Universidad Autónoma de Madrid
Escuela Politécnica Superior
Departamento de Ingeniería Informática

Performance prediction in recommender systems: application to the dynamic optimisation of aggregative methods

Alejandro Bellogín Kouki
Supervisor: Pablo Castells Azpilicueta

Trabajo Fin de Máster

Programa Oficial de Postgrado en Ingeniería Informática y de Telecomunicación.
Universidad Autónoma de Madrid
Junio 2009

Abstract

Performance prediction has gained increasing attention in the Information Retrieval (IR) field since the half of the past decade and has now become an established research topic in the field. Predicting the performance of an IR system, subsystem, module, function, or input, enables an array of dynamic optimisation strategies which select at runtime the option which is predicted to work best in a particular situation, or adjust on the fly its participation as part of a larger system or a hybrid approach. The present work analyses the state of the art on this topic, and restates the problem in the subarea of Recommender Systems (RS) where it has barely been addressed so far.

We research meaningful definitions of performance in the context of RS, and the elements to which it can sensibly apply. In doing so, we take as a driving direction the application of performance prediction to achieve improvements in specific combination problems in the RS field. We formalise the notion of performance prediction, as understood in our research, in specific terms within this frame. Upon this basis, we investigate the potential adaptation of performance predictors defined in other areas of IR (mainly query performance in ad hoc retrieval), as well as the definition of new ones based on theories and tools from Information Theory. Standard performance metrics are taken as the basis for prediction, and alternative metrics are proposed where needed.

The proposed predictors and methods are tested empirically in two types of experiments. First, the correlation between the predictors and performance metrics is measured. Second, the predictors are introduced in two standard aggregative operations in recommender systems, by dynamically weighting the participation of the aggregated components according to the output of the corresponding predictor. The resulting performance of the combined system is then compared to the original performance without the dynamic adjustment.

The results are encouraging since we find four predictors which outperform standard algorithms at all sparsity levels. Moreover, two of them show significant correlation with performance measures. Additionally, the experiments uncover further necessary work and questions for the continuation of the research, such as the definition of appropriate user-level performance metrics, finer and/or alternative correlation analysis methods, refinement and improvement of predictors, or non-trivial transformations of predictor outputs into combination weights.

Table of Contents

List of figures

List of tables

Acronyms

BPA	Basic Probability Assignment
CB	Content Based
CF	Collaborative Filtering
HF	Hybrid Filtering
IDF	Inverse Document Frequency
IF	Item Frequency
IIG	Item Information Gain
IIC	Item-based Item Clarity
IR	Information Retrieval
IUC	Item-based User Clarity
IUF	Inverse User Frequency
MAE	Mean Average Error
MAP	Mean Average Precision
RMSE	Root Mean Square Error
RS	Recommender System
TF	Term Frequency
UF	User Frequency
UIC	User-based Item Clarity
UIG	User Information Gain
UUC	User-based User Clarity

Chapter 1. Introduction

1.1 Motivation

Performance prediction has gained attention in the Information Retrieval (IR) field since the half of the past decade and has now become an established research topic. Predicting the performance of an IR system, subsystem, module, function, or input, enables an array of dynamic optimisation strategies which select at runtime the option which is predicted to work best in that particular situation, or adjust on the fly its participation as part of a larger system or a hybrid approach. Metasearch is a clear example where performance prediction can be applied to, since search results are composed by combining the output of a number of search engines (Fox & Shaw 1993). Personalised retrieval systems (including techniques such as personalised search, collaborative filtering (CF), recommender systems (RS), or retrieval in context) also combine several sources of relevance evidence based on different inputs, such as explicit queries, search history, explicit user ratings, social information, user feedback, context models, etc.

The question of how much the different components of an IR system should come into play in each particular retrieval decision can be a matter of a wide variety of considerations. The relative importance of the subcomponents easily lends itself to being tuned for optimisation. This tuning is in fact indispensable for a good final performance of the aggregate system, and is critical for the good functioning and success of the latter. This is one of the primary applications of performance prediction, since it can be used to weight each component in a way that optimises the quality of the resulting decision.

The question gains increasing relevance today, with the proliferation and variety of open environments and input information channels available to users and IR systems, upon convergent technologies: online newspapers and portals, news aggregators, social network environments, forums, blogs, microblogs, search portals, online encyclopedias and dictionaries, RSS feeds, etc. The number and variety of sources, inputs, criteria, services, strategies, and relevant evidence available to IR systems, along with the fast variability of conditions in such environments, and the wide variations in quality, reliability, availability, etc., of data are such, that achieving effective actions is to a large extent a matter of adequately selecting, combining and/or weighting the right inputs and strategies upon which the retrieval decisions are made, depending on the dynamic conditions and situation at hand. This calls for the research of hybrid approaches with a level of dynamic self-assessment and self-adjustment mechanisms, in order to optimise the resulting effectiveness of the retrieval systems, by opportunistically taking advantage of high-quality data when available, but avoiding to stick to fixed strategies when they can be predicted to yield poor results under certain conditions.

Performance prediction is based on the analysis and characterisation of the evidence used by an IR system to assess the relevance (utility, value, etc.) of retrieval objects (documents, goods, etc.) at execution time, upon which the system makes decisions about the selection, recommendation, or ranking of the information items presented to the user (Cronen-Townsend et al. 2002). The most classic and basic retrieval scenario involves a user query and a collection of documents as the basic input to form a ranked list of search results, but other additional elements can be taken into account to select and rank results (Baeza-Yates & Ribeiro-Neto 1999). The user context (current tasks, query logs, preferences, etc.), global properties of the document collection, comparisons with respect to other reference elements such as historic data, or the output from other systems, among others, are some examples of the different sources of information that a predictor may draw evidence from.

The present work analyses the state of the art on performance prediction in IR, and restates the problem in the subarea of Recommender Systems where it has barely been addressed so far. We research meaningful definitions of performance in the context of RS, and the elements to which it can sensibly apply. In doing so, we take as a driving direction the application of performance prediction to achieve improvements in two specific combination problems in the RS field, namely, neighbour weighting in CF, and the dynamic combination of CF and content based (CB) filtering in hybrid recommendation systems.

Our research approach firstly consists of studying the adaptation of query performance predictors proposed in the IR literature (e.g. (Cronen-Townsend et al. 2002)) into the RS context, taking into account the important differences in the nature of the retrieval spaces. Complementarily, we research the definition of new predictors drawing from Information Theory (as some of the predictors used in IR in fact do). Finally, we test the predictors in the selected combination problems and measure the effect on the final performance.

1.2 Problem definition

As explained earlier, the general primary research problem addressed in this work is the definition of performance predictors in the RS field. This involves identifying systems, components or elements of RS techniques whose performance we aim to predict. This in turn requires defining what performance means for these components, and selecting or defining computable metrics for them – while performance has well-established definitions and metrics in the literature for standard cases, such as a recommender system as a whole, there is not an equivalent notion for specific elements, functions or subsystems, and therefore this is addressed here as part of the research problem. On the other hand, the identification of interesting elements for their performance to be predicted, and the definition of what performance specifically means for each of them, would be too loose a goal unless additional requirements are stated. In our case, the framing aim for the problem is the use of the predictors in specific combination problems in RS. As a second-phase task, achieving and measuring specific improvements resulting from this approach is itself an additional goal of the work.

The dynamic combination problem that is taken as a driving direction for our research can be stated, at the most abstract level, as formulated next. As a matter of fact, the statement that follows shall not make any assumptions about the kind of retrieval system where the combination takes place, and thus could be virtually compatible with retrieval models in any IR area. In Chapter 3, we shall extend and specialise this formal framework for the particular case of recommender systems, where the research presented here shall focus. The formulation and notation are introduced here for clear and unambiguous reference to the key elements of the addressed problems, and will be used in the rest of the document.

The combination problem is thus formulated as follows. Let us assume we wish to answer a retrieval question (or equivalently, make a retrieval decision) based on n retrieval criteria (which are often called decision makers, experts, relevance sources, input systems, IR model components, etc., in the literature of different fields), and some input data, which we may represent as a variable z. The first research question is how the criteria should be combined. This is a largely addressed issue in IR and other fields such as Pattern Matching and Multi-criteria Decision Making, where the combination of classifiers has become an established topic of interest (Kuncheva 2004). In our work, we make some basic assumptions to this respect as follows, despite which the generality of the problem statement remains fairly open.

Let us assume (or require) that each criteria can be represented by a scalar value $g_j(z)$, and the final retrieval answer can also be represented by a scalar function $g(z)$. This is the typical case in IR, where systems are essentially characterised by a retrieval function. In Pattern Matching, g and g_j would be called discriminant functions. How the outputs of g_j and g map to retrieval answers is left open at this level of our problem statement, as far as all g_j conform to a same common scheme, and so does g. For instance, the retrieval question can be "does item d satisfy user needs" (or equivalently, stated as a retrieval decision, "pick d or not, on behalf of the user, given certain user needs"), z can be a query, and $g_j(z)$ can be a Boolean value (0/1 for yes/no), or a value in [0,1] (if the answer is a matter of degree). Or the question can be how to sort a list of documents $\mathcal{D}$ by their degree of relevance to a query q, in which case we might take $z \in \mathcal{D} \times \{q\}$, and sort $\mathcal{D}$ by decreasing value of $g_j(z)$, if the latter value represents the estimated relevance of a document d given a query q, where $z = (d,q)$.

The second research question is whether and how each criterion could be given more or less weight than the other in the final decision. This question is easier to state if g can be expressed as a monotonically increasing function of α_j and g_j for each j, where α_j are free parameters that can be adjusted. A typical (and fairly general) case is when the function is a linear combination (Pennock et al. 2000), though our proposals can be generalised to other combination functions, as long as they allow for a similar parameter-based weighting of criteria:

$$g = \varphi(g_1, \cdots, g_n) = \alpha_1 g_1 + \cdots + \alpha_n g_n$$
$$\sum_{j=1}^{n} \alpha_j = 1$$

Given this formulation, the driving problem of this work is that of finding as good weights α_j as possible, in terms of the resulting quality of the retrieval decision induced by g, which can be expressed as:

$$\{\alpha_j^*\} = \underset{\alpha_j \in \mathbb{R}}{\arg\max} \left(\rho(\varphi) = \rho(\alpha_1 g_1 + \cdots + \alpha_n g_n)\right)$$

where ρ represents a quality (or performance) measure of the decision encoded by g. Without loss of generality, we can suppose this measure is directly proportional to the quality of the decision. As a hypothesis, we assume that a good way to approximate the maximum performance value from ρ is by having α_j be monotonically increasing with respect to the performance $\rho(g_j)$ of each subsystem, that is, the quality of the retrieval decision if it was made based on g_j alone.

Since the definition and computation of a perfect function ρ requires precise and exact knowledge of the real goodness of retrieval decisions, which is in general not available to the system in real settings, it is only possible to compute an estimation, or more precisely, a prediction, of how good a retrieval decision may be.

The problem of predicting the performance of a retrieval system became an established research topic in the IR field by the end of the past decade. In the next chapter, we will review the most significant work in that area. Finding good performance predictors is also one of the main objects of our present work. It can be expressed as finding a function $\gamma_j(z)$ that approximates or predicts $\rho(g_j(z))$. In other words, a good predictor $\gamma_j(z)$ is one which should correlate positively with $\rho(g_j(z))$.

Our approach to the problem, as we shall present in Chapter 3 and 4, delves into issues of uncertainty, as a major factor that determines performance in IR, in the sense that, as a general trend (or hypothesis), the more uncertainty there is in the available input data a system has to make decisions upon, the less accuracy one may expect in the results, and vice-versa. Another main drive in our research is the adaptation of performance prediction methods developed in specific IR settings, such as query performance in ad hoc retrieval, to different areas where the problem has not yet been addressed such as, specifically, Recommender Systems.

Once good performance predictors have been found for the n input systems, the problem remains to actually make α_j a function of the output from the predictors, $\alpha_j = \psi(\gamma_j)$ In our research and experiments so far, we take the simple assignment $\alpha_j = \gamma_j$ (i.e. the identity $\psi(x) = x$) and focus on finding good γ_j predictors.

Having formally defined the research problem addressed here, the underlying research hypothesis this work builds upon (and for some of them, aims to verify) can be summarised as:

- A linear combination is a good and general function to build composite IR systems which combine retrieval submodules.
- A suitable assignment of combination weights, which is monotonically increasing with the predicted performance of the corresponding components, enables an improved performance in a combined retrieval system.
- Performance predictors from the IR field can be adapted to RS and result in effective predictors.
- The effectiveness of performance predictors can be assessed by their correlation to suitable performance metrics.
- The performance of an IR system, or component, on a specific retrieval task is monotonically decreasing with the amount of uncertainty involved in the retrieval problem at hand.

1.3 Research goals

The broad objective of the research presented here is to find predictive methods for the performance of specific components in specific areas of Information Retrieval, and to improve the performance of combined IR methods, based on the dynamic, automatic analysis and prediction of the expected performance of the constituents of the composite method, whereupon the relative participation of each constituent is adjusted, in accordance to its predicted effectiveness. The problem is a recognised research topic in the IR field at large, and ad hoc retrieval in particular. Building on the awareness of the achieved progress so far in that scope, the problem is particularised here to the area of Recommender Systems. Within this general frame, this work has the following specific research objectives:

- **State of the art study and analysis on performance prediction in IR.** Study of different models and approaches to explicitly predict the performance of a system in the state of the art in Information Retrieval. Study of specific techniques and magnitudes, such as query clarity, to approximate, represent, and estimate performance.
- **Identification and analysis of (actual or potential) applications for performance prediction in IR,** such as rank fusion, metasearch, and distributed search, or more specific ones in the area of personalisation.

- **Performance prediction in Recommender Systems.** Study the potential of performance prediction in specific problems and methods in the area of Recommender Systems. Identification and characterisation of combination methods, and definition of a formal framework where performance predictors shall come into play.
- **Adaptation of performance prediction techniques from IR to RS.** Explore the translation of specific effective predictors such as query clarity, researched in ad hoc retrieval, to recommender system problems. Assess the effectiveness of the approach in this new setting in terms of their actual correlation to performance metrics.
- **Definition of new performance predictors in RS.** Complementarily to the adaptation of known techniques, research the definition of new ones, adapted to the RS area, drawing from adjacent fields such as Information Theory, or following heuristic, domain-specific approaches. Assess the effectiveness of the predictors in terms of their correlation to performance metrics.
- **Effective application of performance predictors to combined RS methods.** Introduction of the proposed predictors into combined recommendation methods, achieving an actual improvement of the performance of the combined methods.
- **Experimental work and empiric evidence supporting the proposed models and methods.** Selection, preparation and/or extension of datasets and benchmarks, with priority on standard collections. Methodological study for the experimental approach, setup, and metrics. Definition and execution of experiments. Evaluation of results, assessment of improvement and/or benefits, and analysis of the behaviour of the method. Iterative refinement of proposals, driven by the experimental work.

According to the formulation presented in the previous section, these objectives are equivalent, in practical terms, to finding appropriate approaches or (theoretic, formal, technical) grounds for the definition of γ functions in the literature (not limited to the Information Retrieval field), and their adaptation to the specific aims and problems addressed in our research work, in the area of Recommender Systems. Finally, for the experimental part of the research, additional work is needed in some cases to bring the predictors to computable forms, analyse the results and evaluate them against appropriate benchmarks.

1.4 Document structure

The rest of this document is structured as follows:

- In Chapter 1, the motivation of this work is presented, as well as an initial problem definition and the goals of the research undertaken here.
- Chapter 2 provides an overview of the state of the art in performance prediction and metrics in IR. We also provide a brief introduction to Recommender Systems and rank fusion, which are the main areas where our research is framed.
- In Chapter 3, the proposed methods are presented, including the definition of predictors and the formulation of combination problems. A general formal framework for combined IR methods is defined, in which predictors fit naturally, in order to weight the components being combined. The proposed methods are developed in the area of Recommender Systems, where two specific combined methods are selected, for which specific predictors are proposed and defined.

- Chapter 4 reports on the experimental work, where the proposed techniques are evaluated on a public dataset. Particular attention is paid to the methodological design, in order to properly compare the proposed methods against appropriate (fair and feasible) baselines.
- Chapter 5 provides conclusions drawn from this work, along with potential lines for the continuation of the research.

Chapter 2. State of the art

2.1 Introduction

The IR field is pervaded with cases where relevance and retrieval systems, models or criteria are based on a fusion or combination of sub-models. Metasearch is a clear example, where search results are composed by combining the output of a number of search engines. But standard commercial search engines themselves combine a number of retrieval functions, such as link-based (e.g. PageRank (Brin & Page 1998)) vs. query-based, statistic vs. linguistic, in order to rank their search results. Query-based results in turn combine several relevance criteria, such as the frequency of query terms in the documents, their proximity in the text, the document section where they occur (e.g. title, body, link text, etc.), the font, and so forth. In the area of personalised retrieval, Hybrid Filtering (HF) is the clearest example, especially when defined as a linear combination of a collaborative filtering (CF) and a content-based (CB) recommender (Burke 2002, Cantador et al. 2007). But in fact other Recommender System (RS) approaches, and in general most IR techniques in the personalisation spectrum, such as personalised search (Jeh & Widom 2003), information filtering, relevance feedback (Rocchio 1971), or retrieval in context (Mylonas et al. 2008, Cantador et al. 2008c) represent a family of inherently combinatory retrieval methods, since they combine several sources of relevance evidence based on different inputs: explicit queries, search history, explicit user ratings, social information, user feedback, context models, etc. Aggregate models often appear as inner combinations at finer grain sizes within personalised systems, as in the combination of neighbour ratings to produce recommendations in CF.

The components of composite retrieval models are sometimes combined in a way that makes them difficult or impossible to separate. For instance, certain hybrid recommendation approaches build a unified model where the CB and CF components are inseparable (this is the case of the cascade hybrid recommender (Burke 2002), for example). In many cases though, the combination is more explicit, often taking the form of a mathematical operation (commonly a linear combination) that outputs the combined result as a function of the outputs of the components (Cantador et al. 2008b). In such cases, the relative importance of the subcomponents easily lends itself to being tuned for optimisation. This tuning is in fact indispensable for a good final performance of the aggregate system, and is critical for the good functioning and success of the latter. The task can be addressed in different ways, such as by mere empiric means and ad hoc engineering work. While this is of course necessary at some point, the task can also be formulated and approached in more principled ways. It has indeed been addressed, under different particular forms, as a relevant research problem in the IR field (Fox & Shaw 1993, Montague & Aslam 2001) and adjacent areas such as Multi-Criteria Decision Making (Beg & Ahmad 2003) and classification (Kittler et al. 1998), and connects to important related research topics (such as performance prediction in IR) that will be reviewed in this chapter.

