is a local search technique. This means that any newly created solution is often genotypically close to the original solution. Mutation rarely brings about an increase in performance as it can be viewed as randomly searching a localised area. However, mutation can be useful for reintroducing genetic material (alleles) which may have be eliminated from the population. Rates of mutation are usually kept low (e.g. less than 10% of new individuals may be created using mutation). Once the recombination process is complete each individual's fitness in the new generation is evaluated and the selection process starts again.

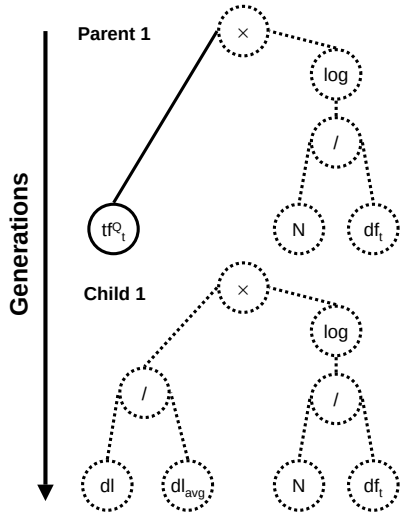

Figure 3.4: Example of mutation in GP

3.2.4 Genotype, Phenotype and Fitness

The *genotype* of an individual is its genetic makeup and consists of the tree shape and its node values. The *phenotype* of the individual is often described as its behaviour and is essentially the solution in its environment. The fitness of a solution is a measure of how good the solution is. Selection occurs based on the fitness only. Fitness is determined by the phenotype which is in turn determined by the genotype. As one can imagine, different genotypes can map to the same phenotype, and different phenotypes can have the same fitness. For most problems modelled by a GP in an unchanging environment, the same genotype will map to the same phenotype, which in turn will have the same fitness. Figure 3.5 shows how the GP terminology maps to the terminology in the problem domain. The genotype is the representation of the actual solution and in this problem domain is the makeup of the term-weighting scheme (i.e. the function form). The phenotype is the behaviour of a term-weighting scheme in its environment and can therefore be viewed as

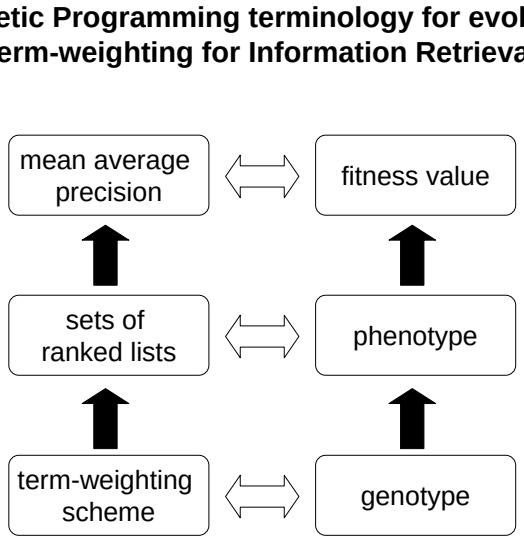

Figure 3.5: Mapping of GP terminology to problem domain

the ranking produced from a specific solution. The fitness function is simply some scalar value that can differentiate one solution from another in terms of performance (i.e. MAP).

The size of the solution space is extremely difficult to measure for many problems. The number of genotypes (assuming binary tree shapes) with n internal nodes (i.e. functions) can be found using the Catalan number $C_n = (2 \cdot n)!/(n! \cdot (n+1)!)$ (Lucas et al, 1993). For every full tree of $2n+1$ nodes, the total number of unique trees creates an enormous space of $C_n \cdot F^n \cdot P^{(n+1)}$ programs, where F is the number of functions and P is the number of terminals. Some of these trees may be functionally equivalent (Gustafson, 2004) due to commutativity, associativity and other properties of binary operations. Even so, the number of possible programs, even for small function and terminal sets, is extremely large. However, this does not necessarily correspond to the number of unique solutions (phenotypes). All that can be determined is that the number of unique phenotypes is less than

or equal to the number of unique genotypes in a static environment.

For any given problem, it is difficult (if not impossible) to measure the number of distinct phenotypes that exist or that can be created by the potential solutions as defined by the parameters, functions and terminals. For example, by removing a single terminal from the terminal set, the number of phenotypes that can be searched may or may not decrease.

3.2.5 Other Characteristics of GP

Premature convergence occurs when the population converges early and to a suboptimal solution. Premature convergence often occurs when one individual, while far from optimal, is significantly better than the rest. The population can converge toward this solution quickly and all solutions will become genotypically similar. Thus, genetic material which may be necessary to reach better solutions is lost and can only be reintroduced to the population via mutation. Mutation rarely creates a fitter solution as it is randomly sampling part of the search space. As a result, rates of mutation are usually kept low while the solution is converging. Keeping the selection pressure low (small tournament size) and having large populations are simple but effective ways of preventing premature convergence. The use of large populations results in GP being computationally expensive. However, this expense is incurred largely so that the extremely large search space can be searched effectively. This large search space is created because of the relaxation of assumptions as to how solutions are characterised and represented by GP.

GP is a computationally expensive learning paradigm and as a result, the allocation of available resources becomes important. While a large population is usually the best way to prevent premature convergence and achieve fit solutions, it must be allowed to run for a sufficient number of generations. An important question in GP is: What is the best way to allocate resources? Restart theory in GP suggests that it is necessary to restart the GP a number of times in order to achieve fit solutions. Research has shown that running a population of size M for G generations N times produces better solutions

than doing one run of a population M for $G \times N$ generations for a number of different problems (Luke, 2001) (i.e. restarting the GP with a similar population size leads to better solutions than letting the GP continue once convergence has seemed to occur). As GP is a non-deterministic algorithm, it cannot be expected to produce a similar solution each time. In many real-world problems, the optimal solution is unknown and thus, it is not possible to know when a certain GP run is converging to-wards the best solution without exhaustively searching the space. This is another benefit of running the GP a number of times. Running the process multiple times can help identify the areas of the search space in which the better solutions lie.

Bloat is a common phenomenon in GP. Bloat occurs when solutions grow in size without a corresponding increase in fitness. As longer solutions take longer to evaluate this will take up a considerably amount of resources. Thus, limiting the size of trees has the effect of reducing bloat, improving generalisation, reducing the search space and increasing the speed of the GP.

3.3 Summary

In this chapter, the basic GP algorithm has been described. It is an expressive form of evolutionary algorithm useful for searching complex spaces for programs or functions, unlike any of the other three main types described. The concepts involved with the selection of a terminal and function set have been presented, together with factors that influence the representation of a possible solution in an environment. Standard methods for the recombination of solutions have also been explained. In particular, the fitness, phenotype and genotype of a solution have been explained and it has been shown how these concepts map to those in the problem domain. Other characteristics and phenomena of this learning paradigm have been discussed.

Chapter 4

Related Research: Term-Weighting Approaches

This chapter presents existing research relevant to this work. In particular, some interesting non-learning approaches to developing term-weighting schemes in IR are outlined in section 4.1. The section details some exhaustive approaches that search for *good* term-weighting scheme. Of particular interest in this section is the introduction of existing axioms for IR. These can be used to search the space of term-weighting schemes in some guided manner or can be used to theoretically motivate resulting solutions. Section 4.2 deals with learning approaches for developing term-weights and term-weighting schemes in IR. The reasons for adopting the GP paradigm to search for term-weighting schemes are motivated further in this section. Furthermore, the merits and limitations of previous approaches applying GP to IR are discussed. The chapter finishes with a summary (section 4.3) of the important points detailed within.

4.1 Non-learning Approaches

In this section, some non-learning approaches that develop term-weighting schemes using a "bag of words" approach for ad hoc retrieval are introduced. These approaches are further broken down into exhaustive (brute-

force) search techniques and analytical approaches.

4.1.1 Exhaustive Searches

There have been many attempts to exhaustively search a limited space of term-weighting functions (Salton and Buckley, 1988; Zobel and Moffat, 1998; Chisholm and Kolda, 1999). An exhaustive search of a limited space of term-weighting functions has been conducted (Zobel and Moffat, 1998) using a set of non-primitive (non-atomic) properties and features that are part of existing retrieval functions. The number of schemes that can be created using the measures is over 1,500,000 (Zobel and Moffat, 1998). Hence, much of the search space is closed and only a subset is explored. They conclude that the search space of similarity measures has a complex landscape making a simple hill-climbing algorithm ineffectual. Using non-primitive features of existing term-weighting schemes in any search (be it exhaustive or not) for term-weighting will bias (or limit) the search toward known forms (and shapes) of term-weighting functions. These approaches to developing retrieval functions are likely to fail, as there is no guarantee that existing parts of functions (which are limited in form at such a coarse level) can be effectively combined to create a high performance weighting function. Another point worth mentioning is that for such exhaustive searches, the parts of the functions to be combined must be quite coarse (non-atomic) so that the search space is quite small. The search space of term-weighting schemes is so large that an exhaustive search of the entire space is infeasible (if not impossible). Furthermore, there is no clear consensus on the features that should be used.

The composition of term-weighting schemes has previously been decomposed into two triples (Salton and Buckley, 1988) as mentioned in chapter two. They perform an exhaustive search of a limited space of weighting functions on small test collections. These term-weighting schemes have since been shown to be quite specific to the smaller collections. The main contribution of that work is the representation of a term-weighting scheme as two triples.

4.1.2 Analytical Approaches

Analytical approaches to term weighting have probably been the most common approaches to developing term-weighting schemes in the past. Many of the models introduced previously, such as the vector space model (Salton et al, 1975), probabilistic model (Robertson and Sparck Jones, 1976) and language model (Ponte and Croft, 1998; Zhai and Lafferty, 2001), have been developed analytically. There has been much interest in theoretically motivating the inverse document frequency (*idf*) measure as originally proposed (Sparck Jones, 1972). Indeed, it can been seen that there is some inconsistency between *idf* and the resolving power of a term (Luhn, 1958). Luhn predicted that the correct weighting for both high frequency and low frequency terms should be low. Indeed, some recent work using exploratory data analysis (Greiff, 1998) has indicated that a flattening of *idf* at both high and low frequencies would lead to an increase in performance.

An approach for developing a measure of term-significance has been attempted using Luhn's idea of resolving power (Zhang and Nguyen, 2005). This approach develops a measure of the most significant terms in a document and is consistent with Luhn's idea of resolving power. A measure of the most significant terms in the collection is also developed in a similar manner. However, their approach has not been tested in an ad hoc retrieval environment.

The $BM25$ scheme and the pivoted document length normalisation scheme outlined previously (chapter two) were developed analytically. They contain tuning parameters (i.e. are parametric) and in fact represent a number (or family) of term-weighting schemes in each case. A specific term-weighting scheme is only recovered by setting the tuning parameters to specific values. It can be argued that these approaches have only found an area of the search space in which some good term-weighting schemes lie. This point will be revisited later in this chapter in a discussion about automatic tuning of parameters.

Incremental approach

Recent research (Amati and Rijsbergen, 2002) has developed parts of a term-weighting scheme incrementally. A family of probabilistic term-weighting schemes has been developed analytically in this incremental three-stage approach. Starting with developing measures for determining the information content of a term (i.e. term-discrimination measures), a complete weighting approach is determined by adding two more methods of normalisation. The first method is a non-linear term-frequency factor which implicitly tends to promote documents containing more distinct query terms (this is what is often called a term-frequency factor). The second method explicitly uses the document length to penalise longer documents.

This approach constrains the shape of a term-weighting function by forcing all three aspects to be present. These assumptions are reasonable as the aforementioned aspects are present in all modern high performance term-weighting schemes. Furthermore, this approach has the advantage of reducing the vast search space, while neither limiting the actual shape of, nor features used within, the constituent functional components of the term-weighting approach. This work further reinforces the validity of the single triple desciption of a term-weighting scheme presented in chapter two.

It is also important to note that in developing a term-weighting approach using such an incremental process, it is important to develop each component function part in the correct order. The term-discrimination part should be developed first, as this is the basic weight of a keyword, taking into account its characteristics in a global context (i.e. all the information about the term in the entire collection). The term-frequency factor can then be developed. Its function is to promote documents with greater occurrences of useful terms. Thirdly, a method of normalisation can be added to the within-document term-frequency aspect using features that reflect the document length.

The correct ordering for developing the component parts of the schemes is an important consideration. Consider the case where a normalisation scheme is developed before incorporating any other aspect. As the probability of relevance is proportional to the length of the document (Singhal et al, 1996),

with no other evidence available, it can be determined that the document length (or normalisation) factor should explicitly promote longer documents. Longer documents are more relevant to topics due to the broad topic they can incorporate. However, it can be noticed that in both the $BM25$ and pivoted document length normalisation schemes, the document length aspect is inversely related to the score of a document, thus aiming to penalise longer documents.

4.1.3 Constraining the Search Space Using Axioms

A novel and elegant approach to formalising useful (if not necessary) characteristics of term-weighting schemes has been developed using constraints (Fang et al, 2004). This work proposes a number of constraints to which all *good* term-weighting functions should adhere. An axiomatic approach (Fang and Zhai, 2005) has further refined this work. This is an important step in developing term-weighting functions as it explicitly details certain operating characteristics of a term-weighting function from basic axioms that are seen as self-evident. It has been shown that when a scheme violates one of the proposed constrains, it typically indicates non-optimality of the scheme (Fang et al, 2004; Fang and Zhai, 2005). However, the search for new functions still involves manually constructing weighting functions that adhere to these constraints.

The constraints (Fang and Zhai, 2005) are briefly introduced using the inductive framework as they will be useful for latter sections of this work. The idea of the inductive framework is to define a base case that describes the score (weight) assigned to a document containing a single term matching (or not matching) a query containing a single term. All other cases can be dealt with inductively, using a document growth function (which describes the change in score when a single term is added to the document) and a query growth function (which describes the change in score when a single term is added to the query). This is an elegant approach to formalising characteristics of a term-weighting function.

This description of term-weighting components is used to formally de-

scribe three axioms (or constraints) that are seen as intuitive in a term-weighting context. Assume $S(Q, D)$ is a function which scores a document D in relation to a query Q in a standard *bag of words* retrieval model. With notation similar in style to (Fang and Zhai, 2005), the existing constraints can be formalised as follows, where $t \in T$ is a term in the set of terms in a corpus and $\delta_t(t, D, Q) = S(Q, D \cup \{t\}) - S(Q, D)$ (i.e. the change in score as t is added to the document D):

Constraint 1: $\forall Q, D$ and $t \in T$, if $t \in Q$, $S(Q, D \cup \{t\}) > S(Q, D)$

Constraint 1 states that adding a new query term to the document must *always* increase the score of that document. This captures the basic behaviour of a term-frequency aspect. This constraint is a more general version of constraints TFC1 and TF-LNC (Fang et al, 2004). This constraint ensures that the basic weight of a term (an *idf* type measure) must be positive and more importantly, that any penalisation due to the document becoming longer (normalisation) must be less than the increase in score due to the term being added.

Constraint 2: $\forall Q, D$ and $t \in T$, if $t \notin Q$, $S(Q, D \cup \{t\}) < S(Q, D)$

Constraint 2 states that adding a non-query term to a document must *always* decrease the score of that document. This constraint is similar to the LNC1 constraint (Fang et al, 2004). This constraint ensures that some sort of normalisation is present and specifies its basic operating principle.

Constraint 3: $\forall Q, D$ and $t \in T$, if $t \in Q$, $\delta_t(t, D, Q) > \delta_t(t, D \cup \{t\}, Q)$

Constraint 3 states that adding successive query terms to a document should increase the score of the document less with each successive addition. This constraint is similar to the TFC2 constraint (Fang et al, 2004). Essentially, the term-frequency influence must be sub-linear. The intuition behind this constraint is that it is ultimately the first occurrence of a term that indicates that the document is on-topic (i.e. related to the query). Due to characteristics of natural language, it is known that when a term first appears in a document, the likelihood of re-appearance increases. Thus, the weight given to successive occurrences of a query term should be reduced.

Weaker Constraint

A weaker constraint, which is non-redundant when constraints 1 and 3 are not satisfied, has previously been developed (Fang, 2007).

Constraint 1.1: $\forall Q, D$ and $t \in T$, if $t \in Q$, $S(Q, D \cup \{t\}) > S(Q, D)$

Constraint 1.1 states that adding a new query term to the document should result in a higher score for the document than adding a non-query term. It is true that if constraint 1 and 2 are satisfied then constraint 1.1 is satisfied accordingly. This constraint has previously been introduced (Fang, 2007) and is included here, as constraint 1 is not unconditionally satisfied by any *efficient* modern term-weighting scheme[1]. Due to the nature of normalisation schemes, if a term-weighting scheme uses the document length explicitly to penalise the document, constraint 1 (and consequently constraint 3) will never be satisfied unconditionally. Consider the case where a term with an extremely low *idf* (term-discrimination) value is added to a document. The penalisation due to the document increasing in length may be more than the increase in weight as the term is added. This is because the document length is used explicitly to penalise documents. As a result this penalises *all* terms which have already appeared in the document. Most if not all modern weighting schemes penalise documents in this way. However, if stop-words are removed the impact of this will be lessened.

Usefulness of constraints

Nonetheless, these constraints are used to check the validity of term-weighting schemes before evaluation. Furthermore, term-weighting schemes which adhere to these constraints are shown empirically to outperform weighting schemes that fail to adhere to one or more of the constraints (Fang et al, 2004; Fang and Zhai, 2005). The constraints are also useful in defining valid bounds on tuning parameters that appear in many existing term-weighting schemes. It should be noted that simply adhering to these constraints does not guarantee a high performance weighting scheme. Rather, it is the violation of one or more of the constraints that indicates that the performance is

[1]See appendix A for a further discussion and solution

non-optimal (i.e. breaks some rule of the proposed model of relevance). It is worth noting that these axioms *typically* only constrain a term-weighting schemes within-document features (i.e. its term-frequency aspect and normalisation aspect). It is also worth noting that the three constraints developed (Fang and Zhai, 2005) are by no means the only valid ones. Indeed, there may be many other useful (theoretically valid) constraints (or axioms) which may be applied to term-weighting schemes.

4.2 Learning Approaches

In this section learning approaches to IR are discussed. Firstly, some approaches that learn tuning parameters in specific term-weighting schemes are discussed. Non-GP approaches that attempt to learn a ranking over a set of document are then discussed. The section concludes with a summary of GP approaches applied to the problem of term-weighting in IR.

4.2.1 Automatic Tuning of Parametric Schemes

Some early approaches to learning (Bartell et al, 1994a,b) have attempted to automatically tune parameters in existing term-weighting functions. They learn a specific instance of a function within a family of functions by learning good tuning parameters in the hope that they are generalisable for unseen data.

Term-frequency normalisation has been one of the central issues in ad-hoc retrieval for many years. Normalisation schemes use a number of features relating to the lengths of documents in a collection to penalise long documents. However, most document normalisation schemes make use of a tuning parameter in order to maximize their performance. In many cases the only way to find the optimal performance for a collection is by manually tuning this parameter. Some research into normalisation has studied the normalisation parameters of the $BM25$ and pivoted normalisation scheme by calibrating them for specific collections (Chowdhury et al, 2002). It is concluded that the normalisation parameters of these schemes need to be calibrated for certain

collections to maximize performance.

Furthermore, it has been determined that the distribution of document lengths in a collection can affect the tuning parameter for specific normalisation functions (He and Ounis, 2003). The idea of a *normalisation effect* is introduced and is shown to be related to the document length distribution in the collection (He and Ounis, 2005a). It is shown to be a collection independent constant. Thus, the normalisation tuning parameter for certain normalisation schemes should be assigned different values for collections that have different document length distributions so that the *normalisation effect* remains constant. An automatic tuning approach to this problem is addressed and shown to be useful over various collections. It is suggested in the literature (He and Ounis, 2005b; Chung et al, 2006) that the query length has an effect on the setting of the tuning parameter in specific normalisation schemes. This phenomenon is closely related to those seen for various smoothing techniques used in language modelling approaches (Zhai and Lafferty, 2001). However, no reason for this difference in normalisation has been presented.

4.2.2 Learning to Rank

There have been more and more approaches that attempt to learn a ranking over a set of documents for a set of topics (queries). Many of these learning approaches do not produce a specific term-weighting scheme as they lack the capability to produce a symbolic representation of the solution. In this section, several learning paradigms that have been adopted to learn term-weights or weighting schemes for IR are discussed.

Support Vector Machines

Support vector machines (SVMs) have predominantly been used in text classification (Kwok, 1998; Leopold and Kindermann, 2002) as they are a supervised linear classifier useful for high dimensional data. Some of these approaches have yielded an increase in performance but few have tried to directly optimise MAP. Some researchers (Joachims, 2002; Cao et al, 2006;

Yue et al, 2007) have adopted SVMs to specifically learn a ranking over a set of documents for a set of training topics. However, the reason for this increase in performance is not presented nor is it known if indeed a term-weighting scheme can be re-modelled to reflect this increase in performance (i.e. a mathematical equivalent term-weighting scheme). This is due to the fact that SVMs learn the dimensional weights in the hyperplane. From this, it is difficult to gain any understanding of an underlying theoretical model of retrieval.

Genetic Algorithms

Traditional genetic algorithms (GAs) have been adopted by some to learn term-weights that are useful for retrieval. One of the first approaches using GAs (Gordon, 1988) models each document as a chromosome (individual) and performs genetic operations on these documents over generations until a useful representation of the documents is evolved. Another approach (Vrajitoru, 1999) models the entire document collection as an individual chromosome and attempts to evolve an entire set of term-weights for each document in the collection within one individual. Other approaches have been adopted and have attempted to discover the most optimal GA parameters for the problem of learning term-weighting in IR for a limited set of resources (Vrajitoru, 2000). However, the approaches mentioned are typically not generalisable as they learn specific weights for terms and do not learn a scheme for weighting terms based on term features. As the optimal set of weights can already be determined given assumptions adopted by using the binary independence retrieval model (BIR) (Robertson and Sparck Jones, 1976) and the non-binary independence model (Yu and Lee, 1986), many GA approaches may simply be discovering some aspect of these weights by implicitly using relevance feedback via the fitness function adopted. This is an interesting problem from an evolutionary learning perspective. However, learning the optimal weights on a specific collection does not lead to these actual weights being reused in a general context.

Neural Networks

A connectionist network (neural network) approach to IR has previously been attempted (Belew, 1989; Kwok, 1995). These approaches typically take a three-layer approach. Typically, one layer consists of the query-terms, which are in turn connected to a second layer consisting of all of the terms in the collection. This layer is connected (via its terms) to a third layer representing the documents that contain those specific terms. This document layer is fed back into the middle layer for all of the terms in the document. Activation starts at the query-term layer and spreads to the second layer and onto the document layer. When the documents reach a certain level of activation, they fire and activation spreads back to the middle layer, which in turn may activate different documents. In this way, the approach has the advantage of returning relevant documents which may have little or no query terms in common with the initial query. However, because of the spread of activations in neural networks, it is difficult to analyse these networks.

4.2.3 GP approaches to IR

It is becoming increasingly clear that researchers are seeing GP as a viable and novel way to solve many difficult real-world problems in the domain of IR. Most of the previously mentioned learning techniques attempt to learn the optimal weights for terms in a retrieval setting, rather than learning a scheme (or function) that can apply useful term-weights automatically. There have been several attempts to evolve the representation of Boolean type queries in order to improve their performance (Kraft et al, 1994; Smith and Smith, 1997; Owais et al, 2005). Typically, these approaches use terms extracted from the relevant documents for a given query and attempt to evolve a Boolean representation of the query which maximizes the return of these relevant documents. This can be viewed as a form of relevance feedback as it may be used in systems where a user has indicated which documents are relevant. A multi-objective approach to this problem has been addressed (Cordón et al, 2002, 2003) which attempts to maximize both precision and recall. Interesting work has also been proposed (Steele and Powers, 1998),

where words semantically related to query terms are chosen via a type of manual thesaurus and a Boolean type query is evolved in order to increase the performance of the query.

There has also been some difficulty in directly optimising the typical evaluation measures used in IR systems (Yue et al, 2007), most notably mean average precision (MAP). The GP approaches previously adopted (Oren, 2002b; Fan et al, 2004; Trotman, 2005; Almeida de et al, 2007; Jen-Yuan Yeh and Yang, 2007) are useful for many reasons. Firstly, they make few assumptions as to the possible constitution of good term-weighting schemes. For example, if *idf* is the optimal type of basic weighting for terms, a GP approach should be able to find this. The use of primitive functions and terminals allow the process to combine useful terminals and search a large function space for correct function forms. Solutions are produced as a symbolic representation, which aids generalisation when these solutions are prevented from growing too large. Furthermore, as a symbolic representation is produced, the solutions can be compared against standard benchmarks and can be analysed mathematically (e.g. using the axioms described previously in this chapter). These GP approaches are a useful form of offline learning. The solutions produced (if non-specific) can be applied to any ad hoc retrieval setting.

Oren

One of the first approaches evolving term-weighting schemes using GP (Oren, 2002b,a) uses non-atomic features of the terms, documents, queries and collection to evolve term-weighting functions for use in IR. Using parts of existing functions as terminals in the GP can be viewed as a type of seeding or biasing, as prior knowledge as to the makeup of a good weighting function is assumed. Furthermore, using such non-primitive measures can limit the search space as these non-primitive measure may not be reducible to their more primitive component measures. This work uses a small document collection (1,239 documents and 70 queries) to evolve functions using a population of 100 individuals run for 150 generations. The term-weighting schemes found tend to be quite specific but often compare favourably to stan-

dard *tf-idf* type solutions on the training data. Nonetheless, this was the first attempt using GP to evolve term-weighting schemes for IR and showed that GP is an interesting and feasible approach for such a problem.

Fan et al.

Another approach evolving term-weighting schemes (Fan et al, 2004) which assumes a simple query term weighting (i.e. tf_t^Q) has also been attempted. The function set in this approach is limited and the terminal set, although primitive, does not span the entire range of primitive measures. Although the term-weighting functions are learned using only short queries, the training and test collections are larger than those used in previous research. The functions produced are shown to perform slightly better than a standard version of $BM25$ for short queries on general data. This approach uses typical GP parameters and a population of 200 individuals for 30 generations. However, due to a lack of analysis of the resulting solutions, it is unclear whether this GP approach has discovered any new properties useful in retrieval or whether it has discovered a tuned version of an existing function (e.g. $BM25$).

Trotman

An approach using primitive atomic features of terms, documents, queries and the document collection has also been attempted (Trotman, 2005). This approach uses a large extensive terminal and function set with little or no constraints on the search space. This approach explores a much larger search space than previous approaches and importantly uses primitive measures throughout. The approach also uses sizeable test collections and importantly uses longer queries for training. This is an important factor in developing a term-weighting approach that can correctly learn the relative weights between different terms in the query. The approach uses a seeded population of 100 individuals (96 random and 4 existing functions) run for 100 generations 13 times. It is reported that two of the resulting term-weighting functions achieve a significant increase in MAP over the default $BM25$ scheme on many of the test collection used. However, an in-depth analysis of the best

functions found was not conducted.

Almeida et al.

More recently, a GP approach optimising MAP for specific collections has been attempted (Almeida de et al, 2007). This approach specifically uses non-primitive parts of existing term-weighting schemes as terminals. This approach also includes complex features from each part of the triple of a term-weighting scheme (namely term-discrimination, term-frequency and normalisation components). The lack of primitive measures in the terminal can leave a large area of the search space unexplored. Due to the complex nature of the terminals used, it is difficult to analyse any function produced. Thus, it is extremely hard to deduce anything about the nature of ad hoc retrieval as a consequence. The approach adopted can be viewed as a learning approach to fusion, as it combines and aggregates existing retrieval functions. One of the main advantages of GP is that it produces a symbolic representation of a solution which is often generalisable. This study may have somewhat neutralised both of these advantages. A similar approach (Jen-Yuan Yeh and Yang, 2007) uses somewhat complex features and can also be viewed as a learning approach to fusion.

Summary of GP Approaches to IR

These four approaches develop entire term-weighting schemes. However, none of these works formally analyse the solutions produced, nor do they attempt an analysis of the ranked lists produced by a solution. Oren (2002b) indicates that the solutions produced from his approach are most likely not generalisable. Fan's (2004) approach uses only short queries for training and it is not certain if the term-weighting schemes produced are generalisable, especially for longer queries. Of the approaches adopted to date that of Trotman's (2005) seems to be have made fewer assumptions and explored a large search space using primitive measures. In a later chapter (chapter nine), one of the best schemes presented in each of three of these approaches is evaluated on unseen test data and tested for constraint satisfaction. This

will aid in the comparison between the approach adopted in this work and the GP approaches previously adopted.

4.3 Summary

In this chapter, some recent research relevant to this work has been discussed. Both learning and non-learning approaches to developing term-weighting schemes have been discussed. An exhaustive search of a limited search space of possible term-weighting functions has been shown to be infeasible. Furthermore, these approaches continue to adopt the triple representation of a term-weighting scheme. Of particular importance is the axiomatic approach which formalises necessary characteristics of good term-weighting schemes. The adherence to known axioms can be useful in theoretically motivating term-weighting functions developed using the incremental GP adopted in this work.

Learning approaches have been described which range from learning tuning parameters in existing term-weighting functions to developing entire term-weighting functions. Some learning approaches only learn the actual document term-weights in some vector representation and are consequently not generalisable. Therefore, GP is potentially an extremely useful learning paradigm for navigating the large complex search space of term-weighting schemes.

Chapter 5

System Design and Experimental Setup

This chapter discusses the design of the system and describes the experimental setup used to test the hypotheses identified in chapter one. Section 5.1 describes the design and flow of the GP system. Some typical GP parameters used in the experiments are also outlined in this section. Section 5.2 describes the experiments undertaken in the upcoming chapters to empirically test the hypotheses. This section (5.2) maps the upcoming experimental chapters to each of the hypotheses. The document collections used for training, validation and testing are outlined in section 5.3. The benchmark solutions used in many of the experiments are outlined in section 5.4 together with details of the manual tuning conducted to optimise their performance. As the encoding and possible representation of a solution is crucial in the GP process, the selection of the terminal and function sets are presented in section 5.5. A brief discussion of the fitness function used is also included in this section, before the main points of the chapter are summarised in section 5.6.

