Query characterisation in Information Retrieval: performance, difficulty, uncertainty and rank fusion

Alejandro Bellogín Kouki
Universidad Autónoma de Madrid
alejandro . bellogin @ uam . es

June 18, 2008

1 Introduction

The Master's Degree in Computer Science at the Computing Engineering School of UAM includes a course on *Information Retrieval* (IR) in which the classic retrieval models are studied (Vector Space Model, Probabilistic Model, etc), along with rank fusion techniques and other topics, such as Web IR and personalization. This course does not cover however the topic of Language Models, which is a quite specific area that nonetheless has recently gained considerable interest in the IR community. Language Models for IR consist of an innovative probabilistic framework to retrieve and rank documents, which has also been used for query analysis [25].

In this work we continue a previous one about the *Language Models in Information Retrieval* which I have achieved in the first semester this year, towards problems in the area of query analysis and characterisation, drawing techniques from Language Modeling, Information Extraction and Natural Language Processing. The goals of the present work are:

- An extensive and in-depth state-of-the-art study in query characterisation, and in particular the prediction of query performance and difficulty.
- A revision and analysis of theoretic approaches for modeling uncertainty, aimed to clarify and synthesize the involved and related concepts.
- Identify and elaborate the possible applications of these techniques in problems such as rank fusion and IR personalisation.
- Start prospective experiments with the aim of enhancing the understanding of the studied techniques, and opportunities for the research of novel approaches not tested in the literature.

This report pays particular attention to the concept of uncertainty and ambiguity, since these are important query characteristics. The connection with the previous work is in the fact that one of the proposed approaches in the literature, which we find of particular value, for capturing query uncertainty and predicting query performance has been based on Language Models.

We have studied the different possibilities, proposed or explored up to date, to model and handle vague information. These alternatives are reported in this work, after some introductory background, and a glossary of relevant notions and terminology, along with the different techniques proposed in the literature, associated with those definitions. After that, further techniques are described and classified in groups based on different criteria.

Since metasearch (or rank aggregation in general) is one of the envisioned areas of application of query analysis and characterisation techniques, some basic concepts from that area have been revised and are summarized in this report as well. In addition to this, we have started to study the relation between some well-known uncertainty models (such as Dempster-Shafer's theory, fuzzy models, entropy models, etc) and Information Retrieval. In this work we get into this wide area and begin an analysis we expect to be useful in the near future. Finally, as part of this work, I have conducted some prospective experiments aimed to observe and better understand the studied techniques, and explore their potential application in innovative approaches addressing new problem areas.

In section 3, I present some recurrent definitions appeared in the literature in the form of a glossary as well as different query types according to different aspects. Section 4 describes the most important query types. In section 5 different ways for modelling uncertainty are presented. Section 6 describes the most important performance predictors proposed in the area, and in section 7 an introduction to the state-of-the-art in rank fusion is given. Section 8 describes a model for personalised IR using clarity measures. In section 9 I report the experiments carried out and the results drawn from them. The report ends with some conclusions drawn from the study, possible lanes for continuation towards further research, and an appendix where the meetings with the tutor are recorded, in addition to a summary of the TREC datasets, and a summary of the relation between the main authors in the literature on the relevant topics.

2 Query characterisation

Dealing effectively with poorly-performing queries is a crucial issue in information retrieval systems. Actually, performance prediction provides some information that can be useful in many ways [60, 57]:

- From the user perspective, it provides valuable feedback that can be used to direct a search: 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 predicted difficulty). This way, the search engine can use the query predictor as a target function for optimizing 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 of the query results.

- For distributed information retrieval, the estimation can be used to decide which search engine to use, or how much weight to give it when its results are combined with those of other engines.

Because of its multiple potential applications, quantifying the ambiguity of queries has been a major research goal in the area of query analysis, but it has received several names in different contexts and with distinct nuances. In the next section we review such names, emphasizing the differences between them, the techniques used and the variables considered in each context.

3 Glossary

In this section we collect some relevant definitions in the query characterisation area, defining the most common names used in the literature when discussing about ambiguity. As G. J. Klir says in [29]:

> "[...] the broad concept of uncertainty is closely connected with the concept of information. The most fundamental aspect of this connection is that uncertainty involved in any problem-solving situation is a result of some information deficiency pertaining to the system within which the situation is conceptualized. There are various manifestations of information deficiency. The information may be, for example, incomplete, imprecise, fragmentary, unreliable, vague or contradictory. In general, these various information deficiencies determine the type of associated uncertainty."

This quote is a first definition of uncertainty. In table 1 we summarize the list of concepts that we have found relevant in the scope of this work, and the meaning of which we aim to clarify, as far as possible, in our glossary.

Clarity

Cronen-Townsend *et al.* [13] defined the **query clarity** as a degree of (the lack of) the **query ambiguity**. In [12] the authors define query ambiguity 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 usage associated with the query and the generic language of the collection as a whole. This measure (clarity score) is defined by these authors 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:

$$\begin{aligned}
P(w|Q) &= \sum_{D \in R} P(w|D)P(D|Q), \quad P(Q|D) = \prod_{q \in Q} P(q|D) \\
P(w|D) &= \lambda P_{ml}(w|D) + (1-\lambda)P_{coll}(w) \\
\text{clarity score} &= \sum_{w \in V} P(w|Q) \log_2 \frac{P(w|Q)}{P_{coll}(w)}
\end{aligned}$$

Concept	**Measure**	**Formula**	**Description**	**Reference(s)**
Query clarity	Clarity score	$\sum_{w \in V} P(w\|Q) \log_2 \frac{P(w\|Q)}{P_{coll}(w)}$	Degree to which the query retrieves documents in the given collection with similar word usage	[13] [12]
Query difficulty	Info$_{\text{DFR}}$	$\sum_{t \in Q} -\log_2 \text{Prob}(\text{Freq}(t\|\text{TopDoc})\|\text{Freq}(t\|\text{Coll}))$	Amount of information gained after a first-pass ranking	[2]
Specificity	Query scope	$-\log{(N_Q/N)}$	Percentage of documents that contain at least one query term in the collection	[45] [23] [38]
Uncertainty	Ranking robustness	$\frac{1}{K}\sum_{i=1}^{K} \text{SimRank}(L(Q,G,C), L(Q,G,T(i)))$	Expected similarity between a fixed ranked list and a random list	[60]
Hardness	Jensen-Shannon divergence	$\frac{1}{m}\sum_j KL(p_j\|\|\bar{p})$	Measure the stability of ranked results in the presence of perturbations of the scoring function	[3]

Table 1: Summary of the described measures and their respective expression.

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

It is important to note that the clarity score is neither bounded nor centered in zero. In figure 1 we can see a representation of the function $z = x \cdot \log_2 x/y$ when $x, y \in (0, 1]$. Since the clarity score is calculated adding numbers in this way $\left(\sum_w P(w|Q) \log_2 \frac{P(w|Q)}{P_{\text{coll}}(w)}\right)$ it can be seen that a supremum (infimum) of this score is proportional to the vocabulary size and that it can be positive or negative[1]

With this definition, we can see that this measure is system-independent, since it only evaluates the coherence of the ranked list of documents (post-retrieval) for a given query. For example, in [12] the authors explicitly show the relation between λ and the collection:

$$\lambda = \frac{||D||}{||D|| + \mu}, \text{ prior } \mu$$

The degree of ambiguity of a query with respect to the collection of documents being searched is often closely related to query performance. They found a strong

[1] Actually, from a technical point of view, we can find a negative clarity value for a term $\left(P(w|Q) \log_2 \frac{P(w|Q)}{P_{\text{coll}}(w)}\right)$ when $P(w|Q) < P_{\text{coll}}(w)$ and positive otherwise.

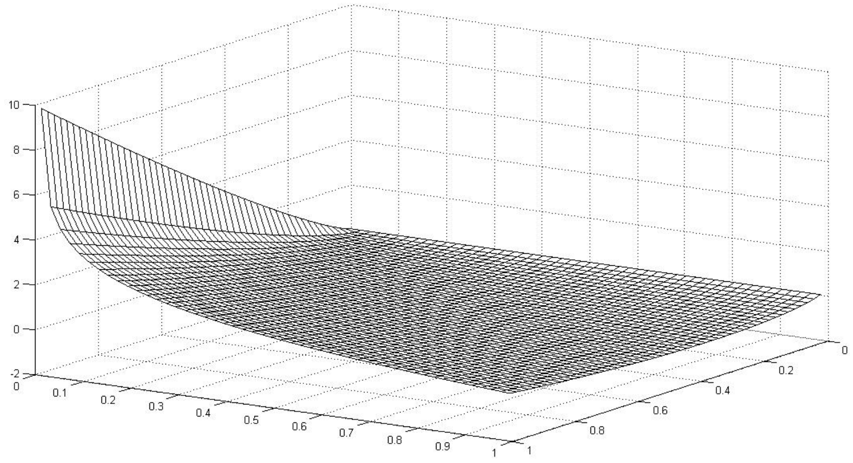

Figure 1: Surface generated by the function $x \cdot \log_2 x/y$ when $x, y \in (0, 1]$. It can be seen that the maximum is found when $y \approx 0, x \approx 1$ and the minimum when $y \approx 1, x \approx 0.5$.

correlation between the clarity score of a test query with respect to the appropriate test collection and the performance of that query. Because of that, the clarity score method has been widely used in the area for query performance prediction [1, 2, 3, 7, 23, 43, 57, 60, 61].

Some applications of the clarity score measure include query expansion (anticipating poorly performing queries which shoul not be expanded), improving performance in the link detection task in topic detection and tracking by modifying the measure of similarity of two documents [36], and document segmentation [16].

Difficulty

In [2], Amati *et al.* proposed the notion of **query difficulty** to predict query performance. In this 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. This Info_{DFR} is defined as

$$\text{Info}_{\text{DFR}} = \sum_{t \in Q} -\log_2 \text{Prob}(\text{Freq}(t|\text{TopDocuments})|\text{Freq}(t|\text{Collection}))$$

For the implementation of this predictor, two different expansion models were tried:

$$
\begin{aligned}
\text{Info}_{\text{KL}}(t) &= \frac{\text{Freq}(t|\text{TopDocs})}{\text{TotFreq}(\text{TopDocs})} \cdot \log_2 \frac{\text{Freq}(t|\text{TopDocs}) \cdot \text{TotFreq}(C)}{\text{TotFreq}(\text{TopDocs}) \cdot \text{Freq}(t|C)} \\
\text{Info}_{\text{Bo2}}(t) &= -\log_2 \left(\frac{1}{1+\lambda}\right) - \text{Freq}(t|\text{TopDocs}) \cdot \log_2 \left(\frac{1}{1+\lambda}\right) \\
\lambda &= \text{TotFreq}(\text{TopDocs}) \cdot \frac{\text{Freq}(t|C)}{\text{TotFreq}(C)}
\end{aligned}
$$

where TopDocs denotes the pseudo-relevant set and C denotes the whole collection.

Query difficulty is system-dependent, since the probability that appears in the formula is not used directly, but it is normalized by considering the probability of the observed term-frequency only in the set of documents containing the term. This measure has also been applied to query expansion, making it possible for this technique to be selective, by avoiding the application of query expansion on the set of worst (difficult) topics.