The question of how much the different components of an IR system should come into play in each particular retrieval decision can be a matter of a wide variety of considerations, such as the reliability of the criterion being implemented by the component, the reliability of the implementation, the quality of its available input, the quality of its output, the relative importance of the criterion for each user, and so forth. But in a sense, the issue can be ultimately reduced to a matter of how effective each component would be by itself in the retrieval decision at hand. If we knew in advance how good or bad is the contribution of each subcomponent, in terms of the effectiveness and quality of the decision it advocates for, we might turn on or off (or gauge somewhere in between) each component in a way that optimises the quality of the resulting decision. Since this information is in general not fully or explicitly available to the retrieval system, it is only possible to aim at a prediction or assessment of the effectiveness (accuracy, reliability) of its subcomponents.

In this chapter, we survey the state on the art on all such problems, involved in the research goals addressed in this work: performance prediction in IR, performance evaluation and metrics (Section 2.2), alternative information quality measures with potential use in IR (Section 2.3), and common (and/or relevant for our research) IR combination problems, mainly in the areas of rank fusion (Section 2.4) and recommender systems (Section 2.5). Finally, Section 2.6 summarises and discusses on the analysed areas and techniques.

2.2 Performance prediction

Broadly speaking, performance refers to the quality of the output of a system in response to a particular input. Performance prediction in IR has been mostly addressed in the area as a query performance issue. Query performance refers to the performance of the IR system in response to the query. It also relates to the appropriateness of a query as an expression for a user information need. Dealing effectively with poorly-performing queries is a crucial issue in IR. Performance prediction provides tools that can be useful in many ways (Zhou & Croft 2006, Yom-Tov et al. 2005a):

- From the user perspective, it provides valuable feedback that can be used to direct a search, e.g. by rephrasing the query or providing relevance feedback.
- From the perspective of a retrieval system, performance prediction provides a means to address the problem of retrieval consistency. The consistency of retrieval systems can be addressed by distinguishing poorly performing queries based on performance prediction techniques. Based on that, a retrieval system can invoke alternative retrieval strategies for different queries (query expansion or different ranking functions based on the predicted difficulty). Thus, the search engine can use the query predictor as a target function for optimising the query.
- From the perspective of the system administrator, she can identify queries related to a specific subject that are difficult for the search engine, and expand the collection of documents to better answer insufficiently covered subjects (for instance, adding more documents to the collection). It also allows simple evaluation means for query results.
- For distributed information retrieval, performance estimations can be used to decide which search engine (and/or database) to use, or how much weight to give it when its results are combined with those of other engines.

These are the main applications of performance prediction in IR systems. The prediction methods researched in the literature use a variety of available data, such as a query, its properties with respect to the retrieval space (Cronen-Townsend et al. 2002), the output of the retrieval system (Carmel et al. 2006), or the output of other systems (Aslam & Pavlu 2007). According to whether or not the retrieval results are used in the prediction, the methods can be classified into pre-retrieval and post-retrieval approaches, which are described in Subsections 2.2.2, and 2.2.3, respectively. Another relevant distinction is whether the predictors are trained or not.

Besides such distinctions, we shall stand out two important information retrieval subareas in our analysis, because of their importance and current interest: rank fusion and Recommender Systems. Later, in Sections 2.4 and 2.5, we explain how prediction performance can be helpful in these areas.

But first, in order to identify good performance predictors and validate or assess their potential, measures of actual performance need to be defined. Performance evaluation and metrics have been a core research and standardisation issue for decades in the IR field. The next subsection introduces and summarises the main metrics and established methodologies in the area. Based on the latter, predictive methods can be assessed by analysing correlations between predictive outputs and reference performance metric values on common input data.

2.2.1 Performance metrics in IR

The notion of performance lends itself to different interpretations, views and definitions. Although we shall get into some specific considerations on performance metrics in Chapter 4, a full discussion is out of the scope of the present work. See e.g. (Baeza-Yates & Ribeiro-Neto 1999, Herlocker et al. 2004) for an extended discussion on the subject. Different ways of measuring performance have been proposed and adopted in the area (Hauff, Hiemstra & de Jong 2008), the most prominent of which shall be summarized here (later in Section 2.5, further metrics, specialised to Recommender Systems, shall be reviewed).

As a result of several decades of work and research in the IR community, a set of standard performance metrics has been established as a consensual reference for evaluating the goodness of different types of IR systems. These measures generally require a collection of documents and a query (or alternative forms of user input such as item ratings), and assume a ground truth notion of relevancy (traditional notions take this relevance as binary, while others, more recently proposed, consider different degrees of relevance).

One of the most prevalent performance measures in IR is *precision*, which is defined as the ratio of retrieved documents which are relevant for a particular query. In principle, this definition takes all retrieved documents into account, but it can also be measured at a given cut-off rank as the *precision at n* or *P@n*, where just the top-n documents are considered. Another related and widespread measure is *recall*, which is the fraction of relevant documents retrieved by the system. These two measures are inversely related, since increasing one generally reduces the other. For this reason, they are often combined (into e.g. the *F-measure*, *Mean Average Precision* or *MAP*), or the values of one measure are compared at a fixed value of the other measure. A common representation is to plot a curve of precision versus recall, which is usually based on 11 standard recall levels (0%, 10%, ..., 100%).

A problem with MAP when used for poorly performing topics is that changes in the scores of best-performing topics mask changes in the scores of poorly performing topics (Voorhees 2005b). In (Voorhees 2005a), two measures were proposed to study how well IR systems avoid very poor results for individual topics: the *%no measure*, which is the percentage of topics that retrieved no relevant documents in the top ten retrieved, and the *area measure* the area under the curve produced by plotting MAP(X) vs. X, where X ranges over the worst quarter topics; but these measures were shown to be unstable. A third measure was introduced: *gmap*, the geometric mean of the average precision scores of the test set of topics (Voorhees 2006). This measure gives appropriate emphasis to poorly performing topics while being stable with as few as 50 topics.

2.2.2 Pre-retrieval prediction

In this category, performance predictors do not rely on the retrieved document set, but on other information, mainly extracted from the query issued by the user. This approach has the important advantage that the predictions can be produced before the system response is even started to be elaborated. This means in particular that the prediction can be taken into account to improve the retrieval process itself. However, these predictors have the potential handicap, with regards to their accuracy, that the extra retrieval effectiveness cues available after the system response are not exploited (Zhou 2007).

Query performance has been studied from two main perspectives: based on statistic methods, and based on linguistic approaches. Most research on the topic has followed the former approach. Some researchers have also explored IDF (inverse document frequency) related features as predictors. In this section, these three approaches are described.

IDF-related predictors

Inverse document frequency (IDF) is one of the most useful and widely used magnitudes in IR. It is usually included in the IR models to properly compensate for how common a term is. The technique commonly takes an ad hoc, heuristic form, even though formal versions of the function exist (Roelleke & Wang 2008, Aizawa 2003, Hiemstra 1998). The main motivation for the inclusion of an IDF factor in a retrieval function is that terms which appear in many documents are not very useful for distinguishing a relevant document from a non-relevant one. In other words, it can be used as a measure of the specificity of terms (Jones 1972), as an indicator of their discriminatory power. In this way, IDF is commonly used as a factor in the weighting functions for terms in text documents. The general formula of IDF for a term k_j is the following:

$$\mathrm{IDF}_j = \log \frac{N}{n_j}$$

where N is the total number of documents in the system and n_j is the number of documents in which the index term k_j appears.

Some research works on performance prediction have studied IDF as a basis for defining predictors. He & Ounis (2004) propose a predictor based on the standard deviation of the IDF of the query terms. Plachouras et al. (2004) represent the quality of a query term by a modification of IDF, where instead of the number of documents, the number of words in the whole collection is used, and the query length acts as a normalising factor. These IDF-based predictors showed moderate correlation with query performance.

There are several variations of the above formula, including adaptations to other areas. In RS, Breese et al. (1998) modified the weight given to each item in a user similarity function by a multiplying factor they called *IUF*, which is equivalent to the above formula, but n_j counting the number of users rated the item j. Their idea was that commonly liked items are not as useful in capturing similarity as less frequent items. It plays a similar role to that of a performance predictor (taking into account the popularity of an item).

Probabilistic methods

Several techniques measure certain characteristics of the retrieval inputs to estimate performance. For example, so-called **clarity scores** have been used to measure the coherence of a collection with respect to a query. In spite of this original (and fundamental) definition, in (Cronen-Townsend et al. 2002) the authors use it on results sets, as a specific variant explored in order to ease calculations and improve computation time. It also takes advantage of the extra information available after the retrieval (for this reason, sometimes it is classified as a post-retrieval predictor (Zhou 2007, Hauff, Hiemstra & de Jong 2008)). We shall explain this variant here, since this is its most practical application.

Cronen-Townsend et al. (2002) define query clarity as a degree of (the lack of) the query ambiguity. In (Cronen-Townsend et al. 2006), query ambiguity is defined as "the degree to which the query retrieves documents in the given collection with similar word usage". They measure the degree of dissimilarity between the language associated with the query and the generic language of the collection as a whole. This measure is defined as the relative entropy, or Kullback-Leibler divergence, between the query and collection language models (unigram distributions). Analysing the entropy of the language model induced by the query is a natural approach since entropy measures how strongly a distribution specifies certain values, in this case, terms. Cronen-Townsend et al. used the following formulation:

$$P(w|q)=\sum_{d\in R}P(w|d)P(d|q),\quad P(q|d)=\prod_{w_q\in q}P(w_q|d)$$

$$P(w|d)=\lambda P_{ml}(w|d)+(1-\lambda)P_{coll}(w)$$

$$\text{clarity}(q)=\sum_{w\in V}P(w|q)\log_2\frac{P(w|q)}{P_{coll}(w)}$$

with *w* being any term, *q* the query, *d* a document or its model, *R* the set of documents that contain at least one query term (it is possible to use here the whole collection), $P_{ml}(w|D)$ the relative frequency of term *w* in document *D*, $P_{coll}(w)$ the relative frequency of the term in the collection as a whole, λ a free parameter (set to 0.6 in Cronen-Townsend's work), and *V* the entire vocabulary.

The authors find that queries whose likely documents are a mix of documents from disparate topics receive lower score than if they result in a topically-coherent retrieved set. They report a strong correlation between the clarity score of a query and the performance of that query. Because of that, the clarity score method has been widely used in the area for query performance prediction. Some applications of the clarity score measure include query expansion (anticipating poorly performing queries which should not be expanded), improving performance in the link detection task in topic detection and tracking by modifying the measure of similarity of two documents (Lavrenko et al. 2002a), and document segmentation (Diaz & Jones 2004).

As an alternative to query clarity, He & Ounis (2004) propose a simplified version of the clarity score where the query model is estimated by the term frequency in the query:

$$SCS=\sum_{w}P_{ml}(w|q)\log_2\frac{P_{ml}(w|q)}{P_{coll}(w)}$$

$$P_{ml}(w|q)=\frac{qtf}{ql}$$

$$P_{coll}(w)=\frac{tf_{coll}}{token_{coll}}$$

where *qtf* is the number of occurrences of a query term *w* in the query, *ql* is the query length, tf_{coll} is the number of occurrences of a query term in the whole collection, and $token_{coll}$ is the total number of words in the collection.

Diaz & Jones (2004) exploit temporal features of the document (time stamps) for prediction. They find that although temporal features are not highly correlated to performance, using them together with clarity scores improves prediction accuracy. Kwok et al. (2004) build a query predictor using support vector regression, based on training the classifier with features such as document frequencies or query term frequencies. A small correlation between predicted and actual query performances is observed with this approach. He & Ounis (2004) propose the notion of query scope as a measure of the specificity of a query, which is quantified as the percentage of documents that contain at least one query term in the collection, e.g. $log(N_Q / N)$, where N_Q is the number of documents containing at least one of the query terms and N is the total number of documents in the collection). Query scope is effective in inferring query performance for short queries in ad hoc text retrieval. On the other hand, it seems to be very sensitive to the query length (Macdonald et al. 2005).

Linguistic methods

Beyond the approaches with statistic basis, linguistic methods have also been researched. Mothe & Tanguy (2005) extract 16 query features and study their correlation with respect to recall and average precision. The 16 features were classified into three different classes according to the linguistic aspects that are analysed:

- Morphological features:
  - **Number of words**.
  - **Average word length** in the query (measured in numbers of characters).
  - **Average number of morphemes per word**, obtained using the CELEX[1] morphological database. The limit of this method is of course the database coverage, which leaves rare, new, and misspelled words as mono-morphemic.
  - **Average number of suffixed tokens**, obtained using the most frequent suffixes from the CELEX database. After that, the authors tested for each lemma in the topic if it was eligible for a suffix from this list
  - **Average number of proper nouns**, obtained by POS (part-of-speech) tagger's analysis.
  - **Average number of acronyms**, detected by a simple pattern-matching technique.
  - **Average number of numeral values**, also detected by a simple pattern-matching technique.
  - **Average number of unknown tokens**, which are those marked up as such by the POS tagger. Most unknown words happen to be constructed words such as "mainstreaming", "postmenopausal" or "multilingualism".
- Syntactic features:
  - **Average number of conjunctions**, detected through POS tagging.
  - **Average number of prepositions**, detected also through POS tagging.
  - **Average number of personal pronouns**, again detected through POS tagging.
  - **Average syntactic depth**, computed from the results of the syntactic analyzer. It is a straightforward measure of syntactic complexity in terms of hierarchical depth. It simply corresponds to the maximum number of nested syntactic constituents in the query.
  - **Average syntactic links span**, computed from the results of the syntactic analyzer. It is the average pairwise distance (in terms of number of words) between individual syntactic links, over all syntactic links.
- Semantic features:
  - **Average polysemy value**, which is computed as the number of synsets in the WordNet[2] database that a word belongs to, averaged over all terms of the query.

Mothe and Tanguy researched the correlation between these features and precision and recall over datasets with different properties, and found that:

[1] CELEX, English database (1993). Available at www.mpi.nl/world/celex
[2] WordNet, lexical database for the English language. Available at http://wordnet.princeton.edu/

- The only feature positively correlated with both performance measures is the number of proper nouns, although it only correlates in two out of four datasets.
- Many variables do not have a significant impact on any performance measure. Only the more "sophisticated" features are correlated with some measure.
- The only two variables correlated with some metric in more than one dataset are the average syntactic links span (for precision), and the average polysemy value (for recall).

2.2.3 Post-retrieval prediction

In this category, predictors make use of retrieved results. Broadly speaking, techniques in this category provide better prediction accuracy compared to those in the previous category. However, computational efficiency is usually a problem for many of these techniques, and furthermore, the predictions cannot be used to improve the retrieval strategies, as the output from the latter is needed to compute the predictions in the first place.

Using visual features, such as titles and snippets, from a surrogate document representation of retrieved documents, Jensen et al. (2005) train a regression model with manually labelled queries to predict precision at the top 10 documents in Web search. The authors report moderate correlation with respect to precision.

The concept of query clarity has inspired a number of similar techniques. In (Amati et al. 2004), the authors propose the notion of query difficulty to predict query performance. In that work, query difficulty is captured by the notion of the amount of information ($Info_{DFR}$) gained after a first-pass ranking. If there is a significant divergence in the query-term frequencies before and after the retrieval, then the authors make the hypothesis that this divergence is caused by a query which is easy-defined (or correlated with high values of average precision). $Info_{DFR}$ shows a significant correlation with average precision. In spite of this, the authors find no correlation between this predictor and query expansion (which was their main application), concluding that although the retrieval effectiveness of query expansion in general increases as query difficult decreases, very easy queries hurt performance.

More recently, a new concept was coined: ranking robustness (Zhou & Croft 2006). It refers to a property of a ranked list of documents that indicates how stable the ranking is in the presence of *uncertainty* in the ranked documents. The idea of predicting retrieval performance by measuring ranking robustness is inspired by a general observation in noisy data retrieval. The observation is that the degree of ranking robustness against noise is positively correlated with retrieval performance. This is because regular documents also contain *noise*, if we interpret noise as uncertainty. The robustness score performs better than, or at least as well as, the clarity score.

Carmel et al. (2006) find significant correlation between average precision and the distance measured by the Jensen-Shannon Divergence between the retrieved document set and the collection. Vinay et al. (2006) propose four measures to capture the geometry of the top retrieved documents for prediction: the clustering tendency as measured by the Cox-Lewis statistic, the sensitivity to document perturbation, the sensitivity to query perturbation, and the local intrinsic dimensionality. The most effective measure is the sensitivity to document perturbation, an idea similar to the robustness score, in the sense that Vinay et al. issue a perturbed version of the document as a pseudo-query and record the new rank that the original document assumes with the modified query. However, document perturbation does not perform equally well for short queries, and prediction accuracy drops considerably when alternative state-of-the-art retrieval techniques (such as BM25 or a language modelling approach) are used instead of the TF-IDF weighting (Zhou 2007).

Kwok et al. (2005) proposed predicting query performance by analysing regularities, and more specifically, similarity among retrieved documents. The basic idea is that when relevant documents occupy the top ranking positions, the similarity between top documents should be high, based on the assumption that relevant documents are similar to each other. While this idea is interesting, preliminary results were inconclusive. A similar idea can be found in (Grivolla et al. 2005). Grivolla et al. calculate the entropy and pairwise similarity among top results. First, the entropy of the set of the *K* top-ranked documents for a query is computed. The entropy should be higher when the performance for a given query is bad. Second, the mean cosine similarity between documents is proposed, using the base form of TF-IDF term weighting to define the document vectors. Correlation between average precision and the proposed predictors is not consistent along the different systems used in the experiment, although they can still be useful for performance prediction, especially when used in combination.

In (Zhou & Croft 2007) two more techniques are defined for Web search:

- **Weighted Information Gain** measures the change in information about the quality of retrieved results (in response to a query) from an imaginary state that only an average document is retrieved to a posterior state that the actual search results are observed. This predictor is very efficient, and it demonstrates better accuracy than clarity scores.
- **Query Feedback** measures the degree of corruption that results from transforming *Q* to *L* (the output of the channel when the retrieval system is seen as a noisy channel, e.g., the ranked list of documents returned by the system). The authors design a decoder that can accurately translate *L* back into a new query *Q'*, whereupon the similarity between the original query *Q* and the new query *Q'* is taken as a performance predictor, since the authors equates the evaluation of the quality of the channel with the problem of predicting retrieval effectiveness. The computation of this predictor requires a higher computational cost than the previous one, which is a major drawback of this technique.

2.3 Quantitative information-theoretic magnitudes

In Section 2.2 we have shown different performance predictors used in IR. One of the most prominent is the clarity score (Cronen-Townsend et al. 2002), which is closely related to the lack of ambiguity and basically measures the distance between the query and the collection language models (using relative entropy). This concept has motivated several other predictors, as we have already seen in Sections 2.2.2 and 2.2.3.