5.1 Flow of the System

In this section the design of the system is discussed. The system evolves the term-weighting scheme which is the method for assigning weights in the doc-

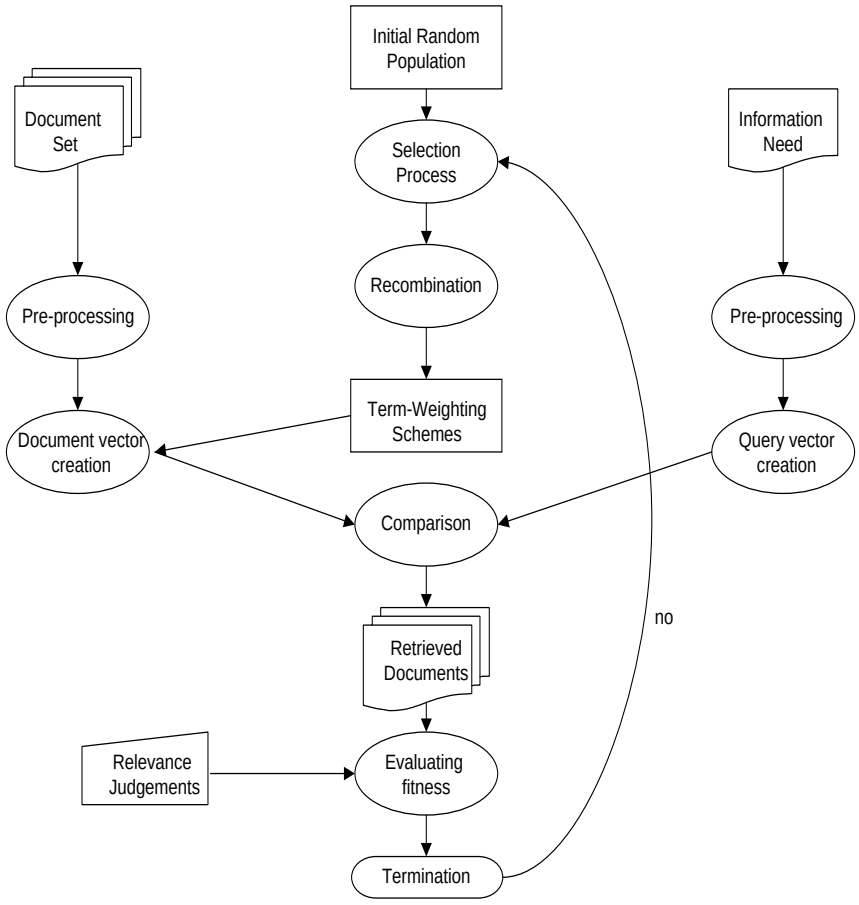

Figure 5.1: Flow of the designed system

ument vector representation. Therefore, before the comparison of each of the document vectors to the query, the weights must be calculated for each term in the document vector. Figure 5.1 shows the flow of the system. The query representation used is simple and remains fixed in this work. Both documents and queries are pre-processed. Primitive features of the document, query and entire collection are made available to the document representation process. It is important to allow the GP to use many atomic integer features that

relate to a term, the documents and the collection. These features should be intuitive and primitive in nature. The initial approach allows the GP to discover if these measures are useful in determining the relevance of a document. The implementation of the system reflects the maximum ease of calculation of such measures. An initial population of term-weighting schemes is generated at random and these schemes are used to weight terms in the documents using the primitive features gathered at the pre-processing stage. Once the weights on the document and query vectors have been calculated and assigned, the system compares the query vector to all the document vectors, producing a ranked list of documents. The fitness is then calculated for each scheme by comparing the ranked list against the human determined relevant documents for each query. If the stopping criterion is not met, a new generation of schemes is created by recombining the genomes of the fitter schemes selected during the selection process. The process of using the population of schemes to assign weights to the document vectors and then evaluating the fitness of these schemes for use in the selection process, continues for N generations.

5.1.1 GP Parameters

Table 5.1: GP parameter settings

Key	Typical Values
Population Size	100 (200)
Number Of Generations	50 (25)
Crossover Probability	90%
Creation Probability	5%
Mutation	5%
Creation Type	Ramped Half and Half
Maximum Depth	6
Selection Type	Tournament Selection
Tournament Size	3
Elitism	Yes

Table 5.1 shows the main parameters in the GP and the typical values used for the experiments. The population size and number of generations are self-explanatory. The crossover probability is the probability that a member of the next generation is created by crossover. The creation probability is the probability that a member of the next generation is created at random. As newly created solutions usually have a lower fitness, this parameter is usually assigned a low value, but is one method of introducing new genetic material to a generation. The creation type is 'ramped half and half', which means that half of the solutions have a maximum depth for all branches and half the solutions are created with various different depths. The maximum depth for creation and crossover are set to the same value to limit the depth of all trees in the population. Often, during crossover the depth of trees will grow quickly. These parameters prevent this and promote generality. The selection type used is tournament selection and the tournament size can be adjusted. Elitism is used whereby the best individual in the generation is automatically copied into the next generation. In choosing the population size and number of generations for the GP, a number of factors should be considered. The size of the search space is obviously crucial. The number of terminals, the number of functions and the depth of the trees (length of the solutions) contribute to the size of the search space. The parameter values used in this research are in line with previous research in this domain (Fan et al, 2004; Trotman, 2005; Oren, 2002b).

5.2 Outline of Experiments Undertaken

5.2.1 Evolving Global and Term-Frequency Schemes

In chapter six, experiments relating to the evolution of several term-discrimination (global) schemes are discussed. In these experiments, a binary weighting is placed on the within-document components. Once validation is complete a suitable term-discrimination scheme is chosen. Using this suitable term-discrimination scheme, a term-frequency component is learned which is dependent on the evolved term-discrimination scheme. This chapter presents

the results from these two phases of learning.

5.2.2 Evolving Normalisation Schemes

In chapter seven, the problem of learning normalisation schemes is addressed. Some preliminary analysis is conducted to aid the understanding of the normalisation problem. This analysis is utilised to construct training data in an attempt to overcome the collection dependence problem that adversely affects many normalisation schemes. Several normalisation schemes are evolved, dependent on the previously evolved parts of the term-weighting triple. The entire scheme produced from the approach is evaluated on unseen test data. Following these three phases of learning a complete evolved term-weighting scheme can be tested against the benchmark term-weighting schemes. The experiments in this chapter (together with those in chapter six) aim to test the first hypothesis [*H1*] which states that the GP approach adopted can find term-weighting schemes that outperform the best benchmarks.

5.2.3 A Phenotypic Analysis of the Search Spaces

In chapter eight, measures of the differences in the phenotype of the solutions are presented. The best solutions from different runs of the GP are used to discover if the GP is finding solutions that promote similar relevant documents (on the training data) or if the GP is finding solutions that are vastly different from one another (i.e. promote different relevant documents). This approach can identify where the solutions lie in relation to each other in the search space. These experiments aim to test the second hypothesis [*H2*] which states that the better solutions produced by the GP are clustered closely in the search space.

5.2.4 A Genotypic Analysis Using Constraints

In chapter nine, a selection of learned term-weighting schemes schemes are analysed to determine if they adhere to existing constraints. It is hoped that the approach adopted in this work finds term-weighting approaches that are

consistent with the existing axioms. Three previous approaches which develop term-weighting schemes using genetic programming (Oren, 2002b; Fan et al, 2004; Trotman, 2005) are analysed. The benchmark schemes and the learned schemes are analysed using known axioms in an attempt to theoretically motivate the schemes resulting from the GP. The analysis and experiments in this chapter aim to test the third hypothesis [*H3*] which states that the schemes evolved using the incremental process can be theoretically validated using the axioms for IR.

5.2.5 GP for Automatic Query Expansion

In chapter 10, the framework is adapted to evolve schemes for automatic query expansion. Schemes that select terms based on co-occurrence measures and measures gathered from the top few documents from an initial retrieval run, are developed using a GP framework. The experiments in this chapter aim to test the fourth hypothesis [*H4*] which states that GP can be used to improve the performance of term-selection schemes in automatic query expansion approaches.

5.3 Document Test Collections

Table 5.2: Document collections

Name	Collection	#docs	dl_{avg}	$\sigma(dl)$
FR	Federal Register (1994)	55,630	387.1	1365.2
LATIMES	LA Times	131,896	251.7	251.9
FT	Financial Times (1991-1993)	138,668	221.8	196.4
FBIS	FBIS	130,471	249.9	554.4
OH89	OHSUMED (1989)	74,869	76.9	61.4
OH90-91	OHSUMED (1990-1991)	148,162	81.4	64.0

Collections from TREC[1] disks 4 and 5 are used as document collections. Table 5.2 shows some characteristics of the document collections used. Subsets of documents from the OHSUMED[2] (Hersh et al, 1994) collection are

[1] http://trec.nist.gov/

[2] http://trec.nist.gov/data/filtering/README.t9.filtering

also used as collections. It can be seen that these documents are shorter and do not vary as much in length as those in other collections.

Table 5.3: Topics

Topics	Avg. Topic Length		
	short	med	long
301-350	2.4	12.4	43.9
351-400	2.4	10.4	32.9
401-450	2.4	9.0	27.5
1-63 (OH)	2.2	5.1	None

Table 5.3 shows the typical lengths of the topics used. TREC topics 301-450 are used with the TREC document collections outlined. For each set of topics, a short query set, consisting of the title field of the topics, a medium length query set, consisting of the title and description fields, and a long query set consisting of the title, description and narrative fields is created. 63 short queries were created for the OHSUMED collections by removing some terms from description field, as there is no title field available for this collection. No long queries are available for the documents in the OHSUMED collection.

Standard stop-words from the Brown Corpus[3] are removed from both documents and queries, and the remaining words are stemmed using Porter's algorithm. No additional words are removed from the narrative fields as is the case in some approaches. Only topics that have relevant documents associated with them are actually used, as performance is undefined for topics with no relevance judgments.

5.3.1 Training Data

The training collections used for each experiment will be detailed as they are presented. Although *some* topics are used for both training and testing, the relevance judgments and documents used in such circumstances are different. Thus, essentially this results in different data for training and testing as frequency characteristics and relevance judgments will vary.

[3]http://www.lextek.com/manuals/onix/stopwords1.html

5.3.2 Validation Data

The schemes that are produced by the GP are validated on the FT collection using topics 301-350 for short, medium and long topic lengths. Essentially, the schemes are validated on 46 short topics, 46 medium length topics and 46 long topics on a sizeable collection. The best scheme from this validation process is then chosen as the most general scheme.

5.3.3 Unseen Test Data

The test data consists of topics 351-450 used with the FT collection (96 topics), topics 351-450 used with the LATIMES collection (95 topics), topics 301-450 used with the FR collection (64 topics), topics 301-450 used with the FBIS collection (115 topics) and topics 1-63 used with both OH89 and OH90-91 (63 topics for each collection).

5.4 Benchmark Solutions

The $BM25$ and pivoted document length normalisation schemes are used as the main benchmark solutions in the experiments. The default values for the tuning parameters are used in these benchmark solutions. A tuned version of the benchmarks is also used. The $BM25$ scheme is tuned using two values of k_1 (1.2 and 2.0, as they are the most commonly reported in the literature) and nine values of b (0, 0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.825 and 1) for each query type (short, medium and long). The best performing k_1 value on all collections was the default value of 1.2 which was used for all the experiments. The best values of b for short, medium and long queries on the test data were 0.125, 0.375 and 0.625 respectively which are used when dealing with these query types.

The pivoted normalisation function is also manually tuned using seven values of s (0, 0.025, 0.05, 0.1, 0.2, 0.3 and 0.4) for each query type. Values greater than 0.4 for s perform poorly (Fang et al, 2004). The best values of s for short, medium and long queries on the test data were 0.025, 0.05 and 0.2 respectively.

5.5 Terminal and Function Sets

Table 5.4: Global terminal set

Terminal	Description
df_t	the number of documents in which term t occurs
cf_t	the number of occurrences of term t in the collection
N	the number of documents in the collection
V	the size of vocabulary of the collection
T	the total number of words in the collection
1	*the constant 1*
0.5	*the constant 0.5*
10	*the constant 10*

Table 5.4 describes the terminal set for the term-discrimination (global) weighting problem. The global terminal set contains simple atomic features that relate to the chacteristics of the terms and documents in a collection-wide context. At this level there is no information about terms in specific documents or lengths of specific documents. As a minimum criteria, the features that create existing *idf* factors should be available to the GP. While this set (Tables 5.4) may not be exhaustive, it is certainly inclusive of the majority of primitive features in these types of schemes.

Table 5.5: Term-frequency terminal set

Terminal	Description
tf_t^D	the number of occurrences of term t in document D
1	*the constant 1*
0.5	*the constant 0.5*
10	*the constant 10*

Table 5.5 shows the terminals that are available to the term-frequency influence component. This set is relatively small as the actual term-frequency is the only variable terminal. Positional (spatial) evidence is not used in

traditional vector systems. One previous approach using GP for this problem uses an 'accumulator' to allow some evidence as to the relative position of the occurrence of each term in a document (Trotman, 2005). A term that occurs at the begining of a document might be more important than if it occurs towards the end of a document. However, using this type of feature violates the original assumptions that were detailed earlier (chapter 2) about term-independent vector based term-weighting schemes and for this study potential spatial evidence regarding term occurrence is rejected.

Table 5.6: Normalisation terminal set

Terminal	Description
dl	the length of a document
dl_{avg}	the average document length in the collection
dl_{dev}	the standard deviation of document lengths in the collection
1	*the constant 1*
0.5	*the constant 0.5*
10	*the constant 10*

Table 5.6 shows the terminal set for the normalisation component. It contains a somewhat limited set of terminals that uses the lengths of documents as features. Other length normalisation factors could have been included (e.g. the vector length or the maximum term-frequency within a document). However, some of these length factors contain less information due to their lack of granularity. For example, using the total number of words in a document is a more accurate description of its length than using the maximum term-frequency. However, it is true that some length features contain information not available to others. For example, given the total length of a document, the number of unique terms in a document (vector length) cannot be known exactly, although it may be estimated from information about the total number of terms in the document. For this study, the most accurate document length feature was chosen and some statistical features of the document lengths throughtout the collection were also calculated. A useful study that measured the benefit of potential terminals in a complete

term-weighting framework on small collections showed no particular benefit of one length factor over another (Cummins and O'Riordan, 2006a) or indeed a benefit in using these length factors in combination.

Table 5.7: Function set

Terminal	Description
$+, -, /, \times$	standard arithmetic operators
$log()$	the natural log
$exp()$	exponential
$square$	the square
$\sqrt{}$	the square-root

Table 5.7 shows the function set used in all of the experiments. Standard arithmetic operators are used together with some non-linear operators. The combinations of such functions allow a large function space to be searched. Previous approaches applying GP to IR (Oren, 2002b; Fan et al, 2004; Trotman, 2005) contain most of these primitive terminals and functions. The terminal and function sets used for the automatic query expansion approaches are detailed in the chapter containing the experimental results for that piece of work.

5.5.1 Fitness Function

The fitness function chosen is mean average precision (MAP). In particular, this measure favours systems which retrieve relevant documents early in the ranking produced. When comparing the fitness of different weighting schemes, a single measure that conveys the effectiveness of a scheme is required. MAP is a viable measure as it is calculated over *all* points of recall and is used in practice as a measure of the performance of IR systems. This is useful even if a certain ranking is poor (i.e. many relevant documents are placed at a low ranking) as the MAP metric supplies evidence as to how poor this ranking is, and more importantly supplies evidence as to where in the search space more favourable rankings may lie.

5.5.2 Statistical Significance

Significance tests are carried out using a one-tailed t-test. The null hypothesis is rejected at the 5% level ($p < 0.05$). A rule of thumb for IR systems is that, for a set of 50 topics, a difference of about 5% absolute MAP is often required in order to see a *significant* increase (Buckley and Voorhees, 2000). Using more topics (as is the case in some of the test collections used here) a significant increase in MAP may be seen with a much smaller difference in absolute MAP. Significance tests may be unable to detect a difference when a difference is present for a number of reasons. The sample size used may be too small to detect a difference or the test itself may not be adequate given the distribution of the data. In general, it is useful to evaluate a new system over a number of different test collections so that a more general view of the system can be determined (Hull, 1993).

5.6 Summary

The design of the system and the experimental setup has been outlined. In particular, the flow of the system has been presented and the GP parameters have been briefly explained. A brief description of all of the upcoming experiments is presented and these are mapped to each of the hypotheses outlined in chapter one.

Importantly, the function and terminal sets for the three stages of the weighting problem have also been defined and have been limited to relatively simple primitive measures. The list of terminals and function used is at least sufficient to enable the GP to find standard benchmark term-weighting approaches.

From an evaluation aspect, document collections and benchmark solutions have been detailed. The fitness function and the need for statistical tests have been briefly motivated.

Chapter 6

Evolving Global and Term-Frequency Schemes

In this chapter results from the experiments that evolve the first two components of a term-weighting scheme are described. The first component of the term-weighting scheme that is evolved is a basic global weighting scheme for terms (term-discrimination scheme) (section 6.1). In the benchmark schemes this is an *idf* factor. Thus, one would expect that a successful GP run would find a scheme at least comparable to the performance of *idf*. These global schemes aim to measure the semantic content of a particular term from their global frequency characteristics. Consequently, these types of schemes are important in other areas of information science.

The second component developed in this chapter is the term-frequency factor (section 6.2). This component determines the weight to apply to the occurrences of query-terms in the document. These term-frequency schemes can be viewed as within-document schemes as they determine the relative importance of term occurrences in a specific document. The results from the experiments that evolved these schemes are presented in the latter half of this chapter.

6.1 Global (Term-Discrimination) Schemes

In the first section of this chapter, a term-discrimination (global) scheme is evolved in the following framework:

$$S(Q,D) = \sum_{t \in Q \cap D} (gw \cdot tf_t^Q) \tag{6.1}$$

where gw is the global scheme. The *idf* functions that appear in the $BM25$ and pivoted document length normalisation function are used as benchmarks against which to compare the evolved schemes. w_1 and w_2 (equations (2.4) and (2.6)) from chapter two can be substituted for gw in this function to complete the benchmark retrieval functions. It has been stated that the *idf* in the $BM25$ (w_1) scheme will often lead to poor results due to a possible negative weight for certain frequent terms (Fang et al, 2004; Fang and Zhai, 2005). As standard stop-words are removed, no negative weights are assigned to terms in long (verbose) queries. Tables 5.4 and 5.7 (chapter 5) are used as the terminal and function set for this global weighting problem.

6.1.1 Training

Table 6.1: Global weighting training collection

Name	Collection	#docs	dl_{avg}	$\sigma(dl)$	Topics	# Topic	length
GLOBAL	OHSUMED (1988)	35,412	72.7	59.2	1-63	63	med

The training collection for this problem consists of approximately 35,000 OHSUMED documents from 1988 and the 63 topics for that collection. Table 6.1 shows some characteristics of the training set. The topics range in length from 2 to 10 words with an average length of 5.1 words. The GP was run seven times with an initial random population of 100 individuals for 50 generations. The best solution produced from each run was retained for validation.

Table 6.2 shows the MAP of seven global evolved weighting schemes (gw) and the benchmarks on the training data. It can be seen that all the evolved schemes are better than the benchmarks (w_1 and w_2) in terms of MAP.

Table 6.2: %MAP for global weightings on training collection

Name	w_1	w_2	gw_1	gw_2	gw_3	gw_4	gw_5	gw_6	gw_7
GLOBAL	19.83	19.98	22.05	21.98	21.60	21.69	20.11	20.11	20.75

Simplification and Validation of Schemes

The seven actual forumlas evolved by the system and their simplifications are as follows:

$$
\begin{aligned}
gw_1 &= \frac{V^2 \cdot \sqrt{cf} \cdot \frac{cf}{df} \cdot \frac{cf}{df}}{df \cdot T} + \sqrt{cf} \\
&= \frac{V^2 \cdot cf^{2.5}}{T \cdot df^3} + \sqrt{cf} \\
gw_2 &= (\frac{-\sqrt{cf}}{-df}) \cdot (log(\frac{0.5}{cf}))^2 \cdot (\frac{\frac{cf}{df}}{log(\frac{0.5}{cf})})^2 \\
&= \frac{cf^{2.5}}{df^3} \\
gw_3 &= \sqrt{(log(\frac{cf}{df}))^2 \cdot ((\frac{N}{df}) \cdot (\frac{N}{df/N}) + 1)} \\
&= \sqrt{(log(\frac{cf}{df}))^2 \cdot \frac{N}{df} \cdot (\frac{N^2}{df} + 1)} \\
gw_4 &= \sqrt{\frac{cf}{df} \cdot \frac{cf}{df} \cdot \frac{cf}{df} \cdot \frac{N}{df}} \\
&= \sqrt{\frac{cf^3 \cdot N}{df^4}}
\end{aligned}
$$

$$
\begin{aligned}
gw_5 &= \sqrt{\sqrt{\frac{0.5}{df}}} \\
gw_6 &= \sqrt{(\sqrt{\frac{\sqrt{df}}{df}})^2} \\
&= \sqrt{\frac{\sqrt{df}}{df}} \\
gw_7 &= \sqrt{\sqrt{\frac{\frac{cf}{N}}{df^2}}} \\
&= \sqrt{\frac{\sqrt{cf/N}}{df}}
\end{aligned}
$$

Table 6.3: %MAP for global weightings on validation data (topics 301-350)

Collection	w_1	w_2	gw_1	gw_2	gw_3	gw_4	gw_5	gw_6	gw_7
FT	21.62	21.57	25.54	24.99	21.03	26.12	22.58	22.58	23.41

Table 6.3 shows the MAP of seven global evolved weighting schemes (gw) and the benchmarks on the validation data. Six of the seven schemes outperform both types of *idf* on the validation data. It can be seen that the best schemes gw_1, gw_2 and gw_4 all contain a similar component (i.e. $\frac{cf}{df}$) which is a measure of density that has been previously proposed (Kwok, 1996; Franz and McCarley, 2000; Cummins and O'Riordan, 2006a). However, it has been incorrected motivated in many studies. This factor is more useful for longer queries and not as useful for shorter queries as was originally proposed (Kwok, 1996). This will be seen empirically in the results section. gw_4 outperforms all other schemes and is chosen as the best general term-discrimination weighting function. This function has been validated over short, medium and long queries (46 topics for each query length) and is a good choice as a basic weighting for terms.

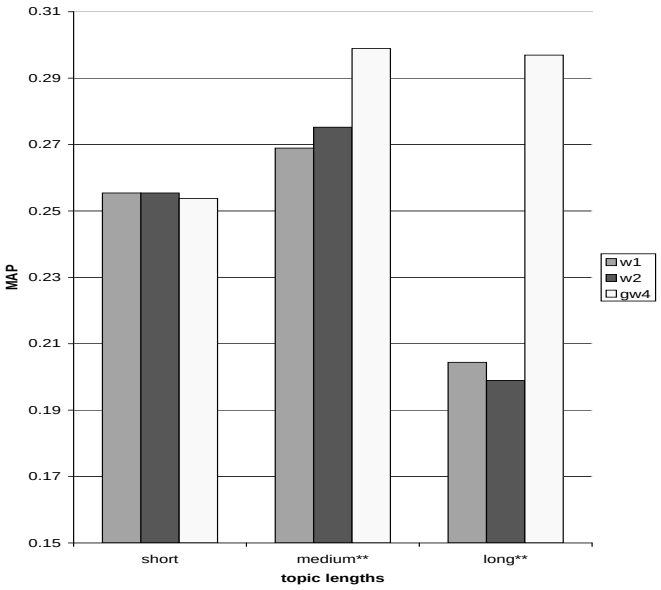

Figure 6.1: MAP of global schemes on FR collection

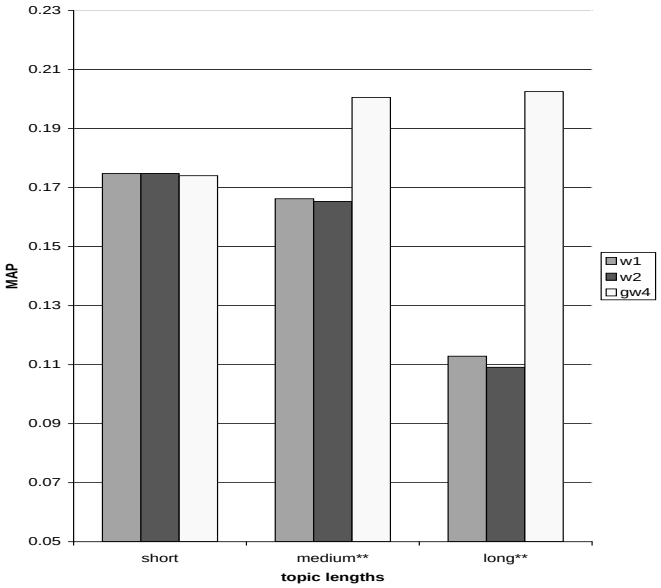

Figure 6.2: MAP of global schemes on FBIS collection

6.1.2 Results and Discussion

Figures 6.1, 6.2, 6.3, 6.4 and 6.5 show results on unseen test data. It can be seen that the evolved scheme (gw_4) remains the best performing global

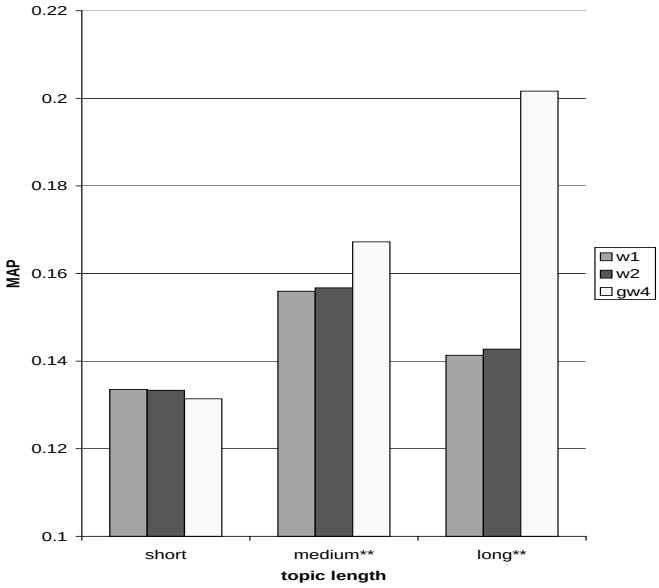

Figure 6.3: MAP of global schemes on LATIMES collection

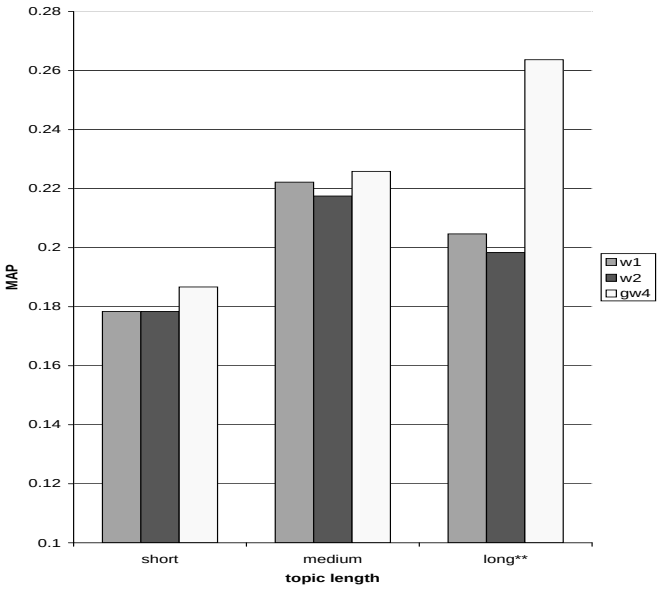

Figure 6.4: MAP of global schemes on FT collection

weighting on various collections. Statistical significance is denoted by two asterisks (**). It can be seen that there is little difference between the three weightings used for short queries. This is because when the query is short,

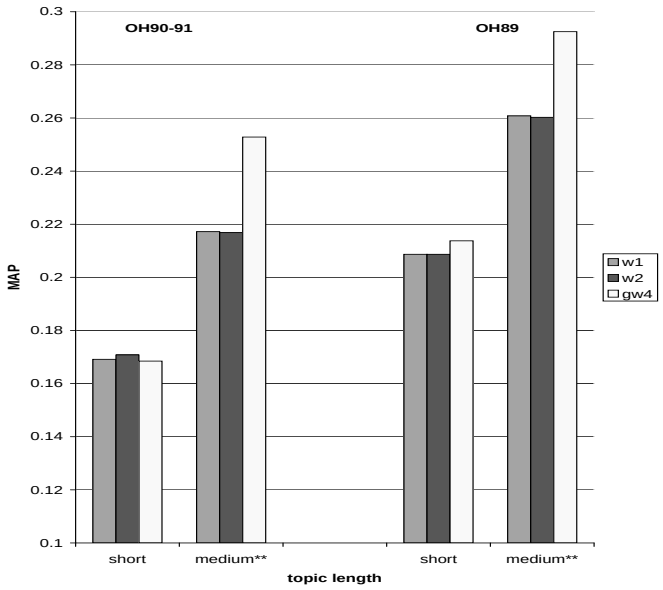

Figure 6.5: MAP of global schemes on OHSUMED collections

global weighting is not as important. For one-term queries, global weighting actually becomes redundant. For medium and long queries, the gw_4 weighting is significantly better ($p < 0.05$) than either types of *idf* used. Both measures of *idf* are similar in terms of performance on all collections. Another interesting point is that as the query length increases from medium to long, both types of *idf* perform worse on all data sets. This suggests that *idf* is incorrectly weighting many terms, and that this becomes more apparent as the topic length increases. It should also be noted that this poor performance is not attributable to the possible negative weighting that w_1 can assign to certain terms; as w_2 can never assign negative weights and performs similarly to w_1. There are indeed many noisy terms in the narrative field of the topic. However, the performance of the evolved weighting always increases as the topic length increases, which suggests that it applies a correct weighting to these types of terms. A previous analysis of global schemes that contain a type of density measure ($\frac{cf}{df}$) indicates that they are consistent with Luhn's theory regarding the resolving power of terms (Cummins and O'Riordan, 2006a). It has been shown that, unlike *idf*, certain terms, that are neither

too rare nor too frequent, receive the highest weight. It is interesting that this has been found using evolutionary techniques.

As many feature extraction techniques and measures of similarity between entire documents use *idf* as their basis, it is interesting that this can be improved upon. *idf* is used in many other areas in information science. The newly evolved measures here are likely to be useful in such areas. For example, in document clustering techniques similarity between documents are calculated prior to retrieval. As documents are typically much longer than the queries being used here, it is likely that there would be a substantial increase in performance using such a measure, as it can be seen that for longer queries the performance increase over *idf* is substantial and significant.

6.2 Term-Frequency Schemes

Now that a new useful global weighting scheme has been validated (gw_4), it can be fixed in the framework and the term-frequency aspect of the formula can now be evolved. In this section, a term-frequency scheme ($tff()$) is evolved in the following framework:

$$S(Q,D) = \sum_{t \in Q \cap D} (tff() \cdot gw_4 \cdot tf_t^Q) \tag{6.2}$$

where $tff()$ is the term-frequency factor. As benchmarks against which to compare the evolved schemes, the $BM25$ and pivoted document length normalisation schemes are used (ignoring normalisation). Thus, for the $BM25$ scheme, b is set to 0 and k is set to 1.2 (its default value). For the pivoted document length normalisation scheme, s is set to 0. The reason for this is that when using these parameter setting for the benchmarks, they both contain only a global and term-frequency element. Table 5.5 and 5.7 are used as the terminal and function set for this term-frequency weighting problem. A brief analysis of the term-frequency factors evolved in the experiments is also included in this section.