Specificity

In the same year (2004) Plachouras *et al.* [45][23][38] defined the **query scope** as a measure of the **specificity** of a query: $-\log(N_Q/N)$, where N_Q is the number of documents containing at least one of the query terms, and N is the number of documents in the whole collection. The authors found that query scope is effective for inferring query performance for short queries in ad-hoc text retrieval. This is because the size of the document set containing at least one of the query terms is an alternative indication of the generality/speciality of a query.

One application of this measure is to find which of the available approaches is most appropriate for a specific query. Plachouras *et al.* considered content-only retrieval, retrieval based on the content of documents and the anchor text of their incoming links, and a combination of the latter approach with a score obtained from the URL length of a document. For this task the authors defined a more general measure:

$$qe = \min\left(\frac{\{\text{number of retrieved documents containing all query terms}\}}{\alpha}, 1\right)$$

qe stands for *query extent*, and it is the number of retrieved documents that contain all the query terms, normalised between 0 and 1 by dividing with a given fraction of the total number of documents in the test collection (α). The authors combine query extent with result extent in order to classify different queries in a particular TREC's task.

$$\text{result extent} = \{\text{number of sites for which size}_j > \mu_{\text{size}} + 2 \times \sigma_{\text{size}}\}$$

where size_j is the number of documents from the jth site, and μ, σ its average and standard deviation. It is system-independent, moreover, its value determines which is the most appropriate retrieval approach for each query, but strongly collection-dependent.

Another application of query scope can be found in [50], where it is used for assigning a measure of uncertainty to each source of evidence (in their work these sources were content analysis and link structure analysis) and then applying Dempster-Shafer's theory of evidence. The estimation of the term scope (a query is considered as a *bag of terms*) is based on defining a probability measure for concepts on top of WordNet's hierarchical structure of concepts.

Uncertainty

More recently (2006) a new concept has arisen: **ranking robustness** [60]. 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 that the degree of ranking robustness against noise is positively correlated with retrieval performance. Regular documents also contain *noise* if we interpret noise as *uncertainty*. This robustness score performs better than or at least as good as the clarity score.

This measure has only been applied to predict retrieval effectiveness. Zhou and Croft consider as inputs to robustness score: a query, a retrieval function (it induces the ranked list that the measure really needs), a document collection and another collection (random collection). Therefore, it is collection-dependent, and if we consider the retrieval function as a black box, it is also system- and query-independent.

Hardness

Another novel and very promising point of view about the performance prediction is the definition of the **query hardness** by Aslam and Pavlu in [3]. This technique is based on examining the ranked lists returned by multiple scoring functions (retrieval engines) with respect to the given query and collection. The authors propose that the results returned by multiple retrieval engines will be relatively similar for *easy* queries but more diverse for *difficult* queries. Actually, they distinguish two notions of query hardness:

System query hardness difficulty of a query for a given retrieval system run over a given collection. To capture the difficulty of the query for a particular system, run over a given collection. It is system-specific.

Collection query hardness difficulty of a query with respect to a given collection. Capturing the inherent difficulty of the query (for the collection) and perhaps applicable to a wide variety of typical systems. It is independent of any specific retrieval system.

Authors suggest that a procedure for estimating query hardness would be useful to alert users about the likelihood of poor results (and propose them to reformulate the query), to employ enhanced search strategies if a difficult query has been found or to combine more accurately input results from distributed systems.

Other notions

Finally, the concepts **query vagueness** and *imprecision* has not been used, to our knowledge, as a specific concept, but as a general one, aggregating all these previous meanings.

Notions from Information Theory

Now we can see some related definitions extracted from Information Theory[17]:

- The **entropy** of a discrete distribution is a measure of the randomness or unpredictability of a sequence of symbols $\{v_1, \cdots, v_m\}$ drawn from it, with associated probability P_i. It can be calculated using the logarithm base 2, in this case it is 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

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

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

  $$\begin{aligned} \delta(x-a) &= \begin{cases} 0 & x \neq a \\ \infty & x = a \end{cases} \\ \int_{-\infty}^{\infty} \delta(x) dx &= 1 \end{aligned}$$

  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.

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

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

  It is closely related to cross entropy, information divergence and information for discrimination. The continuous version is:

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

  The relative entropy is not a true metric because D_{KL} is not necessarily symmetric in the interchange $p \leftrightarrow q$.

  In figure 2 we can see the relation between the defined measures.

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

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

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

- Dempster-Shafer's theory of evidence introduces the concept of uncertainty in the process of merging different sources of evidence, extending in this way the classical probability theory [47, 50, 35]. According to this theory, the set of elements $\Theta = \{\theta_1, \cdots, \theta_n\}$ in which we are interested is called the *frame of discernment*. The goal is to represent beliefs in these sets, defining *belief functions* $Bel : 2^\Theta \longrightarrow [0,1]$. These functions are usually computed based on probability mass functions m that assigns zero mass to the empty set, and a value in $[0,1]$ to each element of the power set of Θ:

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

  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 of each other. 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:

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

4 Query types

In this section we review different classifications used in IR for queries. These classifications can be seen as a first step in order to extract the most relevant query characteristics and investigate their correlation with performance.

Different information needs suggest different types of queries (and different retrieval) issued to the system. The most common type of retrieval is the one of searching

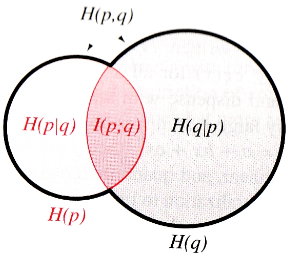

Figure 2: For two distributions p and q, this figure shows the matematical relationships among the entropy, mutual information $I(p;q)$, and conditional entropies $H(p|q)$ and $H(q|p)$. For instance, $I(p;p) = H(p)$, if $I(p;q) = 0$ then $H(q|p) = H(q)$

for documents relevant to the particular need received and return these documents according to an appropriate ranking. In this retrieval the focus is on the whole document, because we do not know what the user is searching for. In this case, natural language processing can help to understand which are the user's needs and what information is contained in every document. For making this possible, it is necessary to have a query written in **natural language**, but nowadays users frequently issue short queries (3 to 4 words) and without any structure [26]. In this particular scenario, the query can be seen as a set of terms that have to be present in the documents retrieved (**boolean** `AND` queries). Most typical techniques smooth this condition and give a weight to each term, even allowing some terms not to appear in the document (`OR` queries).

Leaving aside this language-based classification, a natural and useful partition of queries is found: short against long queries. This is a very basic query characteristic but we have some problems despite its simplicity: from which threshold do we have to consider a query as long, do we have to process the query before we apply the threshold (remove stopwords and punctuation), do longer queries always give more information than shorter queries, ... For example, TREC (see section A) usually provides short title queries and longer description queries in order to test the systems' performance with respect to this feature.

Now we have mentioned TREC, we can look more carefully each track proposed every year in this conference and extract the different information needs inherent to each of these tracks in order to know more about their associated queries.[2]

- Blog track. With this track they want to know which is the behavior in the blogosphere. Tasks: opinion retrieval (opinionated nature of many blogs), blog distillation (feed search).

- Enterprise track. This track has an explicit information need: satisfying a user who is searching the data of an organization to complete some task. It involves new data (email, documents in version control system) and new tasks (search across a single data type or mixed data types) with respect to TREC.

- Legal track. This track is focused on a very specific kind of user: a lawyer. This user needs to retrieve, in a very effective way, documents in digital collections. Tasks: automatic ad hoc, automatic routing, interactive.

[2] The first five mentioned tracks will run this year (2008)

- Million query track. With this track they explore adhoc retrieval on a large collection of documents, and, more important, they investigate about the evaluation: evaluate large number of queries incompletely, rather than a small number more completely. Two tasks: run queries and judge documents.

- Relevance feedback track. Relevance feedback is a very common and known technique in Information Retrieval, with this track they want to provide a framework for exploring the effects of different factors on the success of relevance feedback, allowing comparisons between systems and a common baseline.

- Cross-language track. A track that investigates the ability of retrieval systems to find documents that pertain to a topic regardless of the language in which the document is written.

- Filtering track. The user's information need in this track is stable and some relevant documents are known, but there is a stream of new documents. For each document, the system has to filter the documents to retrieve (make a binary decision) according to a set of user needs represented in *profiles* (all of the information the system has acquired about a specific information need). Tasks: adaptive filtering, batch filtering, routing.

- Genomics track. In this track, a more specific domain is studied: gene sequences and documentation (research papers, lab reports, etc.) Actually, the user's need is to adquire new knowledge in a sub-area of biology linked with genomics information.

- HARD track. The goal of HARD is to achieve High Accuracy Retrieval from Documents by leveraging additional information about the searcher and/or the search context, through techniques such as passage retrieval and using very targeted interaction with the searcher. In some applications, such as information analysis, a more specific requirement exists for high accuracy retrieval (instead of improving the effectiveness of search), i.e. high precision in top documents. For example, in 2005 its topics were the same that Robust Track used.

- Interactive track. A track studying (real) user interaction with text retrieval systems. Four sorts of questions: find any n Xs, find the largest/latest/... n Xs, find the first or last X, comparison of 2 specific Xs. The task consists of answering each question and identifying a minimal set of documents wich supports the answer, within a maximum of 5 minutes.

- Novelty track. A track to investigate systems' abilities to locate new (i.e., non-redundant) information. The basic design is: given a TREC topic and an ordered list of relevant documents, find the *novel* information that should be returned to the user from this set.

- Question answering track. A track designed to take a step closer to information retrieval rather than document retrieval. The QA track last ran in 2007. Tasks: find an exact answer to some question (main task), assemble an answer from information located in multiple documents (list task).

- Robust retrieval track. Task: ad hoc retrieval focused on individual topic effectiveness rather than average effectiveness. The evaluation methodology emphasizes a system's least effective topics.

- SPAM track. The goal here is to provide a standard evaluation of current and proposed spam filtering approaches, thereby laying the foundation for the evaluation of more general email filtering and retrieval tasks. Tasks: on-line filtering (ideal user feedback and delayed feedback) and active learning (classification of new messages).

- Terabyte track. With this track they want to develop an evaluation methodology for terabyte-scale document collections. Sparck Jones and van Rijsbergen proposed a way of building significantly larger test collections by using *pooling*[3], although this technique makes relevance information incomplete TREC has adopted it for this track. Actually, they expect that retrieval algorithms may perform differently at very large scales and that evaluation methodologies will need to be revised to deal more effectively with incomplete relevance information. The main task for the terabyte track is ad hoc informational search. Others tasks: efficiency (query processing times), named page finding.

- Video track. It is devoted to research in automatic segmentation, indexing, and content-based retrieval of digital video. This track became an independent evaluation in 2003, with a workshop taking place just before TREC.

- Web track. The document set used in this track is a snapshot of the World Wide Web, so it is possible to simulate search tasks as if they were being executed online.

After all this gibberish of tracks, we can summarise the most relevant tasks for our purposes, focusing on those defining a specific and different user's need. In the same table appears which predictors have proved to perform properly in that task, if any (table 2).

Type of query	**Good predictors**	**TREC track**
Named page finding	Query scope [45], Weighted Information Gain [61]	Terabyte
Home page	Query scope [45]	Web
Content-based	Weighted Information Gain [61]	Web
Topic distillation	Query scope [45]	Web
Known item finding	Query scope [45]	Web

Table 2: Non-linguistic features found statistically significant correlated with average precision

Besides TREC, there are some more query classifications we can find. There is a lot of research done in the areas of query expansion and relevance feedback, some algorithms have proved to be efficient but usually they felt down with some particular queries: polisemic, ambiguous, long,... queries; in short: difficult queries. These queries are called *bad-to-expand* queries [12], and produce negative improvements after expansion. This is why a very important application of query characterisation is query expansion, because it will detect when the algorithm can expand normally and when it is better do nothing.