These techniques link to notions from Information Theory related to uncertainty and quantitative information analysis, which are also studied in related fields such as Decision Making or classification. Theories that quantify the ambiguity or uncertainty in a particular model have been applied to different fields, such as Economy (Labreuche & Grabisch 2003, Gollier & Treich 2003), Logic (Hunter & Liu 2005, Arieli 2003, Poole & Smyth 2005), or Psychology (Laming 2001). These theories introduce and elaborate on measures such as entropy, mutual information, information gain, relative entropy, or belief functions (from Dempster-Shafer's theory). The performance of an IR system (as information systems in general) is largely dependent on information quality, quantity, uncertainty, and other such magnitudes that are studied in these fields, and therefore a brief, selective revision of the latter is in order here.

These techniques have been usually brought to IR heuristically, although there are some attempts to introduce them in the retrieval model (Hiemstra 1998, Lalmas 1997, Beg &

Ahmad 2003). In the remainder of the current section, we shall revise the most relevant information-theoretic tools which have been or might be used in the context of performance prediction, and to some extent, IR at large.

2.3.1 Entropy

Entropy is well-known to be the basis for powerful theories that quantify the uncertainty and other information magnitudes. The entropy of a discrete distribution is a measure of the randomness or unpredictability of a sequence of symbols $\{v_1, ..., v_m\}$ drawn from it, with associated probability p_i. It can be calculated as (using a base 2 logarithm when measured in *bits*):

$$H = -\sum_{i=1}^{m} p_i \log_2 p_i$$

One bit corresponds to the uncertainty that can be resolved by the answer to a single yes/no question. For a continuous distribution, the entropy is defined as:

$$H = -\int_{-\infty}^{\infty} p(x) \ln p(x) dx$$

We have to note that the entropy does not depend on the symbols themselves, but on their probabilities. When each symbol is equally likely to occur we have the *maximum entropy distribution*, which in the discrete case such an example is the uniform distribution, whereas in the continuous case the main example is the Gaussian distribution. Conversely, if all the probabilities p_i are 0 except one, we have the *minimum entropy distribution*. For example, a probability density in the form of a *Dirac delta* function has the minimum entropy:

$$\delta(x-a) = \begin{cases} 0 & x \neq a \\ \infty & x = a \end{cases}$$

$$\int_{-\infty}^{\infty} \delta(x) dx = 1$$

Some properties of the entropy of a discrete distribution is that it is invariant to shuffling the event labels, and that for an arbitrary function f we have $H(f(x)) \leq H(x)$, that is, processing never increases entropy.

Besides the plain entropy, it is also possible to define a conditional entropy as follows:

$$H(Y \mid X) = \sum_{x \in X} p(x) H(Y \mid X = x) = -\sum_{x \in X, y \in Y} p(x,y) \log\left(p(y|x)\right)$$

This quantifies the remaining entropy of a random variable Y given that the value of the second random variable X is known.

In the field of IR, Cronen-Townsend et al. motivate using entropy for performance prediction: "looking at the entropy of the query language model is a natural approach since entropy measures how strongly a distribution specifies certain values, in this case terms" (Cronen-Townsend et al. 2002). This approach derived in the query clarity definition, since this predictor ignores large and fluctuating contributions due to generic terms, because of its use of the relative entropy between the collection and the query model (see next section).

There are further examples of the use of entropy for predicting the performance of queries. In (Dou et al. 2007), the authors define the click entropy of a query with respect to the percentage of the clicks on each Web page among all the clicks related with that query, linking this definition with ambiguous queries (those with low click entropy). In (Kurland & Lee 2005), the authors note that high entropy language models may be correlated with a larger number of unique terms, which, in turn, has previously been suggested as a cue for relevance.

Entropy has also been used in rank fusion (Beg & Ahmad 2003), where the position of a document is set according to the value of entropy in a given set of documents (entropy minimisation technique). It has even been used for quantifying the information a measure contains about a particular input (hence, evaluating the evaluation measures themselves) (Aslam et al. 2005).

Finally, in RS we find several works where a maximum entropy model is used, due to its ability of combining different knowledge sources subject to some constraints. In (Jin et al. 2005), the authors use a maximum entropy model for hybrid recommendation, where different users' navigational behaviour (such as page-level and task-level patterns) are combined in order to generate the most effective recommendations for new users with similar profiles. Another example is presented in (Pavlov et al. 2004), where the authors use the maximum entropy distribution to model the probability of the next document requested by a user given a particular history.

2.3.2 Kullback-Leibler distance

In Section 2.2.2, we present the clarity score, which basically measures the distance between two language models. Cronen-Townsend et al. (2002) use Kullback-Leibler distance for this calculation, which has become as a standard measure of divergence in the field of IR. A definition of this distance follows, as well as particular applications of it.

If we have two discrete distributions over the same variable x, $p(x)$ and $q(x)$, the relative entropy or Kullback-Leibler divergence is a measure of the distance between distributions:

$$D_{KL}\left(p(x),q(x)\right)=\sum_x q(x)\ln\frac{q(x)}{p(x)}$$

It is closely related to cross entropy, information divergence and information for discrimination. The relation with respect to cross entropy is the following:

$$H(p,q)=H(p)+D_{KL}(p,q)=-\sum_x p(x)\log q(x)$$

The continuous version of the Kullback-Leibler distance is:

$$D_{KL}\left(p(x),q(x)\right)=\int_{-\infty}^{\infty} q(x)\ln\frac{q(x)}{p(x)}dx$$

The relative entropy is not a true metric because D_{KL} is not necessarily symmetric with respect to p and q. In IR, several authors use this distance as a divergence measure (Carpineto et al. 2001, He & Ounis 2004, Zhai & Lafferty 2001). However, since it is not a metric (because of its asymmetry), some authors modify it as follows:

- Lavrenko et al. (2002b) compute a symmetric version by summing the divergence in both directions: $\mathrm{D}_{\mathrm{KL}}(p,q)$ + $\mathrm{D}_{\mathrm{KL}}(q,p)$
- (Sriram et al. 2004, Carmel et al. 2006) and (Aslam & Pavlu 2007) measure the diversity among distributions using the information-theoretic Jensen-Shannon divergence:

$$JS(p_1,\cdots,p_n)=\frac{1}{n}\sum_j D_{KL}\left(p_j,\bar{p}\right), \bar{p} \text{ is the average of } \left\{p_j\right\}$$

2.3.3 Mutual information

If we want to compare two distributions over possibly different variables we can measure the mutual information, which reflects the reduction in uncertainty about one variable due to the knowledge of the other variable:

$$I(p;q) = H(p) - H(p \mid q) = \sum_{x,y} r(x,y) \log_2 \frac{r(x,y)}{p(x)q(y)}$$

where *r*(*x*,*y*) is the joint distribution of the occurrence of values *x* and *y*. The mutual information measures how much the distributions of the variables differ from statistical independence (it is equivalent to the relative entropy between the joint distribution and the product distribution).

Figure 1 shows the relation between some of the measures shown so far (Duda et al. 2000). For two distributions *p* and *q*, this figure illustrates the mathematical relationships among entropy, mutual information *I(p;q)*, and conditional entropies *H*(*p*|*q*) and *H*(*q*|*p*). For instance, *I(p;p)*=*H*(*p*), and if *I(p;q)=0* then *H*(*q*|*p*)=*H*(*q*).

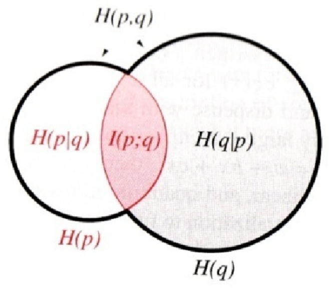

Figure 1. Mathematical relationships among entropy, mutual information, and conditional entropy.

2.3.4 Information gain

The term information gain usually[3] denotes the change in information entropy from a prior state to a state that takes some information as given:

$$IG(Y \mid X) = H(Y) - H(Y \mid X)$$

This is equivalent to the following question: I must transmit *Y*, how many bits on average would it save me if both ends of the line knew *X*? Therefore, it measures the reduction in uncertainty about the value of *Y* when we know the value of *X*.

Different authors have adapted this concept to IR (Pazzani & Billsus 1997, Zhou & Croft 2007, Vinay et al. 2005) and document classification (Montañés et al. 2007). In (Amati & van Rijsbergen 2002), the authors make a simple, direct adaptation of this concept, as follows:

$$IG(t) = \frac{p(t \mid x = t, X = tf) - p(t \mid X = tf)}{p(t \mid x = t, X = tf)} = \frac{p_1 - p_0}{p_1}$$

where $p_1 = \frac{tf + 1}{l(d) + 1}$, $p_0 = \frac{tf}{l(d)}$, *tf* is the number of tokens of term *t* in document *d*, and *l*(*d*) is the length of document *d*. This concept of information gain reflects the portion of infor-

[3] Sometimes Kullback-Leibler divergence is taken as a synonym of this concept. The definition given in this section is drawn from Machine Learning theory.

mation content that is gained with a single token of the term *t* after the observation of *tf* tokens of *t*.

2.3.5 Dempster-Shafer's theory

Dempster-Shafer's theory of evidence introduces the concept of uncertainty in the process of merging different sources of evidence, thus extending the classical probability theory (Shafer 1976, Plachouras & Ounis 2005, Lalmas 1998). In this theory, the set of elements of interest $\Theta = \{\theta_1, ..., \theta_n\}$ is called the frame of discernment. The goal is to represent beliefs in these sets, by defining belief functions $Bel: 2^\Theta \rightarrow [0,1]$. These functions are usually computed based on probability mass functions *m* that assign zero mass to the empty set, and a value in [0,1] to each element of the power set of Θ:

$$m(\varnothing) = 0, \sum_{A \subseteq \Theta} m(A) = 1$$

m is called Basic Probability Assignment (BPA). If $m(A) > 0$ then *A* is called a focal element. The set of focal elements and its associated BPA define a body of evidence on Θ. The belief associated with a set $A \subseteq \Theta$ is defined as:

$$Bel(A) = \sum_{B \subseteq A} m(B)$$

When two bodies of evidence are defined in the same frame of discernment, we can combine them using Dempster's combination rule, under the condition that the two bodies are independent from each other. The rule of combination of evidence returns a measure of agreement between two bodies of evidence. Let m_1, m_2 be the probability mass functions of the two independent bodies of evidence. The probability mass function *m* defines a new body of evidence in the same frame of discernment Θ as follows:

$$m(A) = m_1 \oplus m_2(A) = \frac{\sum_{B \cap C = A} m_1(B) \times m_2(C)}{\sum_{B \cap C \neq \varnothing} m_1(B) \times m_2(C)}$$

Lalmas (1997) introduced this technique in IR, thus defining a model that allows to express uncertainty with respect to the different parts of a document. In (Lalmas 1998) a four-featured model (structure, significance, partiality, uncertainty[4]) is proposed. Lalmas uses the Dempster-Shafer theory to express this model in two steps: first, the initial Dempster's theory is shown to represent structure and significance; second, the refinement function, defined by Shafer, is given as a possible method for representing partiality and uncertainty. The different representations of the document capture the partiality of information. The transformed documents are not actual documents, but consist of more exhaustive representations of the original document. The transformation may be uncertain. A document that requires fewer transformations than another is usually more relevant to the query than the latter.

More recently, Plachouras & Ounis (2005) present the combination of content and link analysis using Dempster-Shafer's theory of evidence, in the context of Web retrieval. Though improvements in retrieval effectiveness are reported, other methods (such as query scope (He & Ounis 2004)) outperform Dempster-Shafer combination of evidence.

[4] The exact information content of a document cannot always be identified appropriately because of the difficulty in capturing the richness and the intensional nature of information. The relevance of a document with respect to a query depends on the existence of information explicit or implicit in the document, so the more uncertainty, the less relevant the document.

2.4 Rank fusion and performance prediction

Rank fusion (also known as "rank aggregation") is a typical problem where performance prediction has been applied. Rank fusion takes place when several retrieval outputs from different sources need to be combined into a single aggregated retrieval result set (Fox & Shaw 1993, Dwork et al. 2001). Examples where rank fusion is applied include metasearch, distributed search, personalised retrieval (where preference-based relevance is combined with query-based results), multi-criteria retrieval, and so forth, as introduced at the beginning of the chapter.

There are two main inherent problems to address in rank fusion: each source may use different methods to return the documents (by similarity or dissimilarity with respect to the query, by counting term frequencies or by computing probabilistic methods, and so on), and the rank scores for each document can be different for each source, or even not be available at all for the rank aggregator (as is the case with commercial Web search engines). For these reasons, the fusion problem is usually divided into two steps: 1) normalisation, where score values (or rank positions, if the latter is not known) are mapped to a common range before the next step, and 2) combination, where, after normalisation, the ranked lists are actually merged into one. The IR literature is profuse in proposed methods to achieve these two steps. An important distinction among them is whether they use rank scores, or only rank positions. Another key aspect is whether or not training data is required. While most methods explicitly separate normalisation and combination, some achieve both in a single operation (Renda & Straccia 2003). When the steps are separate, an important family of combination techniques are based on a linear combination (and variants thereof) of the normalised rank scores to be merged. Performance prediction methods have a clear potential usefulness for rank fusion, more specifically in the combination step, since they may help decide which of the different sources is more trustworthy and select a weight for each of them accordingly in order to merge their results, particularly when the merging function takes the form of (or is based on) a linear combination of score values.

Yom-Tov et al. (2005a, 2005b) were among the first to introduce performance prediction in the context of rank aggregation. The main hypothesis in that work is that queries that are answered well by a search engine are those whose query terms agree on most of the returned documents. The agreement is measured by the overlap between the top results for the full query and the top results for each of the query terms. Building on this, the proposed predictor is based on checking these overlaps between the results of the full query and its subqueries (queries based on a single term from the original query), and it is induced from training queries and their associated relevance judgments using a histogram-based predictor and a decision tree based predictor. In their models the authors use features such as the document frequency of query terms and the overlap of top retrieval results of the full query with those obtained with the individual query terms. They report promising results and showed that their methods are more precise than those used in (Kwok et al. 2004, Plachouras et al. 2004).

Another novel and promising point of view about performance prediction is the definition of query hardness in (Aslam & Pavlu 2007). This technique is based on examining the ranked lists returned by multiple retrieval engines for a given query on the same collection. The authors hypothesise that the results returned by multiple systems would be relatively similar for easy queries but more diverse for difficult queries. The hypothesis and the proposed method are validated by the correlations between predictor outputs and average performance. Although the method is based on a multi-system setting where different engines are available, the findings were not tested at any further application (such as controlling the combination of results).

In (Wu & Mcclean 2006), multiple regression analysis is used to identify the variables (such as the number of results per system, the overlap rate among a group of results, the mean average precision, etc.) that affect performance of data fusion algorithms. In that work, performance prediction requires training data.

Diaz (2007) proposes a technique called spatial autocorrelation for performance prediction when more than one retrieval engine is available. This technique measures the degree to which the top ranked documents receive similar scores (also named as autocorrelation by the author). This approach is based on the cluster hypothesis (Jardine & van Rijsbergen 1971), according to which closely-related documents tend to be relevant to the same query. The autocorrelation of each system is computed as a Pearson's correlation between the scores of that system and some heuristic combination of all the available retrieval functions. The author reports a significant correlation between the autocorrelation of a system and its retrieval performance.

In (Castells et al. 2005) a heuristic personalised model is presented. As an example of rank fusion, it is interesting how the authors outline the introduction of performance predictor, as an automatic combination factor, where concepts such as ambiguity or risk (of user actions in a particular context) would be very useful to infer it.

In the context of rank aggregation more than one source of information (retrieval system) is used. Therefore, to evaluate the performance of the aggregated system, we have to compute some performance combination involving the whole set of retrieval systems. The most common approach is the average (or median) performance, although sometimes the best subsystem is taken as the baseline. Performance calculation is done using the techniques explained in Section 2.2.1 for each retrieval system.

Other possible method is to combine the retrieval systems naively (randomly selecting each source or using a fixed combination factor), evaluate the performance of the result and take it as the baseline (Renda & Straccia 2003). Another approach is the one found in (Dwork et al. 2001), where the authors calculate performance in terms of three distance measures. Each of these measures involucrate different lists for its computation, so there is no need to calculate separately the performance of each source, and then aggregate these performances.

2.5 Recommender systems

Another typical situation where important steps are achieved by aggregative operations is the area of RS. More than many other retrieval systems, RSs combine multiple data and strategies, and the balance is critical for the final performance. In this section, we briefly recall the main concepts in RS, including basic definitions and algorithms, and discuss specific considerations regarding performance in this area.

The aim of recommender systems is to assist users by suggesting "interesting" items from huge databases or catalogues, by taking into account (or inferring) user's priorities or tastes. Three types of systems are commonly recognised, based on how recommendations are made (Adomavicius & Tuzhilin 2005, Cantador et al. 2008a): collaborative filtering, content-based filtering and hybrid filtering. CF recommends the user items that people with similar tastes and preferences liked in the past, CB recommends items similar to the ones the user preferred in the past, and HF combines collaborative and content-based strategies. Although many alternatives are possible, the most common form of ground evidence of user preferences, upon which recommendations are generated, consists of explicit user ratings for individual items. Three more recommendation techniques are also commonly distinguished

(Burke 2002): demographic, utility-based and knowledge-based, although they can be considered secondary to (in fact often integrated into) CF and CB.

Recommender systems have well-known inherent weaknesses. For example, if the system has little rating data about a particular item, i.e., few users have rated it, CF approaches do not perform well; on the other hand, if item descriptions are not available, CB approaches can hardly give accurate recommendations. Table 1 shows a summary of the different problems suffered by CB and CF approaches. HF systems are out of this analysis since they allow compensating the shortcomings of one approach by the strengths of the other, unless both suffer from the same problem.

In the HF approach, the most important decision is how to combine information coming from CF and CB. In (Burke 2002) a detailed taxonomy is presented, where HF approaches are classified into meta-level hybrid recommenders (using the entire generated model by one of the techniques as the input for the other), cascade hybrid recommenders (recommendation is performed as a sequential process), weighted hybrid recommenders (scores are aggregated using linear combination or voting schemes), or switched hybrid recommenders (special case of the previous one, where one technique is turned off whenever the other is turned on).

Problem	**Description**	**CF**	**CB**
Grey sheep	A user whose tastes are unusual compared to the rest of the population	Yes	No
Sparsity	Number of available ratings is small	Yes	No
New item	Items to be recommended must be rated by a substantial number of users	Yes	No
New user	A user has to rate enough number of items in order to infer her preferences	Yes	Yes
Restricted content analysis	Items to be recommended must have data available related with their features	No	Yes
Overspecialisation	All the recommended items are similar to those already rated	No	Yes
Portfolio effect	An item is recommended even if it is too similar to an item previously rated	No	Yes

Table 1. List of common problems in CF and CB systems.