6.2.1 Training

Table 6.4: Term-frequency training collection

Name	Collection	#docs	dl_{avg}	$\sigma(dl)$	Topics	# Topics	Length
LOCAL-TF	LATIMES	32,059	250.1	259.7	301-350	37	med

The training collection for this problem (Table 6.4) consists of approximately 32,000 LA Times documents and topics 301 to 350 that have relevant documents associated with them in that collection. These documents are used to learn a within-document term-frequency factor as these documents are longer and have a higher standard deviation of document length compared to the previous training collection used. Thus, the term-frequencies vary considerably in this environment and therefore, the training collection

Table 6.5: %MAP for term-frequency weightings on training collection

Name	$BM25_{b=0}$	$PIV_{s=0}$	tff_1	tff_2	tff_3	tff_4	tff_5	tff_6	tff_7
LOCAL-TF	25.85	26.62	27.44	27.88	27.99	27.67	23.90	27.22	27.63

contains more varied characteristics which may aid generalisability. Medium length topics are used which range in length from 5 to 35 words with an average length of 9.9 words. The GP was run seven times using a population of 200 for 25 generations. The best solution from each run was retained for validation.

Table 6.5 shows that six of the seven term-frequency evolved weighting schemes (when combined with the previously evolved global scheme) outperform the two benchmarks on the training data. Many of the solutions have a similar performance on the training data. The fifth run (tff_5) produced a constant weighting and has no term-frequency aspect present (the tf_t^D terminal must have been eliminated from the population, or was combined unsuccessfully, in early generations and was not re-introduced successfully).

Simplification and Validation of Schemes

The seven actual forumlas evolved by the system and their simplifications are as follows:

$$
\begin{aligned}
tff_1() &= ((\sqrt{tf_t^D/1})/(tf_t^D \cdot 10)) + ((tf_t^D)/(0.5 + tf_t^D)) \\
&= \frac{\sqrt{tf_t^D}}{tf_t^D \cdot 10} + \frac{tf_t^D}{0.5 + tf_t^D} \\
tff_2() &= \sqrt{log(tf_t^D)^2 + \sqrt{log(tf_t^D)}} + \sqrt{(10^2) + ((-tf_t^D) + (log(tf_t^D)))} \\
&= \sqrt{log(tf_t^D)^2 + \sqrt{log(tf_t^D)}} + \sqrt{100 - tf_t^D + log(tf_t^D)}
\end{aligned}
$$

Table 6.6: %MAP for $tff()$ weightings on validation data (topics 301-350)

Collection	$BM25_{b=0}$	$PIV_{s=0}$	tff_1	tff_2	tff_3	tff_4	tff_5	tff_6	tff_7
FT	25.33	20.59	27.12	27.48	28.02	27.64	26.12	27.85	27.87

$$
\begin{aligned}
tff_3() &= \tfrac{log(0.5)}{tf_t^D} + 10 + \tfrac{log(tf_t^D)}{log(1+tf_t^D)} + (\tfrac{log(tf_t^D)}{log(1+tf_t^D)}/log(log(10))) \\
&= 10 + \tfrac{log(0.5)}{tf_t^D} + \tfrac{log(tf_t^D)}{log(1+tf_t^D)} + \tfrac{log(tf_t^D)}{log(1+tf_t^D)\cdot log(log(10))} \\
tff_4() &= (((((-1)^2)/((1\cdot 0.5)\cdot(-1))) + (((-1)^2)/((tf_t^D/0.5) + (tf_t^D/0.5)))))^2 \\
&= (\tfrac{1}{tf_t^D} - 2)^2 \\
tff_5() &= 1 \\
tff_6() &= \sqrt{\sqrt{tf_t^D/(tf_t^D+0.5)}\cdot\sqrt{tf_t^D/(tf_t^D+0.5)}} \\
&= \sqrt{tf_t^D/(tf_t^D+0.5)} \\
tff_7() &= \sqrt{(((log(1+0.5)) + ((log(tf_t^D))/(\sqrt{tf_t^D})))\cdot(tf_t^D/tf_t^D))} \\
&= \sqrt{log(1.5) + \tfrac{log(tf_t^D)}{\sqrt{tf_t^D}}}
\end{aligned}
$$

Table 6.6 shows that all seven of the schemes outperform the benchmarks on the validation data. It can be seen that some of the functions can be reduced to a relatively simple form and that many have a similar performance. As $tff_3()$ is the best performing scheme on the validation data, it is chosen as the best term-frequency factor.

Relative Term-Frequency Influence

A brief analysis of the term-frequency factors of the evolved schemes show that they assign a similar weighting to terms (at least in the training environment). It has been shown by empirical evidence throughout the literature that the term-frequency factor should typically be non-linear (Robertson et al, 2004). Furthermore, the analysis that follows shows that the evolved term-frequency factors place a lesser influence on term-frequency than the benchmark approaches. Consider the case of a document of average length. Using notation similar to that previously used, if $\delta_t(t, D, Q) = S(Q, D \cup \{t\}) - S(Q, D)$ is the change in weight as a query term ($t \in Q$) is added to the document, then the relative change in score ($R\_S()$) as the term-frequency increases by 1 can be described as follows:

$$R\_S() = \frac{\delta_t(t, D, Q)}{S(Q, D)} \tag{6.3}$$

where $|D \cap t| > 0$ as $R\_S()$ is undefined at $tf_t^D = 1$. This is an important factor in determining how the increase in term-frquency will affect the score of a document (when dealing with a specific term). This factor is important as the actual change in weight alone is unimportant in a ranking situation. Thus, this equation aims to measure the increase in weight as a term is added when compared to the previous weight of the document.

Three evolved functions and the two benchmarks schemes are plotted for a single term t with an increasing term-frequency using this relative term-frequency measure. Figure 6.6 shows this relative increase in weight for a specific query term in a typical term-frequency range (1-10) for the benchmark schemes and three evolved schemes ($tff_3()$, $tff_4()$ and $tff_6()$). Firstly, it can be seen that the relative change in weight is decreasing for the functions chosen. This indicates that as the term-frequency increases, the increase in weight decreases. Many of the schemes evolved have similar characterisics for the term-frequency counts that appear in the training data. It is interesting that the evolved term-frequency functions apply a considerably lower relative weight than either of the term-frequency factors in the benchmarks. The first point (1 on the x-axis) indicates the relative increase in weight as

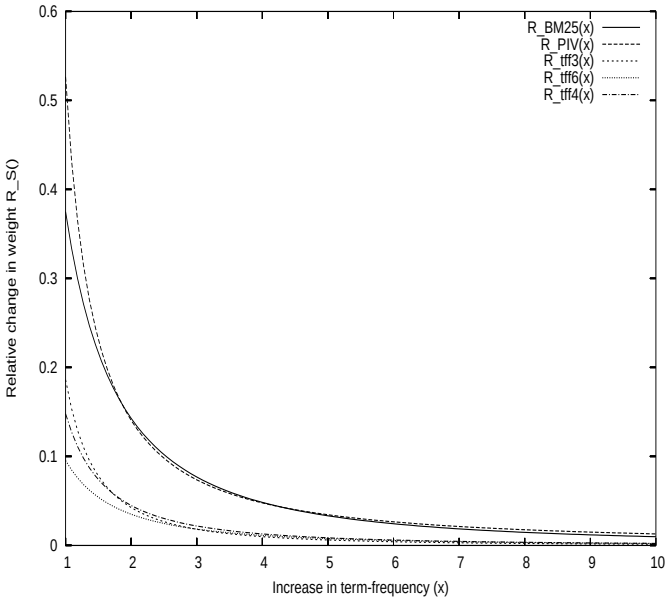

Figure 6.6: Measuring the relative increase in term-frequency

the term-frequency increases from 1 to 2, for a document of average length. This point will account for much of the weight applied to a document as many query terms may only occur once or twice in a document. It can be seen that the pivoted document length normalisation scheme has the largest relative term-frequency influence.

It can be determined in a similar manner that the best scheme $tff_3()$ is similar to $\frac{tf_t^D}{tf_t^D+0.45}$. This is the term-frequency factor of the $BM25$ scheme with a low value for k_1. Thus, a further term-frequency factor ($tff_8()$) is introduced. In the section that follows (section 6.2.2) it is shown from an evaluation apect that the new scheme is empirically similar to the best evolved scheme on the test data:

$$tff_8() = \frac{tf_t^D}{tf_t^D + 0.45} \tag{6.4}$$

This equation is simpler in form than $tff_3()$ and will be used in a later section. This scheme is similar to the term-frequency factor in the $BM25$ scheme when k_1 is set to 0.45. However, this value for k_1 falls below the recommended range of values originally proposed ($1.0 < k_1 < 2.0$) (Hancock-Beaulieu et al, 1996).

Importantly, this scheme depends on the global scheme with which it was evolved. It has previously been reported, when using a similar evolved global term-weighting, that the okapi term-frequency component can be successfully incorperated when k_1 is assigned a value of 0.2 (Cummins and O'Riordan, 2005). Although this is somewhat unexpected, the reason for this lower value could be because of the new density measure (cf/df) contained in the global weighting. This density measure can be thought of as the average term-frequency of a term in the documents which contain the term. This leads to an estimation of term-frequency in the global weighting component. Therefore, it follows from such an argument that the term-frequency influence factor may need to be reduced in order to correctly combine with the new global weighting.

6.2.2 Results and Discussion

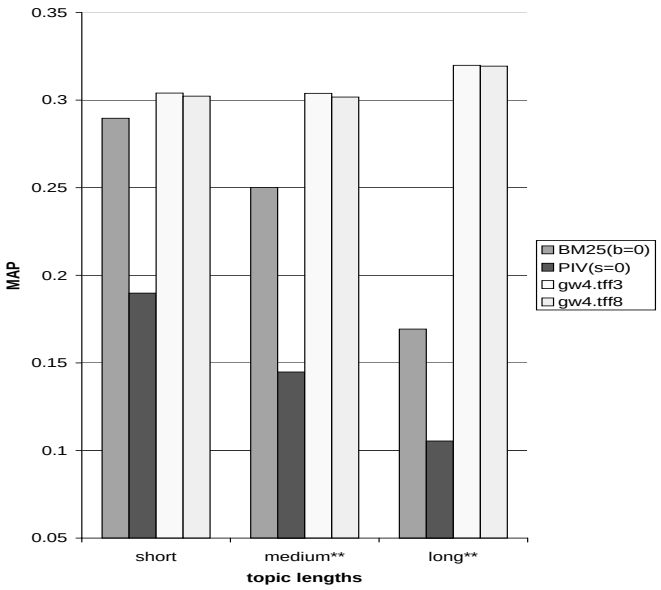

Figure 6.7: MAP of tff schemes on FR collection

Figures 6.7, 6.8, 6.9, 6.10 and 6.11 show the performance of the evolved scheme and the benchmarks on test data. Statistical significance is again denoted by two astericks (**) and is measured against the best benchmark

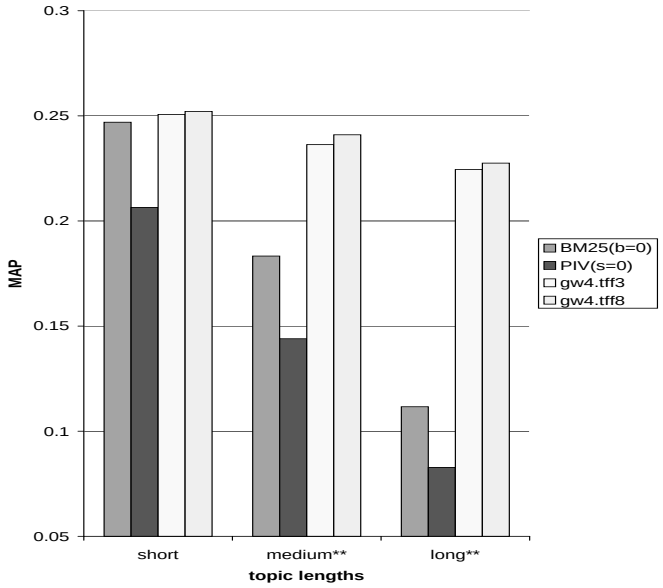

Figure 6.8: MAP of tff schemes on FBIS collection

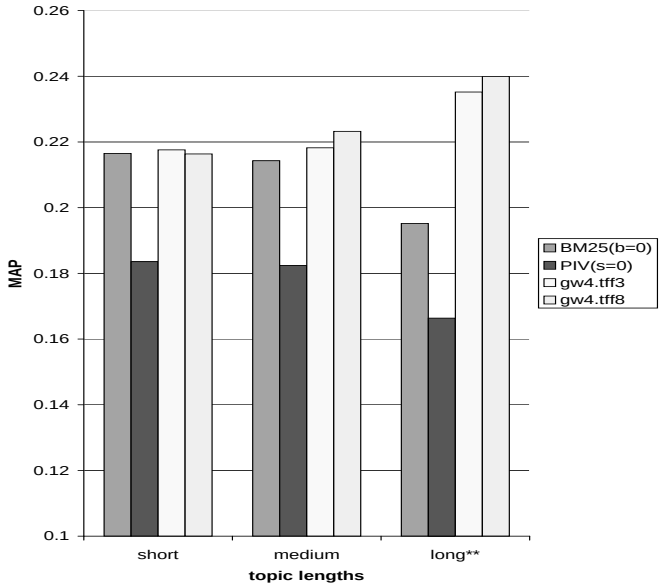

Figure 6.9: MAP of tff schemes on LATIMES collection

($BM25$ when $b = 0$). It can be seen that on short queries the performance of the evolved scheme is similar to that of the $BM25$ scheme (ignoring normalisation). However, on medium and long queries the performance of the

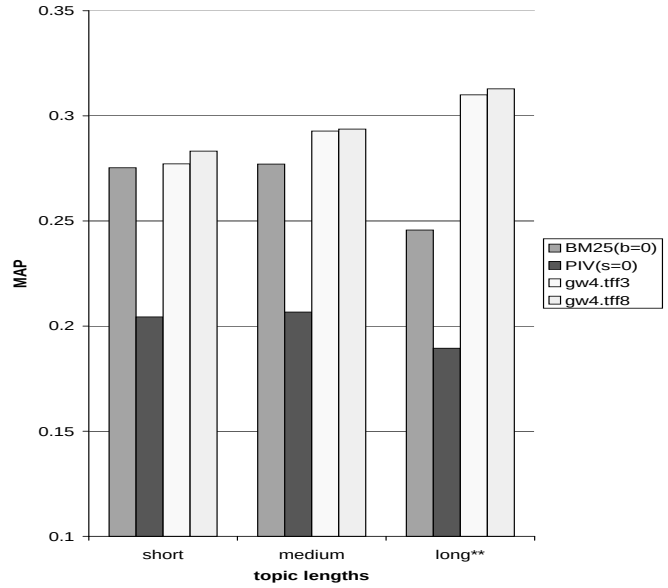

Figure 6.10: MAP of tff schemes on FT collection

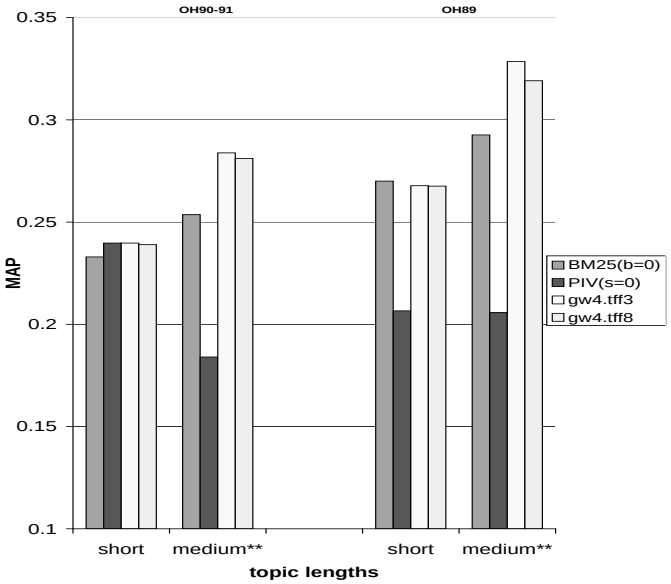

Figure 6.11: MAP of tff schemes on OHSUMED collections

evolved scheme is significantly better than $BM25$ (ignoring normalisation). Another point to note is that the performance of the evolved scheme increases as the query length increases for all but the FBIS collection. It can be seen

that the performance of the $BM25$ and pivoted document length normalisation schemes tends to decrease as noisy terms are added to the topic which mirrors the results seen in the previous section.

Interestingly, the pivoted document length normalisation scheme (ignoring normalisation) performs poorly for all query types on various collections. This would suggest that the term-frequency influence factor in the pivoted document length normalisation scheme is the reason for its poor performance (as its *idf* factor is comparable to the *idf* in the $BM25$ scheme). From the previous analysis, it could be seen that the term-frequency in the pivoted document length normalisation scheme places a higher influence on term-frequency than all other schemes. This would seem to be too large of an influence for the term-weighting component.

It should also be noticed from empirical evaluations that the $tff_8()$ scheme is similar to the evolved scheme. This would tend to validate the analysis in the previous section. Again, many feature extraction techniques use basic $tf{\cdot}idf$ weighting techniques while ignoring normalisation. The schemes develop here are likely to be useful in such circumstances. Furthermore, the empirical evidence has also validated the brief relative term-frequency analysis and has produced a simple term-frequency factor which is similar to the evolved schemes.

6.3 Summary

In this section, term-discrimination (global) schemes have been evolved that outperform traditional *idf* type schemes in an ad hoc retrieval setting. A number of these schemes are presented and are validated on unseen data. These new global schemes contain different characteristics to those of standard *idf* functions. They correctly weight noisy terms in longer more verbose queries which indicates that they are a truer measure of the information contained in a term.

Term-frequency schemes are evolved which are dependent on the best general term-discrimination measure. A brief analysis has shown that these term-frequency schemes apply a lesser weight to the term-frequency aspect

than those of the benchmark schemes. The analysis is used to introduce a simple term-frequency function which has similar characteristics to those of the evolved functions. These schemes (which ignore normalisation) are shown to have an increased performance on medium and long queries. The performance on short queries is comparable to that of the benchmark schemes used thus far. Some reasons for the lack of increase in performance on short queries are also presented.

Chapter 7

Evolving Normalisation Schemes

In this chapter, the third and final component (i.e. normalisation) of the term-weighting framework is completed. As normalisation is known to be a difficult problem, an analysis of the standard $BM25$ normalisation is conducted. This standard $BM25$ normalisation scheme is incorporated into the best evolved term-weighting scheme thus far, in order to better understand the factors that influence normalisation. Section 7.1 presents some detailed analysis that studies some of the factors that affect the tuning parameters currently incorperated into normalisation schemes. This analysis aids in the design of training data that provides a useful general environment for learning normalisation schemes. The results from the experiments that evolve the normalisation schemes are presented in section 7.2. A summary of the main contributions concludes the chapter.

7.1 Normalisation Analysis

Document normalisation is known to be a difficult problem in IR, as tuning is often needed to overcome the collection dependence problem known to affect many normalisation schemes (Chowdhury et al, 2002; He and Ounis, 2003). Therefore, some preliminary experiments are conducted before evolving nor-

malisation functions. This section introduces some preliminary experiments using the $BM25$ scheme and the newer evolved function ($S(Q, D)$ equation (7.1)) that confirm and extend the findings of previous research.

$$S(Q, D) = \sum_{t \in Q \cap D} (ntf_3() \cdot gw_4 \cdot tf_t^Q) \tag{7.1}$$

The $tff_3()$ scheme can be normalised similarly to the $BM25$ scheme which normalises the actual term-frequency. Thus, tf_t^D can be substituted with $tf_t^D/n()$ in the term-frequency factor $tff_3()$, such that $ntf_3()$ is as follows:

$$ntf_3() = 10 + \frac{log(0.5)}{\frac{tf_t^D}{n()}} + \frac{log(\frac{tf_t^D}{n()})}{log(1 + \frac{tf_t^D}{n()})} + \frac{log(\frac{tf_t^D}{n()})}{log(1 + \frac{tf_t^D}{n()}) \cdot log(log(10))} \tag{7.2}$$

where $n()$ is some normalisation function. The normalisation function used with success in the $BM25$ and pivoted document length normalisation function is as follows:

$$n(b) = (1 - b) + b \cdot \frac{dl}{dl_{avg}} \tag{7.3}$$

where b is the level of normalisation to apply such that $0 \leq b \leq 1$. For these preliminary experiments, this normalisation function is used in the evolved function as outlined in equation (7.2). The $BM25$ and $S(Q, D)$ (equation (7.1)) schemes are tested with nine normalisation (i.e. b) values (0, 0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875 and 1) for short, medium and long queries on the test collections.

Figures 7.1 and 7.2 show a typical trend across the collections tested. For the collections shown (LATIMES and FT), it can be seen that for both schemes a lower value of b (low penalisation) tends to lead to a better performance for short queries. The performance tails off for higher values of b for these queries. A midrange value of b results in a higher MAP for medium length queries, while a higher value of b (a higher penalisation) is advantageous when dealing with long queries. This is especially true for the $BM25$ scheme, whose performance is very poor for low values of b on long queries. It

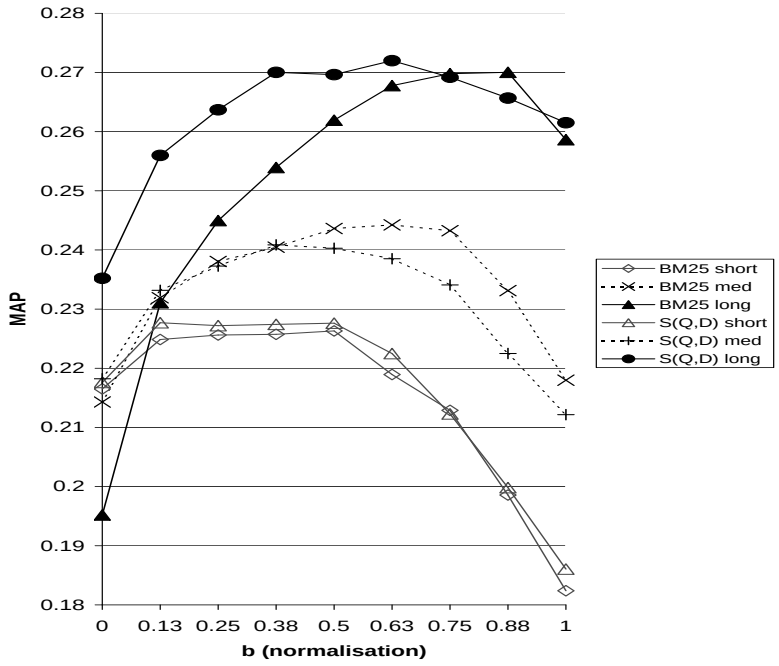

Figure 7.1: $BM25$ and $S(Q, D)$ for varying b on LATIMES

is also worth noting that the $S(Q, D)$ scheme compares quite well to $BM25$ and outperforms it in some cases (especially for long queries for most values of b).

Table 7.1 shows the optimal value of b for each collection for short, medium and long queries for the $BM25$ and $S(Q, D)$ schemes. It can be seen that in most cases the optimal level of normalisation increases as the query length increases (although these are quite coarse intervals of b). The results for the $S(Q, D)$ scheme suggest a similar trend. This phenomenon has been previously reported (He and Ounis, 2003, 2005b; Chung et al, 2006) but neither a further analysis nor a reason for this has been presented.

An analysis of the characteristics of the returned set of documents for each set of queries (short, medium and long) on four of the collections is now presented. The entire set of returned documents (i.e. documents which

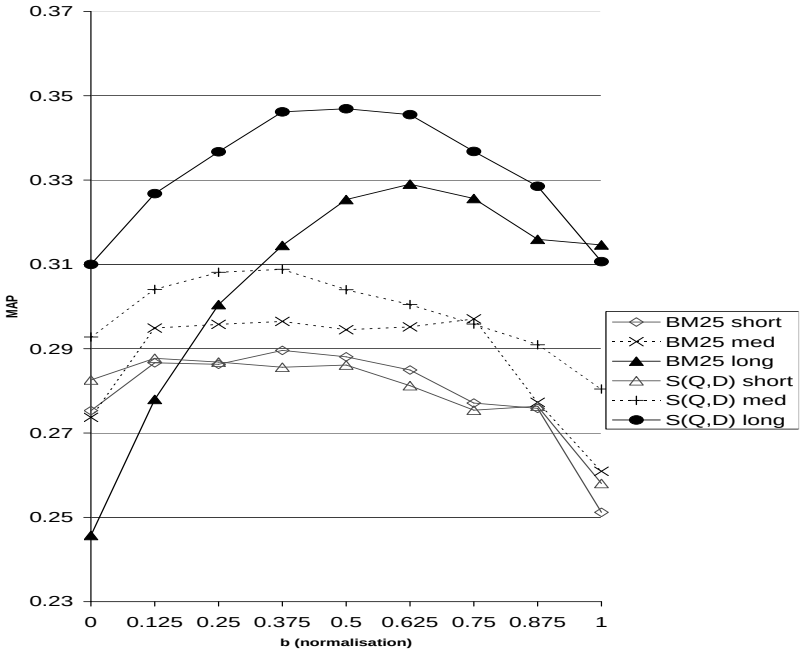

Figure 7.2: $BM25$ and $S(Q, D)$ for varying b on FT

contain at least one query term) for each query on each set of topics is analysed. Figure 7.3 shows that short queries return longer documents on average, while medium and long queries return documents that are closer to the average length document in the entire collection. Furthermore, Figure 7.4 shows that the standard deviation of document length for the returned set of documents is also greater for the shorter queries, indicating that there is a greater variation in the lengths of the documents returned. When the queries are longer, the sets of returned documents become such large samples of the document collections, that the average document length and standard deviation are very similar to those of the documents in the entire collection. The fact that the deviation in document length of the returned documents is higher for shorter queries is an important factor in the normalisation to be applied and is investigated next. The main contributions from this section are that the query length influences the value of b in the $BM25$ normalisation (equation (7.3)) and the fact that queries of different lengths return sets of documents with very different characteristics (which may influence the

Table 7.1: Optimal b per collection for schemes

$BM25$				
Collections	Topics	short	medium	long
LATIMES	301-450	0.125	0.625	0.825
FBIS	301-450	0.125	0.25	0.75
FT	301-450	0.375	0.375	0.625
FR	301-450	0.75	0.625	0.625
OH90-91	1-63	0.625	0.75	-
OH89	1-63	0.375	0.5	-
$S(Q, D)$				
Collections	Topics	short	medium	long
LATIMES	301-450	0.25	0.375	0.75
FBIS	301-450	0.125	0.25	0.25
FT	301-450	0.25	0.375	0.5
FR	301-450	0.25	0.375	0.125
OH90-91	1-63	0.625	0.625	-
OH89	1-63	0.25	0.375	-

normalisation to apply). From a purely theoretical perspective this indicates that this normalisation parameter (b in equation (7.3)) is not as free as was initially modelled. This may not be of great concern in practice as many parameter tuning techniques do currently exist (He and Ounis, 2003). However, in terms of a complete model of term-weighting (which incorporates all query types and collections), it is an important point to note.

7.1.1 Standard Deviation of Returned Documents

This section aims to investigate what effect the changing of the distribution of document lengths in the collection has on the optimal value of b in the $BM25$ normalisation scheme (equation (7.3)). This normalisation function is comprised of a ratio of a specific document to the average document length ($\frac{dl}{dl_{avg}}$). This ratio is then used in a linear function to penalise the document accordingly. However, this ratio tells us nothing about the distribution of

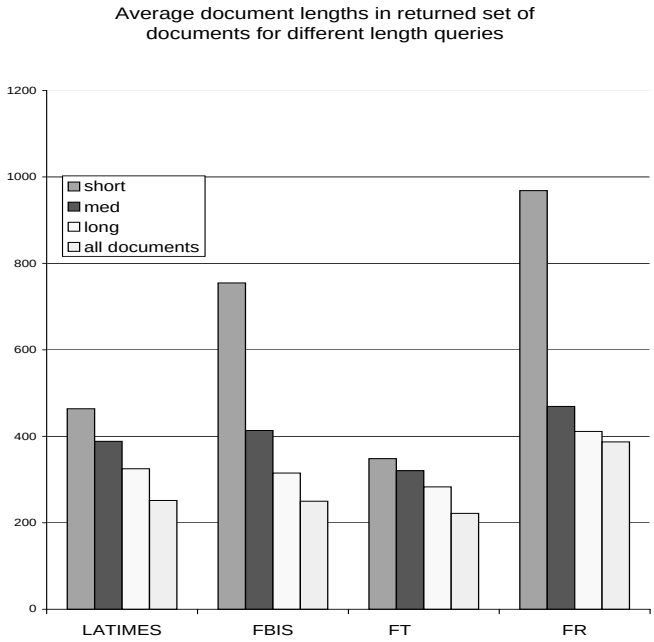

Figure 7.3: Average length of returned documents

documents in the collection or the expected deviation of a document from the average document length. For collections with a high deviation of document length, the ratio ($\frac{dl}{dl_{avg}}$) will vary considerably from very low values (for the shorter documents) to very high values (for the longer documents). Furthermore, from the previous experiment, it was determined that short queries (which return documents sets with a high standard deviation) require a smaller value of b in equation (7.3) for optimal performance. Ultimately, this suggests that the optimal value of b may be inversely related to the deviation of documents in the collection.