[3] Only the top 100 documents for a topic by the participating systems are judged by human assessors and all the documents which are not in the pool are unjudged, forming the qrel set, and assumed to be irrelevant. Therefore, many relevant documents may be missed using such a pooling strategy.

Finally, we can consider a special kind of retrieval, in which documents are timed (the most typical application of this is a news searcher). In this situation we can distinguish two main types of queries: the first type of query favors very recent documents and the other has more relevant documents within a specific period in the past.

Attribute	Types
Language	Natural language, Database, Bag-of-words
Expansion	Bad-to-expand, good-to-expand
Length	Short, long
TREC	name page, home page, content-based, topic distillation, known item finding, navigational
AP	Weak, strong
Time	Recent, past

Efficiency-related tracks

According with [56], there has been no attempt from TREC to build topics that match any particular characteristics, but the effects of topic characteristics on system performance have been analysed both in TREC-2 and TREC-5. This is partly because the topic emphasis was on real user topics, but also because it is not clear what characteristics would be appropiate.

In these analyses, a measure called topic *hardness* was developed. It is defined as an average over a given sets of runs of the precision for each topic after all the relevant documents have been retrieved *or* after a hundred documents have been retrieved, if more than a hundred documents are relevant. As we can see, this measure is oriented toward high-recall performance and how well systems do at finding all the relevant documents.

Despite that, this measure can be used to show correlations between some particular topic characteristic and system performance. In TREC-5, two specific topic characteristics were analysed: length of the topic and number of relevant documents found for that topic. An almost random correlation was found between these characteristics and the hardness (0.19 and 0.14 for number of relevant documents and length, respectively).

This information was accurate in the moment of printing, but nowadays it is out-of-date, because we have already mentioned some tracks focused on topic effectiveness and accuracy:

- Robust track
- HARD track
- Terabyte track
- Legal track

The problem with these tracks is that they are of very recent vintage or have been discontinued. For example, this year will be the third year that the Legal track will run, Robust and HARD last ran was in 2005 and Terabyte in 2006. Despite this, majority of literature still cites these tracks, actually, the most used is the Robust track, with topics of TREC 7, or others similar.

As a future work for the whole community, it is interesting to know whether a topic found to be difficult several years ago is still difficult for current state-of-the-art IR systems. In the Robust tracks of 2003 and 2004, conclusion about this point was that current systems still have difficulty in handling those old difficult topics [7]. However, the Robust track results did not fully answer the basic question underlying its root cause, that is, why are some topics more difficult than others?

5 Modelling uncertainty

In [22] we can find some representations of uncertainty:

- Numeric representations:
  - Probability measures
  - Dempster-Shafer belief functions
  - Possibility measures
  - Ranking functions
- Nonnumeric representations:
  - Plausability measures

All these representations arise because probability has its problems:

- Either one event is more probable than the other, or they have equal probability. It is impossible to say that two events are comparable in likelihood.
- The numbers are not always available

We are going to focus in representations already used in Information Retrieval field, that is, Dempster-Shafer and fuzzy theory (possibility measures is based on ideas of fuzzy logic) [5, 50, 35, 10, 11, 34]. These approaches belong to what is called *Soft Information Retrieval*, set of approaches that aims at applying techniques for dealing with vagueness and uncertainty.

5.1 Fuzzy representation

Fuzzy set theory is a formal framework well suited to model vagueness: in Information Retrieval it has been successfully employed at several levels, in particular for the definition of a superstructure of the Boolean model. In [10] we can find different fuzzy models applied to this field:

- Extended boolean models: fuzzy document representation
- Extended Boolean models: fuzzy extensions of the query language
- Fuzzy Thesauri of terms
- Fuzzy Clustering of Documents

Through these extensions the gradual nature of relevance of documents to user queries can be modelled. Actually, the same author, Crestani, in [11] presents many models helping to give a logical definition and capture the view of relevance:

- Logical models:

  - Modal logic
  - Conceptual graphs
  - Situation theory
  - Channel theory
  - Terminological logic
  - Abductive logic
  - Default logic
  - Belief revision
  - Fuzzy logic

- Logical-uncertainty models (uncertain inference models):
  - Based on probability theory
  - Based on probabilistic datalog
  - Based on logical imaging
  - Based on semantic information theory
  - Based on probabilistic argumentation systems

- Meta-models

Neural networks have also been used in this context to design and implement Information Retrieval Systems that are able to adapt to the characteristics of the IR environment, and in particular to the user's interpretation of relevance [10]. They can be classified as:

- Supervised Learning Techniques. At first, they were used for modeling each document by a unit. Later relevace feedback was added. There has been successful three layers networks: Belew's network (descriptors, documents and documents' authors) and Kwok's network (queries, index terms and documents). Jung and Raghavan attempted to join Vector Space model with learning paradigms of connectionist model.

- Unsupervised Learning Techniques. They have been used mainly for documents or terms clustering and classification. Query expansion is feasable suggesting the user terms that are similar to those she put in the query.

5.2 Dempster-Shafer representation

[50] presents results obtained from content and link analyses which are then combined using Dempster-Shafer's theory of evidence. This theory introduces the concept of uncertainty in the process of merging different sources of evidence, extending in this way the classical probability theory. According to this theory, the set of elements in which we are interested is called the frame of discernment. 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 of each other. The rule of combination of evidence returns a measure of agreement between two bodies of evidence.

With this notation, the frame of discernment in their paper is the set of Web documents in the collection, the scoring functions for the content analysis and the link structure analysis are considered to be the bodies of evidence that will be combined into a single body of evidence in the frame of discernment. Vassilis and Iadh found that Dempster-Shafer theory of evidence is not effective in significantly improving precision, due to either the quality of the sources of evidence, or the appropriateness of the method itself. With respect to the first point, while content-only retrieval is an effective approach for the tasks they experimented with (topic relevance of TREC10 and topic distillation of TREC11), hyperlink analysis has not proved to be equally useful. In addition, the normalisation of the scores used[4] could bias the combination of evidence, since the distribution of hyperlink analysis scores is significantly different from that of the content-only retrieval scores. Instead of Dempster-Shafer theory, different approaches can be used for the combination of evidence.

In [34] we find a model that allows the expression of uncertainty with respect to parts of a document. In [35] a four-featured model is discussed (structure, significance, partiality, uncertainty[5]). 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 less transformations than another one is usually more relevant to the query than the other document. Furthermore, an implementation of the model was performed.

6 Performance prediction

In this section, we show the distinct performance predictors proposed in the literature. Some of them have already been described in the glossary (section 3), but we provide a more fine-grained classification here.

One way to measure the effectiveness of the performance prediction methods is to compare the rankings of queries based on their actual precision (such as MAP) with the rankings of the same queries ranked by their performance scores (that is, the predicted precision). Based on whether retrieval results are needed when computing the performance score, these methods can be classified into three groups: non-retrieval, pre-retrieval and post-retrieval approaches. Another relevant distinction is, whether the predictors are trained or not. In the following sections we describe the approaches proposed in the literature, grouped according to these distinctions.

First of all, we present different performance measures, and after that we describe the different performance predictors proposed in literature.

[4] Content and link analysis scores were normalised as follows:

$$m_c(d_i) = \frac{m_c(d_i)}{\sum_j m_c(d_j)}, m_l(d_i) = \frac{m_l(d_i)}{\sum_j m_l(d_j)}$$

where m_c and m_l denote the bodies of evidence for the content analysis and link analysis

[5] 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.

6.1 Measures

In order to predict the performance of a query, the first step is to differentiate highly performing queries from poorly performing queries [24]. This can be approached using several measures such as the ones we list below.

In the initial robust track [53] two measures were proposed to study how well IR systems are able to avoid very poor results for individual topics: *%no measure* (percentage of topics that retrieved no relevant documents in the top ten retrieved) and *area measure* (area under the curve produced by plotting MAP(X) vs X when X ranges over the worst quarter topics); but these measures were shown to be unstable. A third measure was introduced: *gmap*. Gmap is computed as a geometric mean of the average precision scores of the test set of topics. This measure gives appropiate emphasis to poorly performing topics while being stable with as few as 50 topics [55]. As [54] states, the problem with using MAP as a measure for poorly performing topics is that changes in the scores of best-performing topics mask changes in the scores of poorly performing topics[6]. However, the most commonly used measure to find correlations with is the average precision obtained for each query by each particular system. Recently, we can see a normalized version of average precision that takes into account the topic difficulty [40].

6.2 Non-retrieval approaches[7]

In the field of natural processing language there has been a few attempts to predict performance of queries. One of these is [43], where Mothe *et al.* extract 16 features of the query and study their correlation with respect to recall and average precision. In this study they used TREC 3, 5, 6 and 7 as datasets.

The 16 linguistic features computed were classified in three different classes according to their level of linguistic analysis:

- Morphological features:

  Number of words

  Average word length is the average length of terms in the query, measured in numbers of characters

  Average number of morphemes per word is obtained using the CELEX[8] morphological database. The limit of this method is of course the database coverage, which leaves rare, new, or misspelled words as mono-morphemic.

  Average number of suffixed tokens word , the authors used a bootstrapping method in order to extract the most frequent suffixes from the CELEX database, and then tested for each lemma in the topic if it was eligible for a suffix from this list

  Average number of proper nouns was obtained through the POS tagger's analysis

  Average number of acronyms are detected using a simple pattern-matching technique

[6] For example, the MAP of a run in which the effectiveness of topic A doubles from 0.02 to 0.04 while the effectiveness of topic B decreases 5% from 0.4 to 0.38 is identical to the baseline run's MAP.

[7] This section is a summary of the work carried out by myself this year in the course of *Natural Language Processing*. It is just included here because it is relevant for the subject in hand.

[8] CELEX English database (1993). Available at www.mpi.nl/world/celex

Average number of numeral values are detected using a simple pattern-matching technique

Average number of unknown tokens are those marked up as such by the POS tagger. Most unknown words are constructed words such as “mainstreaming”, “postmenopausal” or “multilingualism”

- Syntactical features:

  Average number of conjuctions detected through POS tagging

  Average number of prepositions detected through POS tagging

  Average number of personal pronouns 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 hierarchy. 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 of the distance between each individual syntactic links (in terms of number of words) over all syntactic links.

- Semantic feature:

  Average polysemy value corresponds to the number of synsets in the WordNet[9] database each word belongs to

Mothe *et al.* found that:

- The only positively correlated feature is the number of proper nouns
- Many variables do not have significant impact on any evaluation measure. Only the more *sophisticated* features appear more than once
- The only two variables found correlated in more than one TREC campaign are the average syntactic links span (for precision) and the average polysemy value (for recall)

We have reproduced their experiments with other datasets and our results are in table 3. In these experiments, we were not able to extract the average number of morphemes per word and the average number of suffixed tokens word because the CELEX database was not available. Nevertheless, we add two new features in our analysis:

Number of hyponyms[10] this number is given directly by Wordnet

Average number of hyponyms

[9] http://wordnet.princeton.edu/

6.3 Pre-retrieval approaches[11]

In this category, performance predictors do not rely on the retrieved document set. The efficiency of this kind of predictor is often high since the performance score can be computed prior to the retrieval process. However, regarding prediction accuracy, these predictors generally have a low performance since many factors related to retrieval effectiveness are not exploited [59].