According to (Breese et al. 1998), CF algorithms can be grouped into two general classes: *memory-based* (or *heuristic-based*) and *model-based*. Memory-based (a.k.a. heuristic) algorithms predict ratings based on the entire collection of previously rated items by the user. If it is a user-based approach, then a user neighbourhood is required, in order to predict a user's rating for an item from the ratings on that item by the user's nearest neighbours. The neighbourhood can be computed using different alternatives, although it is usually determined by the pairwise user similarity as a distance function, and selecting as neighbours the *n* closest users according to this distance. In the item-based approach, the predicted user rating for an item is calculated similarly, using the ratings given by the user on the neighbourhood of the target item. Although it is not frequently used, the item neighbourhood is selected in an analogous way, but using an item similarity function, instead. When no item neighbourhood is used, every item rated by that user is taken into account .

Model-based approaches use the collection of ratings to learn a model, which is then used to make rating predictions. Different approaches have been proposed, based on methods such as probabilistic collaborative filtering (Yu et al. 2004), neural networks (Pazzani & Billsus 1997), maximum entropy models (Pavlov et al. 2004), and others.

Predicting the performance of subcomponents of a recommender system requires specific reference metrics for the performance, against which the predictions can be compared. The standard evaluation metrics in the IR field (introduced in section 2.2.1) can be generally used in this context. However, recommender systems have important specific characteristics of their own (in the way the retrieval problem is stated, handled, and solved, and even the way performance is precisely understood), which call for additional, specialised metrics.

The evaluation of recommender systems has indeed been – and remains – the object of a considerable body of research. See (Mcnee et al. 2006) for an overview and discussion on current methodologies, and new proposed directions. In the RS literature, performance has been usually equated to accuracy, that is, the estimated rating is compared against the actual rating, and so the *mean average error* (MAE, average deviation of the predicted rating from the real rating) and the *root mean squared error* (RMSE, square root of square deviations) are the most widely used measures.

But the definition of performance should take into account the goal of the system itself (Herlocker et al. 2004). For example, Herlocker et al. (2004) identify two main user tasks: annotation in context and finding good items. In these tasks, users only care about errors in the item rank order provided by the system, not the predicted rating value itself. Based on this consideration, researchers have started to use precision and recall to evaluate recommendations, although most works also use MAE or RMSE for comparison with the state of the art. Herlocker et al. encourage considering alternative performance criteria, such as the scalability of the proposed system or the coverage of the method. An alternative evaluation using Machine Learning techniques is proposed in (Bellogín et al. 2008, 2009), which aims to discover what parts of the recommender system are more relevant to provide accurate recommendations to the user.

Some hybrid applications , such as cascade, may require a more specific evaluation, like the one presented in (Burke 2002). Burke uses a cascade technique (collaborative system refines the candidates left under-discriminated by the knowledge-based system), where the goal of the collaborative filtering was not to predict ratings, but rather to improve the quality of the recommendations made by the knowledge-based component.

2.6 Discussion

The topic of performance prediction in IR is an active area of research. There is so far no clear definitive approach, better than the others, and the field remains open for further investigation. Few, if any, applications have been documented to take advantage of principled performance prediction methods, beyond research experiments, and we therefore see an ample potential to be exploited in this area.

The field of Recommender Systems, as a specific case of personalised IR at large, is a particularly propitious area for the introduction of performance prediction techniques, because of the naturally arising need for combination of multiple diverse evidence and strategies, and the uncertainty (and thus the variable accuracy) involved in the exploitation of implicit evidence of user interests. However, to the best of our knowledge, the introduction of performance predictors in this area has not been researched as a formal problem. The issue is nonetheless mentioned recurrently in the literature as a potential idea, which is an evidence of its interest, sometimes addressed as matter of ad hoc, heuristic, and/or manual tuning. For instance, Wang et al. (2006) mention the idea of treating "each individual rating in the user-item matrix as a separate prediction for the unknown test rating", in such a way that similarity towards the test user and towards the test item can be taken as an estimation of the confidence of each individual prediction. In (Herlocker et al. 1999), different variants of the standard CF algorithm are proposed which can be recognised as ad hoc predictors, in particular, significance weighting and variance weighting, which aim to "measure the amount of trust to be placed in a correlation with a neighbour". But as pointed out before, it is introduced as a manual, informal improvement by intuition, trial and error.

In this chapter, we have shown several performance predictors in the field of IR. As we describe in the next chapter, we shall focus on clarity-based predictors, since they have improved performance in several works, and they link to Language Models, a powerful IR theory that has gained increasing significance in the field in the last decade. Besides that, they are more general than other predictors (such as WIG), in so far as they have been adopted in further works, thus proving their adaptability to very different contexts.

Metasearch and personalised IR remain as further suitable areas where performance prediction deserves been researched. We nevertheless focus on RS in this work as a starting point of interest, because of the novelty of the approach in this area, leaving the study of rank fusion problems in other areas (or at more general levels) as future work.

Chapter 3. Proposed approach

3.1 Introduction

In this chapter, we formalise the framework where the combination problem, the performance predictors, and the involved information spaces are identified and the problem is formulated. Upon this, we address the actual definition of performance predictors for several retrieval components in specific combined RS methods, namely, neighbour weighting in CF and the dynamic combination of CB and CF recommenders in hybrid recommendation. The predictors defined here draw from two main areas: Information Retrieval (using techniques such as clarity and others ad hoc predictors, revised in the previous chapter) and Information Theory (such as information gain).

Section 3.2 introduces the formalisation framework to be used in the rest of the chapter. Section 3.3 defines and formulates the specific combination problems in RS that are addressed. Section 3.4 defines a set of performance predictors to be used in the latter, and Section 3.5 summarises which predictors are actually proposed for each of the methods. The effect of the predictors on the methods are tested in the experiments reported in Chapter 4.

3.2 Formal framework

We start by laying out a formal abstract setting upon which the problem is stated and refined, and our proposed solutions are built. The formulation developed henceforth extends the one in the introduction into further detail, suiting the specific areas where the methods will be developed.

Considering a retrieval space consisting of a set $\mathcal{D}$ of items and a set Ω of input data, the retrieval problem can be stated in broad terms as defining a total order $\leq_q$ (a ranking) in a subset of $\mathcal{D}$ for each (or for a given) input value $z \in \Omega$. $\mathcal{D}$ represents a set of elements (documents, movies, books, goods, people, etc., depending on the application domain) among which a selection is to be made, and the elements of Ω represent any situational information that drives that decision (or is relevant for the latter), such as user queries, user history, user ratings, demographic user data, social data, location, time, external events, and/or any further relevant contextual information that the retrieval system is required or capable to take in, depending again on the application domain, and the particular retrieval strategies and techniques being implemented. For instance, in a standard search engine, we would have $\Omega = \mathcal{Q}$ the set of all queries. In a recommender system, we might essentially have $\Omega = \mathcal{U}$, the set of all users (more complex formulations can indeed be considered). In a personalised search system, we might have $\Omega = \mathcal{Q} \times \mathcal{U}$.

The IR field is rich in the variety of approaches and models upon which the ranking $\leq_q$ and its computation are obtained, such as probability (van Rijsbergen 1986, Hiemstra 1998, Robertson 1997), utility theory (Zhai & Lafferty 2006, Pasi & Marques 1999), heuristics (Cao et al. 2006), etc., many of them drawing from theories in adjacent fields (such as Classification and Pattern Matching, Information Theory, Fuzzy Algebra, etc.). Among them, in our present work, we take the case where the ranking is defined in terms of a utility function

$g : \mathcal{D} \times \Omega \rightarrow \mathbb{R}$, that is, $d_1 \leq_q d_2 \Leftrightarrow g(d_1, \mathfrak{z}) \leq g(d_2, \mathfrak{z})$, which matches a large body of models and techniques in the area (such as the VSM or common memory-based recommender approaches). Other formalisms (such as probabilistic models) can moreover be mapped to this form (Gordon & Lenk 1991).

3.2.1 Aggregative IR systems

Our research addresses the case where the retrieval system has a composite nature, that is, it is built as a high-level composition of subcomponents that implement different retrieval criteria or strategies. We assume that this aggregative relation can be expressed by a composition at the functional level. That is, the utility function of the global system can be expressed as $g(d, \mathfrak{z}) = \varphi\ (g_1(d, \mathfrak{z}), ..., g_n(d, \mathfrak{z}))$, where g_j represent the utility functions of the subsystems. Note that it is usual that not all g_j need to use all of d and $\mathfrak{z}$ as input. For instance, in a personalised search system, it may well be the case that we have $g(d,(q,u)) = \varphi\ (g_1(d,q), g_2(d,u))$. The previous formulation nonetheless provides a compact notation and wide generality.

To further focus our research, we assume φ is a linear combination $\varphi\ (s_1, ..., s_n) = \alpha_1\, s_1 + \cdots + \alpha_n\, s_n$. This covers a wide range of important retrieval operators in areas such as metasearch, distributed search, personalised search or, as will be studied here, hybrid recommender systems and core collaborative filtering operators such as the aggregation of neighbour ratings. It is very interesting to note that in (Pennock et al. 2000) the authors prove that the only combination function satisfying several constraints for CF (such as every user is seen as an utility function) is the weighted average sum.

Actually, several examples of such a combination can be found in the area of IR. For example, in rank fusion one of the most used aggregating method is *CombSUM*, in which the score of each document is calculated as a sum over all the scores given to that document by all the sources. In standard document retrieval, the score of each document comes from a (normalised) dot product. Examples from the RS area are discussed in the next section. It is worth noting that this formulation also allows multi-criteria IR, in the sense that personalised retrieval or more complex models are also included. For example, a retrieval system where personalisation and recommendation are incorporated into the basic retrieval algorithm can be understood as a system with three main components (personalisation, recommendation, ad hoc retrieval), where aggregation is considered as a last step.

3.2.2 Predictive dynamic adjustment of combined methods

Given this formulation, our research addresses the problem of dynamically assigning the weights α_j in the combination in such a way that optimises (or improves a baseline in) the performance of the resulting system. The α_j coefficients determine the dominance that each g_j subcomponent shall have in the aggregated retrieval decision. Since the quality and performance of the joint decisions depends on that of g_j to an extent determined by α_j, there is an opportunity to gain performance by favouring the influence of the g_j that supply a better quality output in each situation.

This requires some predictive capability regarding the individual performance of g_j given d and $\mathfrak{z}$ – in fact the prediction can exploit any further information available at retrieval time, which we may account for as part of Ω. If the predicted performance of g_j can be defined and quantified by a function $\gamma_j\ (d, \mathfrak{z}, \Omega)$, then it is possible to set $\alpha_j = \gamma_j\ (d, \mathfrak{z}, \Omega)$ in a way that the components' output is taken into account to the extent that they are predicted to perform better or worse. As we shall see in the sequel, Ω may include for this purpose even further information than has been suggested, such as the output of all g_j for a given input z, that is $g_j(\mathcal{D} \times \{z\})$ for all j, of which function γ might take advantage.

Note also that while we shall ultimately require that the coefficients are such that $\sum_{j=1}^{n} \alpha_j = 1$, we shall obviate (i.e. not satisfy) this condition, without loss of generality, in the rest of specifications defined here, but it shall be enforced in the implementation, by dividing all α_j by their total sum in a last step. This is important as far as the output of g must remain within a range with specific meaning (such as item ratings on a certain scale).

In the next section, we describe how this framework can be used in the field of RS, as well as some specific applications.

3.3 Dynamic fusion in Recommender Systems

We continue the specification of the proposed approach with its particularisation to the area of recommender systems. In this section, we identify two specific core aggregative methods in the area, and show how they fit in the aggregation framework defined in the previous section. Afterwards in section 3.4 we shall define in detail specific proposed predictors to be used in each combination operator.

In the context of recommender systems, the retrieval space $\mathcal{D} = \mathcal{I}$ is the set of all possible items that can be recommended, such as books, movies, restaurants, etc. The input space is $\Omega = (\mathcal{U}, r)$, where $\mathcal{U}$ is the set of all users, and $r : \mathcal{S} \rightarrow \mathcal{R}$ with $\mathcal{S} \subset \mathcal{U} \times \mathcal{I}$ provides a set of user ratings for items (which indicate how a particular user liked a particular item), $\mathcal{R}$ being an ordered set of possible rating values, such as real numbers within a certain range. More complex formulations are common, and indeed needed in real applications, although this simplified version (or equivalent ones) is pervasive in the research literature, as a suitable and powerful abstraction to focus on the essential principles without loss of generality. In this setting, the recommendation problem is, thus, defined as finding the best ranking over $\mathcal{I}$ for every element of $\mathcal{U}$. Instead of finding the whole ranking, the recommender is sometimes just required to find the best item for each user (equivalent to top-1). This can be formulated as follows (Adomavicius & Tuzhilin 2005):

$$g : \mathcal{U} \times \mathcal{I} \rightarrow \mathcal{R}$$
$$\forall u \in \mathcal{U}, i_u^* = \arg\max_{i \in \mathcal{I}} g(u,i)$$

In this scenario, the utility of an item is usually equated to a rating (actual or predicted) by the user for the item. In this way, $g : \mathcal{U} \times \mathcal{I} \rightarrow \mathcal{R}$ is an extension of r, i.e. $g|_{\mathcal{S}} = r$.

In the subsections that follow we shall identify and select two combined methods in the context of RS which will be formulated and treated under this framework. Specific performance predictors will be defined later in Section 3.4 for each of the methods.

3.3.1 Neighbour weighting in CF

In this formulation, each user's neighbour is taken as a retrieval subsystem (or criteria) to be combined. This is an implicit idea in the behaviour of collaborative filtering algorithms[5], since once a neighbourhood is determined, the algorithm predicts the rating of (estimates the utility for) the active user for a particular item taking into account both the rating given to the item by her neighbours, and the similarity between the neighbour and the current user.

According to this, such a formulation fits our aggregative model in the following way[6]:

$$g(u_m, i_k) = \varphi_{N[u_m]}\left(g_1(u_m, i_k), \cdots, g_n(u_m, i_k)\right)$$
$$g_j(u_m, i_k) = \mathrm{sim}(u_m, v_j) \times r(u_j, i_k)$$
$$\alpha_j = \frac{1}{\sum_{v \in N[u_m]} |\,\mathrm{sim}(u_m, v)\,|}$$

where n is the size of the user neighbourhood $N[u_m]$, and $g_j(u_m,i_k)$ represents the utility for item i_k and user u_m according to the neighbour v_j. This utility model is completely equivalent to one of the most common approaches in CF (Adomavicius & Tuzhilin 2005):

$$g(u_m, i_n) = \frac{\sum_{u_j \in N[u_m]} \mathrm{sim}(u_m, u_j) \times r_{j,n}}{\sum_{u_j \in N[u_m]} \left|\mathrm{sim}(u_m, u_j)\right|}$$

Thus in the above interpretation, the combination of neighbour ratings in standard CF algorithms is computed with fixed and equal α_j weights for all neighbours. This is where we propose to make the weights dynamic based on predictions of the performance of each neighbour in its contribution to the recommendation (utility function value) to the current user. Based on the formulation defined in the previous section, we shall propose and test performance predictors γ_j based on properties of user u_j, and set $\alpha_j = \gamma_j(.)$ as the weight of $g_j(u_m,i_k)$. We shall explore alternative strategies in which γ_j depends on different inputs, such as $\gamma_j(u_m,i_k,v_j)$, where the predictor involves information about the current user, the neighbour, and the current item, $\gamma_j(u_m,v_j)$, where information about the item is removed, or $\gamma_j(v_j)$, where only information about the neighbour is used. Different options considered will be presented in Section 3.4, while in Section 3.5, we show which of these predictors are actually used for each application.

3.3.2 Weighted hybrid recommendation

A second major combined method in the area is hybrid recommendation, which combines a CF and a CB algorithm. Hybridisation can be done in several ways but, in agreement with the stated assumptions and scope of our research, we consider the weighted hybrid approach

[5] In (Herlocker et al. 2002), the authors compare performance between those approaches which weight neighbour contributions and those which do not perform this weighting, and they conclude that weighting the contributions of neighbours improves the accuracy of the predictions.

[6] Note that the sim(u_m,v_j) part of $g_j(u_m,i_k)$ above is already acting as a performance predictor weighting the linear combination, where $r_{j,n}$ plays the role of user u_j's criteria regarding the utility of i_k (i.e. she would recommend it or not to u_m to the degree reflected by $r(u_m,i_k)$), where the more u_m and v_j are alike, the better v_j's advice can be predicted to be good for u_m. However, in our proposal, this predictor is understood to be an integral part of the subsystem, and an additional prediction-based weight is introduced.

(Burke 2002), since it allows to introduce combination weights which can be dynamically modified. In the simplest and typical case, hybrid recommendations are produced by weighting and summing the utility values outputs from CF and CB. In a more general view, an arbitrary number of recommenders of different kinds (user-based CF, item-based CF, CB, demographics-based, etc.) can be combined.

Whereas in the previous section the fusion operator is internal to a CF method, here the aggregation takes place at the last step of the recommendation:

$$g(u_m,i_k) = \varphi^{RS}\left(g_{RS_1}(u_m,i_k),\cdots,g_{RS_s}(u_m,i_k)\right)$$
$$g_{RS}(u_m,i_k) = \varphi_{N[u_m]}\left(g_{v_1}^{RS}(u_m,i_k),\cdots,g_{v_n}^{RS}(u_m,i_k)\right)$$

where we have *s* different recommender systems. In order to test in isolation the dynamic hybrid recommendation, we define the following framework:

$$\alpha_j^{RS} = \gamma_j(u_m,i_k,RS)$$
$$\alpha_j = 1 \text{ for each RS}$$

that is, weights of each recommender are not taken into account, and performance predictors will only be used for weighting each recommender, which can use any information coming from that recommender. Namely, if we have only CB and CF, the problem fits in the framework by:

$$g(u_m,i_k) = \alpha_{CB} \times g_{CB}(u_m,i_k) + \alpha_{CF} \times g_{CF}(u_m,i_k)$$
$$\alpha_{CB} = \gamma_{CB}$$
$$\alpha_{CF} = \gamma_{CF}$$

The elements in this formula have already been formalised ($g_{CF}(u_m,i_k)$ is equivalent to the previous $g(u_m,i_k)$), except for $g_{CB}(u_m,i_k)$. This is the utility value of the item i_k for the user u_m according to some content-based algorithm. Since it is unnecessary for our method to formalise this function, we do not explain it here. Nevertheless, a content-based utility function can be derived through a similar process to that shown in the previous section.

The key point in this experiment is that the value of γ_{CB} and γ_{CF} will be (potentially) different for each pair (user, item), instead of the typical scenario when static linear combination is used. As in the previous section, different predictors can be used depending on what variables take into account (current user, current item, or neighbour, as before). However, since we deal now with a content-based recommender, we can extend these options to the content of the item (characterised by some features, such as genre or actors in a movie, for example), the current user profile (as a set of items), or the neighbour profile. Again, the different alternatives are presented in Section 3.4 and 3.5.