To test this hypothesis, a small sample of documents from the LATIMES collection is used with the medium length topics (301-350). A small collection of 8,598 LATIMES documents with an average document length of 28 and a standard deviation of 10 (i.e. low deviation given the average document length) is created. The MAP (using the $S(Q, D)$ scheme) for various values of b on this collection is measured. The characteristics of the collection is then modified by adding 526 extremely long documents. This dramatically

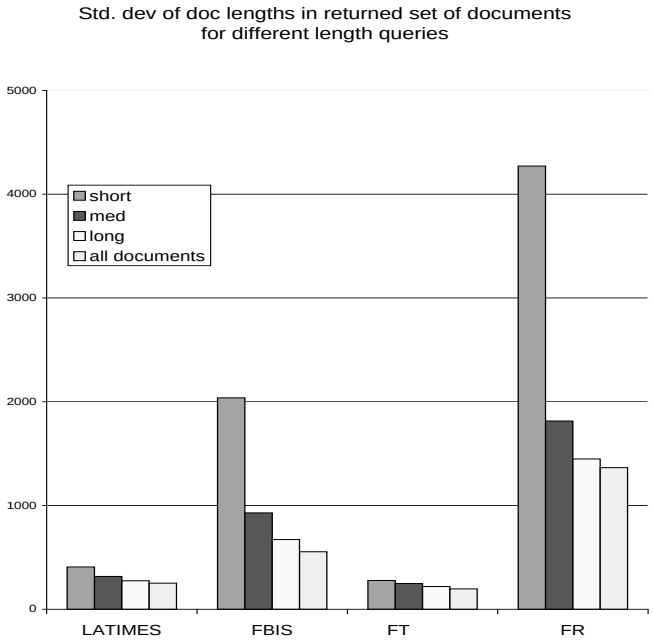

Figure 7.4: Standard deviation of returned documents

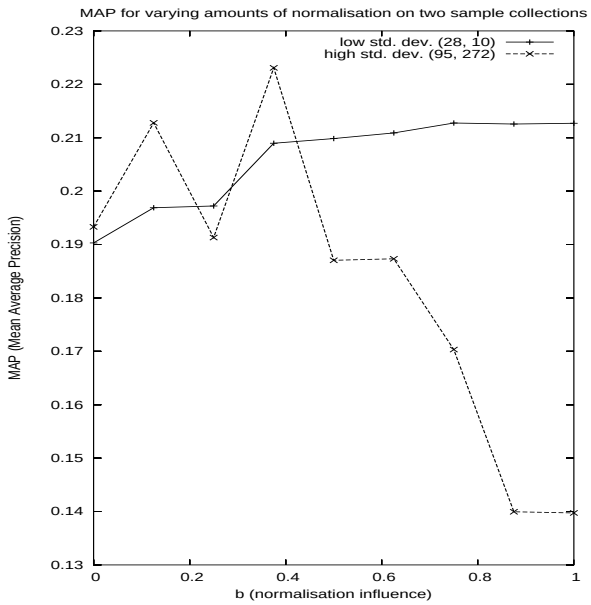

Figure 7.5: Δ of deviation with the $S(Q, D)$ scheme

changes the properties of the collection by adding a few documents. The collection now consists of 9,124 documents with an average document length of 95 and a standard deviation of 272 (i.e. high deviation given the average document length). The MAP (again using the $S(Q, D)$ scheme) for the same values of b is recalculated. As with all the experiments only queries with relevant documents in the sample collection are used. From Figure 7.5, it can be seen that when the standard deviation is low (little variation in document length) normalisation is less important (an expected result in a ranking situation) but still has a benefit at higher levels of b. When the standard deviation increases, the optimal level of b drops sharply as using a high value of b severely penalises the longer documents (some of which are relevant to at least one of the queries). Usually, a high standard deviation indicates a number of very long documents, due to a lower bound (of zero) on the distribution of document lengths.

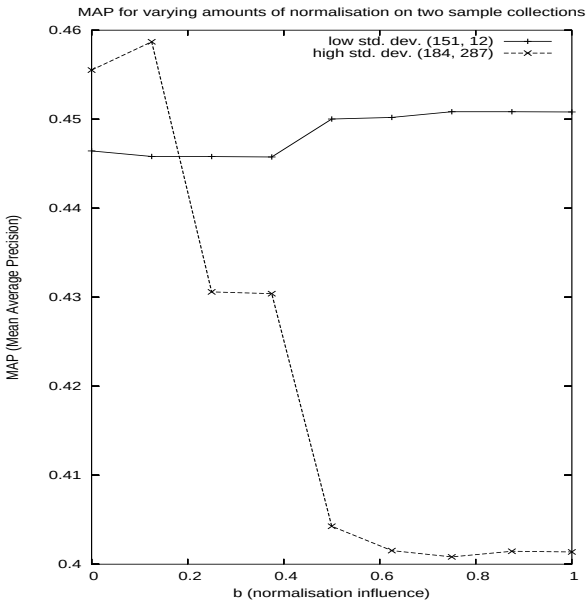

Figure 7.6: Δ of deviation with the $S(Q, D)$ scheme

Figure 7.6 shows the results from a similar experiment using a different 3,710 LATIMES documents with an average length of 151 and standard deviation of 12. A further 48 longer documents are added changing the average

documents length to 184 and the standard deviation to 287. The keys in both figures show the average document length and standard deviation respectively in parenthesis for each collection. It can seen that this phenomenon is the same (albeit contrived and exaggerated in these cases) as that which was observed when using different length queries. Previously, it was shown that shorter queries return a document set with a higher standard deviation. For these short queries, a lower level of b is beneficial. Hence, these two phenomena are actually related and ultimately, it is the deviation of documents lengths that influences the normalisation parameter b in equation (7.3). The short queries create a different distribution of document length in the returned set of documents. This analysis can now aid us in devising an approach to learn normalisation schemes in a suitable training environment.

7.2 Normalisation Schemes

In this section, normalisation schemes ($n()$) are evolved in the framework outlined in the previous section. A normalistion approach similar to the $BM25$ scheme, which normalises the actual term-frequency measure, is assumed. Thus, the aim of this experiment is to evolve normalisation schemes ($n()$) in the scheme described in equations (7.1) and (7.2).

7.2.1 Training

Table 7.2: Training data for Normalisation

Name	Collection	#docs	dl_{avg}	$\sigma(dl)$	Topics	# Topics	Length
NORM1	LATIMES	10,822	168.5	103.9	301-350	24	med
NORM2	LATIMES	12,017	244.7	250.6	301-350	29	med
NORM3	LATIMES	11,329	235.5	367.9	301-350	28	med

The development of training data is crucial to the potential generalisability of the normalisation schemes produced from the GP approach. It is beneficial to have a number of different environments (collections) on which to learn normalisation schemes. Three collections are constructed using an

approach similar to that described in the previous section (7.1.1). A collection of 10,822 documents (NORM1) is created from the LATIMES collections and the queries which have relevant documents associated to them in those collections are used. 1,195 longer documents (between 820 and 1050 words) are added to change the properties of the collection. This is the second collection (NORM2) and consists of 12,017 documents. The third collection (NORM3) consists of the first 10,822 documents and another 507 even longer documents (documents longer than 1050 words). This creates a collection of 11,329 documents again with different characteristics. Table 7.2 shows the characteristics of the three collections. These documents have been chosen based on their length features so that there is one collection with a low standard deviation compared to the average document length, another with a standard deviation close to the average document length and a further collection in which the standard deviation is higher than the average document length. The GP is run seven times and the MAP for the three collections is aggregated and used as the fitness function. Thus, the GP is trying to optimise the MAP for the three collections.

Table 7.3: % MAP for $n()$ weightings on training collection

Collection	$BM25$	PIV	$n_1()$	$n_2()$	$n_3()$	$n_4()$	$n_5()$	$n_6()$	$n_7()$
NORM1,2,3	45.25	42.60	48.78	48.79	49.03	47.07	48.79	48.85	48.79

The results from the training data look promising as all of the evolved schemes outperform the benchmarks. However, it is unknown if the solutions evolved are useful on general data or have overfitted to this training data.

Simplification and Validation of Schemes

The seven actual forumlas evolved by the system and their simplifications are as follows:

$$
\begin{aligned}
n_1() &= \sqrt{(\sqrt{1})/((dl_{avg}/1)/(0.5+dl))} \\
&= \sqrt{\tfrac{dl+0.5}{dl_{avg}}} \\
n_2() &= \sqrt{\tfrac{dl}{dl_{avg}}} \\
n_3() &= \sqrt{(dl+((-1)/(dl_{avg}/(dl+\sqrt{1}))))/(dl_{avg}/(\sqrt{(dl+\sqrt{1})/(dl_{avg}\cdot exp(1))}))} \\
&= \sqrt{(dl-\tfrac{dl+1}{dl_{avg}})/(dl_{avg}/(\sqrt{(dl+1)/(dl_{avg}\cdot exp(1))}))} \\
n_4() &= (1\cdot dl)/(dl_{avg}+\sigma(dl)) \\
&= \tfrac{dl}{dl_{avg}+\sigma(dl)} \\
n_5() &= \sqrt{((-dl)/(-dl_{avg}))} \\
&= \sqrt{\tfrac{dl}{dl_{avg}}} \\
n_6() &= (\tfrac{dl}{dl_{avg}})/((\sqrt{dl})/((log(10))+dl_{avg}))/\sqrt{(((\sqrt{dl})/(\tfrac{dl}{dl_{avg}}))/(\tfrac{dl}{dl_{avg}}))+dl_{avg}} \\
&= \tfrac{(dl/dl_{avg})/(\sqrt{dl}/(log(10)+dl_{avg}))}{\sqrt{(((\sqrt{dl}/(dl/dl_{avg}))/(dl/dl_{avg}))+dl_{avg})}} \\
n_7() &= \sqrt{\tfrac{dl}{dl_{avg}}}
\end{aligned}
$$

Table 7.4: % MAP for $n()$ weightings on validation collection

Collection	$BM25$	PIV	$n_1()$	$n_2()$	$n_3()$	$n_4()$	$n_5()$	$n_6()$	$n_7()$
FT (301-350)	28.25	29.57	29.66	29.67	29.65	29.61	29.67	29.01	29.67

The validation data shows that most, if not all, of the normalisation schemes are comparable in terms of MAP. The lowest MAP achieved on the validation data is by $n_6()$. Many of the formula are also very similar in structure. $n_2()$, $n_5()$ and $n_7()$ are equivalent, while $n_1()$ is structurally very similar. $n_2()$ is chosen as the best normalisation function and is used in the following results section. The benchmark schemes on this validation data are also comparable in terms of MAP. It seems that normalisation (at least for the validation data) aids the benchmark schemes considerably.

7.2.2 Results and Discussion

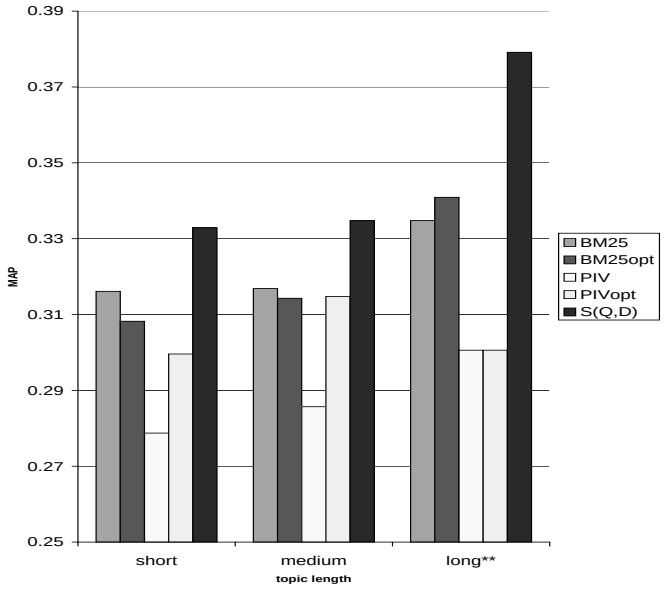

Figure 7.7: MAP of full schemes on FR collection

Figures 7.7, 7.8, 7.9, 7.10 and 7.11 show the performance of the full evolved scheme against the benchmarks. The full evolved scheme is labelled $S(Q, D)$ in these results for simplicity. A tuned version of $BM25$ and the pivoted document length normalisation scheme are also shown. The tuned schemes are tuned for query length only. Therefore, the default $BM25$ may sometimes outperform the tuned $BM25$ (i.e. $BM25_{opt}$) on a specific collection but over all collections for a specific query length the tuned $BM25$

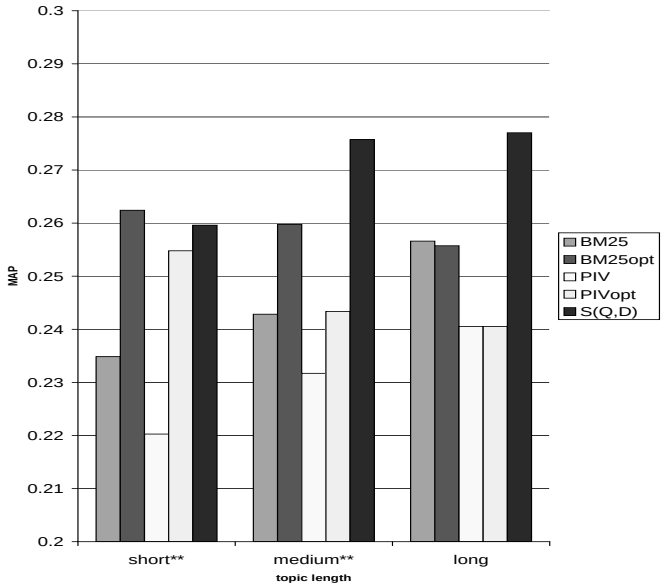

Figure 7.8: MAP of full schemes on FBIS collection

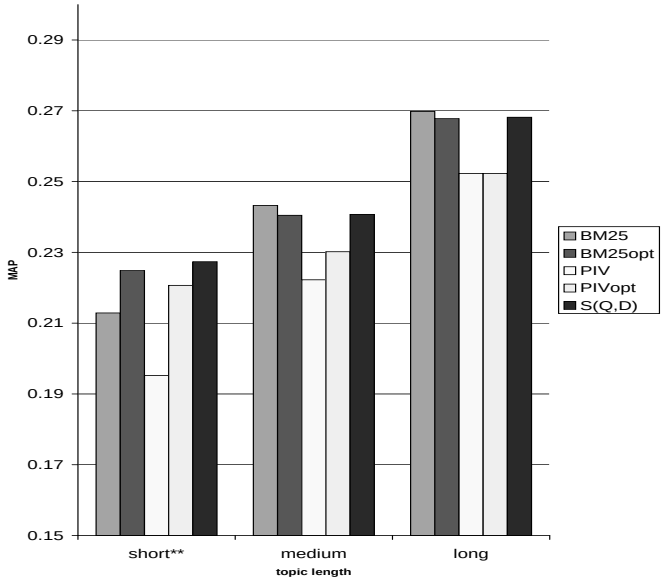

Figure 7.9: MAP of full schemes on LATIMES collection

is optimal. Firstly, the new scheme developed is comparable to, and often significantly outperforms (denoted by (**)) the default $BM25$ on much of the test data. The evolved scheme is non-parametric and therefore should be

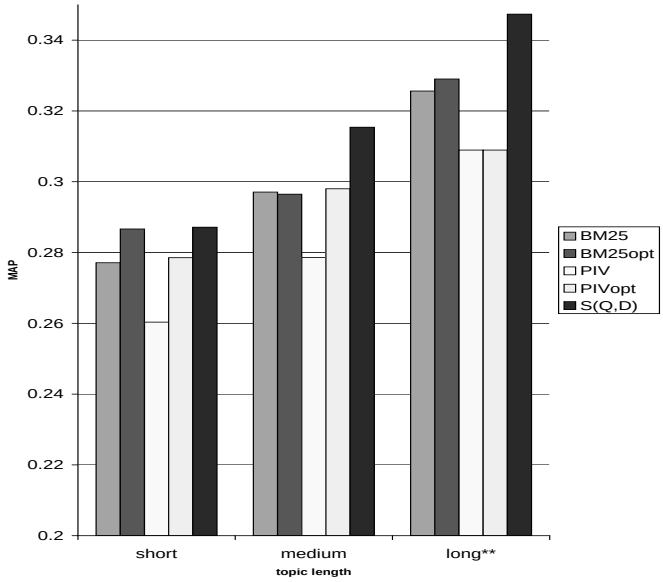

Figure 7.10: MAP of full schemes on FT collection

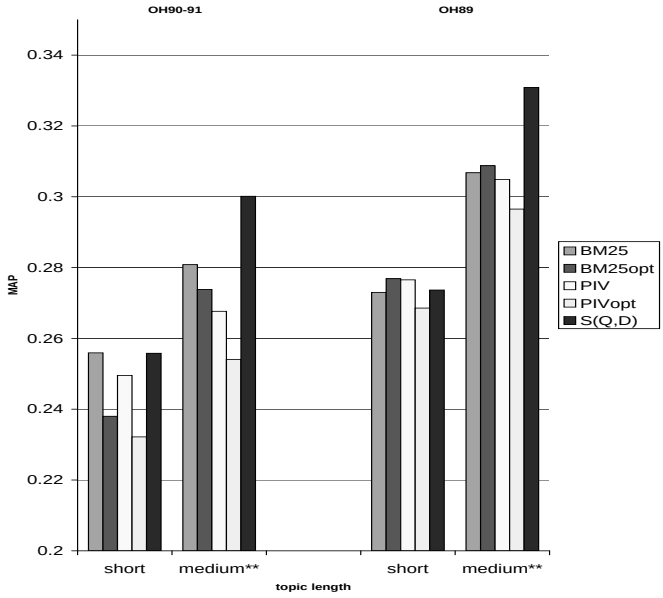

Figure 7.11: MAP of full schemes on OHSUMED collections

compared against one $BM25$ benchmark (i.e. fixed values for its tuning parameters). The significance tests are conducted between one $BM25$ scheme (the default) and the evolved scheme. The newly evolved scheme consis-

tently outperforms the pivoted document length normalisation scheme on all unseen test data. Normalisation increases the performance of both benchmark schemes considerably, while moderately increasing the performance of the evolved scheme. Appendix B shows the precision-recall curves for the complete evolved scheme ($S(Q, D)$) and the default $BM25$ scheme on all the unseen test data.

It was previously hypothesised that the deviation in document length may affect normalisation. However, only one learned scheme uses the standard deviation factor explicitly. Interestingly, six of the seven schemes evolved are sublinear with respect to the document length. This means that as the document grows in length the level of penalisation decreases. The one scheme which is linear with respect to the document length uses the standard deviation in a similar manner to that which had been hypothesised. The sublinearity of these normalisation schemes will be discussed and theoretically motivated in detail in a later chapter.

7.2.3 Validation of Preliminary Analysis

While it has been shown that the evolved scheme often outperforms the standard benchmarks, the validity of the learning approach for normalisation has yet to be shown to be beneficial. In this section, it is shown that many of the normalisation schemes (namely $n_2()$) perform close to the optimal value of b in the standard normalisation scheme (equation (7.3)). An interesting, if not crucial, evaluation is to compare the evolved normalisation scheme used here ($n_2()$) against the standard $BM25$ linear normalisation over all values of b as was used in the previous section. For such an evaluation, the only factor that should vary is the normalisation scheme and therefore, the evolved scheme (equations (7.1) and (7.2)) (for which the $n_2()$ normalisation function was evolved) is used for the following evaluation (as in the preliminary analysis).

Figure 7.12 shows the nonparametric $n_2()$ scheme and the $n(b)$ scheme for nine values of b on a selection of test data using the evolved $S(Q, D)$ scheme outlined at the beginning of the chapter. This example shows that the evolved normalisation is close to the optimal performance of the standard $BM25$

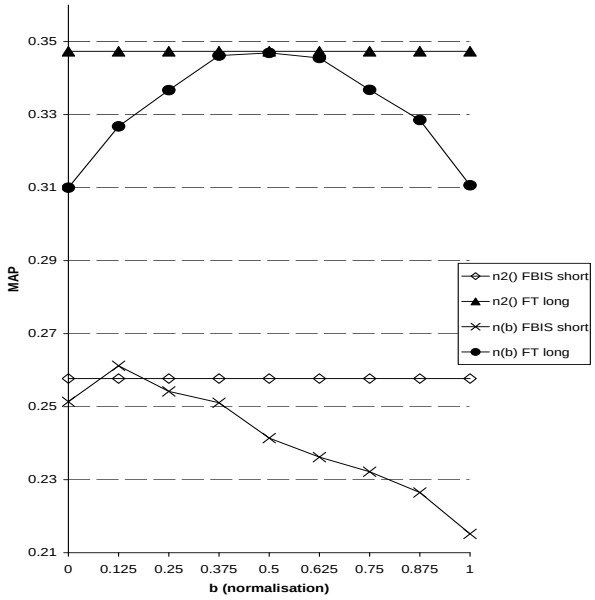

Figure 7.12: MAP of $n(b)$ vs $n_2()$ schemes on FT

linear normalisation for short queries on the FBIS collection (i.e. $b = 0.125$). It also shows that the evolved normalisation scheme slighly outperforms the most optimal setting of the standard normalisation for long queries on the FT collection (i.e. $b = 0.5$). This is encouraging as the learned scheme is nonparametric.

Comparison Metric

For normalisation schemes that contain a tuning parameter, it is useful to know how difficult it is to tune the parameter to achieve a high level of performance. A comparison metric is now presented which compares the nonparametric normalisation to the parametric normalisation. It is useful to know if the optimal setting of the tuning parameter resides in very small range of values within the possible range of values for the parameter (sensitivity) and how much of an increase in MAP the optimal normalisation setting

provides, when compared to other settings for the parameter.

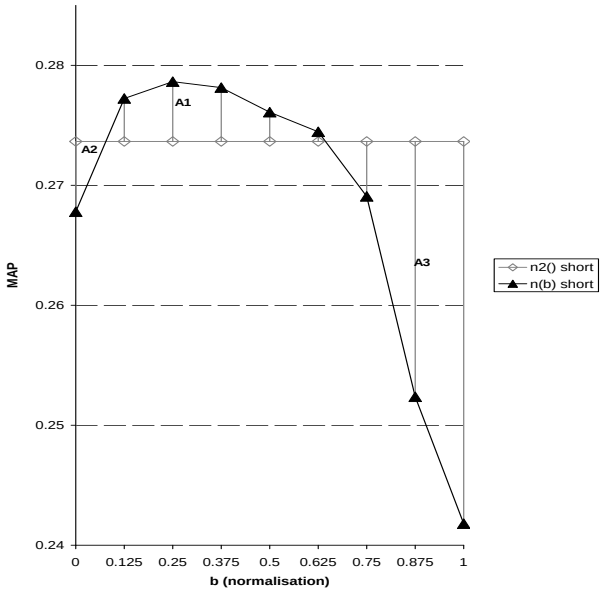

Figure 7.13: MAP of $n(b)$ vs $n_2()$ for short topics on OH89

Figure 7.13 shows the performance of the evolved normalisation scheme (a line) and the standard $BM25$ linear normalisation scheme (a curve). A useful comparison metric can be calculated by measuring the area which lies above the non-parametric normalisation line and below the curve ($A1$), by the area that lies between both ($A1 + A2 + A3$). If b is a completely free parameter, picking a random value of b for an unseen collection would lead to an average value of 50% for this metric. Thus, the comparision metric ($NC()$) can be formulated as follows:

$$NC() = 1 - \frac{A1}{A1 + A2 + A3} \tag{7.4}$$

where $A1$, $A2$ and $A3$ are the areas between the line indicating the performance of the nonparametric scheme ($n_2()$) and the curve indicating the performance of the parametric scheme ($n(b)$) as indicated in Figure 7.13.

Table 7.5: Comparison of $n_2()$ vs $n(b)$ for $S(Q, D)$

	short	medium	long
Collection	%	%	%
FR	99.54	100*	92.37
FBIS	97.01	100*	100*
LATIMES	99.98	99.96	84.76
FT	99.40	100*	99.93
OH89	75.98	78.37	-
OH90-91	94.82	100*	-

Table 7.5 shows this metric for all of the test collections. It can be seen that over many different collections, and especially over different query lengths, the non-parametric function performs very close to the optimal setting of b tuned for the specific collections and often surpasses it (denoted 100%*). Figure 7.13 actually shows the worst performance (75.98%) of the evolved scheme ($n_2()$) when compared with the standard normalisation ($n(b)$) using the new metric. It can be concluded that the evolved normalisation scheme chosen ($n_2()$) performs close to, and often surpasses, the performance of the most optimal setting of b in the standard linear normalisation (7.3) when using the same underlying function (i.e. gw_4 and tff_3). A further analysis of a full evolved term-weighting scheme is conducted in a later chapter.

7.3 Summary

In this chapter an analysis into factors that affect normalisation is presented in order to understand some of the important concepts surrounding normalisation. The characteristics of the returned sets of documents for queries of different lengths is analysed. It is found that it is the deviation of document lengths in the returned sets of documents that affects the level of normalisation to apply (b) when using the linear normalisation function in the $BM25$ scheme. Using this analysis, training data is devised that is used to learn general non-parametric normalisation schemes.

As a result, a full term-weighting scheme is completed and tested on

unseen test data. It is shown that the evolved scheme often significantly outperforms the best benchmark scheme on general test data. Furthermore, a method to compare the standard $BM25$ linear normalisation to the evolved normalisation is designed to validate the development of the training data in this experiment. It is shown that the evolved normalisation compares favourably to the $BM25$ normalisation over all values of its tuning parameter. Chapters six and seven have presented experimental results from an entire term-weighting scheme developed in a three stage incremental process. It has been shown that the final evolved formula provides a superior ranking to the $BM25$ scheme on much of the unseen data for various query lengths without the need for tuning.

Chapter 8

A Phenotypic Analysis of the Search Spaces

A number of questions regarding the evolved term-weighting schemes and the GP process remain unanswered. As GP is a stochastic process, it is interesting to explore the search process itself and the ability of the GP to effectively find areas in which good solutions are located. This chapter deals with determining the closeness of the solutions previously developed, in each of the three term-weighting function spaces (i.e. term-discrimination, term-frequency and normalisation). Firstly, a number of phenotypic distance measures are outlined. Then, using these phenotypic distance measures, trees are developed that show the phenotypic distance between all of the term-weighting schemes in each of the search spaces (function spaces). This framework can be used to develop a representation of where these evolved solutions are located in the solution space. The increase in fitness for each generation of the GP process for the best runs is also included. This chapter ends with some conclusions regarding the GP process and the resulting solutions.

8.1 Phenotype

The phenotype of an individual is often described as its behaviour. Fitness is determined by the phenotype, which in turn is determined by the genotype.

For most problems in an unchanging environment, identical genotypes will map to identical phenotypes which will have the same fitness. Thus, the phenotype of a solution in this retrieval environment can be determined by examining the ranked lists returned given a set of queries.

8.1.1 Phenotypic Distance Measures

Two measures are now presented (Cummins and O'Riordan, 2006b) and are subsequently used for exploring the space of term-weighting schemes. The distance measures presented measure only the parts of the ranked lists that affect the MAP (fitness) of a solution. This is important as the rank of relevant documents is the only direct contributing factor to the fitness of individuals in the GP process and as such is the only part of the ranking that guides the search process. The first metric measures the average difference between the ranks of relevant documents in two sets of ranked lists. This measure will tell if the same relevant documents are being retrieved at, or close to, the same ranks and will indicate if the weighting schemes are evolving toward solutions that produce similar phenotypes. Thus, the distance measure $dist(F_1, F_2)$, where F_1 and F_2 are two term-weighting schemes, is defined follows:

$$dist(F_1, F_2) = \frac{1}{|R|} \sum_{i \in R} \begin{cases} |lim - r_i(F_2)| & \text{if } r_i(F_1) > lim \\ |lim - r_i(F_1)| & \text{if } r_i(F_2) > lim \\ |r_i(F_1) - r_i(F_2)| & \text{otherwise} \end{cases}$$

where R is the set of relevant documents for all queries used and $r_i(F_1)$ is the rank position of relevant document i under weighting scheme F_1. The maximum rank position available from a list is denoted by lim and is usually 1000 (as this is the usually the maximum rank for official TREC runs). Thus, when comparing two schemes this measure will calculate the average difference in rank for a relevant document when ranked by two schemes F_1 and F_2. Although different parts of the phenotype will impact on the fitness to different degrees, they are an important factor in distinguishing the behaviour of the phenotype. The difference in the position of documents at

lower ranks can indicate a difference in the behaviour of two term-weighting schemes, although their performance (fitness) may be similar.

To measure the difference that a change in rank *could* make in terms of MAP, the *dist*() measure is modified so that the change in rank of a relevant document is weighted accordingly. This weighted distance measure, $w\_dist(F_1, F_2)$, is similar to a previous measure (Carterette and Allan, 2005) and is calculated as follows:

$$w\_dist(F_1, F_2) = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{|R_q|} \sum_{i \in R_q} \begin{cases} \left| \frac{1}{lim} - \frac{1}{r_i(F_2)} \right| & \text{if } r_i(F_1) > lim \\ \left| \frac{1}{lim} - \frac{1}{r_i(F_1)} \right| & \text{if } r_i(F_2) > lim \\ \left| \frac{1}{r_i(F_1)} - \frac{1}{r_i(F_2)} \right| & \text{otherwise} \end{cases}$$

where Q is the set of queries and R_q is the set of relevant documents for a query q. This measure considers how the change in rank of a relevant document may affect MAP. It is entirely possible that two ranked lists could be considerably different yet have a similar MAP. This distance measure is more important if one wishes to determine how the difference in the phenotype may affect fitness.

Both Spearman's rank correlation (ρ) and Kendall's tau are not suitable for measuring the parts of the phenotype that contribute exclusively to fitness in a training environment. For example, consider two ranked lists in which all the relevant documents for a query are positioned at the same ranks. If some non-relevant documents are positioned at different ranks, both of the aforementioned measures would indicate there is some difference in these solutions in the training environment. However, this would not be the case as they have actually placed the same relevant documents in the exact same positions and the GP process has actually identified the same solution. It is not known how likely this senario may be. Distance trees are included that use one of these measures of distance (i.e. Spearman's rank correlation) for completeness. As the ranked lists returned by schemes are all positively correlated using Spearman's rank correlation (ρ), the distance measure used is $1 - \rho$. This is because lists which are identical are given a value of 1 and lists which are not correlated are given a value of 0 for Spearman's rank

correlation. As there are many ranked lists produced by each solution, they are simply averaged over the set of queries giving a single average correlation metric. Spearman's rank correlation may be better able to identify how a term-weighting scheme is likely to perform in a general retrieval environment as it includes all documents, which would be relevant to at least some as yet unpresented topic. The distance measures presented earlier can show if certain relevant documents are being favoured by the GP process in its training environment.

8.1.2 Cluster Representation in Training Environment

Neighbour-joining is a bottom-up clustering method often used for the creation of phylogenetic trees. This method is adopted to produce trees that represent solutions that are from *different* runs of the GP. The algorithm requires knowledge of the distance between entities that are to be represented in the tree. A distance matrix is created for the set of entities using a distance measure and the tree can then be produced from the resulting data. This clustering technique is used to visualise the phenotypic distance between the best solutions produced by the GP. For example, if there are N entities (solutions), an $N \times N$ distance matrix can be created using one of the distance measures. Then, using this distance matrix, trees can be created using a suitable drawing package (Choi et al, 2000) which represents the data and can provide a visualisation of where the solutions are located in relation to each other in the environment in which the solutions were evolved. This model is well suited to the evolutionary paradigm and can be used to visualise the distance between the term-weighting solutions in the environments in which they were evolved.