Some researchers have used IDF-related (inverse document frequency) features as predictors. For example, He and Ounis [23] proposed a predictor based on the standard deviation of the IDF of the query terms. Plachouras [44] represented the quality of a query term by Kwok's inverse collection term frequency. These IDF-based predictors showed some moderate correlation with query performance.

Diaz and Jones [16] have tried time features for prediction. They found that although they are not highly correlated to performance, using these time features together with clarity scores improves prediction accuracy. Kwok *et al.* [33] built a query predictor using support vector regression. For features, they chose the best three terms in each query and used their log document frequency and their corresponding frequencies in the query. They observed a small correlation between predicted and actual query performance. He and Ounis [23] proposed the notion of query scope for performance prediction, which is quantified as the percentage of documents that contain at least one query term in the collection $(-\log{(N_Q/N)}$, 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 a collection). Query scope is effective in inferring query performance for short queries in ad-hoc text retrieval, and it seems to be very sensitive to the query length [38].

In table 4 there are some results that show the correlations we have found using the query scope and the average precision in TREC 8, 9 and 2001.

6.4 Post-retrieval approaches

In this category, predictors make use of retrieved results in some manner. Generally speaking, techniques in this category provide better prediction accuracy compared to those in previous category. However, computational efficiency can be an issue for many of these techniques.

Using visual features, such as titles and snippets, from a surrogate document representation of retrieved documents, Jensen *et al.* [28] trained a regression model with manually labeled queries to predict precision at the top 10 documents in the Web search. The authors reported moderate correlation with precision. Elad Yom-Tov *et al.* [57] proposed a histogram-based predictor and a decision tree based predictor. The features used in their models were the document frequency of query terms and the overlap of top retrieval results between using the full query and the individual query terms. Their idea was that well-performing queries tend to agree on most of the retrieved documents. They reported promising prediction results and showed that their methods were more precise than those used in [33, 44].

There are some other techniques based on measuring some characteristics of the retrieved document set to estimate performance. For example, the clarity score measures the coherence of the retrieved document set. In fact, the initial success of the clarity method has inspired a number of similar techniques. Amati [2] proposed to use the KL-divergence (as one possible probabilistic model) between a query term's frequency in the top retrieved documents and the frequency in the whole collection, which is very

[11] Some descriptions of this and next section have been extracted from [59]

similar to the definition of the clarity score. He and Ounis [23] proposed a simplified version of the clarity score where the query model is estimated by the term frequency in the query, i.e. the authors proposed the following calculation:

$$\begin{aligned} SCS &= \sum_{Q} P_{ml}(w|Q) \log_2 \frac{P_{ml}(w|Q)}{P_{coll}(w)} \\ P_{ml}(w|Q) &= \frac{qtf}{ql} \end{aligned}$$

where qtf is the number of occurrences of a query term w in the query and ql is the query length.

Carmel *et al.* [7] found that the distance measured by the Jensen-Shannon Divergence (JSD) between the retrieved document set and the collection is significantly correlated to average precision. Vinay *et al.* [51] 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,
- the local intrinsic dimensionality

The most effective measure is the sensitivity to document perturbation, an idea similar to the robustness score but it does not perform equally well for short queries and prediction accuracy drops considerably when a state-of-the-art retrieval technique (like Okapi or a language modeling approach) is adopted for retrieval instead of the tf-idf weighting used in their paper [59].

Kwok *et al.* [32] suggest predicting query performance by retrieved document similarity. The basic idea is that when relevant documents occupy the top ranking positions, the similarity between top retrieved documents should be high, based on the assumption that relevant documents are similar to each other. While this idea is interesting, preliminary results are not promising. A similar technique can be found in [21]. Grivolla *et al.* calculate the entropy and pairwise similarity: first, the entropy of the set of the K top-ranked documents for a query is defined as $H = -\sum_{w \in W} P(w) \cdot \log P(w)$ where

$$P(w) = \frac{\sum_{d \in D} N_d(w) + \epsilon}{\sum_{v \in W} \sum_{d \in D} N_d(v) + |W|\epsilon}$$

is the probability of the word w in the document set, D is the set of documents on which to calculate the entropy, W is a lexicon of keywords, $N_d(w)$ the number of occurrences of w in d, and ϵ is a constant used to overcome the well-known zero-frequency estimation problem. This entropy should be higher when the performance achieved for a given query is bad. Secondly, as a score of the same type as the entropy, they defined the mean cosine similarity of the documents (DMS), using the base form of tf-idf term weighting $w_{ij} = tf_{ij} \cdot \log \frac{|D|}{df_i}$ for term t_i in document d_j.

Diaz [15] proposes a technique called spatial autocorrelation for performance prediction. This technique measures the degree to which the top ranked documents (for a given retrieval) receive similar scores by spatial autocorrelation of the retrieval. This approach is based on the cluster hypothesis [27]: closely-related documents tend to be relevant to the same request. A significant correlation between score consistency and retrieval performance was observed in their experiments.

Zhou *et al.* [61] defined two more techniques:

- Weighted Information Gain (WIG) measures the change in information about the quality of retrieval (in response to query Q_i) from an imaginary state that only an average document is retrieved to a posterior state that the actual search results are observed:

$$\begin{aligned} WIG(Q_i, C, L) &= H_{Q_i,L}(Q_S, C) - H_{Q_i,L}(Q_S, D_t) = \\ &= \sum_{s,t} \text{weight}(Q_S, D_t) \log \frac{P(Q_S, D_t)}{P(Q_S, C)} = \\ &= \frac{1}{K} \sum_{D_t \in T_K(L)} \log \frac{P(Q_i, D_t)}{P(Q_i, C)} \end{aligned}$$

  where C is the collection and L the ranked list of documents. The heart of this technique is how to estimate the joint distribution $P(Q_S, D_t)$. Zhou *et al.* decide to adopt Metzler and Croft's Markov Random Field (MRF) model:

$$\log P(Q_i, D_t) = -\log Z_1 + \sum_{\xi \in F(Q_i)} \lambda_\xi \log P(\xi | D_t)$$

  The authors consider two kinds of features: single term features T and proximity features P. Proximity features include exact phrase and unordered window features.

- Query Feedback (QF) measures the degree of corruption that arises when Q is transformed to L (output of the channel when the retrieval system is seen as a noisy channel). They design a decoder that can accurately translate L back into new query Q' and the similarity S between the original query Q and the new query Q' is adopted as a performance predictor.

In table 4 can be seen correlations found in our experiments by clarity score (*clarity*) and simplified clarity score (*SCS*) using TREC 8, 9 and 2001 as datasets.

6.5 Usage of training data

In [60] the following classification is proposed, based on the need for training data. Since all the techniques have already been described in previous sections, only the classification is given.

- No training data: IDF-related features as predictors (standard deviation of the IDF of the query terms [23], Kwok's inverse collection term frequency [44]), related to the ideas in the clarity score technique (KL-divergence between a query term's frequency in the top retrieved documents and the frequency in the whole collection [2], simplified version of the clarity score where the query model is estimated by the term frequency in the query [23], percentage of documents that contain at least one query term in the collection (query scope) [23], clarity scores extended to include time features [16]), predict query performance by retrieved document similarity [32].

Queries	Pearson	Spearman	Kendall
All	Proper nouns (0.2305), hyponymy (-0.1808), polysemy (-0.1933), normalized polysemy (-0.2799)	Proper nouns (0.2103), polysemy (-0.2089), normalized polysemy (-0.2506)	Proper nouns (0.1726), polysemy (-0.1414), normalized polysemy (-0.1685)
TREC 8	Proper nouns (0.2857), syntactic depth (-0.1201)	Proper nouns (0.3360), syntactic depth (-0.0275)	Proper nouns (0.2772), syntactic depth (-0.0211)
TREC 9	Proper nouns (0.2978), hyponymy (-0.3084), normalized polysemy (-0.3218)	Normalized polysemy (-0.3445), normalized hyponymy (-0.3099)	Normalized hyponymy (-0.2177), normalized polysemy (-0.2276)
TREC 2001	Acronyms (0.3626)	Acronyms (0.2814)	Acronyms (0.2320)

Table 3: Linguistic features found statistically significant correlated with average precision (correlation in parenthesis, the greater absolute value, the more dependance between variables)

- With training data: histogram-based predictor and a decision tree based predictor (features: document frequency of query terms and the overlap of top retrieval results between using the full query and the individual query term) [57], using support vector regression (features: the best three terms in each query, their log document frequency and their corresponding frequencies in the query) [33], regression model with manually labeled queries to predict precision at the top 10 documents (visual features from a surrogate document representation of retrieved documents) [28]

6.6 Note about query types

An important issue related with section 4 is that most work on prediction has focused on the traditional ad-hoc retrieval task where query performance is measured according to topical relevance. In fact, little work has addressed other types of queries such as named-page finding (NP) queries (table 2). Moreover, these prediction models are usually evaluated on traditional TREC document collections which typically consist of no more than one million relatively homogenous newswire articles.

As can be seen in [59], the prediction accuracy of the clarity and robustness score is low compared to Weighted Information Gain and Query Feedback. Zhou suggests that clarity and robustness score have difficulty in adapting to a Web collection, and these predictors need to consider more documents than WIG or Query Feedback to adequately measure the coherence of a ranked list. Further investigation showed that collections with high mean average precision causes low accuracy of the clarity score, since the ranked list retrieved in these collections are more similar in terms of coherence at the level of top N documents, and it is required to increase N. This can be an explanation to the abscence of papers using clarity scores with large datasets (such as WT10G).

Queries	Pearson	Spearman	Kendall
All	SCS (0.2615), clarity (-0.2154)	SCS (0.3519), clarity (-0.3005)	SCS (0.2361), clarity (-0.2003)
TREC 8	Scope (0.4771), SCS (0.6037)	Scope (0.3248), SCS (0.4919), clarity (-0.3268)	Scope (0.2640), SCS (0.3339), clarity (-0.2327)
TREC 9		SCS (0.4402)	SCS (0.3011)
TREC 2001	Clarity (-0.4822)	Clarity (-0.4452)	Clarity (-0.3004)

Table 4: Non-linguistic features found statistically significant correlated with average precision (correlation in parenthesis, the greater absolute value, the more dependance between variables)

In table 5 we can see a comparison of prediction techniques (where QM stands form query language model, CM for collection language model, CB for content-based queries and NP for named-page finding queries) extracted from [59] showing the query type each technique was designed for.

Technique	Key ideas	Designed for
Clarity	KL-divergence between QM and CM	CB
JSD	Jensen-Shannon Divergence between QM and CM	CB
Ranking Robustness	Perturb terms in the top ranked documents	CB and NP
Query Feedback	Similarity between the original query and the new query based on clarity contribution	CB
WIG	The difference between two weighted entropies	CB and NP

Table 5: Non-linguistic features found statistically significant correlated with average precision

6.7 Generalization of clarity score

In order to obtain a more general formula for the clarity score, we rewrite clarity score and WIG formulae as Zhou does in [59]:

$$\begin{aligned} \text{clarity} &= \sum_{w \in V} \sum_{D \in L} P(D|Q)P(w|D) \log \frac{\sum_{D \in L} P(D|Q)P(w|D)}{P_{coll}(w)} \\ WIG(Q_i, C, L) &= \frac{1}{K} \sum_{D_t \in T_K(L)} \sum_{\xi \in F(Q_i)} \lambda_\xi \log \frac{P(\xi|D_t)}{P(\xi|C)} \end{aligned}$$

In this form, both formulae are very similar. Actually, they can be written in the same form as follows:

$$\text{score}(Q, C, L) = \sum_{\xi \in T} \sum_{D \in L} \text{weight}(\xi, D) \log \frac{P(\xi, D)}{P_{\text{coll}}(\xi)}$$

where T is a feature space and L is a ranked list. Besides this, $D \in L \subseteq C$ must be comparable somehow with elements $\xi \in T$, in order to make sensible functions weight(ξ, D) and $P(\xi, D)$.