3.4 Performance predictors for recommender system components

In this section, we present the different performance predictors used along this work. As introduced in previous sections, they are based on the background principle that the amount of uncertainty present in the input data is a good (inverse) predictor of the performance the retrieval system can achieve. This principle is embedded in some techniques already researched and tested in other IR fields (such as ad hoc retrieval) as discussed in Section 2.2, the adaptation of which we study here, in the context of RS. This principle is also the basis for further approaches we consider to build additional predictors. Finally, a list of heuristic predictors shall be selected or defined for the final experimental part of the work.

Thus, this section deals with the definition of the γ functions, the purpose of which is to effectively weight subcomponents of retrieval systems, in the aggregative methods selected and examined in the previous section, in order to improve their performance.

In the next subsection, we define some predictors building on the concept of clarity (borrowed from query performance analysis) introduced in Section 2.2.3. In Section 3.4.2, we use Information Theory in order to define uncertainty detectors. Finally, in 3.4.3 we present other predictors used in this work, not so formally derived, but based on ad hoc concepts employed in IR. Section 3.5 links the list of predictors defined in this section, with the applications of performance prediction of the previous section.

3.4.1 Clarity-based predictors

Inspired by the clarity score defined by Cronen-Townsend et al. in (Cronen-Townsend et al. 2002), we consider its adaptation to a recommender environment. The original clarity score for Web retrieval is defined as follows:

$$\text{clarity}(q) = \sum_{w \in V} p(w \mid q) \log_2 \frac{p(w \mid q)}{p_c(w)}$$

where the three following key elements are involved:

- $w \in V$: the summation is performed over all the words in the vocabulary, since a query (the element of interest) is composed by words.
- $p(w|q)$: this term defines the query language model.
- $p_c(w)$: this term establishes the collection language model.

This predictor measures the lack of ambiguity of a particular query, by computing the distance between the query and the collection language model. In RS, we are interested in how ambiguous a user is in terms of her particular tastes. In the rest of this section, we adapt this concept and present different ways for measuring the lack of user ambiguity (clarity).

In order to translate this concept to RS, we need to map (and subsequently define) the presented elements to corresponding variables of the RS setting in a meaningful way. There are many possible mappings, which we have studied before settling for the formulation proposed herein. First, we consider the definition of a clarity score for a user. In this case, the user is equivalent to the query in the original formulation. Next, we identify the “words” constituting a user. One (common) approach in RS is thinking of a user a set of items – those rated by the user. Based on this rationale, we get the following formula:

$$\text{clarity}(u) = \sum_{i \in I} p(i \mid u) \log_2 \frac{p(i \mid u)}{p_c(i)}$$

Now we need to define the user and collection language models. The latter is trivial, since it is usually defined as the relative frequency of an item in the whole collection:

$$p_c(i) = \frac{1}{|I|}$$

For the user language model, we should use the relative frequencies of items in users, but since each user does not rate the same item more than once, we modify it in order to include in the formulation the rating value of the user for that item:

$$p(i|u) = \lambda \frac{rat(u,i)}{5} + (1-\lambda) p_c(i)$$

Besides that, this term is linearly smoothed with the collection frequency of that item. If we write all this together, we have the following:

$$\begin{aligned} \text{IUC}(u) &= \sum_{i \in I} p(i|u) \log_2 \frac{p(i|u)}{p_c(i)} \\ p(i|u) &= \lambda \frac{rat(u,i)}{5} + (1-\lambda) p_c(i) \\ p_c(i) &= \frac{1}{|I|} \end{aligned}$$

which we define as *item-based user clarity* (IUC).

We have to note that clarity can be defined differently, simply by interpreting the "words" in a different way. For example, a user can be modelled as a set of users who rated similar items to those she rated[7]. In this view, and following similar steps to those explained before, we derive the following formulation[8]:

$$\begin{aligned} \text{UUC}(u) &= \sum_{v \in U} p(v|u) \log_2 \frac{p(v|u)}{p_c(v)} \\ p(v|u) &= \sum_{i:rat(u,i) \neq 0} p(v|i) p(i|u) \\ p(v|i) &= \lambda \frac{rat(v,i)}{5} + (1-\lambda) p_c(v) \\ p_c(v) &= \frac{1}{|U|} \end{aligned}$$

which we define as *user-based user clarity* (UUC). Analogously, we can define item clarities:

- *Item-based item clarity* (IIC):

$$\text{IIC}(i) = \sum_{u \in U} p(u|i) \log_2 \frac{p(u|i)}{p_c(u)}$$

- *User-based item clarity* (UIC):

$$\begin{aligned} \text{UIC}(i) &= \sum_{j \in I} p(j|i) \log_2 \frac{p(j|i)}{p_c(j)} \\ p(j|i) &= \sum_{u:rat(u,i) \neq 0} p(j|u) p(u|i) \end{aligned}$$

---

[7] This can be thought of as an item-based view in CF.

[8] In this process, a simplification has been made: instead of defining $p(v|i)$ as $p(i|v)p(v)/p(i)$, we smooth the normalised rating value linearly with the collection frequency, similarly to the smoothing in the definition of $p(i|v)$.

3.4.2 Information-theoretic predictors

We have found Information Theory concepts such as entropy or mutual information very tough to define properly in a recommender environment. Because of that, we only present here the translation of the information gain concept to RS:

- *User information gain* (UIG):

$$\text{UIG}(u) = \frac{p_1^u - p_0^u}{p_1^u}$$

$$p_1^u = \frac{\left|\left\{r : r = rat(v,i) \neq 0, i \in \left\{i : rat(u,i) \neq 0\right\}, \forall v \in U\right\}\right|}{|R|}$$

$$p_0^u = \frac{\left|\left\{r : r = rat(v,i) \neq 0, i \in \left\{i : rat(u,i) \neq 0\right\}, \forall v \in U\right\}\right|}{|R|} - \frac{\left|\left\{r : r = rat(u,i) \neq 0\right\}\right|}{|R|}$$

where R is the set of all ratings. This formulation is inspired in one of Amati's work (Amati & van Rijsbergen 2002), explained in Section 2.3.4.

- *Item information gain* (IIG):

$$\text{IIG}(i) = \frac{p_1^i - p_0^i}{p_1^i}$$

$$p_1^i = \frac{\left|\left\{r : r = rat(u,j) \neq 0, u \in \left\{u : rat(u,i) \neq 0\right\}, \forall j \in I\right\}\right|}{|R|}$$

$$p_0^i = \frac{\left|\left\{r : r = rat(u,j) \neq 0, u \in \left\{u : rat(u,i) \neq 0\right\}, \forall j \in I\right\}\right|}{|R|} - \frac{\left|\left\{r : r = rat(u,i) \neq 0\right\}\right|}{|R|}$$

3.4.3 Heuristic predictors

In our research we have tested further predictors, not based on formal magnitudes with theoretic grounds, as the ones defined in the previous two subsections, but following heuristic approaches. We implement these predictors either because they are used in the literature (such as IUF), or in order to emphasise some particular aspect (e.g., neighbourhood similarities, item features) of an algorithm in a comparative way.

A first group of predictors is essentially based on Breese's IUF (*inverse user frequency*) (Breese et al. 1998) and some variations over the same concept. Breese et al. define IUF as:

$$\text{IUF}(i) = \log \frac{|U|}{\left|\left\{u : rat(u,i) \neq 0\right\}\right|}$$

The authors used this value to capture the idea that universally liked items are not as useful in capturing similarity as less common items; in this way, they modified the correlation value (similarity function), multiplying it by the IUF factor. Similarly, we define the following functions, with potential predictive power:

- *Inverse item frequency* (IIF): $\text{IIF}(u) = \log \frac{|I|}{\left|\left\{i : rat(u,i) \neq 0\right\}\right|}$

- *Inverse item rating frequency* (IIRF): $\mathrm{IIRF}(i) = \log \frac{|R|}{|\{u : \mathrm{rat}(u,i) \neq 0\}|}$, where R is the set of all ratings.
- *Inverse user rating frequency* (IURF): $\mathrm{IURF}(u) = \log \frac{|R|}{|\{i : rat(u,i) \neq 0\}|}$

In a second group, we have defined predictors related with CB algorithms rather than CF, such as the following:

- *Item features* (IF): $\mathrm{IF}(i) = \sum_{t \in \mathfrak{T}} \text{TF-IDF}(i,t)$, where $\mathfrak{T}$ is the set of different item features. This predictor sums over each feature in the feature space the feature's weight in that item.
- *User features* (UF): $\mathrm{UF}(u) = \frac{\sum_{t \in \mathfrak{T}} \text{TF-IDF}(u,t)}{|\{i : \mathrm{rat}(u,i) \neq 0\}|}$, in this way, we have a summation (over all the different features) of the content of each user (a user is seen as a vector of items), normalised by the number of items that user has rated.

3.5 Implementation and experimental work

In the previous section, different performance predictors have been defined and adapted from different areas. In Section 3.3, we have presented two prototypical operations in RS, namely neighbour rating combination, and CF + CB hybridisation, where performance prediction can be used. Nevertheless, not every predictor of the previous section is appropriate for both operations. For example, feature-based predictors only make sense in CB algorithms (thus they are not applicable to neighbour weighting).

Besides these theoretic questions, for the sake of clarity, in the experiments section we shall not present the results with *all* possible predictors applicable in the two studied operations, but we focus on the combinations which work best (for at least one of the operations). We therefore summarise in this section the list of predictors whose results will be shown for each of the two combination problems. The framing combination problem provides a proper, additional context for the definition of γ and α, which is needed in order to make full sense of them, and make their notation fully specific.

Neighbour weighting

Recalling Section 3.3.1, the combined method is:

$$g(u_m,i_k) = \varphi_{N[u_m]}\left(g_1(u_m,i_k),\cdots,g_n(u_m,i_k)\right)$$
$$g_j(u_m,i_k) = \mathrm{sim}(u_m,v_j) \times r(u_j,i_k)$$
$$\alpha_j = \frac{\gamma_j}{\sum_{v \in N[u_m]} |\mathrm{sim}(u_m,v)|}$$

For this method, the results with the following predictors will be shown:

- *Item-based user clarity*: $\gamma_j = \gamma_j(v_j) = \mathrm{IUC}(v_j)$

- *User-based user clarity*: $\gamma_j = \gamma_j(v_j) = \mathrm{UUC}(v_j)$

Hybrid recommendation

As defined in Section 3.3.2, the combination function is:

$$g(u_m, i_k) = \alpha_{CB} \times g_{CB}(u_m, i_k) + \alpha_{CF} \times g_{CF}(u_m, i_k)$$
$$\alpha_{CB} = \gamma_{CB}$$
$$\alpha_{CF} = \gamma_{CF}$$

In this case, we have two different γ functions, and therefore, each one can be assigned a different performance predictor. Because of this, we analyse these possibilities separately:

- Content-based component (γ_{CB})
  - *Item features*: $\gamma_{CB} = \gamma_{CB}(i_k) = \mathrm{IF}(i_k)$
  - *Item information gain*: $\gamma_{CB} = \gamma_{CB}(i_k) = \mathrm{IIG}(i_k)$
  - *Item-based user clarity*: $\gamma_{CB} = \gamma_{CB}(v_j) = \mathrm{IUC}(v_j)$
- Collaborative component (γ_{CF})
  - *Item-based item clarity*: $\gamma_{CF} = \gamma_{CF}(i_k) = \mathrm{IIC}(i_k)$
  - *User-based user clarity*: $\gamma_{CF} = \gamma_{CF}(v_j) = \mathrm{UUC}(v_j)$

We also consider the static baseline as a predictor.

- Static baseline, with λ parameter: $\gamma_{CB} = \gamma_{CF} = \gamma^{\lambda} = \lambda$

The experiments with the above performance predictors and combination functions are described in the next chapter. Two type of tests are applied for each predictor and its use in the combinations, in order to assess, confirm and measure their good behaviour.

Chapter 4. Experimental work

4.1 Introduction

In this chapter, the proposed predictors and methods presented in previous chapters are tested empirically in two types of experiments. First, the predictors are introduced in the two methods already explained in Section 3.3: neighbour weighting and dynamic hybrid recommender, and the resulting variations in performance are examined. In addition to this, the correlation between predictors and performance metrics is measured and analysed.

The experiments presented here are focused on the predictors that have led to clear and positive results. Among them, we prioritise the analysis for the clarity-based or information-theoretic ones, above other ad hoc predictors. The evaluation of the remaining predictors has not yet reached conclusive results, and is currently under revision as work in progress.

We start this chapter by describing in detail the metrics and datasets used in the experiments, as well as the methodological design, in Section 4.2. The two sections after that report on the experiments and results with the dynamic weighting approaches presented in Chapter 3 for the corresponding fusion problems: Section 4.3 on neighbour weighting in CF and Section 4.4 on CF and CB weighting in hybrid recommender systems, based on performance predictors. In Section 4.5 a summary of the results and some conclusions are reported.

4.2 Experimental approach

4.2.1 Experimental data

All the experiments reported here have been carried out using the MovieLens dataset, and more specifically the so-called "100K" set. The GroupLens research lab[9] has released different datasets of ratings obtained from a real movie recommender system. The 100K dataset contains 100,000 ratings for 1,682 movies by 963 users. Although there are other public datasets which are larger, this is currently, by far, the most used in the research area, and therefore, in order to ease future comparisons with previous works, we opted for this dataset in our experimental work. In future stages we plan to try heavier datasets, but the 100K set allows for shorter (though still heavy) cycles of experimentation and refinement, more appropriate at this stage, while still providing a fair scale statistic significance.

The main variable with respect to which the behaviour of the proposed algorithms is tested is the amount of sparsity, which we relate to the number of available ratings in the dataset: the larger this number, the lower the sparsity. To this purpose, we split the dataset into different partitions (or "cuts") of the data, each of them having a different level of sparsity (by randomly removing ratings). Each cut divides the rating data into a training set and a test set, as is usual in RS evaluation. In order to randomise the partition and average the results, ten random cuts are generated for each sparsity level. This means that each experiment is run ten times, and the ten corresponding results are averaged at each sparsity level.

[9] GroupLens research lab, http://www.grouplens.org

Nine sparsity levels are considered: from 10% to 90%, in increments of 10%. All this amounts to 9 x 10 = 90 different datasets in total. In the displays of results (tables and graphics), we will show the percentage of training data, as an inverse measure of sparsity (test data is created with the rest of the data). Other secondary variables are also selected (to examine the behaviour of results with respect to them), such as the neighbourhood size in CF, the similarity function used in the CF algorithms, and the type of CF algorithm (see Section 2.5). A summary of the main design options is shown in Table 2.

Variable	Options
Type of CF algorithm	Item- or user-based
Similarity function	Cosine, Pearson, Spearman.
Neighbourhood size	Different values, ranging from 1 to the whole population
Sparsity	Different values
Performance measure	MAE, RMSE, MSE, Precision, Recall

Table 2. Experimental options.

The fixed options taken in the experiments are underlined in the table above: user-based is the primary CF algorithm, Pearson is used as the similarity function between users and between items, and MAE is the primary performance metric. Neighbourhood size and sparsity are left as variables, i.e. we study how the results evolve with respect to them. In particular, nine sparsity levels have been taken (10% to 90%), as explained earlier in this section.

Property	Original	10%	20%	30%	40%	50%	60%	70%	80%	90%
Nr users	943	921.50 (± 4.43)	943 (± 0)	943 (± 0)	943 (± 0)	943 (± 0)	943 (± 0)	943 (± 0)	943 (± 0)	943 (± 0)
Nr items	1,682	1,253.80 (±12.30)	1,412.10 (± 14.87)	1,496.50 (± 10.11)	1,540.10 (± 8.80)	1,579.90 (± 7.11)	1,611.50 (± 8.11)	1,629.20 (± 8.07)	1,652.10 (± 4.51)	1,667.60 (± 2.17)
Nr ratings	100,000	9,984.10 (±98.37)	20,038.60 (±126.36)	30,036.40 (±161.31)	39,956.40 (±212.16)	50,012.80 (±186.74)	59,956.40 (±112.10)	69,977.60 (±149.22)	80,009.50 (± 77.62)	89,977.30 (± 78.83)
Density[10]	6.31%	0.86% (± 0.01)	1.51% (± 0.02)	2.13% (± 0.01)	2.75% (± 0.03)	3.36% (± 0.02)	3.95% (± 0.02)	4.55% (± 0.03)	5.14% (± 0.01)	5.72% (± 0.01)
Avg. nr rated items per user	106.04	10.83 (± 0.12)	21.27 (± 0.13)	31.85 (± 0.17)	42.37 (± 0.22)	53.04 (± 0.20)	63.58 (± 0.12)	74.21 (± 0.16)	84.85 (± 0.08)	95.42 (± 0.08)
Max. nr rated items per user	737	79 (± 5.01)	147.10 (± 8.58)	224.10 (± 13.45)	294.10 (±17.86)	370.30 (± 17.02)	439.10 (± 13.24)	515.60 (± 13.15)	587.50 (± 6.69)	661.80 (± 3.85)
Min. nr rated items per user	20	1 (± 0)	1 (± 0)	1.60 (± 0.52)	2.80 (± 1.03)	5.20 (± 0.63)	7.00 (± 0.67)	8.70 (± 1.25)	11.30 (± 1.06)	14.60 (± 0.84)
Max. nr users rating an item	583	62.80 (± 4.32)	119.20 (± 7.00)	170.10 (± 6.66)	238.80 (± 8.22)	288.80 (± 9.84)	350.20 (± 11.75)	403.20 (± 9.51)	471.10 (± 11.15)	523.50 (± 6.10)
Min. nr users rating an item	1	1 (± 0)	1 (± 0)	1 (± 0)	1 (± 0)	1 (± 0)	1 (± 0)	1 (± 0)	1 (± 0)	1 (± 0)

Table 3. Volumetric characteristics of the different datasets and the generated randomised cuts. Standard deviation is shown in brackets.

Table 3 shows some relevant volumetric and statistic properties of the datasets by sparsity cuts. It is interesting to note that most of the properties keep a constant proportion

[10] The density is #ratings / (#users × #items) × 100, i.e. the percentage of cells in the ratings matrix with known values.

along the different cuts. There is one exception to this: the minimum number of rated items per user is not very significant until a 70-80% of the ratings is taken into account.

4.2.2 Metrics

As introduced at the beginning of this chapter, the experimental work reported here concentrates on two main questions:

- Measuring the variations in the final performance of an RS system, resulting from the introduction of performance predictors to dynamically weight aggregative operations of the system, as a test for the benefit that can be drawn from the prediction techniques in this particular application. This can be seen as an indirect test of the goodness of the predictors, focused on their practical usefulness.
- Examining the correlations between predictor values and performance metrics, when applied to the same input data (same cuts, same users, etc.), as an intrinsic test for the goodness of the predictions.