8.2 Term-Weighting Solution Spaces

In this section the fitness of the best runs of the GP for each generation are shown in order to further understand the learning process. It is useful to show the increase in fitness as the GP progresses. Furthermore, as the phenotypic

distance between solutions in an environment can be represented visually, the seven evolved solutions for each of the three term-weighting spaces are placed in the framework together with the benchmarks used. The benchmarks and the seven evolved solutions for each function space are visually represented in the environments in which they evolved.

8.2.1 Evolution of Term-Discrimination schemes

The first function space analysed is the space of term-discrimination schemes. The best schemes evolved on the training collection were shown to outperform *idf* type solutions on unseen data. Figure 8.1 shows the performance of the best performing individual and average fitness of the population for each generation for two runs of the GP. The worst runs of the GP ($gw_5()$ and $gw_6()$) still produced a solution that was better than any randomly created individuals (i.e. those from generation 1). It can be seen that convergence tends to occur quite quickly in this problem domain and that learning tends to stop after about 10 generations for this problem.

Solution Space

The distances between the best scheme from each of the seven GP runs is now visually represented. Figures 8.2, 8.3 and 8.4 show the tree representation of the seven evolved solutions and the two benchmarks solutions using the three different distance measures. The fitness of each scheme on the training data is shown in parentheses. The better evolved solutions on the training data (gw_1 to gw_4) are clustered closer together. For example, $dist(w_1, gw_1)$ is 36.50 and indicates that a relevant document differs by an average of approximately 36 rank positions when ranked by these two schemes on the training data. $w\_dist(w_1, gw_1)$ is 0.0430 and relates to the possible difference in MAP. $1 - \rho$ is 0.5246 ($\rho = 0.4754$) and indicates a weak correlation between the ranked lists produced by the two schemes (w_1 and gw_1). These trees indicate that the solutions are evolving towards ranked lists (solutions) produced by gw_1 as there is an increase in performance around this area of the tree. Solutions from run 5 and 6 (i.e. gw_5 and gw_6) can be seen promote the same documents

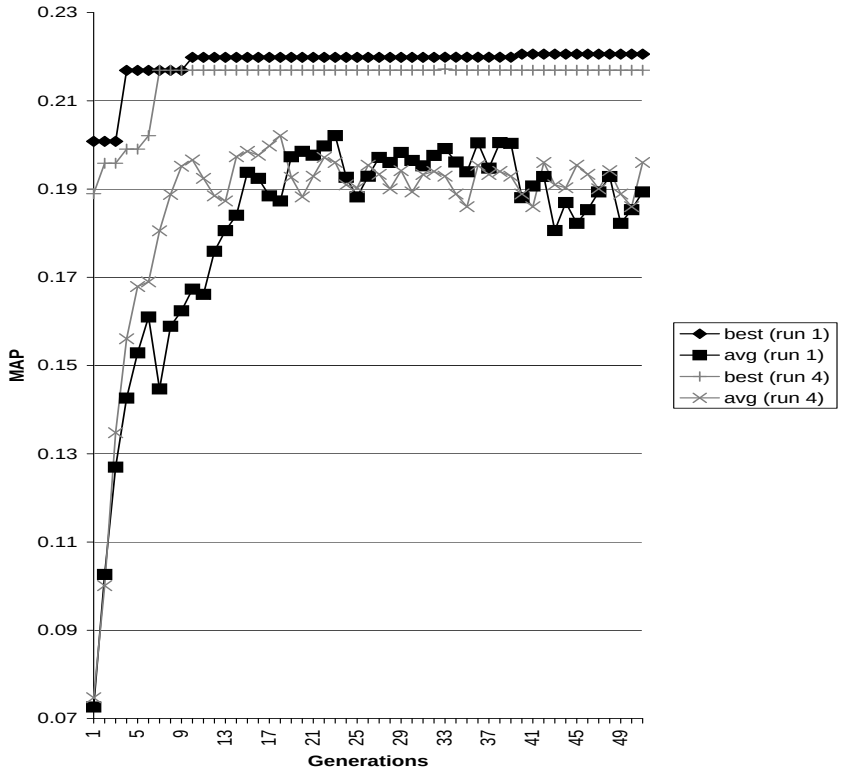

Figure 8.1: Best and average of population for two global weighting runs

as the standard *idf* type solutions. Phenotypically close solutions will have a similar fitness but it is not necessarily true that solutions with a similar fitness will have a similar phenotype (i.e. ranked list). It was previously determined that gw_4 was the better general solution. Although gw_1 and gw_2 performed quite well on the validation data slight overtraining may have occurred in these cases. It can also be seen that the trees produced from all three distance measures tend to create similar shaped trees, although the relative distances are slightly different.

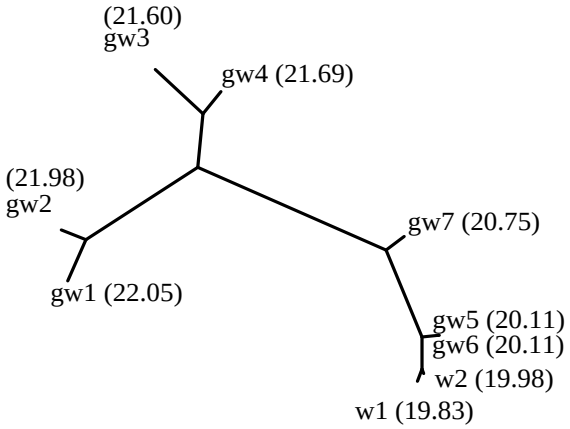

Figure 8.2: Visualisation of global space using $dist()$ measure

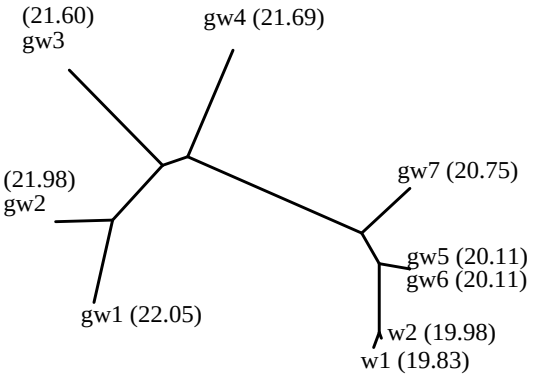

Figure 8.3: Visualisation of global space using $w\_dist()$ measure

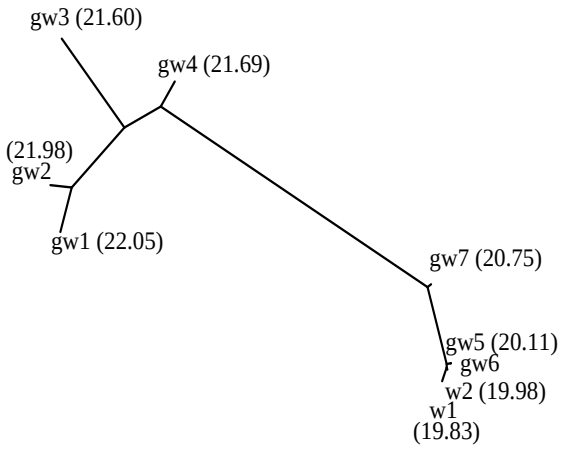

Figure 8.4: Visualisation of global space using $1 - \rho$ measure

8.2.2 Evolution of Term-Frequency schemes

The evolution of the solutions in the term-frequency function space is now examined. Figure 8.5 shows the best and average of the population from two runs of the GP. One of the runs for this problem ($tff_5()$) did not evolve anything useful as it failed to incorporate a term-frequency (tf_t^D) aspect. All other runs of the GP produced solutions that were better than any randomly created individual. Due to the relatively low term-frequency influence, which is beneficial when using the evolved global weighting, as was discussed in a previous chapter, it may be a somewhat difficult problem for GP. It can be seen that learning tends stop after the 15^{th} generation during the best GP run ($tff_3()$ or run 3) for this problem.

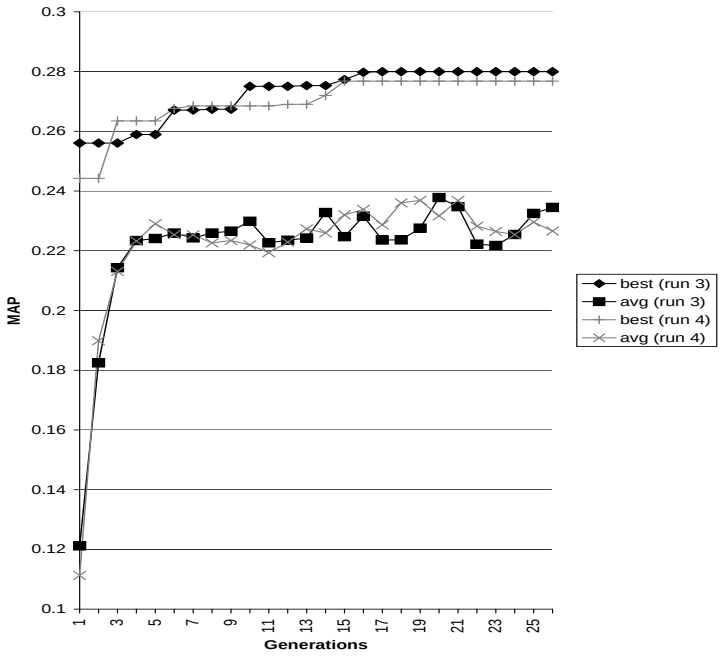

Figure 8.5: Best and average of population for two term-frequency runs

Solution Space

It was previously shown that the best evolved term-frequency factors outperform the benchmark schemes on unseen test data. It was also analytically shown that the better term-frequency schemes contained a similar relative term-frequency aspect. It can be hypothesised that they will be closely clustered in the solution space. Figures 8.6, 8.7 and 8.8 show the tree representations for the seven evolved solutions and the two benchmarks. The fitness of each scheme on the training data is again shown in parentheses. The better evolved schemes on the training data are again clustered very close together. Most runs of the GP evolve solutions which promote the same relevant documents. The entire ranked lists are also very similar, indicating that relevant and non-relevant documents are being retrieved near the same

ranks for the different schemes (indicated by the $1 - \rho$ measure). One of the schemes ($tff_7()$) promotes some different relevant documents but has a comparable performance. This would suggest that there may indeed be some different types of term-frequency schemes (some that promote different relevant documents) that are comparable in terms of performance. It would seem however that in this solution space the schemes which are located close to $tff_3()$ are useful and are at least easier to find (i.e. their genotypes may be simpler and thus more easily found). A binary solution is also shown in the trees (indicated by $tff_5()$ as it had no term-frequency aspect present). This scheme simply represents the evolved global term-weighting scheme (gw_4) in this space.

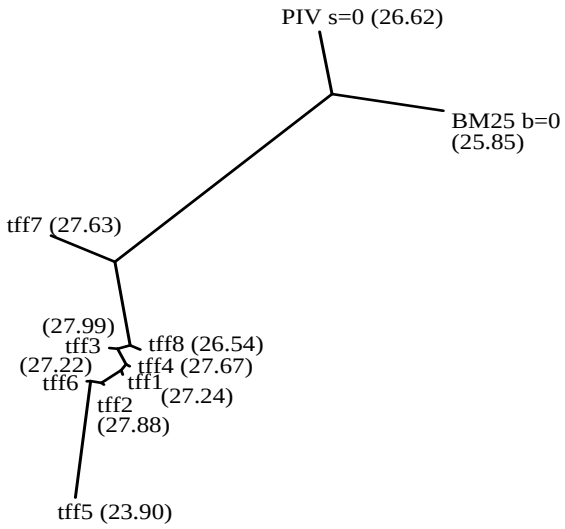

Figure 8.6: Visualisation of term-frequency space using $dist()$ measure

As an example of the distances in these trees, $dist(BM25_{b=0}, tff_3)$ is 94.43 indicating that a relevant document moves about 94 ranks on average when ranked by these two schemes. $w\_dist(BM25_{b=0}, tff_3)$ is 0.0911 which is related to the potential difference in MAP between the two schemes. $1 - \rho$ is 0.5999 ($\rho = 0.4001$) for these two schemes indicating again that there is a weak correlation between the ranked lists produced by the solutions.

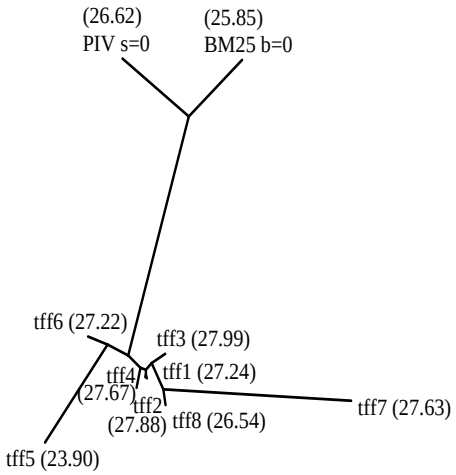

Figure 8.7: Visualisation of term-frequency space using $w\_dist()$ measure

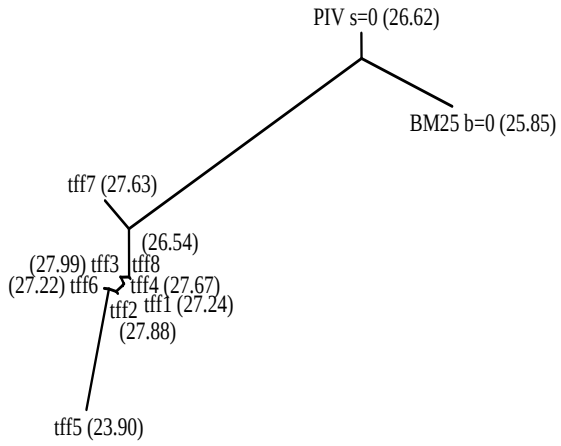

Figure 8.8: Visualisation of term-frequency space using $1 - \rho$ measure

It is worth noting that the term-frequency factor ($tff_8()$) which was tuned manually to behave similarly to the better evolved term-weighting schemes (e.g. $tff_3()$) returns similar relevant documents and has a strong correlation with these types of schemes.

8.2.3 Evolution of Normalisation Schemes

The better GP runs for the normalisation problem are now shown. Figure 8.9 shows the best and average of three runs of the GP for the normalisation problem. Convergence seems to occur quite early for the better runs. However, the best individual in the first generation in some runs has a considerably lower fitness (run 4 or $n_4()$ is one such example). In the runs that have quite a good fitness in the first generation, the $\frac{dl}{dl_{avg}}$ factor, which seems to be very useful for normalisation, has been found quite early. This can also be seen to occur in some form in *all* evolved solutions (shown in the previous chapter). For run 4 (shown), the $\frac{dl}{dl_{avg}}$ function was found after 22 generations and was modified to its final form ($\frac{dl}{dl_{avg}+\sigma(dl)}$) in the last 3 generations.

Solution Space

Figures 8.10, 8.11 and 8.12 show the tree representation of the seven evolved solutions, the two benchmarks and a constant weighting where $n() = 1$ (i.e. no normalisation). The fitness of each scheme on the training is again shown in parentheses. The better evolved schemes on the training data are again clustered very close together. Three runs of the GP evolved to the same function. Another run ($n_1()$) evolved a function that is very similar to these others. The difference between $n_1()$ and $n_2()$ is negligible in terms of ranking. $n_6()$ is a complex function but has a similar behaviour to many of the other schemes. The best scheme on the training data ($n_3()$) seems to have slightly overfitted as $n_2()$ is the best solution on the validation data. This scheme has been evolved three times and has been found quite early in the GP process. The function is relatively simple and even a very complex function like $n_6()$ promotes similar relevant documents. The entire function promotes different relevant documents compared to $BM25$ or the pivoted normalisation scheme,

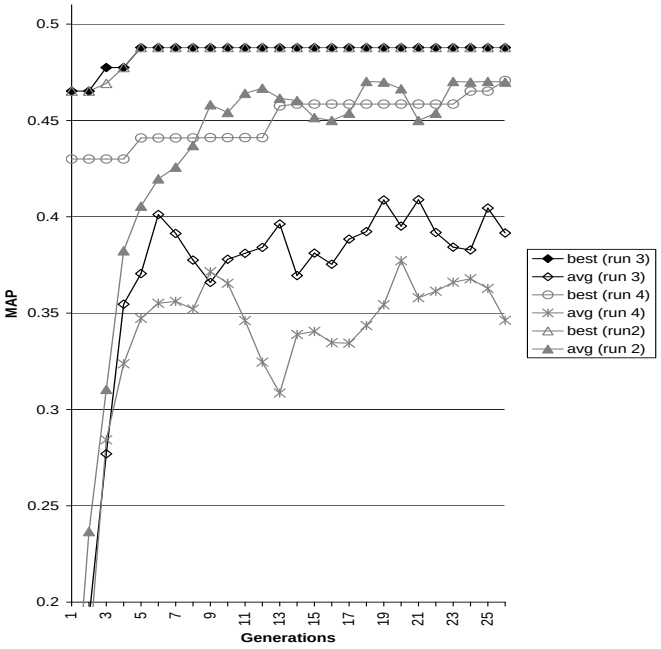

Figure 8.9: Best and average of population for three normalisation runs

as the actual ranked lists are different.

The distances in this environment are as follows: $dist(BM25, n_2())$ is 47.15, indicating that a relevant document is different by about 47 ranks. $w\_dist(BM25, n_2())$ is 0.10689 which is relates to the potential difference in MAP between the two schemes. $1 - \rho$ is 0.23835 ($\rho = 0.76165$) for these two schemes indicating that there is stronger correlation between the ranked lists produced by the solutions than in the previous solution spaces. This again indicates that the application of normalisation brings types of term-weighting schemes closer in terms of ranking. This has been suggested in terms of performance in the previous chapter. However, it can still be seen that the evolved term-weighting schemes are still different from the benchmarks. For the two benchmark schemes, $dist(BM25, PIV) = 29.36$, $w\_dist(BM25, PIV) = 0.0619$ and $\rho = 0.900$ indicating a stronger correla-

tion between the benchmark schemes.

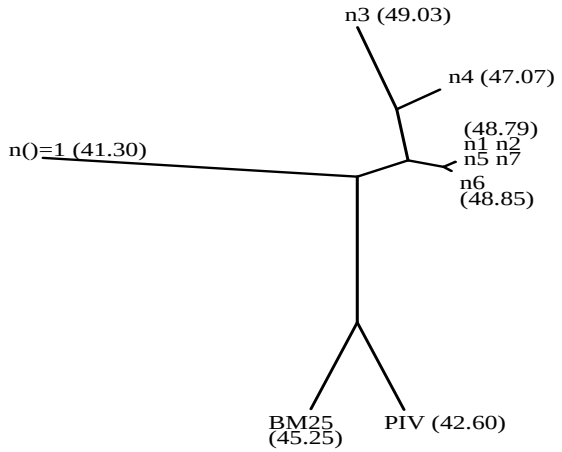

Figure 8.10: Visualisation of normalisation space using $dist()$ measure

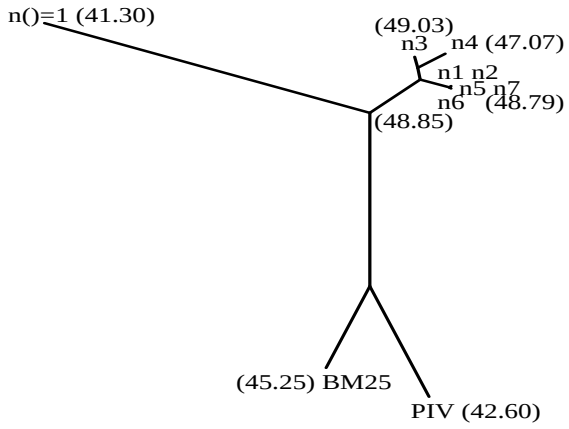

Figure 8.11: Visualisation of normalisation space using $w\_dist()$ measure

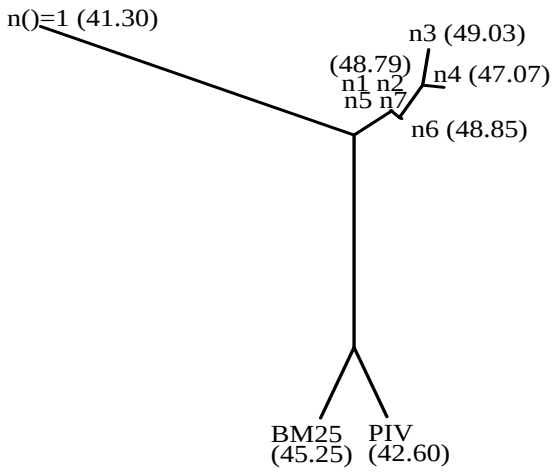

Figure 8.12: Visualisation of normalisation space using $1 - \rho$ measure

8.3 Summary

Two metrics that measure the distance between the ranked lists returned by different term-weighting schemes have been presented. These measures are useful for determining the closeness of term-weighting schemes and for analysing the solutions without the need to analyse the exact form (genotype) of a term-weighting scheme. This framework can be used for all types of term-weighting schemes and complements the GP paradigm adopted.

The distance matrices produced from these distance measures can be used to produce trees. The two measures outlined are quite similar as the trees produced have a similar form indicating that they provide similar information about relative distances between phenotypes. It has been shown that the GP evolves solutions which produce a similar ranking on the training data. This ranking is a better overall ranking than those produced by $BM25$ or the pivoted document length normalisation scheme. The evolved solutions tend to have a similarly high fitness. It has also been shown that these solutions are located in a different area of the solution space than current benchmarks.

Chapter 9

A Genotypic Analysis Using Constraints

In this chapter a detailed analysis of one of the evolved schemes previously outlined is presented. A new axiom which constrains the normalisation aspect of a term-weighting function is outlined and empirically validated in section 9.1. The axiomatic framework previously introduced is utilised to determine if the newly evolved scheme is consistent with the axioms (constraints). An analysis of the benchmark term-weighting schemes and a number of previously learned term-weighting schemes is also conducted using the constraints in section 9.2. Section 9.3 presents empirical evidence to support the axiomatic analysis. A summary of the important contributions conludes this chapter.

9.1 Axioms for Term-Weighting

The existing constraints are now briefly described before introducing a new constraint. The first constraint (constraint 1) states that adding a new query term to the document must *always* increase the score of that document. The second constraint (constraint 2) states that adding a non-query term to a document must *always* decrease the score of that document, while the third constraint (constraint 3) states that adding successive query terms to a

document should increase the score of the document less with each successive addition of that term. A weaker non-redundant constraint (constraint 1.1) states that adding a query term must lead to a higher score than adding a non-query term. These constraints (axioms) have been stated more formally in chapter 4.

9.1.1 A New Axiom for Normalisation

A new constraint is now proposed which aims to avoid over-penalising successive occurrences of non-query terms in documents. $S(Q, D)$ is a function which scores a document D in relation to a query Q in a standard *bag of words* retrieval model. With notation similar in style to the original research (Fang and Zhai, 2005), the new constraint can be formalised as follows, where $t \in T$ is a term in the set of terms in a corpus and $\delta_t^{-1}(t, D, Q) = S(Q, D \cup \{t\})^{-1} - S(Q, D)^{-1}$ (i.e. the change of the inverse score as t is added to the document D):

Constraint 4: $\forall Q, D$ and $t \in T$, if $t \notin Q$, $\delta_t^{-1}(t, D, Q) > \delta_t^{-1}(t, D \cup \{t\}, Q)$.

According to Heaps' law (1978), the appearance of new (unseen) terms in a corpus grows in roughly a square-root relationship (sub-linearly) to the document length (in words). Ultimately, it is the number of unique terms that is the best measure of how broad the topic of the document is likely to be. For example, consider a document that has 9 words ($dl = 9$) and contains 3 unique terms (i.e. vector length of 3). If this document grows in length to 100 words ($dl = 100$), the expected number of unique terms would be approximately 10. Thus, as the document grows in length, the topic broadens sub-linearly. Furthermore, it is the number of occurrences (term-frequency) of these unique terms that indicates the strength of each different aspect (i.e. dimension of the vector) of the document.

Simply using the vector length for normalisation might seem an intuitive approach when considering such an argument. However, using the vector length as the normalisation factor will lead to a violation of constraint 2. Consider a non-query term, which has already appeared in the document. If

this term re-occurs, the weight of the document will not decrease as the vector length remains unchanged. This argument has some similarities with the argument presented in chapter five (section 5.5) regarding the level of granularity of the document length feature chosen for normalisation. This is an unexpected yet elegant property of the evolved normalisation schemes. Consequently, the above constraint avoids over-penalising longer documents by ensuring that the normalisation aspect (measured in repeated terms) is sub-linear. Therefore, as non-query terms appear in a document they should be penalised less with successive occurrences. The normalisation used in term-weighting schemes is inversely related to the weight to apply, and therefore is typically the denominator in such functions. Essentially, the inverse of the score reduction due to non-query terms being added should be sub-linear.

9.1.2 An Empirical Validation

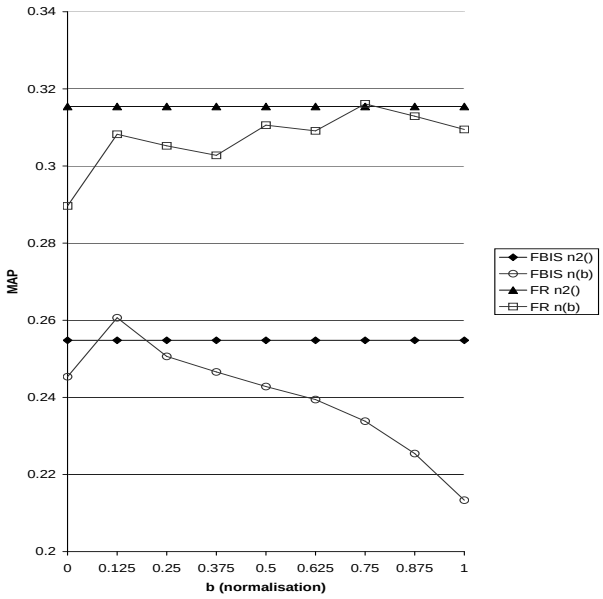

Figure 9.1: MAP of $BM25$ using $n(b)$ vs $n_2()$ for short topics

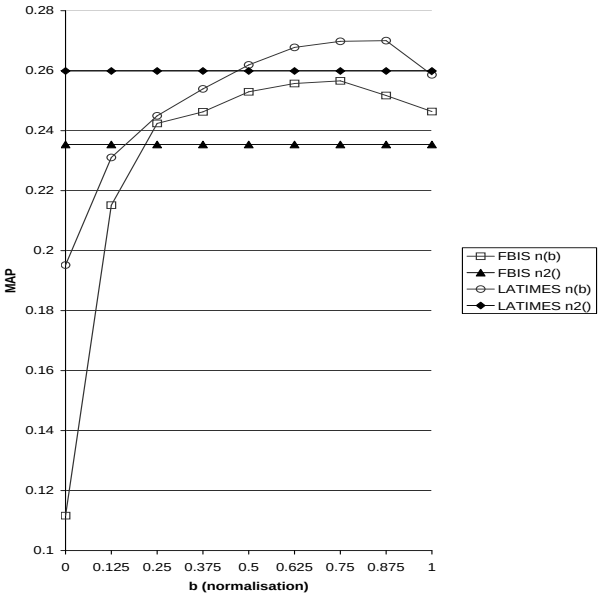

Figure 9.2: MAP of $BM25$ using $n(b)$ vs $n_2()$ for long topics

In this section the new axiom (constraint 4) is empirically validated. The standard $BM25$ scheme is used for this analysis as the normalisation scheme contained within is linear for all values of b and the normalisation scheme was developed in conjunction with the full $BM25$ scheme. The standard $BM25$ scheme using nine values of b (0, 0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.825 and 1) is compared against a modified $BM25$ scheme. The $BM25$ scheme is modified by incorporating an evolved normalisation factor. Thus, the evolved sub-linear normalisation ($\sqrt{dl/dl_{avg}}$) replaces the standard linear normalisation ($(1-b)+b\cdot\frac{dl}{dl_{avg}}$) in the $BM25$ scheme. Although the evolved normalisation was learned in conjunction with a different underlying function, it is incorporated into the $BM25$ scheme. Therefore, the *only* difference between the two versions of the $BM25$ scheme is that the modified $BM25$ scheme adheres to the new axiom (constraint 4) and the original $BM25$ does not. It is worth noting that this is the same experiment that appears

in chapter seven (section 7.2.3), except that the underlying term-weighting function is different.

Table 9.1: Comparison of $n_2()$ vs $n(b)$ for $BM25$

	short	medium	long
Collection	%	%	%
FR	99.99	59.63	44.42
FBIS	95.14	100*	45.17
LATIMES	100*	100*	74.16
FT	100*	100*	100*
OH89	97.76	99.99	-
OH90-91	85.30	100*	-

Figures 9.1 and 9.2 shows a sample of these results. These are included to show the variability of the optimal value of b in the $BM25$ scheme, even for the same query lengths. The comparison metric developed in chapter seven ($NC()$ equation (7.4)) is used to compare the parametic $BM25$ to the modified $BM25$. Table 9.1 details the results from the test collections. It can be seen that the modified scheme ($BM25$ with $n_2()$) often outperforms the optimal setting of b in the original $BM25$ scheme and in most cases performs close to the optimal. For some longer queries the performance is slightly worse than a random setting of b (50%), but in general, and especially when taking into account the results from chapter seven, the results validate both the experimental analysis in chapter seven and the theoretical analysis presented in this chapter. It is encouraging that a non-parametric normalisation function that adheres to the new axiom performs comparably to the *optimal* performance of the linear normalisation function. Furthermore, it is reasonable to assume that the reason that the tuning parameter (b) is needed is because a linear function shape is not correct for normalisation, and thus needs to be tuned depending on the data presented.

9.2 Axiomatic Analysis of Schemes

This section presents an analysis of the evolved term-weighting scheme developed. Furthermore, a brief analysis of three separate term-weighting schemes developed using GP (Oren, 2002b; Fan et al, 2004; Trotman, 2005) is presented. An analysis of the $BM25$ and pivoted document normalisation schemes using the aforementioned constraints is also included. Consequently, four learned functions and two standard benchmarks are analysed. A table detailing constraint satisfaction for each of the term-weighting schemes concludes this section.

9.2.1 Incrementally Evolved Function

One of the term-weighting schemes developed in this work can be written as follows:

$$F1(Q,D) = \sum_{t \in Q \cap D} (ntf(tf_t^D, dl) \cdot \sqrt{\frac{cf_t^3 \cdot N}{df_t^4}} \cdot tf_t^Q) \tag{9.1}$$

where

$$ntf(tf_t^D, dl) = \frac{tf_t^D / n(dl)}{(tf_t^D / n(dl)) + 0.45} \tag{9.2}$$

and

$$n(dl) = \sqrt{\frac{dl}{dl_{avg}}} \tag{9.3}$$

The term-frequency influence factor here (i.e. $\frac{tf_t^D/n(dl)}{(tf_t^D/n(dl))+0.45}$) has been modelled to reflect the effect of an evolved term-frequency influence function by measuring the relative term-frequency (chapter six). A previous evaluation using a term-discrimination scheme very similar to one of the schemes evolved here, showed that the within-document component of the $BM25$ scheme could be incorporated into the new scheme when k_1 was set to 0.2 (Cummins and O'Riordan, 2005). This is again considerably lower than the

default of 1.2.