Once we have this general form, we can find that clarity and WIG differ in the following three aspects:

1. The feature space T:

   - For clarity, the feature space is the whole vocabulary consisting of single terms.
   - For WIG, the feature space is single terms or phrases that extracted from query Q.

2. The $weight(\xi, D)$:

   - For clarity score, $weight(\xi, D) = P(D|Q)P(\xi|D)$
   - For WIG, $weight(\xi, D) = \frac{\lambda_\xi}{K}$ if D is one of the top K documents in L, and 0 otherwise.

   We can see that the $weight(\xi, D)$ for WIG is a almost a constant and is much simpler than that for clarity, which makes WIG free from estimation noise in $P(\xi|D)$ and $P(D|Q)$.

3. $P(\xi, D)$:

   - For the clarity score, $P(\xi, D) = \sum_{D \in L} P(D|Q)P(\xi|D)$
   - For WIG, $P(\xi, D) = P(\xi|D)$.

   This means that the clarity score uses a document model averaged over all documents in the ranked list for $P(\xi, D)$, while WIG uses the actual document model of document D.

7 Rank fusion

Query characterisation methods can be very useful in the context of metasearch, since they can help to decide which of the different sources is more trustworthy, inferring a value for each of them in order to combine their results. Accordingly we give here a short introduction to the topic.

Rank fusion or rank aggregation is needed when you want to combine various result lists from different sources into one list, with no knowledge neither of the process followed by each source to produce those lists nor the data that has been used or the score rank for each element. Examples were rank fusion takes place include, for instance, metasearch, personalised retrieval (combine personalised results with query-based results), multi-criteria retrieval, etc

There are some inherent problems in rank fusion: each source can use different methods to return the documents (by similarity or dissimilarity with respect to the query, counting term frequencies or evaluating the underlying probabilistic methods), the score ranks for each document (if they are known) can be different for each source. For these reasons, fusion process is divided in normalisation (data transformation into a common domain before the next phase) and combination (method for joining the distinct normalised lists into one). There are techniques in each of these phases that use the document rank position, others use the score returned by the source for each document; there are also techniques that use training data.

We will use the following notation:

$\Omega = \{d_1, \cdots, d_n\}$	Document set
$\mathcal{R} = \{\tau_1, \cdots, \tau_k\}$	Ranks to combine
$\tau(d)$	Position of document d in ranking τ
$s_\tau(d)$	Score of d in ranking τ
$\Omega\mathcal{T} \subset \Omega$	Documents retrieved by τ
$\Omega_\mathcal{R} = \cup_{\tau \in \mathcal{R}} \Omega_\tau \subset \Omega$	Documents retrieved by some τ in $\mathcal{R}$
$\bar{s}_\tau(d)$	Normalized score of d in τ
$s_\mathcal{R}(d)$	Combined score of all rankings from $\mathcal{R}$
σ^2	Variance

7.1 Normalisation methods

This is a very important process, since it moves the initial data to a common domain, where the next steps apply.

These methods can be divided in two groups: if they apply over ranking positions or ranking scores. There also some methods that need training data, although the majority do not need them.

Rank based :

$$\begin{aligned}
\text{Rank-sim} \quad &: \quad \bar{s}_\tau(d) = 1 - \frac{\tau(d) - 1}{|\Omega_\tau|} \\
\text{Borda} \quad &: \quad \bar{s}_\tau(d) = \begin{cases} 1 - \frac{\tau(d)-1}{|\Omega|} & , \quad d \in \Omega \\ \frac{|\Omega| - |\Omega_\tau| + 1}{2|\Omega|} & , \quad d \notin \Omega \end{cases} \\
\text{Bayes} \quad &: \quad \bar{s}_\tau(d) = \log \frac{P(\tau(d)|d \text{ is relevant})}{P(\tau(d)|d \text{ is irrelevant})}
\end{aligned}$$

Bayes method is the only one that needs training data, it requires some *a priori* relevance assessments (such as the ones included in TREC collections)

Score based :

$$
\begin{aligned}
\text{Standard} &: \quad \bar{s}_\tau(d) = \begin{cases} \frac{s_\tau(d) - \min_{d' \in \Omega_\tau} s_\tau(d')}{\max_{d' \in \Omega_\tau} s_\tau(d') - \min_{d' \in \Omega_\tau} s_\tau(d')} & , \quad d \in \Omega \\ 0 & , \quad d \notin \Omega \end{cases} \\
\text{Sum} &: \quad \bar{s}_\tau(d) = \begin{cases} \frac{s_\tau(d) - \min_{d' \in \Omega_\tau} s_\tau(d')}{\sum_{d' \in \Omega_\tau} s_\tau(d') - \min_{d' \in \Omega_\tau} s_\tau(d')} & , \quad d \in \Omega \\ 0 & , \quad d \notin \Omega \end{cases} \\
\text{ZMUV} &: \quad \bar{s}_\tau(d) = \begin{cases} \frac{s_\tau(d) - \mu}{\sigma^2} & , \quad d \in \Omega, \sigma^2 \neq 0 \\ 0 & , \quad d \in \Omega, \sigma^2 = 0 \\ -2 & , \quad d \notin \Omega \end{cases} \\
\text{2MUV} &: \quad \bar{s}_\tau(d) = \begin{cases} 2 - \frac{s_\tau(d) - \mu}{\sigma^2} & , \quad d \in \Omega, \sigma^2 \neq 0 \\ 0 & , \quad d \in \Omega, \sigma^2 = 0 \\ 0 & , \quad d \notin \Omega \end{cases} \\
\text{Manmatha} &: \quad \bar{s}_\tau(d) = P(y \text{ is relevant } | s_\tau(y) = s_\tau(x))
\end{aligned}
$$

All of these methods need no training data. However, Manmatha model [39] assumes that the set of non-relevant documents follow an exponential distribution, and a Gaussian distribution for the set of relevant ones. The density parameters are approximated using the Expectation-Maximisation method. The other methods were proposed by Montague and Aslam in [41] and their difference is in the parameter shifted to zero (the minimum, the mean) and the value set to non-retrieved documents. Because of this, ZMUV and 2MUV are outlier-insensitive.

Probabilistic normalisation of score distributions

In [19] we can find a novel score-based normalisation method. This method tries to avoid the distortion due to combine input sources with different individual biases. The authors use the score distributions and calculate an *optimal score distribution* (OSD) in order to map the input scores and obtain comparable distributions.

This model assumes an ideal unbiased scoring function $r(x)$ exist, ranging in $[0, 1]$, its cumulative distribution is $\bar{F}$. Given a scoring function $s_\tau(x)$ and its cumulative distribution F_τ, then the normalised function is $\bar{s}_\tau = \bar{F}^{-1} \circ F_\tau \circ s_\tau$, which is a solution to $P(s_\tau(y) \leq \bar{s}_\tau(x)) = P(r(y) \leq \bar{s}_\tau(x))$. This can be done with the following steps:

1. Compute the score distribution F_τ of each input system τ

2. Find a good approximation to an unbiased strictly increasing distribution $\bar{F} : [0, 1] \rightarrow [0, 1]$.

3. For each $x \in \Omega$ and $\tau \in \mathcal{R}$ normalise the score x:

$$s_\tau(x) \longrightarrow \bar{s}_\tau(x) = \bar{F}^{-1} \circ F_\tau \circ s_\tau(x)$$

4. Combine the normalized scores using some score combination strategy.

First two steps can be done *offline*.

This approach is an alternative to others normalisation methods, since it is better than other techniques. It can be extended using historic data [20].

7.2 Combination methods

We can distinguish two types of methods, according they use the score or the rank of the document:

Score based

$$\begin{aligned}
\text{CombMIN} &: \quad s_{\mathcal{R}}(d) = \min_{\tau\in\mathcal{R}} \bar{s}_\tau(d) \\
\text{CombMED} &: \quad s_{\mathcal{R}}(d) = \text{median}_{\tau\in\mathcal{R}} \bar{s}_\tau(d) \\
\text{CombMAX} &: \quad s_{\mathcal{R}}(d) = \max_{\tau\in\mathcal{R}} \bar{s}_\tau(d) \\
\text{CombSUM} &: \quad s_{\mathcal{R}}(d) = \sum_{\tau\in\mathcal{R}} \bar{s}_\tau(d) \\
\text{CombANZ} &: \quad s_{\mathcal{R}}(d) = \frac{1}{h(d,\mathcal{R})} \sum_{\tau\in\mathcal{R}} \bar{s}_\tau(d), \\
& \quad h(d,\mathcal{R}) = \text{ number of input systems that retrieve } d \\
\text{CombMNZ} &: \quad s_{\mathcal{R}}(d) = h(d,\mathcal{R}) \sum_{\tau\in\mathcal{R}} \bar{s}_\tau(d)
\end{aligned}$$

These are some of the most popular and effective combination algorithms to date. They need no training data. CombMNZ and CombSUM are the best methods [37], proposed by Fox and Shaw [49].

There are other methods that requires training data: Bartell (describes different strategies to set the weights, such as the *Conjugate Gradient method* [4]) and Vogt (linear combination and neural net fusion methods [52]).

Rank based The two methods belonging to this category need no training data. One of them uses Markov chains, and the other is the weighted Borda method.

The Markov chain model [18] where the set of states Ω_R and the transition matrix is computed by different strategies, based on the rankings produced by the different input systems. Some specific models to define the transitions are, given the current state $d \in \Omega_R$:

- MC_1: from current state d choose uniformly from all the sources a state d' such that $\tau(d') \geq \tau(d)$ for some τ
- MC_2: first choose uniformly one source τ including d, and then a state d' in τ such that $\tau(d') \geq \tau(d)$
- MC_3: equivalent to the previous one but choosing uniformly the state d', so that if $\tau(d') \geq \tau(d)$ hold the transition os done d', otherwise we stay in d
- MC_4: choose uniformly a state d', if $\tau(d') \geq \tau(d)$ for the majority of $\tau \in R$ then move to d', else, stay in d

The probability values for each state in the stationary distribution of the Markov chains defined by these models is taken as the score that determines the fused rankings. In other words, if $P : \Omega_R \to [0, 1]$ is a stationary distribution then $s_R(d) = P(d)$. Between the four presented models, MC_1 and MC_4 tend to be the top ones.

Weighted Borda model is based in Borda model, but votes are weighted taking into account the quality of the source.

Hybrid methods In this case we only have the logistic regression model, proposed by Savoy[46] where ranks and scores are combined as follows:

$$\begin{aligned} s_R(d) &= \frac{1}{1+e^{-\alpha-\beta\cdot u(d)}} \\ \beta\cdot u(d) &= \sum_{\tau\in R}\beta_{\tau,1}\cdot\tau(d)+\beta_{\tau,2}\cdot s_\tau(d)+\beta_{\tau,3}\cdot\sigma^2_\tau(d) \end{aligned}$$

where $\alpha, \beta_{\tau,i}, i = 1, 2, 3$ are parameters to be learnt by each source τ and σ^2_τ is the variance of the normalised relevance scores for the source τ.