The first set of experiments (evaluation of dynamic combinations based on performance prediction) allows for straightforward comparisons of the recommendation algorithm against a statically weighted combination, and its consequences are easy to interpret and explain.

The second type of experiments, correlation analysis, is very popular in performance prediction in IR, since it avoids complicated experiments (e.g., a particular algorithm has just to be selected and tested against some simple dataset). However, it requires a sensible evaluation measure at the level where the predictor is introduced, for which in some cases there is currently no standard in RS, a problem which we address here.

But first, correlation analysis requires a correlation function to be defined. We shall primarily use the Pearson correlation for this purpose, which detects linear dependencies between variables. It corresponds to the covariance of the two variables divided by their standard deviation, thus ranging from -1 to +1:

$$\text{Pearson}(x,y) = \frac{\text{cov}(x,y)}{\sqrt{\text{var}(x)\,\text{var}(y)}}$$

When this value is positive, the dependence is direct, otherwise it is inverse. If the correlation is +1, then there is a perfect positive linear relationship, and 0 would indicate no relation at all. In the next sections, we show the computation of correlations between the scores of different predictors and measures. Since we need a measure score for each user, we use the measures shown above at a user level. Additionally, only the significant correlations are shown, which means, in this case, that only those which p-value (probability of observing this particular correlation, assuming the null hypothesis is true, that is, there is no correlation between the two variables) is less than 0.05 appears in the tables.

Both sets of experiments (dynamic combination and correlation) require the selection (or definition) of performance metrics. As explained in Section 2.5, several standard metrics are available in the field. In all of our experiments we use MAE, since it is the most widely accepted and used (at least as a primary measure, sometimes complemented with more ad hoc ones). In some cases here, we complement MAE with additional metrics which allows to better uncover specific effects in performance by the proposed techniques. The definition of MAE is as follows:

$$\text{MAE} = \frac{1}{MN}\sum_{m=1}^{M}\sum_{n=1}^{N}\left|r_{m,n} - p_{m,n}\right|$$

That is, MAE computes the deviation of the predicted rating $p_{m,n}$ from the real rating $r_{m,n}$ (extracted from a test set), averaging it over all the users and items. Therefore, the lower the MAE is, the more accurate is the recommender algorithm.

In our experiments, we define two more measures. The first one is needed in the context of neighbour rating combination in CF. The purpose of predictors in this problem is to assess how useful are specific neighbour's ratings as a basis for computing a rating for the active user in the basic CF formula. A performance predictor for a neighbour need thus to be contrasted to a measure of how "good" is the neighbour's contribution to the global community of users in the system. Since, to the best of our knowledge, there is no such function in the literature (and certainly not a standard one), we need to propose and introduce a fair and sound one. The measure we propose, named *Neighbour Goodness* of a user (how "good a neighbour" she is to her surroundings), is defined as the difference in performance of the RS when including vs. excluding the user (her ratings) from the dataset (the performance of an item could be analogously defined).

For instance, based on MAE, this is defined as follows:

$$\text{neighbour\_goodness}(u_m) = \sum_{u_j \in \mathcal{U}_m} \text{AE}_{\mathcal{U}_m}(u_j) - \sum_{u_j \in \mathcal{U}_m} \text{AE}_{\mathcal{U}}(u_j)$$

$$\text{AE}_{\mathcal{V}}(u_m) = \sum_{i_k \in I} \left| r_{m,k} - p^{\mathcal{V}}_{m,k} \right|, \mathcal{V} \subseteq \mathcal{U}$$

$$\mathcal{U}_m = \{ u_j \in \mathcal{U} : j \neq m \} \subseteq \mathcal{U}$$

where $p^{\mathcal{V}}_{m,k}$ denotes the predicted rating for user u_m and item i_k using $\mathcal{V}$ as the global community, $\mathcal{V}$ being any set or subset of users. Note that $\sum_{u_j \in \mathcal{U}_m} \text{AE}_{\mathcal{U}_m}(u_j)$ is the MAE of the system when leaving out user u_m.

This measure thus quantifies how much a user affects (contributes to or detracts from) the total amount of MAE of the system, since it is computed in the same way as MAE, but leaving out the user of interest (in the first term, it is completely omitted, in the second term, the user is only involved as a neighbour). In this way, we measure how well a user contributes to the rest of users, or put informally, how better or worse is the world (in the sense of how well recommendations work) with and without the user.

The second new proposed measure, named *Error Differential on Discriminative Weighting*, and noted as Δ_γ, is needed to compensate for the lack of a suitable standard dataset where HF approaches can be evaluated. As will be explained later in Section 4.4 on the HF experiments, MovieLens is the most widely used dataset for CF, but it (or any other public CF dataset) does not provide suitable data for CB recommendation. As a result, the MAE performance line of CB approaches systematically stay at a significant distance below CF, which makes the evaluation and comparison of hybrid methods impractical: no matter how good or bad is a HF strategy, the less it happens to use CB, the better performance it shows. In other words, the best weight for CB is always 0, or the closest possible to that, regardless of any other considerations. This makes it impossible to compare, only in terms of absolute MAE, how well or badly a (dynamic or static) HF system assigns weights to its two components.

Therefore, rather than the direct MAE, what we shall measure in the experiments on HF is, given a weight α_{CB} (which fixes its α_{CF} counterpart because of the normalisation constraint) how well the dynamic HF system selects the user for which CB should take this weight. Even if any weight above 0 is a bad decision, we can compare how worse, or less bad, it is than a static assignment of the same α_{CB} for all users. We do so by subtracting the partial average error of each user in our dynamic hybrids from the total MAE of a static hybrid that takes the same α_{CB} as the user has been assigned in the hybrid (in fact, for easier reading, we compute this difference as its percent ratio against the static MAE), and we average this over all users.

This can be formally expressed as follows:

$$\Delta_\gamma = \frac{100}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{\text{MAE}\left(S\left(\alpha_u(u)\right)\right) - \text{AE}_\gamma(u)}{\text{MAE}\left(S\left(\alpha_u(u)\right)\right)}$$

$$\alpha_u = \frac{\alpha_{CB}}{\alpha_{CB} + \alpha_{CF}}$$

$$\gamma = \left(\gamma_{CB}, \gamma_{CF}\right)$$

where $S(\alpha)$ denotes a static hybrid recommender with a weight of α for the CB component and 1-α for CF. AE(u) denotes (similarly to its use in neighbour goodness) the average recommendation error on a particular user (without averaging as in standard MAE) when the hybrid algorithm is used. Contrary to AE and MAE, the higher Δ_γ is, the better our methods behave, positive and negative values indicating better and worse behaviour than static hybrids, respectively. Note that a good Δ_γ does not indicate whether the dynamic hybrid is better or worse than CB or CF – in fact, on MovieLens, as explained earlier, hybrids are always better or equal to CB, and worse or equal to CF. What the magnitude reflects is that the different non-static weights of CB and CF in our dynamic HF systems are better distributed, in terms of resulting overall performance than static methods. The need and meaning of this measure will be explained again in Section 4.4 when presenting the experiments on HF.

4.2.3 Implementation

All the performance predictors and recommendation algorithms in the experiments have been implemented in Java, using the Taste[11] library and the state-of-the-art recommendation algorithms implemented in this library. We extended its capabilities for specific needs of our experiments, such as measuring the average error (AE) for individual users. We may underline that Taste performs well, is widely used in the literature, and its performance is indeed not a variable that affects the experiments, as it is used in both the methods and the baselines.

As mentioned earlier, Pearson correlation is chosen in all the experiments as the basic user (and item) similarity function. The most popular functions to compute the similarity between two users are the cosine similarity and correlation-based similarity (Pearson and Spearman). Which of them performs better is a domain-dependent issue (Breese et al. 1998, Clements et al. 2007) but the Pearson correlation function is the best performing on MovieLens. Its definition is as follows:

$$\text{sim}\left(u_m, u_j\right) = \frac{\sum_{i_n \in I_{m,j}} \left(r_{m,n} - \overline{r}_m\right)\left(r_{j,n} - \overline{r}_j\right)}{\sqrt{\sum_{i_n \in I_{m,j}} \left(r_{m,n} - \overline{r}_m\right)^2} \sqrt{\sum_{i_n \in I_{m,j}} \left(r_{j,n} - \overline{r}_j\right)^2}}$$

where $\overline{r}_j$ denotes the average of all ratings by user u_j.

[11] Taste: collaborative filtering engine for Java, http://taste.sourceforge.net/

Most of the computational load of the proposed methods can be accounted for in off-line processes. While we have not yet undertaken a formal analysis of computational costs, the informal observation is that our methods do not add a significant extra load compared to the baselines. The off-line part is considerably heavy, the main burden corresponding to the operations required by the experimental approaches themselves: repetition of runs by several sparsity levels, randomization, storage of intermediate data, computation of an array of performance metric values (e.g. neighbour goodness is a particularly heavy one). Typically, an experimental cycle could take over 5-6 days minimum, using in the order of three standard PCs and two average-performance servers.

4.3 Neighbour weighting in CF

The dynamic aggregation of neighbour ratings based on a prediction of their performance, when seen as individual recommenders, (as defined in Sections 3.4 and 3.5) is tested against the basic CF algorithm without dynamic weights. Figure 2 shows the results for two clarity-based predictors (user-based and item-based), when taking neighbourhood sizes of a) 100 and b) 500 users respectively. Each graphic shows performance values (MAE) for the nine cuts described earlier in this chapter.

Our method clearly improves the baseline with the smaller neighbourhoods (by up to 9% for 60-80% cuts), and gets almost equal performance with 500 user neighbourhood. This indicates that our method works particularly well when limited neighbourhoods are used, and the improvement fades down to the baseline as they are enlarged. This means that our method is more efficient than the static option with respect to this variable, i.e. that it is able to get better results out of more economic neighbourhood sizes. Enlarging neighbourhoods comes at an important computational cost (of O($k·n·m$) order, k being neighbourhood size, n the total number of users, and m the number of items in the system) in a CF system. Computational cost being one of the well-known barriers in the area (Goldberg et al. 2001), achieving equal (or improved) performance at a lower cost is a relevant result. Let us recall that the total number of users in this dataset is 943, which means that 100 users is about a 10% of the total user community. CF systems described in the literature commonly take neighbourhood sizes of 5 to 500 users for this dataset, being 50 to 200 the common range (Wang et al. 2006, Xue et al. 2005).

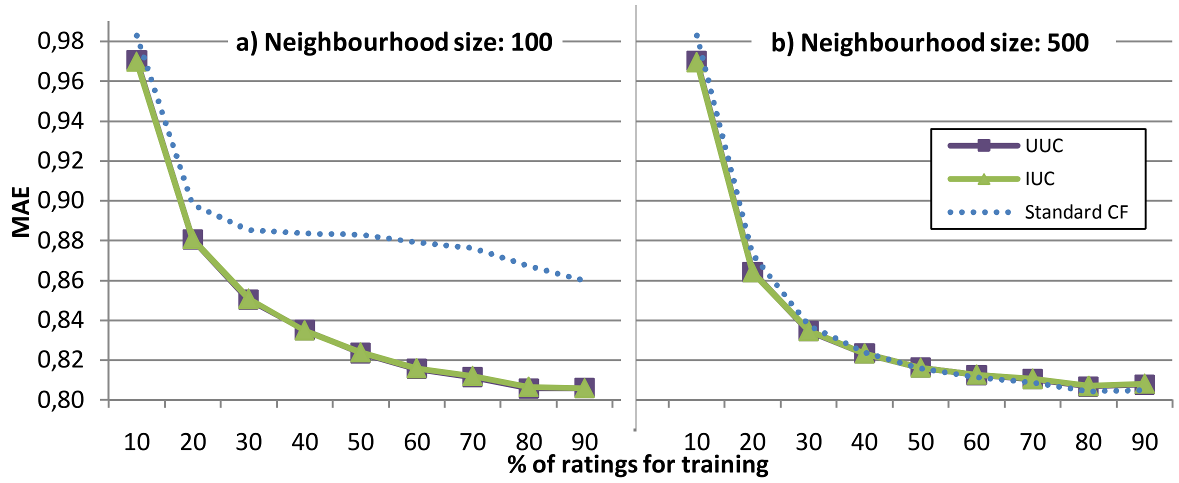

Figure 2. Performance comparison of CF with dynamic, clarity-based neighbour weighting, and standard CF, using neighbourhoods of a) 100 users and b) 500 users.

The trend in the evolution of performance with neighbourhood size, is clear in Figure 3, showing the behaviour of clarity-based predictors at different sizes, setting the sparsity cut at a) 60% and b) as a double check, 80% (which are standard ranges in the RS literature). It can be seen that the shape of the curves in both figures is very similar, evidencing the consistent superiority of clarity-based adjusted CF with small to medium (i.e. usual) neighbourhood sizes (e.g. over 10% improvement at size = 50 users). With neighbourhoods of only 100 users, it can be seen that our method performs as well as the baseline algorithm with 250 to 300 users, which exemplifies the benefit in terms of the computational cost (time and memory) needed to achieve the same performance.

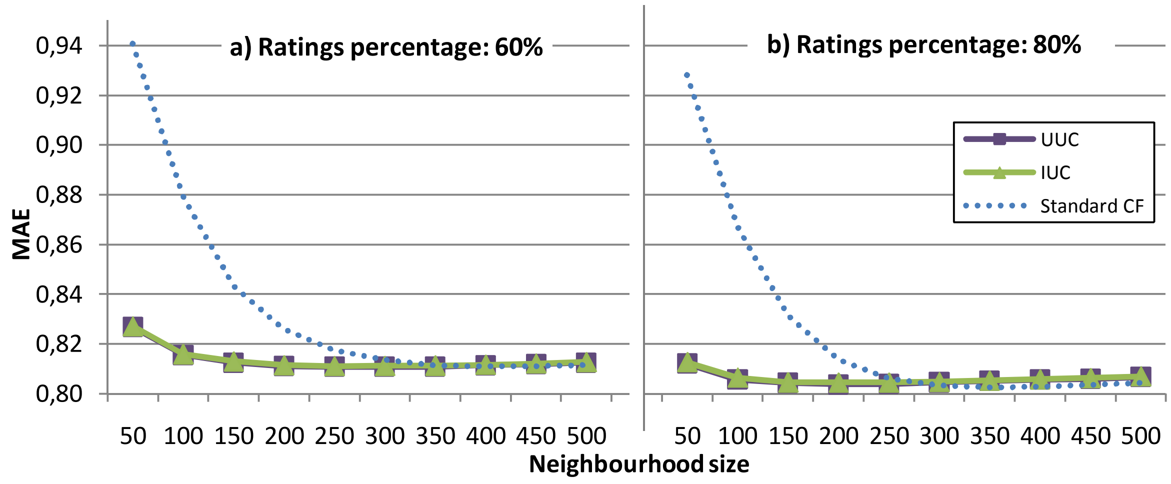

Figure 3. Comparison of standard of CF with dynamic, clarity-based neighbour weighting, and standard CF, using neighbourhoods varying from 50 to 500 users, at a) 60% cut, b) 80% cut.

We also observed that our predictors largely improve IUF as defined in (Breese et al. 1998) and explained in Section 2.2.2 (we omit these results here since, as a matter of fact, IUF does not even lead improve standard CF). (Herlocker et al. 1999) hypothesise that IUF ignores the fact that a user who disagrees with the popular feeling provides a lot of information, which may explain this result.

Correlation analysis

As discussed at the beginning of Section 4.2.2, in addition to measuring final performance improvements as reported above, it is also relevant to analyse the direct correlation between the predictors and performance metrics, on a per-input basis. In the case of the IUC and UUC predictors, the input is a user and an item, respectively. Therefore, in order to examine potential correlations we need to find a function that relates the final performance of the RS to the same input: a user or an item, seen as components of a combined recommender model. This motivates the use of *Neighbour Goodness* defined in Section 4.2.2, as a measure of *user performance* as a neighbour. Recall this measure quantifies how a user affects the total amount of MAE, so that a well performing user should relate to higher values of this measure (and vice-versa), reflecting to what degree the whole community gets better (or worse) results when the user is included as a potential neighbour.

Table 4 shows the correlation between the UUC predictor and user performance. It confirms the hypothesis that UUC is a good performance predictor, since it shows a direct correlation between both, meaning that the higher the value of the measure (well performing user), the higher the value of the predictor (clear user). An exception to this is when only a 10% of ratings is used, where the correlation appears as negative. This can nonetheless be a spurious effect, which can be due to the lack of data in the training set at this cut.

Performance predictor	% of ratings								
	10	20	30	40	50	60	70	80	90
UUC	-0.23	0.21	0.26	0.22	0.21	0.20	0.19	0.18	0.15
IUC	-0.24	0.17	0.19	0.15	0.14	0.13	0.13	0.13	0.09

Table 4. Pearson correlation values, when they are found to be statistical significant at a level of 5%, between the Neighbour Goodness and the UUC and IUC predictors.

To contextualise these correlation values, a good result in the literature corresponds with a value between 0.20 and 0.40 (He & Ounis 2004, Mothe & Tanguy 2005, Carmel et al. 2006), while the best predictors achieve values above 0.45 (Cronen-Townsend et al. 2002, Jensen et al. 2005). Our results are not as high as this, but still they are within (or close to) acceptable levels, especially in the UUC predictor. Additionally, we analysed the correlation measured by the Spearman and Kendall functions with similar results, supporting the direct correlation.

4.4 Hybrid recommendation

We describe in this section the comparative experiments on the dynamic approach based on performance prediction, defined in Section 3.5 for the combination of CB and CF in HF, against standard HF with static combination.

To experiment with HF techniques, two issues need to be addressed. First, while reference implementations of standard CF methods are publicly available (Taste, CoFE[12], Cofi[13], Vogoo[14]), there are not similar resources for CB approaches. Therefore, as is common practice in HF research, we have implemented our own version of a state of the art CB recommender, following the general established guidelines in the literature. As far as it does not introduce significant deviations or strong particularities, the implementation of the CB component does not affect or interfere with (i.e. it is essentially not a relevant variable in) the experimental conditions.

The second issue is that while HF is known to perform better in practice that CB or CF alone, on the MovieLens dataset it is not trivial to build a CB system that gets good enough performance compared to CF, at least when measured by MAE. Figure 4 shows that CB performance is consistently and considerably worse than CF at all points, even for sparse data cuts where CF has its worst performance (the "sparsity problem" in CF). Though we have optimised our CB implementation as far as possible, this is as good as one may expect to get. This is due to the fact that the data in MovieLens, as in all other public datasets in the area, are oriented to CF, and provide comparatively limited data that can be exploited for CB strategies to make their best. As a consequence, it is difficult to build an effective HF system that improves CF on MovieLens. As mentioned earlier in Section 4.2.2, the best HF tends to be the one which gives 0 weight to CB, and it is difficult to build sensible experiments on this basis (we may beat a static HF system, but one which makes little sense as being worse than its CF component). Put in simple terms, an interesting HF system can be built when the

[12] Collaborative Filtering Engine, for Java, http://eecs.oregonstate.edu/iis/CoFE
[13] Java-based Collaborative Filtering library, http://www.nongnu.org/cofi
[14] Recommendation Engine and Collaborative Filtering, for PHP, http://www.vogoo-api.com

performance curves of CB and CF cut at some point(s), which is not the case with MovieLens.