Inductive Document and Query Functions

Using inductive growth functions (Fang and Zhai, 2005), the newly evolved function can be re-written and analysed with regard to the constraints outlined in that work. The idea of this inductive framework is to define a base case function that describes the score (weight) assigned to a document containing a single term matching (or not matching) a query containing a single term. All other cases can be dealt with inductively using two separate functions. A *document growth function* describes the change in the score when a single term is added to the document, while a *query growth function* describes the change in the score when a single term is added to the query. The inductive growth functions for the newly evolved weighting approach are now described. The parts of the newly evolved function which are constrained by the framework adopted are also indicated. The constraints (axioms) outlined in the axiomatic framework are then used as criteria for judging the potential effectiveness of the scheme prior to evaluation.

In the inductive framework, $\{q\}$ describes a term added to the query and $\{d\}$ describes a term added to a document. The base case simply describes the weight given to a one-term query matching (or not matching) a one-term document and is described as follows:

$$S(Q, D) = f(q, d) = \begin{cases} weight(q) = weight(d) & q = d \\ penalty(q, d) & q \neq d \end{cases}$$

where $S(Q, D)$ scores a document D in relation to a query Q. The function assigns a score of $weight(q)$ to the document when d matches q, otherwise it assigns a penalty score of $penalty(q, d)$. The evolved term-weighting function can be rewritten as follows using notation similar in style to the original work (Fang and Zhai, 2005), where $\{x, y\} \in \mathbb{Z} > 0$ refer to the term-frequency and the document length respectively:

$$S(Q,D) = \sum_{t \in Q \cap D} (ntf(x,y) \cdot \sqrt{\frac{cf_t^3 \cdot N}{df_t^4}} \cdot tf_t^Q) \tag{9.4}$$

where $ntf(x,y)$ is as equation 9.2 and $n(y) = \sqrt{\frac{y}{dl_{avg}}}$ (equation 9.3). The base case function can be written as follows:

$$weight(q) = \sqrt{\frac{cf_t^3 \cdot N}{df_t^4}} \cdot ntf(1,1) \tag{9.5}$$

which accurately describes the weight assigned to a one-term query matching a one term document and was learned in the framework. The following case describes the weight given to a query term that does not match a document term:

$$penalty(q,d) = 0 \tag{9.6}$$

It should be noted that $penalty(q,d) = 0$ because of the way the framework is modelled and not as a result of the GP process itself. Thus, the search is constrained to those which do not penalise terms explicitly for not occurring. The following query growth function ($g()$) describes the change in weight assigned to a document as a term is added to the query:

$$g() = S(Q,D) + S(\{q\},D) \tag{9.7}$$

This is similar to the pivoted normalisation query growth function as the weight grows linearly as terms are added to the query. This query growth function is imposed by the framework adopted, as it can be seen that query terms are weighted in a simple manner. It has been previously noted that this is a simple form of growth function. However, there has been no justification for a more complex form. The following function is the document growth function and can be written in a somewhat similar manner to that of the $BM25$ weighting:

$$h() = \sum_{t \in Q \cap D - \{d\}} (S(Q,\{t\}) \cdot \frac{ntf(tf_t^D, dl+1)}{ntf(1,1)}) + S(Q,\{d\} \cdot \frac{ntf(tf_d^D+1, dl+1)}{ntf(1,1)})$$

This document growth function has been learned in stages and was restricted in a certain sense by the properties and features initally supplied to the learning approach, but as is identified in the next section, it was not constrained by the axioms developed in previous research. This re-writing process helps to examine the difference between this scheme, the $BM25$ scheme and the pivoted document normalisation scheme. It can also be useful for developing new schemes in a similar analytical approach to the original work (Fang and Zhai, 2005).

Satisfaction of Constraints

Due to the nature of most modern term-weighting schemes, if the document length is used explicitly to penalise the document, constraint 1 (and consequently constraint 3) will never be satisfied unconditionally[1]. However, if stop-words are removed, the likelihood of this occurring will be lessened. For this analysis, this phenomenon (i.e. the typically small penalisation of existing terms in the document) will be ignored as per previous work (Fang and Zhai, 2005). Therefore, constraint 1 will be deemed satisfied if the score, attributable from the term, and not the document as a whole, increases for every occurrence of that term. This proviso leads to the recovery of the more specific term-frequency constraints in the original work by Fang (2004). Similarly, constraint 3 will be deemed satisfied if the score increase, attributable from a particular term, grows sub-linearly. Nonetheless, the inductive framework and constraints therein are still preferable as they are more general, more intuitive and are incorporated in a more elegant framework.

The newly evolved scheme satisfies all previously existing constraints and the newly postulated constraint. It is worth noting that as there are no tuning parameters in this function, the constraints are satisfied unconditionally. Firstly, it can be seen that the term-discrimination part of this function always produces a positive value. In fact, all of the term-discrimination factors evolved (global term-weighting schemes) produce a positive value.

The first constraint (constraint 1) states that adding a new query term

[1]See Appendix A for a complete explanation

to the document should always increase the score of a document. In the newly developed scheme it can be seen that that $ntf(x{+}1, y{+}1) > ntf(x, y) \ \ \forall x, y > 0$. It is trivial to show that $ntf(x + 1, y) > ntf(x, y) \quad \forall x, y > 0$ which is if the length aspect of the document is ignored. As the term-frequency ($ntf(x, y)$) is normalised using $\frac{x}{n(y)}$, when x increases by 1, y will increase by 1. Thus, when $n(y)$ is sub-linear (as is the case in the formula) or is linear with $n(0) > 0$, this will be satisfied. Due to the method of normalisation used and because the term-discrimination scheme used will always return a positive value, this constraint is more readily satisfied. Indeed, all of the evolved term-discrimination schemes evolved return a positive value. This is a fundamental property that can be derived from this axiom regarding the term-discrimination schemes.

The second constraint (constraint 2) states that adding a non-query term must decrease the score of a document. It is true that $ntf(x, y + 1) < ntf(x, y)$ for the scheme as the normalisation scheme identified ($n(y)$), increases $\forall y > 0$. This will decrease the score of a document. This constraint enforces some sort of document normalisation (penalisation). As the first two constraints hold it is obvious that $ntf(x + 1, y + 1) > ntf(x, y + 1)$ which simply indicates that adding a query term to a document will achieve a higher score than adding a non-query term to a document (constraint 1.1).

The third constraint (constraint 3) states that adding successive query terms to a document will increase the score of the document less with each successive occurrence. Essentially the term-frequency influence must be sub-linear. It can be shown that $(ntf(x+1, y+1) - ntf(x, y)) > (ntf(x+2, y+2) - ntf(x+1, y+1)) \ \forall x, y > 0$ is true for all x and y in the formula. Indeed, it can be shown that five of the seven evolved term-frequency factors are sub-linear.

The fourth constraint (constraint 4) states that the inverse of the score reduction due to successive non-query terms should be sub-linear. Basically, $\forall x, y > 0 \ (ntf(x, y + 1) - ntf(x, y))^{-1} > (ntf(x, y + 2) - ntf(x, y + 1))^{-1}$ must be true for the constraint to be satisfied. It can be seen that this is true of the evolved term-weighting scheme developed. It is also interesting that if the normalisation function ($n(y)$) is sub-linear with respect to y and

the term-frequency is normalised using $\frac{x}{n(y)}$ (which is used in the framework), then the first constraint (constraint 1) will always be satisfied. Indeed, six of the seven evolved normalisation schemes are sub-linear. This gives some empirical support to the new axiom.

9.2.2 BM25

The $BM25$ weighting scheme is presented here again for completeness. The score of a document D in relation to a given query Q can be calculated as follows:

$$BM25(Q,D) = \sum_{t \in Q \cap D} (\frac{tf_t^D \cdot tf_t^Q}{tf_t^D + k_1 \cdot ((1-b) + b \cdot \frac{dl}{dl_{avg}})} \cdot log(\frac{N - df_t + 0.5}{df_t + 0.5})) \tag{9.8}$$

k_1 is set to 1.2 by default, while b has a default value of 0.75.

Analysis

It can be noted that the idf component in the $BM25$ ($log(\frac{N-df_t+0.5}{df_t+0.5})$) function will return a negative value when $df_t > \frac{N}{2}$ and thus violates constraints 1 and 3 in certain circumstances. However, it typically violates these constraints when stop-word removal is not used, as very frequent terms would otherwise be removed. Interestingly, the new constraint outlined here (constraint 4) is violated by $BM25$ and the pivoted normalisation scheme (outlined next). It can be seen that constraint 4 is violated when the following normalisation function is used in both $BM25$ and the pivoted normalisation scheme (for any value of b), as this is linear with respect to dl:

$$n_b = ((1-b) + b \cdot \frac{dl}{dl_{avg}}) \tag{9.9}$$

This analysis suggests that when using this function, b needs to be tuned on each specific collection. If the collection contains some long documents compared to the average document length, it would be important to have a low value for b as it would otherwise unfairly penalise these longer documents.

9.2.3 Pivoted Document Length Normalisation

The pivoted document length weighting scheme (Singhal, 2001) calculates the score of a document D in relation to a given query Q as follows:

$$PIV(Q,D) = \sum_{t \in q \cap d} (\frac{1 + log(1 + log(tf_t^D))}{(1-s) + s \cdot \frac{dl}{dl_{avg}}} \cdot log(\frac{N+1}{df_t}) \cdot tf_t^Q) \qquad (9.10)$$

where s is the normalisation parameter referred to as the slope and has a default value of 0.2.

Analysis

It can be noted that the *idf* component in the pivoted document length normalisation ($log(\frac{N+1}{df_t})$) function will always return a positive value and the term-frequency factor is sub-linear. However, due to the manner in which normalisation is incorporated, constraints 1 and 3 are not satisfied unconditionally for all values of $s < 1$. In this formula, the normalisation may grow and offset any gain in the term-frequency factor. A further discussion on normalisation is included in Appendix A. This scheme satisfies constraint 1 and 3 in most circumstances (i.e. typically while $s < 0.4$ (Fang et al, 2004)). It can also be seen that the normalisation function used in this function is linear and thus violates constraint 4. Again, this suggests that it needs to be heavily tuned depending of the collection.

9.2.4 Oren

One of the schemes outlined in one of first approaches adopting GP to IR (Oren, 2002a) can be re-written as follows:

$$F2(Q,D) = \sum_{t \in Q \cap D} (\frac{tf_t^D}{tf_t^D + df_t + dl \cdot (1 + 0.436 \cdot \frac{tf_t^D}{tf_{max}^D} \cdot (cf_{max} + log(cf_{max})))}) \qquad (9.11)$$

where tf_{max}^D is the frequency of the most common term in D, cf_{max} is the frequency of the most common term in the collection.

Analysis

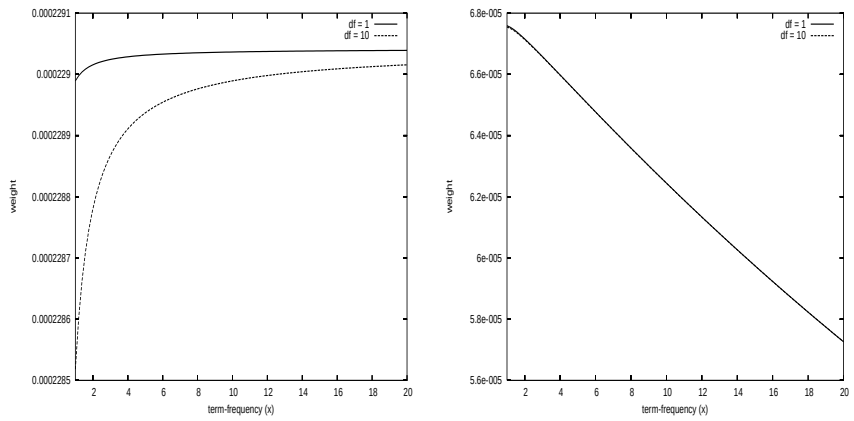

Figure 9.3: Change in weight for $F2$ for scenarios 1 and 2 respectively

This function can be written as $\frac{tf_t^D}{tf_t^D+df_t+dl\cdot(1+\frac{tf_t^D}{tf_{max}^D}\cdot K_1)}$ where K_1 is a constant such that $K_1 > 0$. Consider the simple case where a document consists of multiple occurrences of the same query-term (scenario 1). In such a circumstance, $x = tf_t^D = dl = tf_{max}^D$. The function can then be re-written as $\frac{x}{x+df_t+x\cdot(1+K_1)}$ where df_t is a constant for a particular term. Figure 9.3 shows the change in score for the first 20 occurrences of the query term in this scenario. An extra occurrence of a query term will make the score of the document higher and the term-frequency aspect will grow sub-linearly. In these circumstances, constraint 1 and 3 are satisfied. Now, consider a document with 100 terms ($dl = 100$) and a maximum term-frequency of 30 for one of the terms ($tf_{max}^D = 30$) already occurring in the document (scenario 2). The function can then be written as $\frac{x}{x+df_t+(100+x)\cdot(1+\frac{x}{30}\cdot K_1)}$. Figure 9.3 also shows the change in weight for the first 20 occurrences of another query term added to the document in this scenario. It can be seen that in this circumstance that the score of the document (attributable from that query term) actually decreases. This violates constraint 1 and 3. Constraint 2 is satisfied as the score of a document will always decrease as non-query terms are added due to dl in the denominator. It can be shown that constraint 1.1 is also satisfied. However, it can be determined that the normalisation component is linear in nature (i.e. the normalisation component dl is linear

in the denominator) violating constraint 4. As a result this function is likely to perform poorly in general, and even worse for long queries where a number of different query terms will interact incorrectly.

9.2.5 Fan et al

The best function outlined from another GP approach (Fan et al, 2004) can be re-written as follows:

$$F3(Q,D) = \sum_{t \in Q \cap D} \left(\frac{log(tf_t^D \cdot X)}{vl + 2 \cdot tf_{max}^D + 0.373}\right) \cdot tf_t^Q \tag{9.12}$$

where

$$X = \left(tf_{avg}^D + \frac{tf_t^D}{log(tf_t^D \cdot 2 \cdot tf_{avg}^D)} + \frac{tf_t^D \cdot N \cdot tf_{avg}^D \cdot (tf_{max}^D + vl)}{df_t^2}\right) \tag{9.13}$$

where tf_{avg}^D is the average term-frequency in D and vl is the length of the document vector (unique terms).

Analysis

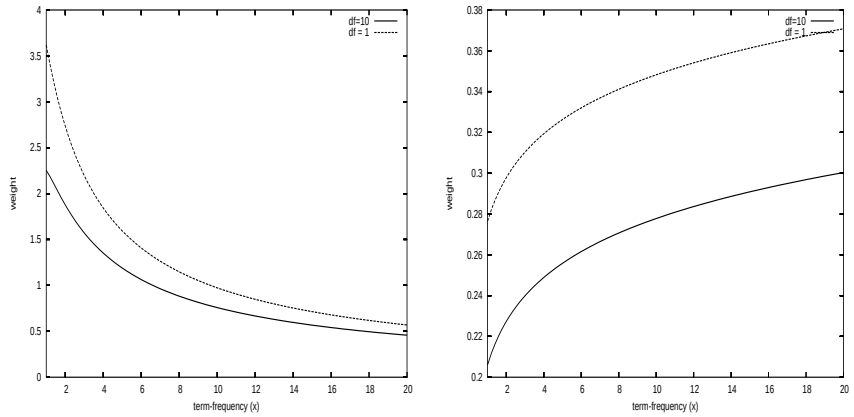

Figure 9.4: Change in weight for $F3$ for scenarios 1 and 2 respectively

Again, consider the simple case where a document consists of multiple occurrences of the same term. In such a circumstance, $x = tf_t^D = dl = tf_{max}^D = tf_{avg}^D$ and $vl = 1$. The weight of the document can be written as $log(x \cdot (x + \frac{x}{log(x \cdot 2 \cdot x)} + \frac{x \cdot N \cdot x \cdot (x+1)}{df_t^2}))/(1 + 2 \cdot x + 0.373)$. Figure 9.4 shows the score change when $N = 100,000$ and when $df_t = 1$ and when $df_t = 10$. It

can be seen that constraints 1 and 3 will be violated in this simplistic first scenario. Now consider the more common case where a document already contains a number of terms (e.g. $dl = 100$, $tf^D_{max} = 30$ and $vl = 4$). Figure 9.4 shows the score change (attributable from a particular query term) when it is added to this type of document. The score change can be re-written as $log(x \cdot (\frac{100+x}{5} + \frac{x}{log(x \cdot 2 \cdot \frac{100+x}{5})} + \frac{x \cdot N \cdot \frac{100+x}{5} \cdot (30+5)}{df_t^2}))/(5 + 2 \cdot 30 + 0.373)$. In this second scenario, a new occurrence of a query-term results in a higher score and the increase in score as the term-frequency increases is sub-linear. As a result, constraints 1 and 3 are conditionally satisfied. As the normalisation used is the number of unique terms (vector length), constraints 2 and 4 are violated. If a non-query term which has already appeared in the document re-occurs, the weight of the document will not decrease as the vector length remains unchanged. Even if the document length factor used was changed to the document length (i.e. dl), constraint 4 would still be violated, as the normalisation factor is linear in nature.

9.2.6 Trotman

One of the best performing schemes in the GP approach adopted by Trotman (2005) can be re-written as follows:

$$F4(Q,D) = \sum_{t \in Q \cap D} (log_2 |\frac{N - log_2|N|}{2 \cdot df_t}| \cdot \frac{cf_t}{df_t} \cdot \frac{tf_t^D \cdot tf_t^Q \cdot cf_{max}}{max(C_3, C_4 + \frac{C_1 \cdot (log|C_2 + tf_t^Q| + cf_t) \cdot dl}{N \cdot dl_{avg}}) + tf_t^D}) \quad (9.14)$$

where cf_t is the frequency of t in the entire collection of N documents. C_1, C_2, C_3 and C_4 are constants of value 33.40102, 23.94623, 1.2 and 0.25 respectively.

Analysis

Firstly, the $log_2|\frac{N - log_2|N|}{2 \cdot df_t}|$ part can lead to a negative weight for terms with a high document frequency. This leads to constraints 1, 1.1 and 3 being violated in circumstances similar to the $BM25$ scheme. Also, it can be seen that when $C_3 > C_4 + \frac{C_1 \cdot (log|C_2 + tf_t^Q| + cf_t) \cdot dl}{N \cdot dl_{avg}}$, which typically occurs when cf_t is low (i.e. for rare terms), the within-document part of the formula reduces to

$\frac{tf_t^D}{1.2+tf_t^D}$ which is a primitive form of the $BM25$ local weighting. This form of the function has no normalisation component and thus, violates constraint 2 in these circumstances. However, this form conditionally satisfies constraints 1 and 3 as adding a query term to a document always increases its score (constraint 1) and is always sub-linear (constraint 3).

When $C_3 < C_4 + \frac{C_1 \cdot (log|C_2 + tf_t^Q| + cf_t) \cdot dl}{N \cdot dl_{avg}}$, which typically occurs when the collection frequency for a term is high (i.e. more common terms) a different form of the function is used. Consider a typical case when tf_t^Q is 1 and N is large (e.g. 100,000). In such a case, this reduces to approximately $0.25 + \frac{3.2+cf_t}{3000} \cdot \frac{dl}{dl_{avg}}$. For high values of cf_t, this will exceed C_3 (i.e. 1.2). Interestingly, this can be re-written as $\frac{tf_t^D}{0.25+K_2 \cdot \frac{dl}{dl_{avg}}+tf_t^D}$ (where K_2 is a global constant for a particular term) which contains a normalisation form similar to the $BM25$ scheme. When the function takes this form constraints 1, 1.1, 2 and 3 are satisfied. However, the normalisation scheme is not sub-linear and thus does not conform to the new constraint (constraint 4).

9.2.7 Summary of Constraint Satisfaction

Table 9.2: Constraint satisfaction

		Constraints				
Rank	Scheme	1.1	1	2	3	4
1	$F1(Q,D)$	Yes	Yes*	Yes	Yes*	Yes
2	$BM25(Q,D)$	Cond.	Cond.	Yes	Cond.	No
3	$PIV(Q,D)$	Yes	Cond*	Yes	Cond*	No
4	$F2(Q,D)$	Yes	Cond*	Yes	Cond*	No
5	$F4(Q,D)$	Cond.	Cond.	Cond.	Cond.	No
6	$F3(Q,D)$	Yes	Cond*	No	Cond*	No

Table 9.2 shows the constraints that each scheme satisfies. The conditional satisfaction (denoted "Cond.") indicates that the constraint is satisfied in some circumstances (as noted in the analysis) but does not unconditionally satisfy the constraint. "Cond*" indicates a weaker conditional satisifaction. For example, constraint 3 will only be violated by the $BM25$ scheme for cer-

tain high frequency terms (stop-words), while violations of constraint 3 are inherently more probable for $F2$ as the function shape is incorrect. Schemes that satisfy constraints 1 and 3 (i.e. ignoring the typically small penalisation of previously existing terms in the document) are denoted as "Yes*". Schemes that adhere to a constraint unconditionally are denoted "Yes". Table 9.2 also shows how the schemes can be ranked based on how many constraints they satisfy. $BM25$ is ranked ahead of PIV as the constraints violated in each case are for different reasons. Thus, PIV will typically break constraints more often than $BM25$ will. $F2$ will also break constraints more often than PIV because of the different conditions that lead to violations.

It should be noted that this ranking is coarse as it is not known if violations of different constraints lead to equal levels of suboptimality. It is also unknown if the schemes identified are specific to a type of query or indeed the specific environment in which they were trained. Nonetheless, given these details of constraint satisfaction it seems an intuitive and possibly useful way of ordering the schemes by expected performance. Furthermore, it is difficult to know exactly how many constraint violations would happen in a typical retrieval setting without building a mathematical model that could measure the number of violations on an actual document collection.

9.3 Empirical Comparison

In this section experiments are presented which empirically validate the previous analysis. The use of stop-word removal in the experiment leads to constraints 1 and 3 being satisfied for $BM25$ and $F4$ on the collections used here. This has been determined empirically.

9.3.1 Experimental Results

The results in Tables 9.3 and 9.4 show that $F1$ outperforms most of the other schemes on the *various* test data. The remaining schemes tend to perform in accordance with the rank in Table 9.2, except for $F2$ which performs quite poorly on most of the test data. Statistical significance is measured against

Table 9.3: %MAP on unseen test collections (I)

	LATIMES (351-450)			FT91-93 (351-450)			FBIS (301-450)		
#docs	(131896)			(138668)			(130471)		
#topics	(95)			(96)			(115)		
Schemes	short	medium	long	short	medium	long	short	medium	long
$F1$	22.76	24.62	27.52	28.95**	31.94	34.98**	27.07**	29.06**	28.61
$BM25$	21.28	24.32	26.97	27.71	29.70	32.56	23.48	24.28	25.66
PIV	19.52	22.22	25.23	26.03	27.86	30.89	22.02	23.17	24.05
$F2$	11.92	11.17	11.94	17.02	16.72	14.44	14.45	10.85	07.67
$F4$	22.22	24.55	26.44	28.23	29.13	31.04	22.44	24.76	26.10
$F3$	11.74	05.78	02.49	22.71	12.23	06.95	25.39	20.50	10.76

Table 9.4: %MAP on unseen test collections (II)

	FR (301-450)			OH89 (1-63)		OH90-91 (1-63)	
#docs	(55630)			(74869)		(148162)	
#topics	(64)			(63)		(63)	
Schemes	short	medium	long	short	medium	short	medium
$F1$	33.11	33.81	39.27**	27.56	32.80**	25.53	30.07**
$BM25$	31.61	31.68	33.47	27.29	30.67	25.54	28.08
PIV	27.87	28.56	30.06	27.65	30.48	24.95	26.76
$F2$	14.98	14.12	12.56	20.26	15.70	17.70	13.51
$F4$	26.55	27.05	28.55	26.10	30.72	23.19	26.27
$F3$	29.21	18.76	08.50	06.97	00.90	06.60	01.05

the $BM25$ benchmark scheme. Two astericks (**) denotes a statistically significant increase over $BM25$. It should be noted that $F1$ is a slightly different function than in chapter seven as the term-frequency factor is slightly different. Therefore, the significance tests are slightly different for one of the collections.

9.3.2 Discussion

The existing axioms and the newly postulated axiom are useful estimators of term-weighting optimality. They can be useful in estimating the performance

of a scheme. An interesting result is that many of the learned approaches conditionally adhere to some of the constraints. This would suggest that they may have learned useful methods for weighting terms in the training environment but that their training data is quite specific (i.e. the constraints are satisfied for the characteristics on the training data but are not unconditionally satisfied).

These results can aid in the learning of term-weighting functions. Small collections (less than 10,000 documents) should be avoided when aiming to learn *generalisable* term-weighting schemes. Indeed, it has already been shown that the term-discrimation (global) part of a term-weighting scheme can indeed be learned on a small collection but it is typically the within-document (local) part of these schemes that is not generalisable (Cummins and O'Riordan, 2006a). $F2$ (Oren, 2002a) was evolved on a document collection of less than 1000 documents. It can be seen that the performance of this scheme is quite poor and gets worse as more terms are added to the query.

Furthermore, it is advisable to use medium or long queries when learning term-weighting schemes. Using only short queries in training (Fan et al, 2004) will most likely lead to very specific term-weighting functions as short queries do not provide as much information about how terms should interact with each other (particularly in a term-discrimination context). This can be seen as the performance of $F3$ decreases as the query length increases for all collections. It may be worth noting that using medium or long queries will lead to an increase in training time.

To overcome the collection dependance problem (which typically affects the type of normalisation to use), it is advisable to use multiple varied training collections indexed seperately in order to learn schemes that will adhere to the constraint specified here. Another possible approach would be to use vastly different query lengths as it has previously been shown that short queries return documents sets with a higher deviation of document length. However, when adopting such an approach the query length should *not* be explicitly used as a feature in the term-weighting scheme. This is because it has been shown that the difference in query length creates different char-

acteristics in the returned set of documents and a tuning parameter (b) is needed only if the normalisation function shape is linear.

9.4 Summary

A new normalisation constraint to which the newly evolved scheme adheres is outlined and both theoretically and empirically validated. The full evolved scheme is analysed using the axiomatic approach to IR. This term-weighting scheme is decomposed into document and query growth functions using the axiomatic framework and is shown to satisfy previously known constraints. The validity of the existing constraints is further enforced as the term-weighting scheme described was learned using a purely empirical learning approach.

It has been shown that the three stage process does not enforce adherence to the axioms. However, the resulting schemes do tend to adhere to the axioms more readily than if the functions were evolved in their entirety.

A number of other term-weighting functions are analysed and shown to adhere to the some of the axioms. One of the schemes is consistent with more axioms than the others. Interestingly, this scheme is the incrementally evolved term-weighting scheme. Finally, an evaluation of the term-weighting schemes validates the analysis. This chapter concluded the work into term-weighting schemes for traditional text-based IR systems for adhoc retrieval.

Chapter 10

GP for Automatic Query Expansion

In this chapter, schemes for the selection and re-weighting of terms in two standard query expansion (QE) approaches are evolved in a "bag of words" retrieval model. This chapter can be viewed somewhat independently as it deals with a different weighting problem from that previously discussed. Section 10.1 describes the two basic approaches currently used to automatically expand queries with potentially useful terms. Section 10.2 outlines the framework and experiments in which schemes for the selection of expansion terms are evolved. The terminal and function sets for both QE approaches are also outlined in this section. The results of the experiments are detailed in section 10.3. Finally section 10.4 summarises the main points of the chapter.

10.1 Automatic Query Expansion

Many relevant documents may never get returned by an IR system simply because the vocabulary of the author and that of the searcher are different. The example used in a previous chapter supposed a user searching for information about fixing a "leaking tap". Many potentially relevant documents may contain the word "faucet" instead of "tap". This *mismatch* typically excludes many documents from the returned set and often limits the recall of

the system. Automatic query expansion deals with automatically adding extra terms to the query based on some heuristics and resubmitting the newly expanded query to the IR system.

10.1.1 Local QE Benchmark

In this approach, the set of terms (E) to add to the query are chosen based on frequency characteristics of the terms in the top few documents (P) of an initial retrieval run. The top P documents are assumed relevant and the Robertson/Sparck-Jones weight, developed for the probabilistic model of IR (w_{rsj} in equation (2.7)), is often used to determine the usefulness of the terms (Robertson and Walker, 1999). A simple but effective term-selection scheme (Robertson and Walker, 1999) is $TSV_t = pdf_t \cdot w_{rsj}$ (equation (2.8)). This selection scheme simply chooses terms with a high discrimination weight (w_{rsj}) that appear in many of the top P documents. A number of terms ($|E|$) is then chosen based on the TSV_t score and these are added to the query. The weight applied to these expanded terms is the w_{rsj} weight instead of the *idf* weight. The number of terms ($|E|$) and number of top ranked documents ($|P|$) deemed relevant are usually fixed. In recent years, local query expansion has shown promise and can typically increase the MAP of certain easier queries. Research has shown that improved results are achieved when the weights of the terms added to the query using this approach are reduced to about a third of their value (Billerbeck et al, 2003). This will be the benchmark used in the experiments for this approach ($TSV\frac{1}{3}$). Some other experiments have used a multiplier of 2.5 which similarly increases the weight of the original query terms as these are likely to be more important (Walker et al, 1997).

10.1.2 Global QE Benchmark

This method of query expansion has been less successful than its local counterpart on larger TREC type test collections (Harman, 1993). Term-term (or co-occurrence) relationships are often measured using the cosine similarity measure (equation 2.9). Choosing expansion terms in isolation can often ig-

nore the concept of the query. Therefore, it is often beneficial to measure the possible expansion terms against each term in the query and aggregate the values (Qiu and Frei, 1993). The cosine similarity measure is used in these experiments as a measure of similarity between terms. The vectors that describe these terms contain binary values indicating the presence or absence of the term in the document. Thus, documents become the descriptors of the terms and the vector length becomes the number of documents in the collection. Therefore, the benchmark scheme for measuring the quality of an expansion term t against the *entire* query is as follows:

$$\overline{cos(Q,t)} = \frac{1}{|Q|} \cdot \sum_{q \in Q} \frac{df_{q,t}}{\sqrt{df_q \cdot df_t}} \tag{10.1}$$

This measures the similarity between each term in the query Q against a possible expansion term t. The average is then used to select and weight terms in the framework. The top ranked terms in the collection (which should be related to the concept of the entire query) are then added to the query.

10.2 Experimental Design

This section introduces the experimental design of both the local and global GP approaches to query expansion.