Score-based combination can also be combined with rank-based normalisation. For instance, the Bayes normalisation followed by CombSUM is competitive with score-based techniques.

8 Application of clarity measures to personalised IR

We have seen previously the formula of a general score function:

$$\text{score}(Q, L, C) = \sum_{\xi\in T}\sum_{D\in L}\text{weight}(\xi, D)\log\frac{P(\xi, D)}{P_{\text{coll}}(\xi)}$$

where T is a feature space and L is a ranked list.
We have to note that

- In the personalisation scenario there is no query, and
- the list L can be unordered, since the ordering is used to choose some top-N when the list is too long

The second observation simplifies the model we are going to build, although, as we are going to show, this notion can be added to it without too much effort. Without any doubt, the first observation requires a deeper understanding of the problem to solve.

In the personalisation space we have three objects:

- A set of users $U = \{u\}$
- A set of items $I = \{i\}$
- A mapping between these sets: $m : (u, i) \longrightarrow w_{ui}$, where w_{ui} stands for the weight that users u have for the item i (preference for that item)

Our goal is to translate these objects into the concepts needed by the score function. Depending on what we want to measure, we will have different choices, but it sounds sensible to take I as the collection, since items can be documents, and we have a situation like the one in which clarity score is used [12]. Now the problem is still the query. If we consider that we have no query, then we should have to take as the list L all the collection, but this measure will say nothing, because we want to detect differences

[12] Actually, we can take U as the collection, but in this case we will be able to measure different things we are not going to consider at this point.

between the list and the collection. Since we want to involve the user (we want to apply the model to personalise), we can try to take each user as the query, if we interpret the mapping function m as a retrieval function that returns a list of items (elements from the collection) given a user (the query). Proceeding in this way, we can build the list L:

$$L_u = \{i : \exists w \neq 0 : m(u, i) = w\}$$

Actually, this list can be sorted according to the weights for each item. Moreover, if a more fine-grained model is wanted, two lists can be created: one with only the positive preferences and other with the negative ones. At this point we have not considered the feature space T yet. In this space we have one constraint: we need its elements been comparables with respect to the elements of the collections (the items). Our approach considers this space as an ontological space, where each $\xi \in T$ is a concept. Using this approximation, the weight function is easily defined as the weight given by an annotator for a concept in an item, and the probability of relevance can be calculated similarly as with documents: relative frequency of a concept in an item and the overall frequency in the collection.

Once this model has been created, we can give an interpretation to the score returned by such a function: it will measure the coherence (with respect to the whole collection) of the list of preferences (profile) for a user. This means that a list very coherent has concepts very correlated, and, from the point of view of personalisation, it will be easy for the method find items relevant for the user, however, it will be very difficult for it to find items related with concepts unknown by the user (protfolio effect). Therefore, a first application of this measure can be the *adaptation* of the recommendation algorithm to take these information into consideration.

We have to note that this model is very general, and admits different levels of detail. For example, in the context of movie recommendation, each item (a movie) has a lot of features that can be of interest for the user: director, actor, genre, etc. It is not required for this model to choose only one set of features, we can deal with this issue in the following way:

$$\begin{aligned} T_1 &= \{\text{directors}\}, T_2 = \{\text{actors}\}, \cdots, T_n = \{\text{genres}\} \\ \vec{T} &= \begin{pmatrix} T_1 & \cdots & T_n \end{pmatrix} \\ \vec{S} &= \begin{pmatrix} S_1 & \cdots & S_n \end{pmatrix} \end{aligned}$$

where $\vec{S}$ is the score vector for a user, where each of its components is the score given by the previous formulation and a specific set of features. Another possible approach can be to do some kind of linear combination of these scores in order to return only one number for each user:

$$S_u = \sum_i \lambda_i S_i, \lambda_i \in \mathcal{R}$$

The main problem of this aproximation is its lack of bijectivity, what makes impossible to give explanations for a calculated score.

9 Experiments

We have considered the following experiments:

1. Implementation of some of the techniques read in the literature, and compare some examples with the ones mentioned in the papers.

2. Given a collection with a list of predefined queries (such as TREC), calculate the query clarity for each query and split the queries according to their score. Then, we can calculate somehow (parametric vs non-parametric estimation) a distribution for each segment of clarity. When a new query is received, its clarity score is calculated and the corresponding distribution is assigned.

3. For rank fusion, we may want to give more weight to the system that produces a higher clarity score given a query.

In these experiments, when we say *clarity score* we can try any of the other techniques proposed in the area. Actually, we have results from the first two experiments (using clarity score), and the third one has been developed but not finished at the time of writing.

We have also found interesting to apply these ambiguity-driven methods to other fields, in a novel manner, such as:

4. Personalisation: these techniques are able to show which users are more focused in what, and which are interested in almost everything (concepts in ontology, items in a recommender system, ...), even discovering relations between users (analysing their similarity).

5. Folksonomies: using clarity scores and the other related techniques it is possible to emerge some kind of structure from a given set of tags, studying similarity between tags or even classifying them into more general or more concrete classes, as a first step for building an ontology.

Here, we can test quality of these techniques against others used in the literature (although not directly applied in this type of experiments, such as LSA[14] or Normalized Compression Distance[9]).

We have begun modelling the first experiment (see section 8) and some results are shown later on. It is important to note that all the experiments aim to be prospective and by no means conclusive.

9.1 Test collection analysis

In order to make our experiments comparable with the rest of experiments found in the literature, we have carried out an analysis about the datasets, query-sets and measures used in some (important) works. Besides, we have noted which of those papers give enough information to repeat *in the same conditions* the experiment, i.e. if the authors only provide some graphics, it is more difficult to compare (or even impossible) against them than if they give an in-depth study, with a complete table for example.

Because of this, we have combined this information in the tables 6 and 7, focusing on papers related with our experiments (ambiguity and metasearch). In these tables, *contrastable* refers to an experiment whose functionality can be implemented and tested, even with different datasets; *reproducible* refers to the experimental setup, i.e. if all the initial conditions of the experiment are provided in order to repeat it.

Based on this table, we have found that TREC 8 collection allows comparisons with other works in metasearch, whereas Robust Track 2004 has been used when authors want to predict query difficulty.

Reference	Dataset	Query-set	Metrics	Contrastable	Reproducible
[24]	TREC disk4&5 (minus CR)	TREC2004 Robust Track (topics 301-450, 601-700)	r and p-value of the linear dependence between the average precision and each of the predictors	Yes	Yes
[2]	Idem	100 statements (among these, 50 difficult topics from all 150 queries of previous TRECs)	MAP, P@10	Yes	No (which topics?)
[60]	TREC1-3, TREC4, Robust04, Terabyte04, Terabyte05	All of each track (titles)	Kendall's rank and Pearson's correlation (with average precision), coefficient of determination and standardized regression coefficients (dependent variable is average precision)	Yes	Yes

Table 6: Information about different experiments carried out in the study of ambiguity, clarity and difficulty

These are the reasons why we have used the TREC 8 collection for our main experiments, despite we use query difficulty predictors.

9.2 Description of the experiments

At the beginning, we considered two experiments:

- Cluster queries according their clarity score
- In a metasearch context, use the clarity score to weight the results of each search engine

Both experiments have some advantages and disadvantages:

- The first work is a natural extension of another work made in this group [19], besides this it would allow testing a modification of that work prepared in an-

Reference	Dataset	Query-set	Metrics	Contrastable	Reproducible
[41]	Ad hoc track of TREC3, TREC5, TREC9. Subset of the TREC5 data defined by Vogt	All of each track (and random set of n engines)	Average precision	No (only curves provided)	No (random choice)
[39]	Ad hoc track of TREC3, TREC4, TREC5, TREC9	All of each track (and top n engines)	Average precision	Yes	Yes
[20]	Web track of TREC8, TREC9, TREC9L, TREC2001	All of each track	Average precision (averaged over the 4 collections)	Yes	Yes
[58]	TREC8 collection	249 topics	P@10, MAP, number of queries with no relevant results in the top 10 results (%no)	Yes	Yes
[3]	TREC5, TREC6, TREC7, TREC8, Robust04, Terabyte04, Terabyte05	All queries (among $2, 5, 10, 20$ (random choice and repeated 10 times), or all retrieval runs available)	Query average AP, query median AP (for collection query hardness), median-system AP (for system query hardness), Kendall's τ, correlation coefficient ρ	Yes	Yes
[7]	.GOV2	100 topics (from 2004 and 2005 terabyte tracks)	Average and median precision, Pearson and Spearman correlation	Yes	Yes

Table 7: Information about different experiments carried out in the rank aggregation field

other course (final work in *Information Retrieval*) consisting of estimating non-parametrically the distributions used in [19]. The problem here is that we need a lot of training and testing with respect to which are the best clusters. We could try different predictors.

- The second experiment would be easier to develop (because it does not require any training phase) but it has been carried out in a very similar way by other authors [58].

While thinking about which could be the best experiment we have found another one. It is focused in metasearch, and it would rank documents (from different sources) according to the clarity score using a voting-like mechanism:

- We start from an empty list L, used for aggregating documents
- For each source we have a document d_i which is the first document not used in that list at the moment
- In each step, we take the document $d \in \{d_i\}$ that maximizes $CS(L \cup d, q)$, where q is the query issued by the user, L is the aggregated list until that moment and CS is the clarity score.

As we said before, we can also try different difficulty predictors as a replacement of the clarity score.

9.3 Results

At the end of this work, four experiments have been carried out:

1. Comparison between linguistic and non-linguistic performance predictors, correlations found between average precision and these predictors. These results have been shown in section 6.
2. Implementation of clarity-driven personalisation model explained in section.
3. Given a set of queries for testing, they are clustered according to their clarity value, and these clusters are used to discriminate which scores have to be taken into account when the source distribution is being build.
4. Use the clarity score to weight each source according to the clarity each one assigns to the query.

All these experiments have been tested using the TREC 8, 9 and 2001 datasets and Terrier was used as the indexing and retrieving tool. Implementation of the clarity score was simplified as follows:

- Sum was not carried out over every term in vocabulary, but only in terms appearing in relevant documents.
- As Cronen-Townsend *et al.* did in [12], we also set a maximum size for set R (in our case: 100).
- Only the subset WT2G was used (instead of whole WT10G).

These simplifications were done in order to obtain some results in a short period of time.

Results related with the fourth experiment listed were not finished by the end of this work. Results related with the second and third one are presented below.

9.3.1 Clarity-driven personalisation model

In this experiment, we did some tests with the Movielens dataset[13]. During these experiments, a simplification was made: only the *genre* feature was used. But a problem came out: how do we have to interpret the query (user preferences)? We found this answer was completely conditional on how the items were interpreted. We had two options, given a feature (in our case, movie genre):

- Each item is seen as a set of genres. This means that the query is also seen as a set of genres (each movie the user like is replaced with its genres), where some decision has to be taken about repetitions.
- Each item is seen as a genre, or equivalently, as a set of movies. In this case, the user is seen as a set of movies (no change made).