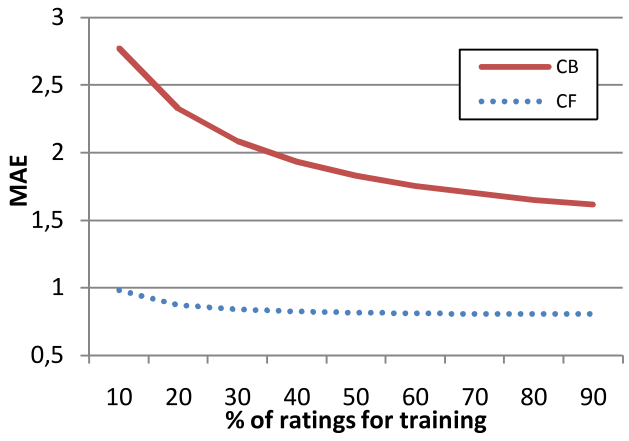

Figure 4. Performance comparison of standard CB and CF on MovieLens.

Figure 5 illustrates this situation. It shows the performance of our best three dynamic methods against the static HF. Though we have tested static weights ranging from 0.1 to 0.9, we only show the two best ones (0.1 and 0.2, that is, the lowest ones as expected). It can be seen that our methods beat the static hybrids, but this is hardly significant, as performance improves consistently as the CB weight decreases, no matter how the weight decrement is decided. Any arbitrary predictor (applied to CB) that just outputs lower values than another (applied to CF), will perform better only for this reason. It can also be observed that since static recommenders always use the same combination factor, these two curves run almost parallel to each other.

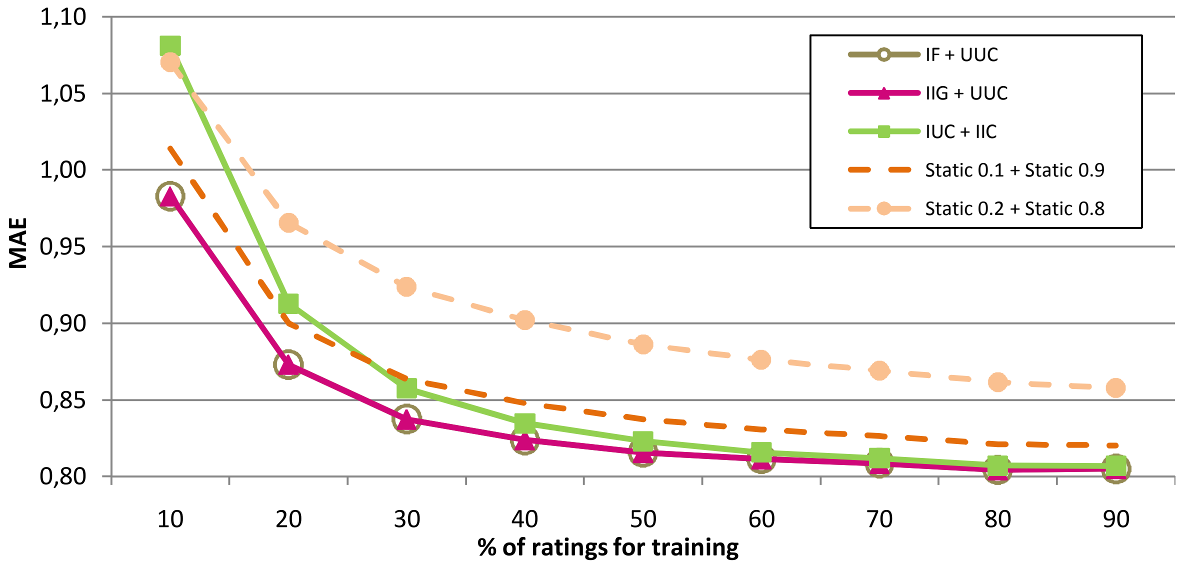

Figure 5. Comparison of MAE performance for static and dynamic hybrid recommenders.

Based on these considerations above, we conclude that MAE is not discriminative enough in this context, and we consider the use of a more specific measure, namely the Δ_γ, measure explained in Section 4.2.2. This measure reflects how better the hybrid combination weights are chosen by dynamic hybrid recommenders compared to static ones. Recall that positive values of this magnitude correspond to improvements with respect to the static algorithms.

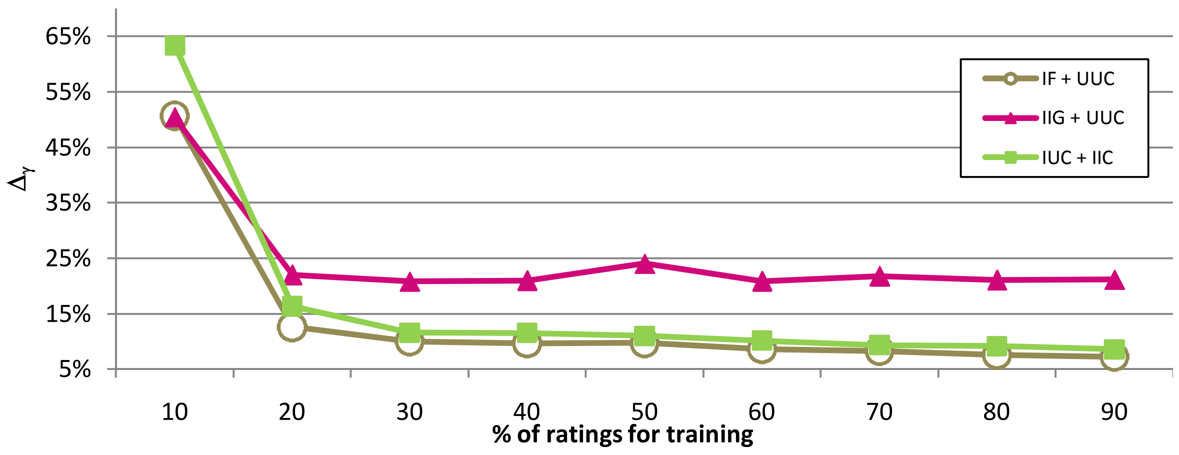

Figure 6. Error differential on dynamic weighting at different sparsity levels.

In this figure, the performance gain is almost constant at the different levels of sparsity (for each dynamic hybrid recommender). A drastic favourable raise can be seen for very high sparsity. Improvements in these situations are relevant, since sparsity is a pervading situation, and a relevant problem, in real recommender systems.

The general conclusion that can be drawn from this experiment is that our dynamic hybrid recommenders achieve a suitable weighting of their components, outperforming static approaches by more than 8% at all sparsity levels (three dynamic hybrids), with a maximum of almost 65% (for the combination of IUC and IIC).

Correlation analysis

Table 5 and 6 show the correlations between the predictors used in this experiment and the average error (AE) measure. Since the predictors estimate the performance of CB and CF recommendation for a particular user, we should take a performance measure of how good the recommendation is for that user (i.e. on the same input). Although not standard in the literature (since it is not usual to break down performance on a per-user basis), the average error (AE) on a user is appropriate here – in fact it is just a partial (user-based) constituent of MAE. As AE is inversely correlated with performance (in the sense that higher values represent worse performance), and for coherence with respect to the previous section we reverse its sign, so that good predictors should be the ones with positive correlations.

Performance predictor	% of ratings								
	10	20	30	40	50	60	70	80	90
UUC	-0.44	0.07	0.19	0.20	0.18	0.17	0.17	0.16	0.12
IIC	0.21	0.34	0.31	0.27	0.24	0.21	0.18	0.16	0.13

Table 5. Pearson correlation values, when found to be statistical significant at a level of 5%, between the AE performance measure (CF) and the predictors used for weighting the CF component: UUC and IIC.

It can be observed that clarity-based predictors (IUC, UUC, and IIC) are positively correlated at all sparsity levels, both as CB and CF predictors. Spearman and Kendall correlations confirm this situation with values up to 0.52 (Spearman for IIC).

Performance predictor	% of ratings								
	10	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**
IIG	-0.09	-0.07	-0.08	-0.08	-0.10	-0.10	-0.11	-0.10	-0.10
IUC	0.17	0.18	0.16	0.15	0.14	0.14	0.12	0.10	0.09
IF	0.06	0.00	N/A	N/A	-0.07	-0.07	-0.01	-0.03	0.08

Table 6. Pearson correlation values, when found to be statistical significant at a level of 5%, between the AE performance measure (CB) and the predictors used for weighting the CB component: IIG, IUC, and IF.

Nevertheless, the other predictors (IIG and IF) do not show a consistent direct correlation. The correlation values for these predictors are in the interval [-0.1, 0.1], which does not reflect significant (positive or negative) correlation (a similar result is found with Spearman and Kendall correlations). This part of the results is thus not conclusive and requires further investigation. Aspects to be further studied and/or reconsidered include using other correlation coefficients that account for non-linear dependencies (not detected by Spearman and Kendall), the choice of alternative performance measures, which may reveal significant correlations, and the refinement and improvement of the predictors. Often a refinement of a predictor, without changing its underlying principles, may uncover its actual predictive power (as in the case of "improved clarity" by Hauff, Murdock & Yates (2008)). As to the performance metrics, the average error of the CB recommender may not be the best option, for reasons related to its generally bad performance on MovieLens as explained earlier. Actually, the correlation between the AE of CF and CB is negative (on average, -0.14), which may explain why these predictors correlate almost randomly with respect to the CB's AE.

4.5 Summary

The experiments in this chapter confirm the predictive power of clarity-based techniques in recommender systems, both by their correlation with performance metrics, and as a basis for dynamically weighting linear combination operations in RS algorithms. The results are particularly positive in data sparsity (as in the hybrid experiment) and small neighbourhood (as in neighbour weighting in CF) situations.

Other predictors have been tested in hybrid recommendation experiments which outperform static recommenders, but they do not show a clear direct correlation with respect to the proposed performance measure. This situation demands further research along three main points:

- The performance measure against which predictor correlation is measured is indeed a variable in the experiment, and can thus be changed to find a significant correlation not evidenced thus far.
- Improvement and/or finer-grain refinement of predictors. It is possible that the predictor value distribution has a particular form that hides its correlation to performance. A key aspect here can be to address the definition of proper ψ functions (introduced in Section 3.2) in order to properly transform the output of the predictor into combination weights. Several detailed observations in the experiments (not shown here) are suggesting that the simplifying assumption $\psi(x) = x$ made so far (i.e. taking the output of predictors directly as α weights) is likely to be far too simplistic and to require express attention.

- Consideration of alternative correlation tests, that detect non-linear, or other kind of dependencies, between predictors and performance. An alternative, simple test would be to check if performance is monotonic with respect to the predictors.

The rest of predictors defined in Chapter 3 have not yet resulted in clear results so far. They are currently under revision, which is work in progress at the time of this writing, and part of the continuation of the research presented here.

Chapter 5. Conclusions

5.1 Summary

Performance prediction is a relatively recent research topic in IR, in which we see an important potential to face the current challenges posed by the growth, diversification of information sources, and take advantage of the opportunities enabled by the convergence of technologies and infrastructures, upon which worldwide spaces connect to each other and to the personal sphere.

This work addresses the problem of performance prediction in the context of Recommender Systems, where the integration of inputs and subsystems is particularly intense. We have studied the state of the art on the topic in IR, and the possibility to adapt techniques such as query clarity (Cronen-Townsend et al. 2002) into the RS domain, where users and items (and ratings), instead of documents and queries, make up the problem space. Additionally, we have researched further prediction methods borrowing from Information Theory (as clarity in fact does).

The definition of predictors is focused on their use in two classic combination methods in RS, namely the combination of neighbour ratings in RS, and the combination of CF and CB in HF. The proposed predictors are defined under the framing purpose to obtain dynamic weighting methods for the latter two operations, in an approach where the higher the expected performance of a component (neighbour, CB, CF) is, the larger the assigned weight to the component in the combination it makes part of.

With the purpose of providing empiric support for the proposed techniques, we have shown two types of experiments, one testing the performance improvement in the combined methods, as a result of dynamically adjusting the weights in the combination; and one studying the direct correlation of performance predictors and performance metrics. The results are generally positive, validating the proposed approach, and raising further questions to be addressed in future (or ongoing) research.

5.2 Contributions of this work

The specific contributions of the work presented here can be summarised as:

- **A formal framework** for the introduction of performance predictors in combined IR functions. The framework is defined at a general level for IR systems, and further particularised for RS.
- **Adaptation of query clarity** techniques to RS. Four different translations have been developed, depending on the actual object of interest (user vs. item) and on how this object is modelled (e.g. a user can be modelled as a set of rated items, as a set of users who rated common items, etc.).
- **Definition of new performance predictors** for RS, based on Information Theory (information gain), and heuristic approaches (user/item feature for CB recommendation).

- **Experimental validation** of the proposed methods. A twofold methodology has been followed: a) performance analysis of combined systems where predictors are introduced for dynamic weighting of subcomponents; and b) analysis of correlation between the predictors and performance metrics. The results support the evidence that incorporating performance prediction in RS improves the performance of the combined system. The results are particularly positive for high data sparsity (as in dynamic hybrids) and small neighbourhoods (as in neighbour weighting in CF).
- **Two new performance measures** are proposed: *Neighbour Goodness* and *Error Differential on Discriminative Weighting*. The former quantifies how much a user affects (contributes to or detracts from) the total amount of MAE of the system, and the latter reflects how better the hybrid combination weights are chosen by dynamic hybrid recommenders compared to static ones.

5.3 Discussion

Performance prediction is a recent research area in IR, but as we have exposed in this work, there is large room for improvement (since there is no clear winner) and several possibilities for innovation. Our research is concerned with personalised IR (of which RS is a particular case), where performance prediction has not been yet exploited. The problem gains relevance in a technological scope, as we are witnessing today, where the scale, openness, heterogeneity, diversification, and highly dynamic nature of information streams and spaces demands the combination of several criteria, systems, and strategies for IR to be effective. We hence see the optimisation of these combinations in dynamic ways as an important opportunity for innovation with high potential impact on IR technology in many areas.

We have seen that the adaptation of query performance predictors from IR to RS is possible, and leads to good results. Uncertainty analysis proves also to be a good approach for predicting performance in RS. Actually, recent work (Zhu et al. 2009) confirms uncertainty analysis as an area of interest in the research community. Having good performance predictors in RS has an intrinsic interest per se. The system may decide to make a recommendation only when the predicted performance is above some threshold, and other decisions can be similarly made, or gauged. Similar strategies can be devised in personalised search (e.g. should preferences be input in the returned results or not, or how much), and IR at large.

In this work, we have used the output of the predictors directly as the combination weights of two RS functions, rather than using the output to define a threshold. The experiments show that the discrimination of useful users can be improved by measuring their expected performance. Neighbour selection (or, equivalently, weighting) is a relevant problem in CF, and improvements on this point are thus relevant in the area. In our first experiment, we show that incorporating performance predictors in a CF environment outperforms the baselines from the literature. Moreover, the enhancement is more significant with small neighbourhoods.

New performance measures have been proposed and introduced in our experiments with hybrid recommendation systems, as a complement of standard performance measures, in order to compare our dynamic HF algorithms against static ones. In terms of MAE, dynamic algorithms outperform static hybrids, although this comparison is not really meaningful, because of the dataset constraints that make HF difficult to test in terms of MAE (since the best hybrid is one in which the CB participation is zero, as CF always and largely outperforms HF on this dataset). The proposed error differential measure shows an improvement from a 10%, to more than 60% when little training data is available, which is an unsolved, interesting problem for RS researchers.

In conclusion, our results so far confirm that performance prediction is a useful approach for achieving specific improvements in specific problems in RS. Several future research lines lay ahead. The ones we plan to address are summarised in the next section.

5.4 Future work

The research presented here uncovers new problems and several directions for the continuation of this work. Among them, we may highlight the following.

Large scale experiments. The experiments reported here are based on average scale versions of the MovieLens dataset (for the sake of agile and practical experimental iteration times), which are the most widely used in the literature. However, once our methods reach stable optimal versions, the experiments will be repeated with the heaviest dataset versions, such as Large MovieLens (1 million of ratings) or Netflix[15] (more than 100 million of ratings), in order to get more precise insights with considerably higher statistical significance.

Improvement of predictors and definition of new ones. Our current results have shown good performance with almost no change from the original formulation of the predictors in IR. We plan to build a wider, comprehensive testbed to study the dependence of predictors on further variables (such as the λ parameter in the clarity predictor, the base of the logarithm, and other parameters). Besides this, we shall undertake an extensive analysis of further potential predictors in order to explore their applicability to the area of RS, such as Jensen-Shannon Divergence or Weighted Information Gain, which have obtained good results in IR. While we have so far studied the adaptation of the most important query performance predictors from IR, we plan to extend this in an exhaustive, systematic manner, covering most or all the proposed techniques in the literature.

Comprehensive analysis of predictors. In this work, we disregard the ranges and shape of predictors, i.e., we make the assumption that $\psi = \textit{identity}$, which is too simplistic. This is a point of revision with, in particular, predictors which so far have shown inconclusive results.

Creation of specific datasets. In the experimental section, we show the need for an appropriate dataset for HF, since the existing public datasets in RS are focused on CF, resulting in a bad performance of CB algorithms, which makes it difficult to evaluate HF techniques. Building a more suitable dataset where CB has a comparable performance to CF is not a trivial task and may imply a loss of generality (by using non-standard data), but it would enable far better experimental conditions in this area.

Research of performance measures. We have introduced new performance measures in this work, but we have not yet undertaken a detailed analysis of their properties and behaviour. Different measures can be considered to evaluate the performance of a single user, since this issue is not explicitly covered in the literature, at least in a principled, reusable way. Proper measures are key to understand and define better specific predictors, and to draw further insights from correlation analysis. A special stress on this type of analysis is indeed a

[15] Dataset provided by Netflix, an online DVD-rental service, http://www.netflixprize.com

general trend in performance prediction research in IR (Hauff, Hiemstra & de Jong 2008), and a good direction to drive successive predictor refinements.

Extension of formal framework. We have presented a model which is general enough for covering very different areas of IR. However, only the specific specialisation to RS has been addressed. As future work, we plan to address the extension of the formalisation, both to more general levels encompassing a larger span of IR techniques and models, and to other specific areas of interest, as the ones mentioned in the next paragraph. We shall likewise study the possible connection to unified formal frameworks proposed in IR, such as (Zhai & Lafferty 2006), based on Utility Theory, or (Wang et al. 2006), based on Language Models.