10.2.1 Evolving Local QE Selection Functions

The GP approach adopted evolves the scheme used to select and weight terms for use in the expanded query in order to improve the retrieval of the system. For each query expansion scheme, each term in the top P documents from the initial retrieval run is rated on how useful it is. Firstly, it is necessary to choose a value for $|P|$ and decide how many terms ($|E|$) to add to the original query. Previous research has indicated that values for $|P|$ should be between 8 and 16 and values for $|E|$ should lie between of 7 and 42. Values of $|P| = 10$ and $|E| = 16$ are used as these lie within the best parameter

ranges (Walker et al, 1997; Billerbeck and Zobel, 2003). In the approach adopted, the term selection schemes are also used to weight the expansion terms. Therefore, the top 16 terms should be the most important and terms lower down the rank should not modify the query as much (and should be weighted accordingly). $|E|$ is set to 16 so that solutions can be evolved in a reasonable time, as longer queries take longer to process. It is intuitive that the weight of an expansion term should be a function of the usefulness of the expansion term. It is also logical to assume that the weight of the expansion term is also related to the weighting scheme applied to the original query terms (i.e. the default BM25 scheme). Thus, the following formula is used to score the complete expanded query $(E \cup Q)$ in relation to a document D:

$$S(E \cup Q, D) = BM25(Q, D) + \sum_{t \in E} LSF_t \cdot BM25(t, D) \tag{10.2}$$

where Q is the original query, E is the set of expansion terms, LSF_t is the local selection function (to be evolved) and $BM25(t, D)$ is the $BM25$ score of a single term t in the document D. Thus, a weighting of 1 for LSF_t would indicate that the expansion term is as important as if it had occurred in the original query. Therefore, the GP can also learn the correct weighting for the expansion terms, which is a function of the usefulness of the expansion term. This is different from the way in which terms are selected and re-weighted using the benchmark Robertson/Sparck-Jones weight.

10.2.2 Evolving Global QE Selection Functions

In this approach, the scheme used to select and weight terms based on co-occurrence characteristics is evolved. Again, it is assumed that the weight of an expanded term is a function of $GSF(Q, t)$ (i.e. the similarity of a term to the entire query). Similar to the local expansion framework, the expansion term is also related to the weighting scheme applied to the original query terms (i.e. the default BM25 scheme). Thus, the following formula is how the system scores the complete expanded query $(E \cup Q)$ in relation to a

document D:

$$S(E \cup Q, D) = BM25(Q, D) + \sum_{t \in E} GSF(Q, t) \cdot BM25(t, D) \tag{10.3}$$

where again Q is the original query and E is the set of expansion terms. Again, a weighting of 1 for $GSF(Q, t)$ would indicate that the expansion term t is as important as if it had occurred in the original query. The term-term similarity measure ($sim(q, t)$) is the similarity to be evolved in the framework as follows:

$$GSF(Q, t) = \sum_{q \in Q} sim(q, t) \cdot tf_q^Q \tag{10.4}$$

where tf_q^Q is the frequency of query term q in the query Q. The $GSF(Q, t)$ value is calculated for a subset of the terms in a collection and the top few are chosen to be added to the query. This approach is computationally expensive and as a result only the top 8 terms are chosen to add to each query.

10.2.3 Terminal and Function Sets

Local QE Terminal set

The terminal and function set for the local expansion approach is determined by considering the characteristics of the terms in the set of pseudo-relevant documents and the characteristics of these terms in the entire collection. It is important to keep the terminals as primitive (atomic) as possible so that there are fewer assumptions as to how the relevance of terms, documents and pseudo-relevant documents are related. The GP should be allowed to discover the best way to combine these to improve the performance on the training data. Table 10.1 and Table 10.2 show the terminal and function set used in these local query expansion experiments. It can be seen that most of the terminals are primitive and easy to calculate once an initial retrieval run has been conducted.

Table 10.1: Local QE terminal set

Terminal	Description
N	no. of documents in the collection
P	no. of documents in pseudo-relevance set (i.e. 10)
cf_t	frequency of term t in the collection
df_t	document frequency of term t in the collection
pcf_t	frequency of term t in the set of pseudo-relevant documents
pdf_t	number of pseudo-relevant documents containing t
V	vocabulary of collection (no. of unique terms)
C	size of collection (no. of words)
U	vocabulary of pseudo-relevant document set
S	size of pseudo-relevant document set in words

Table 10.2: Function set for QE approaches

Function	Description
$+, -, /, \times$	standard arithmetic operators
$log()$	the natural log
$square$	the square
$\sqrt{}$	the square-root

Global QE Terminal set

The terminal and function set for the global expansion approach is determined by considering the characteristics of the documents in which the query terms and possible expansion terms co-occur throughout the entire collection. It is also important to consider the characteristics of each query term and each possible expansion term in isolation. Table 10.3 shows the terminal set chosen. The set of documents in which both t (a possible expansion term) and q (a query term) co-occur is T_{qt}. This terminal set is quite large as there is a number of potential combinations of features that can be gathered regarding the terms in the documents in which potential expansion terms co-occur. Table 10.2 is also used as the function set for the global query

Table 10.3: Global QE terminal set

Terminal	Description
cf_q	frequency of a query term (q) in the collection
cf_t	frequency of a non-query term (t) in the collection
df_q	no. of documents in which query term (q) appears
df_t	no. of documents in which non-query term (t) appears in
N	no. of documents in a collection
S	no. of words in the collection
$\|Q\|$	no. of terms in the query Q
bin_{qt}	no. of documents in T_{qt}
$prod_{qt}$	sum of the product of the $tf's$ ($\sum_{D \in T_{qt}} tf_t^D \cdot tf_q^D$)
min_{qt}	sum of the minimum of the $tf's$ ($\sum_{D \in T_{qt}} min(tf_t^D, tf_q^D)$)
sum_{qt}	sum of the sum of the $tf's$ ($\sum_{D \in T_{qt}} tf_t^D + tf_q^D$)
cof_q	sum of the $tf's$ for q in T_{qt} ($\sum_{D \in T_{qt}} tf_q^D$)
cof_t	sum of the $tf's$ for t in T_{qt} ($\sum_{D \in T_{qt}} tf_t^D$)
W_{qt}	total no. of words in T_{qt}
1	*the constant* 1
0.5	*the constant* 0.5

expansion experiments.

10.2.4 Document Test Collections

Table 10.4: Characteristics of document collections for QE

Collection	# docs	words/doc	# queries	words/query
Medline	1,033	56.8	30	11
CISI	1,460	47.8	76 (112)	26.8
Cranfield	1,400	59.6	225	8.8
LISA	6,004	36.3	35	20.9
NPL	11,429	18.8	93	6.8
OHSU88	70,825	75.3	63	4.9
OHSU89	74,869	76.9	63	4.9
LATIMES$\frac{1}{2}$	65,138	250.8	44 (301-350)	9.9
FBIS$\frac{1}{2}$	61,578	257.2	36 (351-400)	7.9

Table 10.4 shows some of the characteristics of the document test collections[1] used in these experiments. The document collections are much smaller than those used in previous experiments due to the computational cost of conducting these query expansion experiments. The Medline collection is used as the training collection for both experimental approaches. The TREC collections denoted $\frac{1}{2}$ indicate that only half of the collection is used. Medium length queries (as used in previous experiments) are used with OHSU88, OHSU89, LATIMES$\frac{1}{2}$ and FBIS$\frac{1}{2}$.

10.2.5 GP Parameters

Local QE Approach Parameters

The local query expansion experiments are run for 100 generations with an initial random population of 1000. The solutions are trained on the entire Medline collection and query set. Trees are limited to a depth of 8 as the terminal set is larger than that previously used. When testing the evolved schemes after training, query terms can be selected and re-weighted like the other standard benchmark schemes.

Global QE Approach Parameters

The global query expansion experiments are run for 70 generations with an initial population of 2000. Populations of less than 500 for this problem tend to converge prematurely as the terminal set is quite large. Trees are limited to a depth of 10 simply because of the much larger terminal set for the global expansion problem. Many of the parameters have been chosen following preliminary experiments.

Common Parameters

For both approaches, an elitist strategy is used where the best performing individual is copied into the next generation. The tournament size is set to 4. The aim of the experiments is to discover general natural language

[1] http://www.dcs.gla.ac.uk/idom/ir_resources/test_collections/

characteristics for query expansion that will aid retrieval performance. As selected terms should be based on some normalised level, (for example from 0 to 1, indicating how much of the original BM25 weight should be assigned to the expanded term) original query terms are not added to the expanded query during training. The fitness function used for both the experiments is MAP and each experiment was run four times.

10.3 Experimental Results

This section outlines the experimental results for both approaches and includes a brief discussion of some of the characteristics of the best solutions.

10.3.1 Local QE Results

Figure 10.1 shows the best solution from each generation of the four runs of the GP. It can be seen that the best solutions from the randomly created solutions in the first generation are between 58% and 61% MAP. This is quite high compared to the initial retrieval run ($BM25$ shown in Table 10.5). This is because the terms selected from the top ranked documents come from documents that have a high similarity with the query. The following solution is the best evolved selection function (LSF_4 from run 4) after 4 runs of the GP on the Medline collection:

$$LSF_4 = \sqrt{\frac{(\frac{pcf}{V} \cdot log(pdf) \cdot pcf^2) + (\frac{P}{\sqrt{df}} \cdot log(pdf) \cdot log(pcf))}{log(\frac{P}{\sqrt{df}} \cdot log(log(pcf)) \cdot V)}} \qquad (10.5)$$

Table 10.5 shows the performance of the expanded queries using the best benchmark ($TSV\frac{1}{3}$) and the best evolved selection scheme (LSF_4). The column titled $BM25$ is the performance of the unexpanded query in the initial run. Two astericks (**) indicate that the MAP is significantly better than the benchmark expansion scheme ($TSV\frac{1}{3}$). The dagger (†) indicates that the MAP is significantly better than the unexpanded original query ($BM25$). It is encouraging that the MAP increases on many unseen document collections.

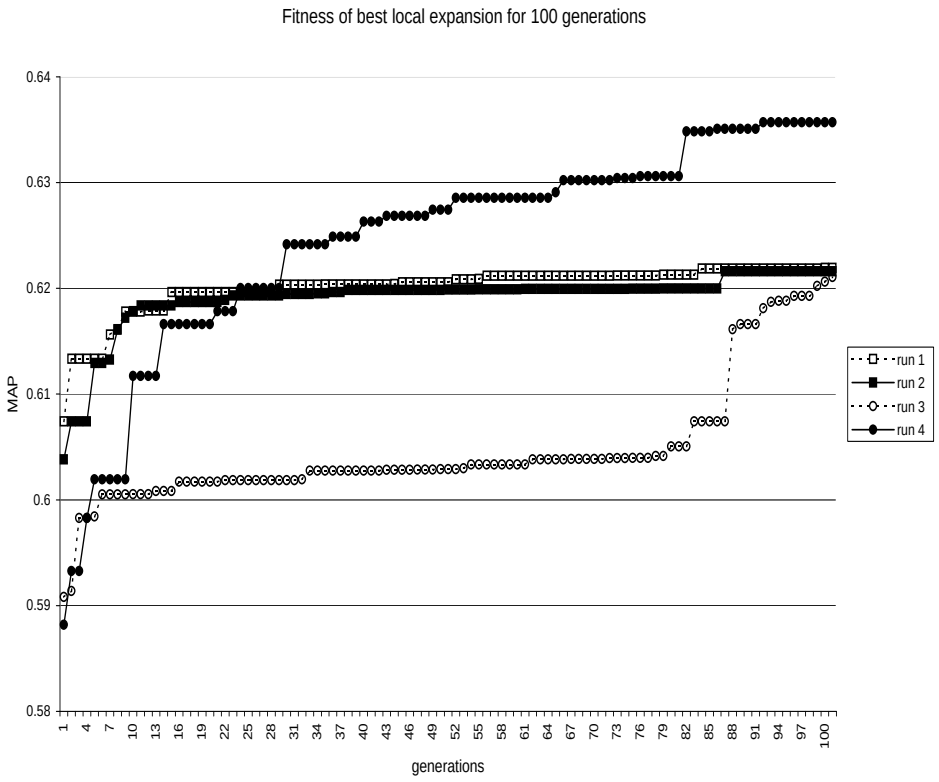

Figure 10.1: Increase in fitness for local QE approach during training

Table 10.5: %MAP for expanded queries using $TSV\frac{1}{3}$ and LSF_4

Collection	Qrys	$BM25$	$TSV\frac{1}{3}$	LSF_4
Medline	30	53.43	60.78†	64.20†**
CISI	76	23.08	24.41†	24.93†
Cranfield	225	42.23	43.90	43.38
LISA	35	35.00	38.14	36.95
NPL	93	28.75	28.62	28.77
OHSU88	63	32.78	36.61†	37.00†
OHSU89	63	30.69	31.30	33.98†**
LATIMES$\frac{1}{2}$	44	30.85	31.26	34.18**
FBIS$\frac{1}{2}$	36	22.09	22.67	25.10

Although the training set used is quite small, it can be seen that the term-selection properties for collections of various sizes are quite similar as the best scheme found on the training set increases the MAP on the larger collections.

Table 10.6: Scores for expansion terms for topic 301 on LATIMES$\frac{1}{2}$

Topic 301					Topic 301				
Terms	LSF_4	df_t	pcf_t	pdf_t	Terms	$TSV\frac{1}{3}$	df_t	pcf_t	pdf_t
anti-terror	0.44	12	3	2	activ	55.5	4916	29	10
organ	0.44	7035	47	10	organ	51.5	7035	47	10
hoodlum	0.38	23	3	2	crimin	22.4	2170	23	6
racket	0.38	341	18	3	repres	20.6	6615	10	7
fbi	0.37	812	25	4	includ	17.4	1811	19	8
rico	0.36	231	35	2	white-collar	17.4	94	3	3
crime	0.35	2681	36	5	justic	17.1	2053	15	5
activ	0.35	4916	29	10	cooper	16.9	2134	7	5
white-collar	0.34	94	3	3	terror	16.7	673	5	4
crimin	0.33	2170	23	6	civil	16.2	2427	16	5
mobster	0.33	67	5	2	act	16.1	5840	14	6
indict	0.30	779	15	3	fbi	16.0	812	25	4
law	0.30	6995	32	6	crime	15.7	2681	36	5
mafia	0.30	110	5	2	anti-terror	15.0	12	3	2
justic	0.30	2053	15	5	law	14.9	6995	32	6
mob	0.30	244	11	2	identifi	14.7	3259	5	5

Table 10.6 shows the top 16 terms added to query 301 for the LATIMES collection. The title of query 301 is "*International Organized Crime*" and the description is as follows: "*Identify organizations that participate in international criminal activity, the activity, and, if possible, collaborating organizations and the countries involved*". This is preprocessed to the following: "*intern organ crime identifi organ particip intern crimin activ activ collabor organ countri involv*". It is interesting to see that the GP evolves a scheme (LSF_4) which weights terms on a suitable scale for expansion. The first term selected ('*anti-terror*') is a non-query term and gets added to the query with 0.44 of its $BM25$ weight. The second term selected ('*organ*') appears in the original query three times and is also added with a weight of about 0.44, indi-

cating that its weight in the re-formulated query will be 3.44 times its $BM25$ weight. It is interesting to note that although many terms are common to the top 16 terms for both selection schemes, the ranking is different. As more terms are added to a query, the performance should plateau as the weight given to the expansion terms should be weighted correctly to reflect the usefulness of the term. This property is investigated next. It is also interesting that many of the terms are added to the query with an average weight of about 0.33 times the $BM25$ weight. This is similar to the benchmark scheme used here (Billerbeck et al, 2003).

Expanding Queries by More Terms

To test whether the evolved expansion scheme (LSF_4) correctly weights expansion terms, the scheme was tested by allowing various numbers of terms to the original query. The evolved scheme was originally evolved by adding the top 16 terms to each query on the Medline collection. Figure 10.2 shows both the evolved selection scheme and the best benchmark scheme ($TSV\frac{1}{3}$) for varing numbers of expansion terms on both the OHSU88 and OHSU89 collections. Queries with up to 48 expanded terms in multiples of 8 terms were evaluated.

Figure 10.2 shows that the evolved selection scheme is quite stable as more terms are added to the original query. The benchmark scheme is quite erratic for various numbers of terms. This would seem to indicate that the weighting assigned to these terms is not correct for the benchmark expansion scheme (i.e. bad expansion terms are getting an incorrectly high weight or good expansion terms are not getting a sufficiently high selection value). For the evolved selection value (LSF_4), it can be determined that a term occurring in only one pseudo-relevant document will get a zero weighting because of the $log(pdf)$ part of the numerator and in effect is not added to the query. Thus, the number of terms available for selection is limited to terms that occur in at least two pseudo-relevant documents. It is also interesting to note that the LSF_4 scheme will only select terms that occur more than three times in the set of pseudo-relevant documents. This is due

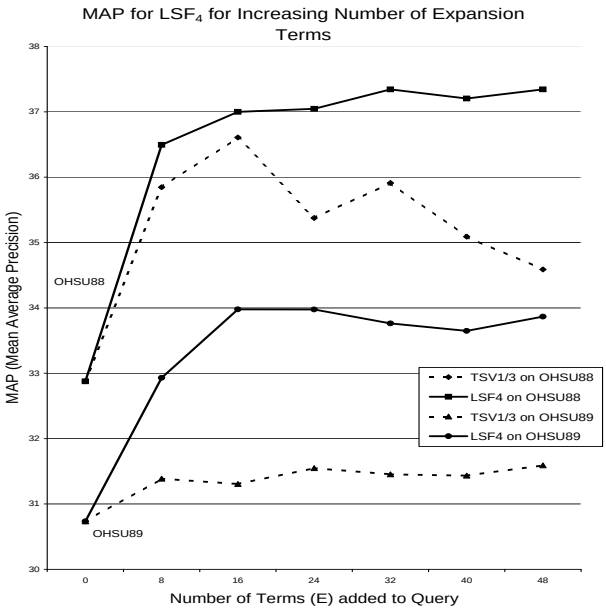

Figure 10.2: MAP for varying number of Terms ($|E|$)

to the $log(log(pcf))$ part of the denominator of the function. This leads to a very selective type of expansion as for some queries, only a small number of expansion terms meet the minimum criteria. Studies have shown that selective query expansion leads to more of an improvement when compared to massive query expansion (Robertson et al, 1995).

These characteristics have been learned when the number of pseudo-relevant documents chosen is 10 and may not be generalisable for many values of P. However, it has been demonstrated that by selecting 16 terms and also using the selection value to weight the term in the expanded query, a general and stable selection scheme can be learned. Furthermore, the GP has evolved a type of automatic thresholding into the weighting scheme that is different for each query and is dependent on the quality of expansion terms available to it.

10.3.2 Global QE Results

Figure 10.3 shows that the best solutions from the randomly created first generation achieve a MAP of between 56% and 57% on the training collec-

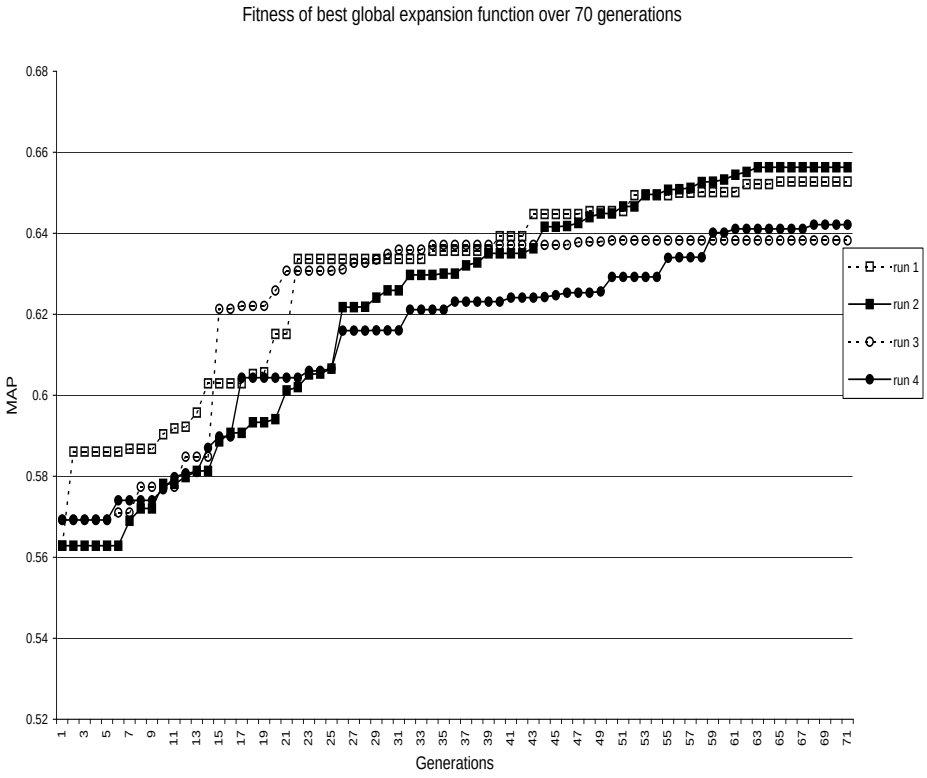

Figure 10.3: Increase in fitness for global QE approach during training

tion. This is poorer than those from the local approach because terms from the entire corpus are randomly being added to the query initially and will have little relation to the query. However, all four runs resulted in a better performance on the training set in the final generation because the possible terms were not limited to those from the top 10 documents of an initial run (i.e. there is a larger number of possible expansion terms to choose from). The following is the best formula from the 4 runs of the GP:

$$
\begin{aligned}
GSF_2 = \; & (((min)/((((((log((df_q^2)))) \cdot ((\sqrt{(log(bin))})/(((df_q^2))/ \\
& (log(cf_t \cdot S))))) + (((-((cof_t \cdot cf_t)/(log(sum))))/((\sqrt{(log(bin))}) \cdot \\
& (\sqrt{(\sqrt{S})}))) \cdot ((((df_q^2))/((\sqrt{N})/(log(|Q|)))) + (cof_q/(\sqrt{df_t}))))) + \\
& (-(((log(1)) \cdot ((log(sum)) \cdot ((df_q^2)))) + ((1 + W)/(((df_q^2))/ \\
& (log(cf_t \cdot S)))))))) \cdot cof_t)
\end{aligned}
$$

Table 10.7: %MAP for expanded queries using best evolved solution

Collection	Docs	Qrys	BM25	$\overline{cos}$	GSV_2
Medline	1,033	30	53.42	57.54	65.63**
CISI	1,460	76	23.08	23.17	23.56
Cranfield	1,400	225	42.23	42.40	38.53
LISA	6,004	35	35.00	34.90	32.23
NPL	11,429	93	28.75	29.28	25.84

Table 10.7 shows the MAP for the original query and the expanded queries on the smaller collections included in this research. The high computational cost of conducting the global based QE approach prohibited the use of the larger collection with the resources available. There is a significant increase in MAP on the Medline collection (i.e. the training set). This confirms previous concept-based approaches (Qiu and Frei, 1993) which also show a similar increase on this collection. However, this evolved scheme seems to be specific to that collection as there is no substantial improvement on any other collection. In general, this type of query expansion is not as effective as its local counterpart. For many queries the terms added reduce the performance. Even the benchmark scheme used does not signifcantly outperform the unexpanded queries ($BM25$).

Nonetheless, it is interesting that specific global expansion schemes can be learned using GP. The terms added to the queries for the evolved scheme seem to be of a similar topic on the training collection. For example, Table 10.8 shows the terms added to two queries from the Medline collection for the benchmark ($\overline{cos}$) and the best evolved scheme (GSF_2). The weights assigned to the top 8 most similar terms as determined by both solutions are also shown. The 21^{st} Medline query ("*language development in infancy and preschool age*") is preprocessed to the following: "*languag develop infanc preschool ag*". Its eight most similar terms, according to the evolved solution, are shown. Similarly, the terms added to the 23^{rd} query ("*infantile autism*"), which is preprocessed to "*infantil autism*", are also shown. It can be seen that the evolved scheme promotes terms that seem to be related to the query

Table 10.8: Scores for expansion terms for two sample Medline queries

Query 21

Terms	GSF_2	Terms	$\overline{cos}$
deaf	1.24	children	0.19
learn	1.03	infant	0.13
spoken	0.95	child	0.12
sentenc	0.77	speech	0.12
word	0.74	development	0.11
children	0.69	disord	0.11
teach	0.64	individu	0.11
impair	0.62	psycholog	0.11

Query 23

Terms	GSF_2	Terms	$\overline{cos}$
autist	3.24	autist	0.47
mental	2.75	mental	0.30
schizophrenia	2.01	child	0.29
contact	1.55	children	0.26
child	1.40	schizophrenia	0.23
situat	1.37	ego	0.23
innat	1.34	emot	0.22
psychot	1.21	psychot	0.22

concept. It also provides a weighting that is related to the quality of the expansion term. It can promote different forms of query terms that the stemming algorithm has failed to conflate (e.g. "*autism*" and "*autist*" have the same stem). The benchmark scheme ($\overline{cos}$) shows the average cosine correlation between the expansion term and the set of query terms. However, although solutions can be evolved that correctly find good expansion terms for a query, these solutions seem to be specific. Many global approaches have failed to achieve an adequate level of performance because the co-occurrence relationship is defined at a document level. It has been suggested that co-occurrence should be determined at a closer proximity (i.e. at paragraph or sentence level). It can also be seen that the solution evolved is very long. This could also be a reason for its apparent overtraining on the Medline

collection.

10.4 Summary

Two approaches that attempt to overcome the problem of term-mismatch in IR have been outlined. GP has been used to develop functions that automatically select and weight terms for use in these two approaches. Local query expansion schemes have been evolved that outperform standard benchmarks and importantly increase the performance of many queries on some of the larger test collections. The best evolved scheme correctly weights expansion terms for use in the re-formulated query. The best scheme also evolved an automatic thresholding technique to limit the number of potential expansion terms available.

The global approach to query expansion is less successful in terms of generalisation (performance on unseen test data), although specific expansion formulas can be learned that significantly increase the performance on the training data. Overall the global approach adopted in these experiments is not as successful as the local QE approach. However, the global approach does select terms that are similar to the topic of the query on the training data and applies a useful weight to these terms.

Chapter 11

Conclusion

This work has examined an evolutionary based approach to learning term-weighting schemes in IR. The work has demonstrated a number of experiments and presented results that validate the learning framework adopted. This work contributes to a number of areas in GP and IR. This concluding chapter details these contributions as well as outlining a number of avenues for future work.

Term-Weighting Function Performance

This work has outlined a structured method of incrementally learning term-weighting schemes in IR. This has been shown to be both theoretically and empirically valid. The framework has proven useful in analysing and determining useful components in term-weighting schemes. A number of new term-discrimination schemes have been developed and are shown to significantly outperform *idf* type solutions. The better schemes adopted contain aspects which adhere to Luhn's theory of resolving power. The new solutions are better estimators of the semantic content of a term and are likely to be useful in other areas in information science.

The term-frequency schemes developed, dependent on the best term-discrimination scheme, are shown to be less influential in the weighting process when compared to the term-frequency schemes used in the benchmarks. This is somewhat unexpected although it may be surmised that it is because

of the new density measure (cf/df) contained in the global weighting. This density measure can be thought of as the average term-frequency of a term in the documents which contain the term. This leads to an estimation of term-frequency in the global weighting component. Therefore, it may be expected that the term-frequency influence factor may be reduced.

The entire term-weighting function is completed by adding a normalisation scheme. Many factors that affect normalisation have been analysed and the length of the query has been shown to affect normalisation for a specific type of scheme (linear). It has been determined that it is the length of the query that affects the distribution of document lengths in the returned set of documents. This difference in distribution affects the normalisation tuning parameter of linear shaped functions. This is an interesting finding. The completed scheme compares favorably to the benchmarks without the need of tuning parameters. In fact, it often outperforms a tuned version of the best benchmark ($BM25$) without modification on unseen test data. The first hypothesis [*H1*] has been proven as the term-weighting schemes developed significantly outperform $BM25$ on much of the data presented. Moreover, it can be seen that there is a number of further contributions in each part of the three component parts of the term-weighting framework. This further validates the three stage learning process adopted.

Analysis of the Phenotypes

GP has been shown to be a useful and novel method of searching for useful term-weighting schemes. An analysis of the solution spaces of the three function types has shown that the better solutions tend to be clustered together. This supports the theory that the best term-weighting schemes are closely clustered in the framework adopted and that these are indeed different than the current benchmarks. In the term-discrimination function space, four of the seven final solutions contained the cf/df factor and these solutions' phenotypes were clustered closer to each other that the idf type solutions.

In the term-frequency function space, five of the seven final solutions had a similar term-frequency influence. This influence is lower than either of the

benchmarks and again these solution are shown to return similar relevant documents.

In the normalisation function space, six of the seven methods of normalisation developed were sub-linear with respect to the document length factor used. The same scheme was effectively evolved a number of times for normalisation. It can therefore be concluded that the second hypothesis [*H2*] is true and that the GP can guide the search to similar useful areas of the search space for the GP parameters outlined in this work.

Theoretical Validity of the Genotypes

The newly developed term-weighting schemes are shown to satisfy a number of axioms developed for IR. A new axiom that is satisfied by many of the evolved normalisation schemes is theoretically and empirically validated. A comparison of previously learned schemes shows that the incremental process aids adherence to these axioms, although importantly does not force this. Interestingly, all four main axioms cannot be adhered to unconditionally by modern term-weighting schemes. A solution to this phenomenon is also presented in Appendix A. A term-weighting approach which unconditionally adheres to all constraints can be created although it may be inefficient. Furthermore the correlation between function performance and constraint satisfaction has been reinforced by this research.

It can be concluded that the incremental approach adopted aids in the adherence to the axioms, as the term-weighting schemes evolved in other GP approaches do not adhere to as many axioms. It is also shown that the incrementally evolved solution adheres to more axioms that any of the analytically developed term-weighting benchmarks. Therefore, the third hypothesis [*H3*] has been proven true as it can be seen that the evolved term-weighting schemes are theoretically valid.

Automatic Query Expansion

This work has presented two ways of developing schemes which learn to select terms for use in an expanded query. Both approaches adopted are shown to

increase performance over standard benchmark methods on the training data. For the local QE approach a useful and novel term-selection scheme has been evolved that contains some interesting thresholding properties. The scheme evolved for this local QE approach increases performance over the standard benchmark on a number of unseen collections.

However, on general unseen test data the global QE schemes are not shown to improve upon the standard benchmarks. In this case the final hypothesis [*H4*] has not been proven to be true for the global QE part of this study. As the training collections used in this piece of research are quite small, it may not be possible to learn suitable schemes for term-selection on these collections. It could also be that the benchmark selection schemes are close to optimal given the features used in the global QE approach and that more evidence or more refined features may be needed to further improve performance.