An example of both situations follows. Given the movies *Cinderella* and *The sound of music* with associated genres animation, children's and musical, for the first one, and musical, for the second one. A user has preferences about these two movies. The different options are:

- Cinderella = { animation, children's, musical }, TSM = { musical }. The user is seen as U = { animation, children's, musical }.
- The user is U = { Cinderella, TSM }, both movies have no modification because now the items are the genres: Animation = { Cinderella, ... }, Musical = { Cinderella, TSM, ... }, and so on.

We tried our clarity-model for both configurations. Three different clarity scores were calculated:

1. A clarity for each movie (it only has sense in the second case, when each item is a set of movies). In this situation, movies with highest clarity values belong to *film-noir* genre, which is the least popular genre (only a 1.2% of total). Some examples:
    - The movie *Agnes Browne* (comedy and drama) has a clarity value of -0.4130 (the lowest value)
    - *Light It Up* (drama) has a value of -0.3416
    - *Hotel de Love* (comedy and romance) has a value of 0.0879
    - *Destination Moon* (sci-fi) has a value of 0.7181
    - *Force of Evil* (film-noir) has a value of 2.85 (the highest value)
2. For each user and each item is a set of movies. In this situation, user preferences had to be limited to their first three movies, since this user query would be too long and clarity would then be very small.
3. For each user and each item is a set of genres. Here we consider no repetition, what it means that if a user likes two movies from the same genre, this genre only will appear once.

By movie		**By genre**	
User id	Value	User id	Value
1026	3.1181	1946	2.2305
172	3.1181	5096	2.2305
4974	3.1181	3804	2.1955
5999	3.1161	4433	2.1825
2373	3.1156	4238	2.1753
3623	3.1155	1800	2.1650
700	3.1089	2507	2.1580
1034	3.1037	2764	2.1455
4948	3.1024	1053	2.1418
5362	3.0997	180	2.1340
3110	3.0922	889	2.1340
304	3.0868	1310	2.1340
1387	3.0834	1548	2.1340
5055	3.0813	3546	2.1340
4950	3.0801	4369	2.1340
4873	3.0802	4856	2.1340
4920	3.0800	1465	2.1340
5537	3.0744	5847	2.1340
6034	3.0702	3623	2.1339
597	3.0689	1128	2.1234

Table 8: Top 20 “clearest” users. Note that user 3623 appears in both lists. *By movie* refers to the model where the items are sets of movies.

The last two experiments are very similar, but the results are slightly different. In table 8 we can see the top 20 users for each situation. We can find here that both lists are not the same, rearranged list.

We have to note that there are groups of users with the same value of clarity (this is more likely to happen when each item is a set of genres), and inside these groups clarity is sorted according to the movie id.

9.3.2 Score normalisation based on clarity-oriented clustering

In this section we show the results found in one of our experiments. This experiment has the following steps:

- For each query, find its clarity value
- Cluster queries according to their clarity value (in our case, we had two clusters)
- Apply probabilistic normalisation of score distributions (section 7.1) to each cluster
- Calculate the MAP of the results obtained for each cluster and when no clustering is done and compare them.

[13] http://www.grouplens.org/system/files/ml-data_0.zip

The algorithm presented in section 7.1 and based on [19] has some parameters we have had to choose for our experiment. More specifically, we have chosen CombSUM as a combination method, combination of 2-size systems sets repeating 5 times each sample. With this configuration, we obtained the results summarised in table 9.

Method	**TREC 8**	**TREC 9**	**TREC 2001**
Normal	0.3734	0.1928	0.3273
Clarity^B	0.4566	0.2511	0.3577
Clarity^W	0.2942	0.1342	0.2848

Table 9: MAP for different normalisation methods. The separation when clarity is used is given by the median value of all the query clarities involved in each track. If the superscript is B the cluster used in the normalisation is fromed with the less ambiguous queries (greater clarity value).

In this table we find some interesting conclusions:

- The unambiguous queries improve their performance when they are normalised only using scores belonging to unambiguous queries.
- Using the median as a cluster generator give similar results in all tracks.

10 Future work

This work has been very productive, since a lot of techniques has been handled and a lot of discussions about them have happened. From these discussions, future research lanes have been opened:

- Define a more general clarity score (or prediction scorer), not restricted to a given query and a collection, but one able to compare complete collections, or even more: complete ordered collections. This could solve the problem of the low correlation between clarity score and average precision when dealing with web (large) collections. A possible approach to incorporate the document ranking into the formula is through a *probability of being seen* or something similar.
- Continue investigating in fuzzy theory, information theory and multicriteria decision making theory.
- Try a different paradigm for the query difficulty prediction: given a normalised distribution (from a source or a set of sources) for a given query, infer the difficulty of that query. This will require a lot of training experiments to know which are the parameters more correlated with the query difficulty in the context of metasearch, such as the combination and normalization methods.
- Check how the clarity score behaves with a dynamic collection, like in the voting-like experiment described in section 9.2.
- Improve the clarity-driven personalisation model taking into account positive and negative preferences and the rating values.

- Use ambiguity predictors in order to measure similarity between users. This is applicable when dealing with folksonomies (comparing tags and tagged items for each user) or for comparing user profiles (defined as a set of concepts). Another application of such a measure can produce some rules when creating groups of users, these groups can be used in collaborative areas, such as recommender algorithms (collaborative filtering) or cooperative work (groupware).

Another interesting task has been untested: combine different predictors linearly or with the aid of genetic algorithms. Actually, in [59] Zhou combined two predictors by a simple linear combination, and they found that the corresponding Pearson's correlation coefficient was increased. Other researchers also report that the combination of multiple prediction features can provide better prediction accuracy than anyone when used in isolation. And in general, performance prediction should be done using a combination of resources, if this is computationally possible, due to the fact that they capture different aspects of the retrieval process that have a major impact on retrieval effectiveness.

11 Conclusions

We have reported here our study, findings, and perspectives of an extensive revision of research addressing the problem of query characterisation as a means to improve retrieval performance.

Some baselines have been found (i.e. clarity score), but other models and approaches for the analysis of uncertainty-related features can be explored as well, such as the fuzzy models, or more specialised vagueness modeling approaches, which we plan to investigate, seeking for new models or, hopefully, solutions to the problems studied and/or proposed in this work.

As interesting or likely more than the conclusions and preliminary findings of the work so far, are he many potential research lanes stemming from this point, several of which are proposed and outlined in this report. The community is indeed already using state-of-the-art techniques such as the ones reported, and applying them in different fields (OLAP [6], temporal retrieval [16], NLP [43], query expansion [8], high accuracy retrieval techniques [48] and more). Metasearch and personalised IR are two of the potential areas we envision in our future research work for the application of the techniques studied here.

References

[1] James Allan and Hema Raghavan. Using part-of-speech patterns to reduce query ambiguity. In *25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 307–314, 2002.

[2] Giambattista Amati, Claudio Carpineto, and Giovanni Romano. Query difficulty, robustness, and selective application of query expansion. *Advances in Information Retrieval*, pages 127–137, 2004.

[3] Javed A. Aslam and Virgiliu Pavlu. Query hardness estimation using jensen-shannon divergence among multiple scoring functions. In *ECIR*, pages 198–209, 2007.

[4] B. Bartell. *Optimizing Ranking Functions: A Connectionist Approach to Adaptive Information Retrieval*. PhD thesis, Department of Computer Science & Engineering, The University of California, San Diego, 1994.

[5] Gloria Bordogna and Gabriella Pasi. Handling vagueness in information retrieval systems. In *ANNES '95: Proceedings of the 2nd New Zealand Two-Stream International Conference on Artificial Neural Networks and Expert Systems*. IEEE Computer Society, 1995.

[6] Burdick, Doug, Deshpande, Prasad, Jayram, T., Ramakrishnan, Raghu, Vaithyanathan, and Shivakumar. Olap over uncertain and imprecise data. *The VLDB Journal*, 16(1):123–144, January 2007.

[7] David Carmel, Elad Yom-Tov, Adam Darlow, and Dan Pelleg. What makes a query difficult? In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 390–397, New York, NY, USA, 2006. ACM.

[8] Claudio Carpineto, Renato de Mori, Giovanni Romano, and Brigitte Bigi. An information-theoretic approach to automatic query expansion. *ACM Trans. Inf. Syst.*, 19(1):1–27, January 2001.

[9] R. Cilibrasi and P. Vitanyi. Clustering by compression. *IEEE Transactions on Information Theory*, 51(4):1523–1545, 2005.

[10] F. Crestani and G. Pasi. *Soft Information Retrieval: Applications of Fuzzy Set Theory and Neural Networks*, pages 287–315. Physica Verlag (Springer Verlag), Heidelberg, Germany, 1999.

[11] Fabio Crestani and Mounia Lalmas. Logic and uncertainty in information retrieval. *Lectures on information retrieval*, pages 179–206, 2001.

[12] Cronen-Townsend, Steve, Zhou, Yun, Croft, and W. Precision prediction based on ranked list coherence. *Information Retrieval*, 9(6):723–755, December 2006.

[13] Steve Cronen-Townsend, Yun Zhou, and W. Bruce Croft. Predicting query performance. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 299–306, New York, NY, USA, 2002. ACM.

[14] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[15] Fernando Diaz. Performance prediction using spatial autocorrelation. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 583–590, New York, NY, USA, 2007. ACM.

[16] Fernando Diaz and Rosie Jones. 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*, pages 18–24. ACM Press, 2004.

[17] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, November 2000.

[18] Cynthia Dwork, Ravi S. Kumar, Moni Naor, and D. Sivakumar. Rank aggregation methods for the web. In *World Wide Web*, pages 613–622, 2001.

[19] Miriam Fernández, David Vallet, and Pablo Castells. Probabilistic score normalization for rank aggregation. In *28th European Conference on Information Retrieval (ECIR 2006)*, pages 553–556. Springer Verlag Lecture Notes in Computer Science, Vol. 3936, April 2006.

[20] Miriam Fernández, David Vallet, and Pablo Castells. Using historical data to enhance rank aggregation. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 643–644, New York, NY, USA, August 2006. ACM Press.

[21] Grivolla, Jourlin, and De Mori. Automatic classification of queries by expected retrieval performance. In *Predicting Query Difficulty - Methods and Applications, SIGIR 2005*, 2005.

[22] Joseph Y. Halpern. *Reasoning about Uncertainty*. MIT Press, October 2003.

[23] Ben He and Iadh Ounis. Inferring query performance using pre-retrieval predictors. In *String Processing and Information Retrieval, SPIRE 2004*, pages 43–54, 2004.

[24] Ben He and Iadh Ounis. Query performance prediction. *Information Systems*, 31(7):585–594, November 2006.

[25] Djoerd Hiemstra. *Using language models for information retrieval*. PhD thesis, 2000.

[26] Bernard J. Jansen, Amanda Spink, and Tefko Saracevic. Real life, real users, and real needs: a study and analysis of user queries on the web. *Information Processing and Management*, 36(2):207–227, 2000.

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

[28] Eric C. Jensen, Steven M. Beitzel, David Grossman, Ophir Frieder, and Abdur Chowdhury. 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*, pages 615–616, New York, NY, USA, 2005. ACM.

[29] George J. Klir. *Uncertainty and Information: Foundations of Generalized Information Theory*, volume 43. Wiley, New York, January 2007.

[30] K. L. Kwok. A new method of weighting query terms for ad-hoc retrieval. In *SIGIR '96: Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 187–195, New York, NY, USA, 1996. ACM.