Extension to new areas. As stated in the introduction, the problem addressed here arises in many areas, such as personalised search, and context-based retrieval, and meta-search.. Extending the work to such areas requires adaptations in both the formal framework and the specific predictors. Further datasets are also needed, meeting specific requirements for each domain. In case of lack of public datasets, the ad hoc creation of specific ones is an alternative.

Bibliography

Adomavicius, G. & Tuzhilin, A. (2005), 'Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions', *IEEE Transactions on Knowledge and Data Engineering* **17**(6), 734–749.

Aizawa, A. (2003), 'An information-theoretic perspective of tf-idf measures', *Information Processing & Management* **39**(1), 45–65.

Amati, G., Carpineto, C. & Romano, G. (2004), 'Query difficulty, robustness, and selective application of query expansion', *Advances in Information Retrieval* pp. 127–137.

Amati, G. & van Rijsbergen, C. J. (2002), 'Probabilistic models of information retrieval based on measuring the divergence from randomness', *ACM Trans. Inf. Syst.* **20**(4), 357–389.

Arieli, O. (2003), Preferential logics for reasoning with graded uncertainty, pp. 515–527.

Aslam, J. A. & Pavlu, V. (2007), Query hardness estimation using Jensen-Shannon divergence among multiple scoring functions, *in* 'ECIR', pp. 198–209.

Aslam, J. A., Yilmaz, E. & Pavlu, V. (2005), The maximum entropy method for analyzing retrieval measures, *in* 'SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval', ACM, New York, NY, USA, pp. 27–34.

Baeza-Yates, R. & Ribeiro-Neto, B. (1999), *Modern Information Retrieval*, Addison Wesley.

Beg, M. M. S. & Ahmad, N. (2003), 'Soft computing techniques for rank aggregation on the world wide web', *World Wide Web* pp. 5–22.

Bellogín, A., Cantador, I., Castells, P., Ortigosa, A. (2008), Discovering Relevant Preferences in a Personalised Recommender System using Machine Learning Techniques. Proceedings of the Preference Learning Workshop (PL 2008), at the 8th European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2008). Antwerp, Belgium.

Bellogín, A., Cantador, I., Castells, P., Ortigosa, A. (2009), Discerning Relevant Model Features in a Content-based Collaborative Recommender System. Springer-Verlag. To appear.

Breese, J. S., Heckerman, D. & Kadie, C. (1998), Empirical analysis of predictive algorithms for collaborative filtering, *in* 'Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence', pp. 43–52.

Brin, S. & Page, L. (1998), 'The anatomy of a large-scale hypertextual web search engine', *Computer Networks and ISDN Systems* **30**(1–7), 107–117.

Burke, R. (2002), 'Hybrid recommender systems: Survey and experiments', *User Modeling and User-Adapted Interaction* **12**(4), 331–370.

Cantador, I., Bellogín, A. & Castells, P. (2007), Modelling Ontology-based Multilayered Communities of Interest for Hybrid Recommendations. In Proceedings of the 1st International Workshop on Adaptation and Personalisation in Social Systems: Groups, Teams, Communities (SociUM 2007), at the 11th International Conference on User Modelling (UM 2007). Corfu, Greece.

Cantador, I., Bellogín, A. & Castells, P. (2008*a*), 'A multilayer ontology-based hybrid recommendation model', *AI Commun.* **21**(2-3), 203–210.

Cantador, I., Bellogín, A. & Castells, P (2008*b*), News@hand: A Semantic Web Approach to Recommending News. In Proceedings of the 5th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH 2008). Hannover, Germany. LNCS, vol. 5149. Springer-Verlag, pp. 279-283.

Cantador, I., Bellogín, A. & Castells, P. (2008*c*), Ontology-based Personalised and Context-aware Recommendations of News Items. In Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI 2008). Sydney, Australia, December, pp. 562-565.

Cao, Y., Xu, J., Liu, T. Y., Li, H., Huang, Y. & Hon, H. W. (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, New York, NY, USA, pp. 186–193.

Carmel, D., Yom-Tov, E., Darlow, A. & Pelleg, D. (2006), What makes a query difficult?, *in* 'SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval', ACM, New York, NY, USA, pp. 390–397.

Carpineto, C., de Mori, R., Romano, G. & Bigi, B. (2001), 'An information-theoretic approach to automatic query expansion', *ACM Trans. Inf. Syst.* **19**(1), 1–27.

Castells, P., Fernández, M., Vallet, D., Mylonas, P. & Avrithis, Y. (2005), Self-tuning personalized information retrieval in an ontology-based framework, *in* R. Meersman, Z. Tari & P. Herrero, eds, 'SWWS'05: In proceedings of the 1st International Workshop on Web Semantics', Vol. 3762, Springer, pp. 977–986.

Clements, M., de Vries, A. P., Pouwelse, J. A., Wang, J. & Reinders, M. J. T. (2007), Evaluation of neighbourhood selection methods in decentralized recommendation systems, *in* 'Workshop on Large Scale Distributed Systems for Information Retrieval (LSDS-IR)'.

Cronen-Townsend, S., Zhou, Y. & Croft, B. W. (2002), Predicting query performance, *in* 'SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval', ACM Press, New York, NY, USA, pp. 299–306.

Cronen-Townsend, Steve, Zhou, Yun, Croft & W. (2006), 'Precision prediction based on ranked list coherence', *Information Retrieval* **9**(6), 723–755.

Dean, J. & Ghemawat, S. (2004), MapReduce: Simplified data processing on large clusters, pp. 137–150.

Diaz, F. (2007), Performance prediction using spatial autocorrelation, *in* 'SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval', ACM, New York, NY, USA, pp. 583–590.

Diaz, F. & Jones, R. (2004), Using temporal profiles of queries for precision prediction, *in* 'SIGIR '04: Proceedings of the 27th annual international conference on Research and development in information retrieval', ACM Press, pp. 18–24.

Dou, Z., Song, R. & Wen, J.-R. (2007), A large-scale evaluation and analysis of personalized search strategies, *in* 'WWW '07: Proceedings of the 16th international conference on World Wide Web', ACM Press, New York, NY, USA, pp. 581–590.

Duda, R. O., Hart, P. E. & Stork, D. G. (2000), *Pattern Classification (2nd Edition)*, Wiley-Interscience.

Dwork, C., Kumar, R. S., Naor, M. & Sivakumar, D. (2001), Rank aggregation methods for the web, *in* 'World Wide Web', pp. 613–622.

Fox, E. A. & Shaw, J. A. (1993), Combination of multiple searches, *in* 'TREC', pp. 243–252.

Goldberg, K., Roeder, T., Gupta, D. & Perkins, C. (2001), 'Eigentaste: A constant time collaborative filtering algorithm', *Inf. Retr.* **4**(2), 133–151.

Gollier, C. & Treich, N. (2003), 'Decision-making under scientific uncertainty: The economics of the precautionary principle', *Journal of Risk and Uncertainty* **27**(1), 77–103.

Gordon, M. D. & Lenk, P. (1991), 'A utility theoretic examination of the probability ranking principle in information retrieval.', *Journal of the American Society for Information Science* **42**(10), 703–14.

Grivolla, Jourlin & Mori, D. (2005), Automatic classification of queries by expected retrieval performance, *in* 'Predicting Query Difficulty - Methods and Applications, SIGIR 2005'.

Hauff, C., Hiemstra, D. & de Jong, F. (2008), A survey of pre-retrieval query performance predictors, *in* 'CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management', ACM, New York, NY, USA, pp. 1419–1420.

Hauff, C., Murdock, V. & Yates, R. B. (2008), Improved query difficulty prediction for the web, *in* 'CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge mining', ACM, New York, NY, USA, pp. 439–448.

He, B. & Ounis, I. (2004), Inferring query performance using pre-retrieval predictors, *in* 'String Processing and Information Retrieval, SPIRE 2004', pp. 43–54.

Herlocker, J., Konstan, J. A. & Riedl, J. (2002), 'An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms', *Inf. Retr.* **5**(4), 287–310.

Herlocker, J. L., Konstan, J. A., Borchers, A. & Riedl, J. (1999), An algorithmic framework for performing collaborative filtering, *in* 'SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval', ACM Press, New York, NY, USA, pp. 230–237.

Herlocker, J. L., Konstan, J. A., Terveen, L. G. & Riedl, J. T. (2004), 'Evaluating collaborative filtering recommender systems', *ACM Trans. Inf. Syst.* **22**(1), 5–53.

Hiemstra, D. (1998), A linguistically motivated probabilistic model of information retrieval, *in* 'ECDL '98: Proceedings of the Second European Conference on Research and Advanced Technology for Digital Libraries', Springer-Verlag, London, UK, pp. 569–584.

Hunter, A. & Liu, W. (2005), Measuring the quality of uncertain information using possibilistic logic, pp. 415–426.

Jardine, N. & van Rijsbergen, C. J. (1971), 'The use of hierarchic clustering in information retrieval', *Information Storage and Retrieval* **7**(5), 217–240.

Jeh, G. & Widom, J. (2003), Scaling personalized web search, *in* 'WWW '03: Proceedings of the 12th international conference on World Wide Web', ACM Press, New York, NY, USA, pp. 271–279.

Jensen, E. C., Beitzel, S. M., Grossman, D., Frieder, O. & Chowdhury, A. (2005), Predicting query difficulty on the web by learning visual clues, *in* 'SIGIR '05: Proceedings of the

28th annual international ACM SIGIR conference on Research and development in information retrieval', ACM, New York, NY, USA, pp. 615–616.

Jin, X., Zhou, Y. & Mobasher, B. (2005), A maximum entropy web recommendation system: combining collaborative and content features, *in* 'KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining', ACM Press, New York, NY, USA, pp. 612–617.

Jones, K. S. (1972), 'A statistical interpretation of term specificity and its application in retrieval', *Journal of Documentation* **28**(1), 11–20.

Kittler, J., Hatef, M., Duin, R. P. W. & Matas, J. (1998), 'On combining classifiers', *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **20**(3), 226–239.

Kuncheva, L. I. (2004), *Combining Pattern Classifiers: Methods and Algorithms*, Wiley-Interscience.

Kurland, O. & Lee, L. (2005), Pagerank without hyperlinks: structural re-ranking using links induced by language models, *in* 'SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval', ACM Press, New York, NY, USA, pp. 306–313.

Kwok, K. L., Grunfeld, L., Dinstl, N. & Deng, P. (2005), Trec 2005 robust track experiments using pircs, *in* 'Online Proceedings of 2005 Text REtrieval'.

Kwok, K. L., Grunfeld, L., Sun, H. L. & Deng, P. (2004), Trec 2004 robust track experiments using pircs, *in* 'Online Proceedings of 2004 Text REtrieval'.

Labreuche, C. & Grabisch, M. (2003), Modeling positive and negative pieces of evidence in uncertainty, pp. 279–290.

Lalmas, M. (1997), Dempster-Shafer's theory of evidence applied to structured documents: Modelling uncertainty, *in* 'SIGIR', ACM, pp. 110–118.

Lalmas, M. (1998), Information retrieval and Dempster-Shafer's theory of evidence, *in* 'Applications of Uncertainty Formalisms', Springer-Verlag, pp. 157–176.

Laming, D. (2001), Statistical information, uncertainty, and bayes' theorem: Some applications in experimental psychology, pp. 635–646.

Lavrenko, V., Allan, J., Deguzman, E., Laflamme, D., Pollard, V. & Thomas, S. (2002*a*), Relevance models for topic detection and tracking, *in* 'Proceedings of the second international conference on Human Language Technology Research', Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 115–121.

Lavrenko, V., Allan, J., Deguzman, E., Laflamme, D., Pollard, V. & Thomas, S. (2002*b*), Relevance models for topic detection and tracking, *in* 'Human Language Technology 2002', pp. 104–110.

Macdonald, C., He, B. & Ounis, I. (2005), Predicting query performance in intranet search, *in* 'Predicting Query Difficulty - Methods and Applications, SIGIR 2005'.

Mcnee, S. M., Riedl, J. & Konstan, J. A. (2006), Being accurate is not enough: how accuracy metrics have hurt recommender systems, *in* 'CHI '06: CHI '06 extended abstracts on Human factors in computing systems', ACM, New York, NY, USA, pp. 1097–1101.

Montañés, E., Quevedo, J. R., Combarro, E. F., Díaz, I. & Ranilla, J. (2007), 'A hybrid feature selection method for text categorization', *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* **15**(2), 133–151.

Montague, M. & Aslam, J. A. (2001), Metasearch consistency, *in* 'SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval', ACM, New York, NY, USA, pp. 386–387.

Mothe, J. & Tanguy, L. (2005), Linguistic features to predict query difficulty, *in* 'Predicting Query Difficulty - Methods and Applications, SIGIR 2005'.

Mylonas, Ph., Vallet, D., Castells, P., Fernández, M., and Avrithis, Y. (2008), 'Personalized information retrieval based on context and ontological knowledge'. *Knowledge Engineering Review* 23(1), special issue on Contexts and Ontologies, pp. 73-100.

Pasi, G. & Marques, R. A. (1999), 'A decision making approach to relevance feedback in information retrieval: A model based on soft consensus dynamics', *International Journal of Intelligent Systems* **14**(1), 105–122.

Pavlov, D., Manavoglu, E., Giles, C. L. & Pennock, D. M. (2004), 'Collaborative filtering with maximum entropy', *Intelligent Systems, IEEE* **19**(6), 40–47.

Pazzani, M. & Billsus, D. (1997), 'Learning and revising user profiles: The identification of interesting web sites', *Mach. Learn.* **27**(3), 313–331.

Pennock, D. M., Horvitz, E. & Giles, L. C. (2000), Social choice theory and recommender systems: Analysis of the axiomatic foundations of collaborative filtering, *in* 'Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence', AAAI Press / The MIT Press, pp. 729–734.

Plachouras, V., He, B. & Ounis, I. (2004), University of Glasgow at TREC2004: Experiments in Web, Robust and Terabyte tracks with Terrier, *in* 'Proceedings of the 13th Text REtrieval Conference (TREC 2004)'.

Plachouras, V. & Ounis, I. (2005), 'Dempster-Shafer theory for a query-biased combination of evidence on the web', *Information Retrieval* **8**(2), 197–218.

Poole, D. & Smyth, C. (2005), Type uncertainty in ontologically-grounded qualitative probabilistic matching, pp. 763–774.

Renda, E. M. & Straccia, U. (2003), Web metasearch: rank vs. score based rank aggregation methods, *in* 'SAC '03: Proceedings of the 2003 ACM symposium on Applied computing', ACM Press, New York, NY, USA, pp. 841–846.

Robertson, S. E. (1997), 'The probability ranking principle in IR', pp. 281–286.

Rocchio, J. J. (1971), *Relevance Feedback in Information Retrieval*, The SMART Retrieval System–Experiments in Automatic Document Processing, Prentice Hall, Englewood Cliffs, NJ, chapter 14.

Roelleke, T. & Wang, J. (2008), Tf-idf uncovered: a study of theories and probabilities, *in* 'SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval', ACM, New York, NY, USA, pp. 435–442.

Shafer, G. (1976), *A Mathematical Theory of Evidence*, Princeton University Press.

Sriram, S., Shen, X. & Zhai, C. (2004), A session-based search engine, *in* 'Proceedings of the 27th annual international conference on Research and development in information retrieval', ACM Press, pp. 492–493.

van Rijsbergen, C. J. (1986), 'A non-classical logic for information retrieval', *Comput. J.* **29**(6), 481–485.

Vinay, V., Cox, I. J., Milic-Frayling, N. & Wood, K. (2005), Evaluating relevance feedback algorithms for searching on small displays, pp. 185–199.

Vinay, V., Cox, I. J., Milic-Frayling, N. & Wood, K. (2006), On ranking the effectiveness of searches, *in* 'SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval', ACM Press, New York, NY, USA, pp. 398–404.

Voorhees, E. M. (2005*a*), Overview of the Trec 2004 robust retrieval track, *in* 'Proceedings of the Thirteenth Text REtrieval Conference, TREC 2004', pp. 70–79.

Voorhees, E. M. (2005*b*), 'The Trec robust retrieval track', *SIGIR Forum* **39**(1), 11–20.

Voorhees, E. M. (2006), 'The Trec 2005 robust track', *SIGIR Forum* **40**(1), 41–48.

Wang, J., de Vries, A. P. & Reinders, M. J. T. (2006), Unifying user-based and item-based collaborative filtering approaches by similarity fusion, *in* 'SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval', ACM Press, New York, NY, USA, pp. 501–508.

Wu, S. & Mcclean, S. (2006), 'Performance prediction of data fusion for information retrieval', *Information Processing & Management* **42**(4), 899–915.

Xue, G.-R., Lin, C., Yang, Q., Xi, W., Zeng, H.-J., Yu, Y. & Chen, Z. (2005), Scalable collaborative filtering using cluster-based smoothing, *in* 'SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval', ACM Press, New York, NY, USA, pp. 114–121.

Yom-Tov, E., Fine, S., Carmel, D. & Darlow, A. (2005*a*), Learning to estimate query difficulty: including applications to missing content detection and distributed information retrieval, *in* 'SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval', ACM Press, New York, NY, USA, pp. 512–519.

Yom-Tov, E., Fine, S., Carmel, D. & Darlow, A. (2005*b*), Metasearch and federation using query difficulty prediction, *in* 'Predicting Query Difficulty - Methods and Applications, SIGIR 2005'.

Yu, K., Schwaighofer, A., Tresp, V., Xu, X. & Kriegel, H. P. (2004), 'Probabilistic memory-based collaborative filtering', *IEEE Transactions on Knowledge and Data Engineering* **16**(1), 56–69.

Zhai, C. & Lafferty, J. (2001), Model-based feedback in the language modeling approach to information retrieval, *in* 'CIKM '01: Proceedings of the tenth international conference on Information and knowledge management', ACM Press, New York, NY, USA, pp. 403–410.

Zhai, C. & Lafferty, J. (2006), 'A risk minimization framework for information retrieval', *Inf. Process. Manage.* **42**(1), 31–55.

Zhou, Y. (2007), Retrieval Performance Prediction and Document Quality, PhD thesis, University of Massachusetts.

Zhou, Y. & Croft, B. W. (2006), Ranking robustness: a novel framework to predict query performance, *in* 'CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management', ACM, New York, NY, USA, pp. 567–574.

Zhou, Y. & Croft, B. W. (2007), Query performance prediction in web search environments, *in* 'SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval', ACM, New York, NY, USA, pp. 543–550.

Zhu, J., Wang, J., Taylor, M. & Cox, I. (2009), Risky business: Modeling and exploiting uncertainty in information retrieval, *in* 'ACM SIGIR 2009'.