However, the analysis conducted regarding the form of these term-selection schemes is quite brief and a more rigorous analysis would indeed be required to claim that these evolved QE approaches are superior to the benchmarks used.

11.1 Future Work

From an application point of view, it would be useful to determine if the term-discrimination (global) schemes developed are useful in other areas of information science. Methods of feature extraction and measures of information content are used extensively throughout the domain. Document clustering methods and document classification problems are those most likely to benefit from these types of scheme. Due to the way the term-weighting schemes have been developed (i.e. incrementally), a scheme (which ignores normalisation) can be adopted and used for text classification or document clustering, similar to the way simple *tf-idf* schemes have been used. In these text classification and document clustering methods, the documents are typically compared to each other in a vector framework using a term-weighting approach. However, as the documents are typically longer than queries and

are often of similar length, the global and term-frequency methods outlined in this work could be very useful in such areas. Length aspects (normalisation) are often ignored in such methods as basic non-tunable features are often more useful. Moreover, the schemes developed in this work have been shown to model relevance and similarity more consistently on various length documents and queries without the need for tuning.

It has been reinforced that there is a correlation between function performance and constraint satisfaction. The fact that all of the axioms are so rarely adhered to unconditionally may be useful. By developing an inductive model that counts the number of constraint violations on actual document collections, a test collection and set of queries may be mathematically shown to violate a certain number of constraints for a given term-weighting scheme. A numerical score corresponding to the number of times a document (and by extension a test collection) violates the constraints for a given set of queries may be used as a numerical score to test the optimality and relative performance of a term-weighting scheme. This may be a useful tool in measuring the performance of such schemes because relevance judgments are not used in the process.

As the inductive axiomatic framework may better model the human determination of relevance, it would be interesting to further explore this area. The inductive approach seems to mirror the human experience of reading a document. A term-weighting scheme which unconditionally adheres to all constraints can be created (as outlined in Appendix A). Although it may be inefficient, it would be interesting to create such a scheme and test it to further validate, extend and advocate the axiomatic approach based in this inductive framework.

It would be interesting to apply GP to other problems areas in IR and information science. Some limited work regarding query expansion has been conducted here but much more can be accomplished in this area. Furthermore, the query expansion schemes developed in the latter stages of this work may be validated by the axiomatic approach to semantic term matching (Fang and Zhai, 2006) in a similar manner to the term-weighting analysis. This would neatly fit with the work completed to date.

Appendix A

On Violations and Satisfaction of Constraints

A.1 Violations of Constraints 1 and 3

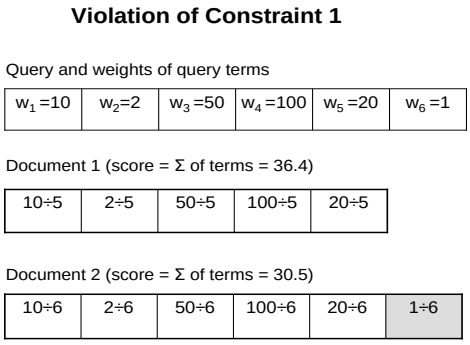

Figure A.1: Violation of constraint 1

Due to the type of the normalisation schemes used in modern term-weighting functions, when a term-weighting scheme uses the document length explicitly to penalise the document, constraint 1 (and consequently constraint 3) can *never* be satisfied unconditionally. Consider the case where a term with an extremely low *idf* value (i.e. where the term has negligible semantic

content) is added to a document. The penalisation due to the document increasing in length will more than offset the increase in weight as the term is added (as all existing terms in the document are penalised by the document length accordingly). For the $BM25$ scheme this will only tend to happen for terms with a very low *idf* value (terms that appear in close to half of the documents).

Figure A.1 shows a set of query terms with some basic weights applied to them. Document 1 contains 5 of the query terms while document 2 contains 6 of the query terms. The normalisation part used in the example is simply the document length. The normalisation (division by the document length) reduces the weight of *all* of the existing terms in the document and therefore, the score of a document may not increase as a query term is added. In the example shown, the score of document 1 is calculated by summing up the scores of the 5 query terms (36.4). The score of document 2 is calculated similarly (summing up the 6 query terms). As the query term added to document 2 has a very low term-discrimination weight ($w_6 = 1$) compared to the other query terms, the increase in weight due to this query term being added does not offset the increase in penalisation. The score of document 2 is only 30.5, although document 2 is created by adding a query term to document 1. However, the potential for violations of the type just described may be more prevalent in different types of term-weighting schemes. It is worth noting for the discussion presented in this section that the efficiency of these types of schemes is $O(N \times |Q|)$ where $|Q|$ is the length of the query vector (usually less than 20 for even the longest queries, but often only 2 or 3 for shorter queries) and N is the number of documents in the collection. This is because only query terms appearing in the document are used to determine the score of a document. This efficiency is important as test collections are becoming extremely large.

A.1.1 Choosing Normalisation

The document length is typically used explicitly to penalise documents. $S1(Q, D)$ and $S2(Q, D)$ describe two possible ways of normalising a doc-

ument used in modern weighting schemes.

$$S1(Q,D) = \sum_{t \in Q \cap D} (\frac{tff()}{n()} \cdot w(t)) \tag{A.1}$$

where $w(t)$ is the term-discrimination aspect, $tff()$ is the term-frequency aspect and $n()$ is some normalisation aspect. Other functions, such as the $BM25$ scheme, penalise the actual term-frequency as follows:

$$S2(Q,D) = \sum_{t \in Q \cap D} (tff(\frac{tf_t^D}{n()}) \cdot w(t)) \tag{A.2}$$

Both of these approaches to normalisation lead to the same potential violations of constraints 1 and 3. Furthermore, the first method of normalisation presented ($S1(Q,D)$ in equation A.1) violates constraints 1 and 3 for more reasons. As the normalisation ($n()$) is independent of the term-frequency influence ($tff()$), it may grow to such a degree that the penalisation more than outweighs the increase in weight that the term-frequency provides.

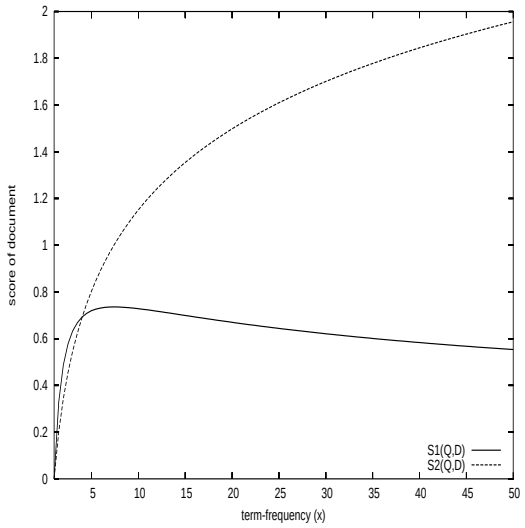

Figure A.2: Change in score for different methods of normalisation

Consider these two somewhat similar methods of applying normalisation (i.e. $S1(Q,D)$ and $S2(Q,D)$) from an inductive perspective. Now, let x define the term-frequency for a query term. Consider a document that is

made up of successive occurrences of this term. In such a case, x also defines the document length. Let $log(x)$ be the term-frequency factor and $\sqrt{x}$ be the normalisation aspect. In isolation they would appear to adhere to the aforementioned constraints (i.e. the term-frequency is sub-linear and the normalisation is sub-linear). Figure A.2 shows that $S1(Q, D)$ (i.e. $log(x)/\sqrt{x}$) does not always increase for successive occurrences of query-terms. $S2(Q, D)$ does adhere to this constraint in the simplest inductive case. Thus, when normalisation is explicitly used to penalise the document score, it should be applied to the actual term-frequency as $S2(Q, D)$ (i.e. $log(x/\sqrt{x})$) to help satisfy constraint 1 for the simplest inductive case.

A.2 Satisfaction

The problem identified (potential violations of constraints 1 and 3) in section A.1 can be overcome by penalising documents in an similar manner to how documents are promoted when query terms occur. In the following solution, the document length is not used explicitly to penalise a document, but when a term is found which does not occur in the query, the document is penalised as follows:

$$S3(Q, D) = \sum_{t \in D} \begin{cases} tff() \cdot w(t) & \text{if } t \in Q \\ -b \cdot tff() \cdot w(t) & \text{if } t \notin Q \end{cases}$$

where b is some constant factor. In this formulation it can be seen that the document length is not used explicitly to penalise the document (i.e. there is no $n()$ function used). As non query terms occur, a weight is substracted from the overall score. Furthermore, this penalisation can be different for different types of non-query terms. In this framework, normalisation is implicit in the weighting scheme, and not explicit, as is usually the case. This type of weighting scheme has previously been explored (Jung et al, 2000). As no summary description of the document length is explicitly used, this may lead to better normalisation and subsequent retrieval (Jung et al, 2000). Just as important words are more heavily weighted, words of a high term-

discrimination value that are not in the query lead to a higher penalisation as they indicate that the topic of the document varies considerably (i.e. may relate to many other subjects). This type of normalisation deals with each term separately and as such deals with the semantic content of the entire document and not just of the query (via its query terms).

Consider two documents (D_1 and D_2) that match a similar number of query terms and are of similar length. If D_1 contains non-query terms that are of negligible semantic content (low term-discrimination) and D_2 contains non-query words that have a high term-discrimination, it may be better to rank D_1 higher than D_2 as its topic is not as broad (i.e. the subject of D_1 is not associated to as much off-topic material as D_2). As such it is probably more useful to the user. This intuitively seems like a desirable property in retrieval. With such a formulation it is easy to see that constraint 1 and constraint 2 are adhered to. Consequently, it must adhere to the weaker constraint 1.1. As more of the same non-query terms are added, the increase in penalisation is also reduced. This approach, however, is less efficient as the entire document vector must be examined, instead of the much shorter query vector. The efficiency of this scheme is $O(N \times |D|)$ where $|D|$ is the length of the document vector (on average this is about 150 for the collections used in the experiments described herein). Thus, there is a trade-off between unconditionally satisfying constraints 1 and 3 and the efficiency of the approach adopted.

Appendix B

Further Evaluation

B.1 Precision-Recall Curves

The following figures shows the 11-point precision-recall curves for the default BM25 scheme and the entire evolved scheme as outlined in chapter 7 ($S(Q, D)$). The figures show the precision for all query types (short, medium and long) for each test collection.

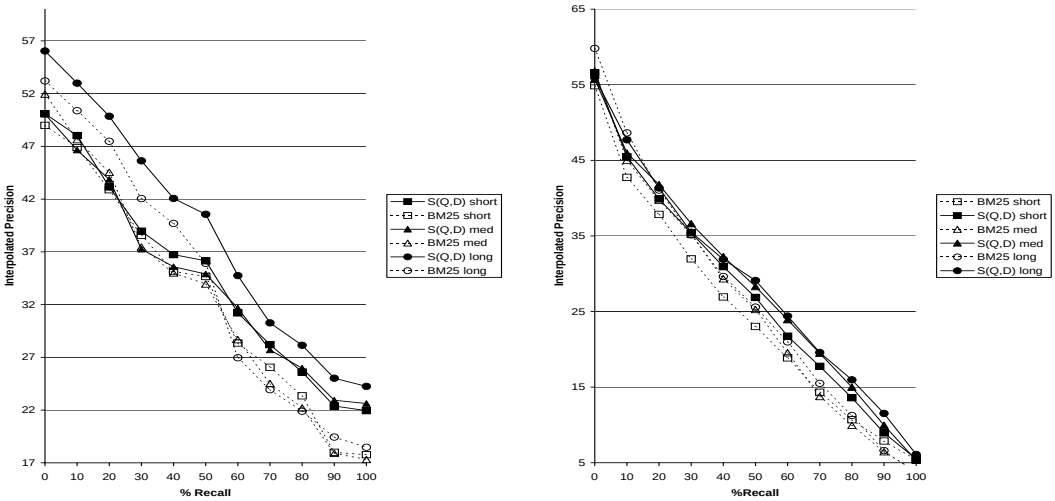

Figure B.1: Precision-Recall for on FR and FBIS respectively

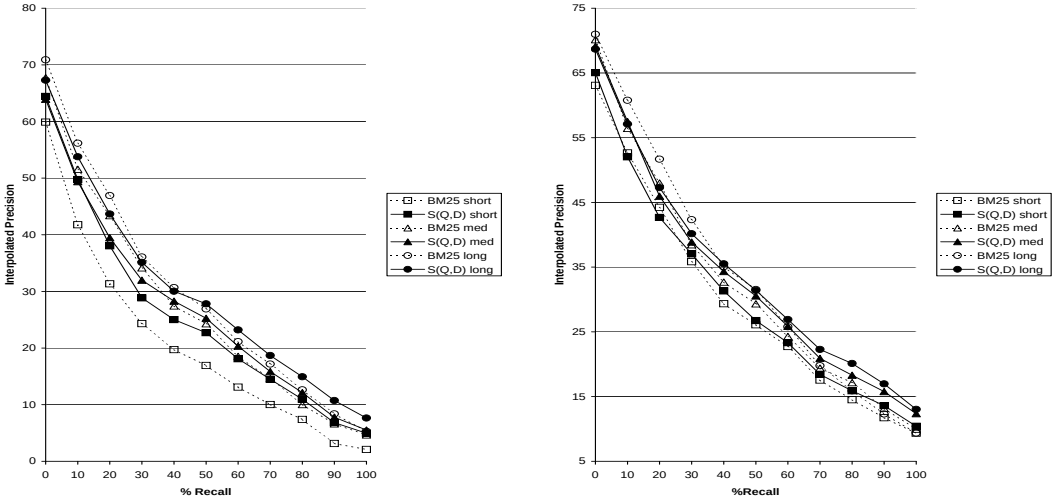

Figure B.2: Precision-Recall for on LATIMES and FT respectively

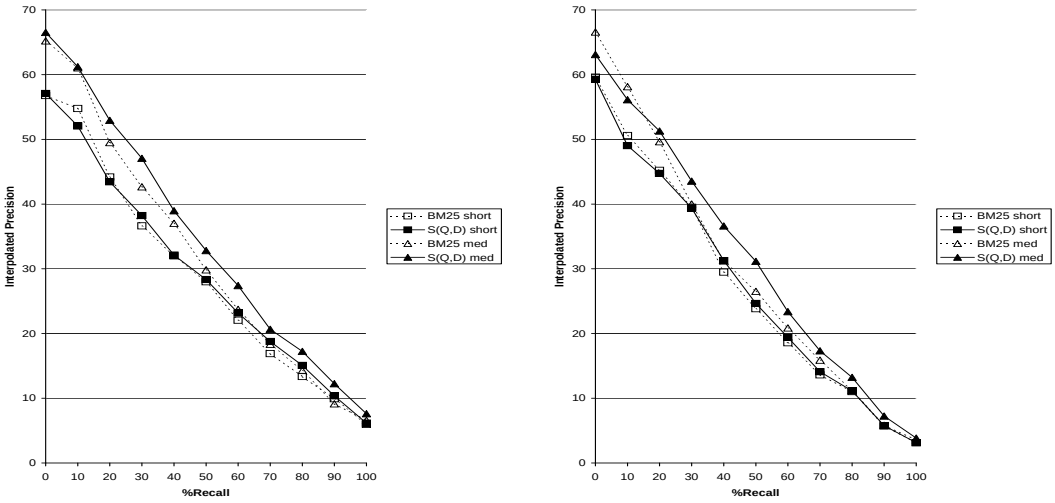

Figure B.3: Precision-Recall for on OH89 and OH90-91 respectively

Bibliography

Almeida de HM, Gonçalves MA, Cristo M, Calado P (2007) A combined component approach for finding collection-adapted ranking functions based on genetic programming. In: SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press, Amsterdam, The Netherlands, pp 399–406

Amati G, Rijsbergen CJV (2002) Probabilistic models of information retrieval based on measuring the divergence from randomness. ACM Trans Inf Syst 20(4):357–389

Baeza-Yates RA, Ribeiro-Neto BA (1999) Modern Information Retrieval. ACM Press / Addison-Wesley

Bartell BT, Cottrell GW, Belew RK (1994a) Automatic combination of multiple ranked retrieval systems. In: SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval, Springer-Verlag New York, Inc., Dublin, Ireland, pp 173–181

Bartell BT, Cottrell GW, Belew RK (1994b) Learning the optimal parameters in a ranked retrieval system using multi-query relevance feedback. In: Symposium on Document Analysis and Information Retrieval, Las Vegas

Belew RK (1989) Adaptive information retrieval: using a connectionist representation to retrieve and learn about documents. In: SIGIR '89: Proceedings of the 12th annual international ACM SIGIR conference on Research

and development in information retrieval, ACM Press, Cambridge, Massachusetts, United States, pp 11–20

Billerbeck B, Zobel J (2003) When query expansion fails. In: SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, ACM Press, Toronto, Canada, pp 387–388

Billerbeck B, Scholer F, Williams HE, Zobel J (2003) Query expansion using associated queries. In: CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management, ACM Press, New Orleans, LA, USA, pp 2–9

Blair DC, Maron ME (1985) An evaluation of retrieval effectiveness for a full-text document-retrieval system. Commun ACM 28(3):289–299

Buckley C, Voorhees EM (2000) Evaluating evaluation measure stability. In: SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press, Athens, Greece, pp 33–40

Cao Y, Xu J, Liu TY, Li H, Huang Y, Hon HW (2006) Adapting ranking svm to document retrieval. In: SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press, Seattle, Washington, USA, pp 186–193

Carterette B, Allan J (2005) Incremental test collections. In: CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management, ACM Press, Bremen, Germany, pp 680–687

Chisholm E, Kolda TG (1999) New term weighting formulas for the vector space method in information retrieval. Tech. rep., Oak Ridge National Laboratory

Choi JH, Jung HY, Kim HS, Cho HG (2000) Phylodraw: a phylogenetic tree drawing system. Bioinformatics 16(11):1056–1058

Chowdhury A, McCabe MC, Grossman D, Frieder O (2002) Document normalization revisited. In: SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press, Tampere, Finland, pp 381–382

Chung TL, Luk RWP, Wong KF, Kwok KL, Lee DL (2006) Adapting pivoted document-length normalization for query size: Experiments in chinese and english. ACM Transactions on Asian Language Information Processing (TALIP) 5(3):245–263

Cordón O, Herrera-Viedma E, Luque M (2002) Evolutionary learning of boolean queries by multiobjective genetic programming. In: PPSN VII: Proceedings of the 7th International Conference on Parallel Problem Solving from Nature, Springer-Verlag, London, UK, pp 710–719

Cordón O, Herrera-Viedma E, Lpez-Pujalte C, Luque M, Zarco C (2003) A review of the application of evolutionary computation to information retrieval. International Journal of Approximate Reasoning 34:241–264

Cummins R, O'Riordan C (2005) An evaluation of evolved term-weighting schemes in information retrieval. In: Herzog O, Schek HJ, Fuhr N, Chowdhury A, Teiken W (eds) CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management, ACM press, Bremen, Germany, pp 305–306

Cummins R, O'Riordan C (2006a) Evolving local and global weighting schemes in information retrieval. Information Retrieval 9(3):311–330

Cummins R, O'Riordan C (2006b) A framework for the study of evolved term-weighting schemes in information retrieval. In: Stein B, Kao O (eds) TIR-06 Text based Information Retrieval, Workshop. ECAI 2006, Riva del Garda, Italy

Darwin C (1859) The Origin of the Species by means of Natural Selection, or The Preservation of Favoured Races in the Struggle for Life. Penguin Books, London, UK

Deerwester SC, Dumais ST, Landauer TK, Furnas GW, Harshman RA (1990) Indexing by latent semantic analysis. Journal of the American Society of Information Science 41(6):391–407

Fan W, Gordon MD, Pathak P (2004) A generic ranking function discovery framework by genetic programming for information retrieval. Inf Process Manage 40(4):587–602, DOI http://dx.doi.org/10.1016/j.ipm.2003.08.001

Fang H (2007) An axiomatic approach to information retrieval. PhD thesis, University of Illinois at Urbana-Champaign, Urbana-Champaign, Illinois, USA

Fang H, Zhai C (2005) An exploration of axiomatic approaches to information retrieval. In: SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press, Salvador, Brazil, pp 480–487

Fang H, Zhai C (2006) Semantic term matching in axiomatic approaches to information retrieval. In: SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, ACM, Seattle, Washington, USA, pp 115–122

Fang H, Tao T, Zhai C (2004) A formal study of information retrieval heuristics. In: SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press, Sheffield, United Kingdom, pp 49–56

Fogel LJ, Owens AJ, Walsh MJ (1966) Artificial Intelligence through Simulated Evolution. John Wiley, New York, USA

Franz M, McCarley JS (2000) Word document density and relevance scoring. In: SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference, ACM Press, Athens, Greece, pp 345–347

Gordon M (1988) Probabilistic and genetic algorithms in document retrieval. Commun ACM 31(10):1208–1218

Greiff W (1998) A theory of term weighting based on exploratory data analysis. In: Proceedings of the 21st International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '98), Melbourne, Australia

Gustafson S (2004) An analysis of diversity in genetic programming. PhD thesis, School of Computer Science and Information Technology, University of Nottingham, Nottingham, England

Hancock-Beaulieu M, Gatford M, Huang X, Robertson SE, Walker S, Williams PW (1996) Okapi at trec-5. In: Voorhees EM, Harman DK (eds) Proceedings of the 5th Text REtrieval Conference (TREC-5), NIST

Harman D (1993) Overview of the first TREC conference. In: SIGIR '93: Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press, Pittsburgh, Pennsylvania, United States, pp 36–47

He B, Ounis I (2003) A study of parameter tuning for term frequency normalization. In: CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management, ACM Press, New Orleans, LA, USA, pp 10–16

He B, Ounis I (2005a) A study of the dirichlet priors for term frequency normalisation. In: SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press, Salvador, Brazil, pp 465–471

He B, Ounis I (2005b) Term frequency normalisation tuning for bm25 and dfr models. In: ECIR, pp 200–214

Heaps HS (1978) Information Retrieval: Computational and Theoretical Aspects. Academic Press, Inc., Orlando, FL, USA

Hersh W, Buckley C, Leone TJ, Hickam D (1994) Ohsumed: an interactive retrieval evaluation and new large test collection for research. In: Proceedings of the 17th annual international ACM SIGIR conference on Research

and development in information retrieval, Springer-Verlag New York, Inc., Dublin, Ireland, pp 192–201

Hull D (1993) Using statistical testing in the evaluation of retrieval experiments. In: SIGIR '93: Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press, Pittsburgh, Pennsylvania, United States, pp 329–338

Jen-Yuan Yeh HRK Jung-Yi Lin, Yang WP (2007) Learning to rank for information retrieval using genetic programming. In: SIGIR'07 workshop on learning to rank for information retrieval (LR4IR-2007), pp 11–18

Joachims T (2002) Optimizing search engines using clickthrough data. In: KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM Press, Edmonton, Alberta, Canada, pp 133–142

Jones KS, Walker S, Robertson SE (2000) A probabilistic model of information retrieval: development and comparative experiments - part 2. Information Processing and Management 36(6):809–840

Jung Y, Park H, Du D (2000) A balanced term-weighting scheme for effective document matching. Tech. Rep. TR008, Department of Computer Science, University of Minnesota, Minneapolis, Minnesota

Koza JR (1992) Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge, MA, USA

Kraft DH, Petry FE, Buckles WP, Sadasivan T (1994) The use of genetic programming to build queries for information retrieval. In: Proceedings of the 1994 IEEE World Congress on Computational Intelligence, IEEE Press, Orlando, Florida, USA, pp 468–473

Kwok JT (1998) Automated text categorization using support vector machine. In: Proceedings of ICONIP'98, 5th International Conference on Neural Information Processing, Kitakyushu, JP, pp 347–351

Kwok KL (1995) A network approach to probabilistic information retrieval. ACM Trans Inf Syst 13(3):324–353

Kwok KL (1996) A new method of weighting query terms for ad-hoc retrieval. In: Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press, Zurich, Switzerland, pp 187–195

Leopold E, Kindermann J (2002) Text categorization with support vector machines. how to represent texts in input space? Mach Learn 46(1-3):423–444

Losee RM (2000) When information retrieval measures agree about the relative quality of document rankings. Journal of the American Society of Information Science 51(9):834–840

Lucas JM, van Baronaigien DR, Ruskey F (1993) On rotations and the generation of binary trees. J Algorithms 15(3):343–366

Luhn H (1958) The automatic creation of literature abstracts. IBM Journal of Research and Development pp 159–165

Luke S (2001) When short runs beat long runs. In: Spector L, Goodman ED, Wu A, Langdon WB, Voigt HM, Gen M, Sen S, Dorigo M, Pezeshk S, Garzon MH, Burke E (eds) Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001), Morgan Kaufmann, San Francisco, California, USA, pp 74–80

Mitra M, Singhal A, Buckley C (1998) Improving automatic query expansion. In: Research and Development in Information Retrieval, pp 206–214

Oren N (2002a) Improving the effectiveness of informtation retrival with genetic programming. Master's thesis, Faculty of Science, University of the Witwatersrand, South Africa

Oren N (2002b) Reexamining tf.idf based information retrieval with genetic programming. In: SAICSIT '02: Proceedings of the 2002 annual research

conference of the South African institute of computer scientists and information technologists on Enablement through technology, South African Institute for Computer Scientists and Information Technologists, Port Elizabeth, South Africa, pp 224–234

Owais SSJ, Kromer P, Snasel V (2005) Evolutionary learning of Boolean queries by genetic programming. In: Eder J, Haav HM, Kalja A, Penjam J (eds) ADBIS 2005, Advances in Databases and Information Systems, Tallinn, Estonia, vol 152

Ponte JM, Croft WB (1998) A language modeling approach to information retrieval. In: SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press, Melbourne, Australia, pp 275–281

Porter M (1980) An algorithm for suffix stripping. Program 14(3):130–137

Qiu Y, Frei HP (1993) Concept-based query expansion. In: Proceedings of SIGIR-93, 16th ACM International Conference on Research and Development in Information Retrieval, Pittsburgh, US, pp 160–169

Rijsbergen CJV (1979) Information Retrieval. Butterworth-Heinemann, Newton, MA, USA

Robertson S, Zaragoza H, Taylor M (2004) Simple bm25 extension to multiple weighted fields. In: CIKM '04: Proceedings of the thirteenth ACM international conference on Information and knowledge management, ACM, Washington, D.C., USA, pp 42–49

Robertson SE (1977) The probability ranking principle in ir. Journal of Documentation pp 294–304

Robertson SE, Sparck Jones K (1976) Relevance weighting of search terms. Journal of the American Society for Information Science 27(3):129–146

Robertson SE, Walker S (1999) Okapi/keenbow at trec-8. In: Voorhees EM, Harman DK (eds) Proceedings of the 8th Text REtrieval Conference (TREC-8), NIST, pp 151–162

Robertson SE, Walker S, Hancock-Beaulieu M, Gull A, Lau M (1995) Okapi at TREC-3. In: Voorhees EM, Harman DK (eds) Proceedings of the 3rd Text REtrieval Conference (TREC-3), NIST, pp 21–30

Roussinov D, Fan W, Neves FAD (2005) Discretization based learning approach to information retrieval. In: CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management, ACM, Bremen, Germany, pp 321–322

Ruthven I, Lalmas M (2003) A survey on the use of relevance feedback for information access systems. Knowl Eng Rev 18(2):95–145

Salton G, Buckley C (1988) Term-weighting approaches in automatic text retrieval. Information Processing & Management 24(5):513–523

Salton G, Wong A, Yang CS (1975) A vector space model for automatic indexing. Commun ACM 18(11):613–620

Salton G, Fox EA, Wu H (1983) Extended boolean information retrieval. Commun ACM 26(11):1022–1036

Sanderson M, Zobel J (2005) Information retrieval system evaluation: effort, sensitivity, and reliability. In: SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press, Salvador, Brazil, pp 162–169

Saracevic T (1997) Relevance: a review of and a framework for the thinking on the notion in information science. Readings in information retrieval pp 143–165

Singhal A (2001) Modern information retrieval: A brief overview. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering 24(4):35–43

Singhal A, Buckley C, Mitra M (1996) Pivoted document length normalization. In: SIGIR '96: Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press, Zurich, Switzerland, pp 21–29

Smith MP, Smith M (1997) The use of genetic programming to build boolean queries for text retrieval through relevance feedback. Journal of Information Science 23(6):423–431

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

Steele R, Powers D (1998) Evolution and evaluation of document retrieval queries. In: Powers DMW (ed) NeMLaP3/CoNLL98: New Methods in Language Processing and Computational Natural Language Learning, ACL Association for Computational Linguistics, Flinders University, Adelaide, Australia, pp 163–164

Trotman A (2005) Learning to rank. Information Retrieval 8:359 – 381

Voorhees EM, Harman D (2000) Overview of the eighth text REtrieval conference (TREC-8)

Vrajitoru D (1999) Genetic programming operators applied to genetic algorithms. In: Banzhaf W, Daida J, Eiben AE, Garzon MH, Honavar V, Jakiela M, Smith RE (eds) Proceedings of the Genetic and Evolutionary Computation Conference, Morgan Kaufmann, Orlando, Florida, USA, vol 1, pp 686–693

Vrajitoru D (2000) Large Population or Many Generations for Genetic Algorithms ? Implications in Information Retrieval, Physica-Verlag, pp 199–222

Walker S, Robertson SE, Boughanem M, Jones GJF, Jones KS (1997) Okapi at TREC-6 automatic ad hoc, VLC, routing, filtering and QSDR. In: Voorhees EM, Harman DK (eds) Proceedings of the 6th Text REtrieval Conference (TREC-6), NIST Special Publication 500-240, pp 125–136

Whitley D (2001) An overview of evolutionary algorithms: practical issues and common pitfalls. Information and Software Technology 43(14):817–831

Yu CT, Lee TC (1986) Non-binary independence model. In: SIGIR '86: Proceedings of the 9th annual international ACM SIGIR conference on Research and development in information retrieval, ACM, Palazzo dei Congressi, Pisa, Italy, pp 265–268

Yu CT, Lam K, Salton G (1982) Term weighting in information retrieval using the term precision model. Journal of the ACM (JACM) 29(1):152–170

Yue Y, Finley T, Radlinski F, Joachims T (2007) A support vector method for optimizing average precision. In: SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press, Amsterdam, The Netherlands, pp 271–278

Zhai C, Lafferty J (2001) A study of smoothing methods for language models applied to ad hoc information retrieval. In: SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, ACM, New Orleans, Louisiana, United States, pp 334–342

Zhang J, Nguyen TN (2005) A new term significance weighting approach. J Intell Inf Syst 24(1):61–85

Zipf GK (1949) Human Behavior and the Principle of Least Effort. Addison-Wesley (Reading MA)

Zobel J, Moffat A (1998) Exploring the similarity space. SIGIR Forum 32(1):18–34