[31] K. L. Kwok. An attempt to identify weakest and strongest queries. In *Predicting Query Difficulty - Methods and Applications, SIGIR 2005*, 2005.

[32] K. L. Kwok, L. Grunfeld, N. Dinstl, and P. Deng. Trec 2005 robust track experiments using pircs. In *Online Proceedings of 2005 Text REtrieval*, 2005.

[33] K. L. Kwok, L. Grunfeld, H. L. Sun, and P. Deng. Trec 2004 robust track experiments using pircs. In *Online Proceedings of 2004 Text REtrieval*, 2004.

[34] Mounia Lalmas. Dempster-shafer's theory of evidence applied to structured documents: Modelling uncertainty. In *SIGIR*, pages 110–118. ACM, 1997.

[35] Mounia Lalmas. Information retrieval and dempster-shafer's theory of evidence. In *Applications of Uncertainty Formalisms*, pages 157–176. Springer-Verlag, 1998.

[36] Victor Lavrenko, James Allan, Edward Deguzman, Daniel Laflamme, Veera Pollard, and Stephen Thomas. Relevance models for topic detection and tracking. In *Proceedings of the second international conference on Human Language Technology Research*, pages 115–121, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[37] Joon H. Lee. Analysis of multiple evidence combination. In *SIGIR '97: Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 267–276, New York, NY, USA, 1997. ACM Press.

[38] Craig Macdonald, Ben He, and Iadh Ounis. Predicting query performance in intranet search. In *Predicting Query Difficulty - Methods and Applications, SIGIR 2005*, 2005.

[39] R. Manmatha and H. Sever. A formal approach to score normalization for meta search. In *Human Language Technology Conference (HLT 2002)*, pages 88–93, 2002.

[40] Stefano Mizzaro. The good, the bad, the difficult, and the easy: Something wrong with information retrieval evaluation? In *Advances in Information Retrieval , 30th European Conference on IR Research, ECIR 2008*, Lecture Notes in Computer Science, pages 642–646. Springer, 2008.

[41] Mark Montague and Javed A. Aslam. Relevance score normalization for metasearch. In *CIKM '01: Proceedings of the tenth international conference on Information and knowledge management*, pages 427–433, New York, NY, USA, 2001. ACM.

[42] Josiane Mothe and Ludovic Tanguy. Linguistic analysis of users' queries: towards an adaptive information retrieval system.

[43] Josiane Mothe and Ludovic Tanguy. Linguistic features to predict query difficulty. In *Predicting Query Difficulty - Methods and Applications, SIGIR 2005*, 2005.

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

[45] Vassilis Plachouras, Iadh Ounis, Cornelis J. van Rijsbergen, and Fidel Cacheda. University of glasgow at the web track: Dynamic application of hyperlink analysis using the query scope. In *TREC*, pages 646–652, 2003.

[46] J. Savoy, A. Le Calve, and D. Vrajitoru. Report on the trec-5 experiment: Data fusion and collection fusion, 1988.

[47] Glenn Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

[48] Chirag Shah and Bruce W. Croft. Evaluating high accuracy retrieval techniques. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 2–9, New York, NY, USA, 2004. ACM.

[49] Joseph A. Shaw and Edward A. Fox. Combination of multiple searches. In *Text REtrieval Conference*, pages 0+, 1994.

[50] Plachouras Vassilis and Ounis Iadh. Dempster-shafer theory for a query-biased combination of evidence on the web. *Information Retrieval*, 8(2):197–218, April 2005.

[51] Vishwa Vinay, Ingemar J. Cox, Natasa Milic-Frayling, and Ken Wood. 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*, pages 398–404, New York, NY, USA, 2006. ACM Press.

[52] Christopher C. Vogt and Garrison W. Cottrell. Fusion via a linear combination of scores. *Information Retrieval*, 1(3):151–173, 1999.

[53] Ellen M. Voorhees. Overview of the trec 2004 robust retrieval track. In *Proceedings of the Thirteenth Text REtrieval Conference, TREC 2004*, pages 70–79, 2005.

[54] Ellen M. Voorhees. The trec robust retrieval track. *SIGIR Forum*, 39(1):11–20, June 2005.

[55] Ellen M. Voorhees. The trec 2005 robust track. *SIGIR Forum*, 40(1):41–48, June 2006.

[56] Ellen M. Voorhees and Donna K. Harman, editors. *TREC: Experiment And Evaluation in Information Retrieval*. MIT Press, Cambridge, MA, 2005.

[57] Elad Yom-Tov, Shai Fine, David Carmel, and Adam Darlow. 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*, pages 512–519, New York, NY, USA, 2005. ACM Press.

[58] Elad Yomtov, Shai Fine, David Carmel, and Adam Darlow. Metasearch and federation using query difficulty prediction. In *Predicting Query Difficulty - Methods and Applications, SIGIR 2005*, 2005.

[59] Yun Zhou. *Retrieval Performance Prediction and Document Quality*. PhD thesis, University of Massachusetts, September 2007.

[60] Yun Zhou and Bruce W. Croft. Ranking robustness: a novel framework to predict query performance. In *CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 567–574, New York, NY, USA, 2006. ACM.

[61] Yun Zhou and Bruce W. Croft. 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*, pages 543–550, New York, NY, USA, 2007. ACM.

Appendix A TREC datasets

This appendix describes some features about the dataset used in the experiments. A review about two specific tracks are also provided.

TREC[14] stands for *Text REtrieval Conference*. It is a yearly workshop, and it consists of a set of tracks (areas of focus in which particular retrieval tasks are defined). These tracks have several purposes:

- New research areas: the first running of a track often defines what the problem really is, and a track creates the necessary infrastructure (test collections, evaluation methodology, etc.) to support research on its task.
- Robustness of core retrieval technology (if the same techniques are appropriate for a variety of tasks).
- Provide tasks that match the research interests of groups.

Every year new tracks are proposed, and past tracks are submitted, with different collections or different topics.

An IR test collection consists of three parts:

- A set of documents,
- a set of questions (called topics in TREC), that can be answered by some of the documents, and
- the right answers (called relevance judgments) that list the documents that are relevant to each question.

NIST[15] (TREC's sponsor) is very concerned about evaluation metrics, because of this they provide specific metrics to compare different systems, and investigate about more stable measures. Actually, they provide a software (trec_eval) that returns some classical IR measures such as precision, recall, average precision or precision at N given a ranking and the relevance judgments.

We are interested in the Robust Retrieval track because it is focused on **individual topic effectiveness** rather than average effectiveness.

A.1 Information about disk4

The document collections consist of the full text of various newspaper and newswire articles plus government proceedings.

The format of the documents on the TREC disks use a labeled bracketing expressed in the style of `SGML` (Standard Generalized Markup Language). `SGML` DTD's are included on each disk. The different datasets on the disks have identical major structures but have different minor structures. Every document is bracketed by `<DOC></DOC>` tags and has a unique document identifier, bracketed by `<DOCNO></DOCNO>` tags. The datasets have all been compressed using the `UNIX` compress utility and are stored in chunks of about 1 megabyte each (uncompressed size).

The contents of the disks are as follows:

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

[15] National Institute of Standards and Technology: http://www.nist.gov

Collection	Documents	Size
Congressional Record of the 103rd Congress	30000	235 MB
Federal Register (1994)	55000	395 MB
Financial Times (1992-1994)	210000	565 MB

A.2 Information about disk5

The format of the documents on the TREC disks use a labeled bracketing expressed in the style of SGML (Standard Generalized Markup Language). SGML DTD's are included on each disk. The different datasets on the disks have identical major structures but have different minor structures. Every document is bracketed by <DOC></DOC> tags and has a unique document identifier, bracketed by <DOCNO></DOCNO> tags. The datasets have all been compressed using the UNIX compress utility and are stored in chunks of about 1 megabyte each (uncompressed size).

Collection	Documents	Size
Foreign Broadcast Information Service	130000	470 MB
Los Angeles Times (1989-1990)	130000	475 MB

Appendix B Relation between seminal authors

In this appendix we show how the principal papers reference each other. This information is useful to discover if, in a particular moment, there is a technique developed but excluded from the community. For instance, [13] is the more cited, but [43] and [42] and only cited once.

(First) Author	Reference	Year	Citations
Allan	[1]	2002	Cronen-Townsend [13]
Amati	[2]	2004	Cronen-Townsend [13], Kwok [30]
Aslam	[3]	2007	Amati [2], Carmel [7], Cronen-Townsend [13], Kwok [31], Yom-Tov [58, 57], Zhou [60]
Carmel	[7]	2006	Cronen-Townsend [13], He [23], Mothe [43], Yom-Tov [57]
Cronen-Townsend	[13]	2002	Kwok [30]
He	[23]	2004	Amati [2], Cronen-Townsend [13]
Kwok	[30]	1996	
Kwok	[31]	2005	Amati [2], Yom-Tov [57]
Mothe	[43]	2005	Cronen-Townsend [13]
Mothe	[42]	2005	Carmel [7]
Yom-Tov	[57]	2005	Amati [2], Cronen-Townsend [13], He [23]
Zhou	[60]	2006	Amati [2], Cronen-Townsend [13], He [23], Yom-Tov [57]
Zhou	[61]	2007	Carmel [7], Cronen-Townsend [13], He [23], Yom-Tov [57]

Appendix C Meetings with the tutor

25/02/2008 (2 hours) Detailed description and clarification of the goals of the work, confirmation of temporal schedule and detailed workplan.

12/03/2008 (1h.) Review of the literature read at that moment and refinement of short-term goals set: create a glossary and study about some possible experiments.

28/03/2008 (1h.) Revision and discussion of the draft glossary in progress. Starting to record all type of queries (based on TREC's tracks, mainly) for a possible section.

09/04/2008 (1h.) Revision of planning and past tasks. Explanation of some possible experiments. New task: investigate about the robust track.

16/04/2008 (1h.) Revision of planning: bibliography must be finished for the next meeting. Organization of future tasks (as milestones, with tentative dates). Search datasets for experiments.

23/04/2008 (1.5h.) Sections to be finished: TREC, query types, Dempster-Shafer. Discussion about the next experiment-oriented tasks: general survey about datasets used in literature, and find out if the experiments described are likely to be reproducible or, at least, comparable with our future experiments. Analysis about the methodology (dataset, query-set, measure) and a technique for the metasearch experiment (and choose one out of the two discussed ones).

07/05/2008 (1h.) Discussion about methodology and techniques. Complete analysis collection table. Another possible experiment is discussed. For the next session the experiment has to be completely defined.

12/05/2008 (1h.) Three experiments were selected and defined in further detail, to be conducted for this work: metasearch, query clustering and query clarity by section. For the next session, we have to check the literature and find out whether anyone has already tried to directly apply clarity to metasearch.

22/05/2008 (1h.) It would be relevant to find out who cites whom. A preliminary analysis about personalization to be prepared.

29/05/2008 (1h.) Discussion about using clarity in the personalisation model. High priority for experiments.

10/06/2008 (1.5h.) Detailed discussion about the experiments. Future work defined.

18/06/2008 (1h.) Revision of report draft and final presentation

Appendix D Time dedicated by student

Searching and extending bibliography 5 hours

Reading bibliography 2 months (120 hours)

Experiments (design, preparation, realisation, analysis) 1 month (50 hours)

Writing report 1 week (30 hours)

Prepare slides 8 hours

Meetings with tutor 14 hours