Chapter 6

AN OVERVIEW OF EVALUATION METHODS IN TREC AD HOC INFORMATION RETRIEVAL AND TREC QUESTION ANSWERING

Simone Teufel
Computer Laboratory
University of Cambridge, United Kingdom
Simone.Teufel@cl.cam.ac.uk

Abstract This chapter gives an overview of the current evaluation strategies and problems in the fields of information retrieval (IR) and question answering (QA), as instantiated in the Text Retrieval Conference (TREC). Whereas IR has a long tradition as a task, QA is a relatively new task which had to quickly develop its evaluation metrics, based on experiences gained in IR. This chapter will contrast the two tasks, their difficulties, and their evaluation metrics. We will end this chapter by pointing out limitations of the current evaluation strategies and potential future developments.

Keywords Information retrieval; Question answering; Extrinsic evaluation; Intrinsic evaluation; Precision; Recall; Accuracy; MRR; Composite evaluation metrics.

1 Introduction

Why is evaluation of systems in natural language processing and related fields important? Language-related tasks are cognitively hard; for tasks which are interesting, systems do not perform anywhere near the 100% mark. Typically, one cannot assess if an increase in quality has taken place by subjectively looking at the output of two systems (or a possibly improved system and its predecessor). More complicated, objective evaluation strategies for monitoring performance differences are needed, and community agreement on these strategies is also important. For instance, the existence of an objective, agreed-upon, simple evaluation metric (namely word error rate, or WER) in the field of

L. Dybkjær et al. (eds.), Evaluation of Text and Speech Systems, 163–186.

automatic speech recognition (ASR) has substantially helped monitor progress in the task, and thus advance the field: the WER of the state-of-the-art ASR systems continues to go down year by year, for harder and harder sub-variants of the original task. Similar situations have arisen in optical character recognition (OCR) and information extraction. Evaluation can also direct future research: it can signal when a certain ceiling has been reached (i.e., when all state-of-the-art systems perform similarly), and can help define interesting new tasks. For instance, the filtering task in information retrieval (IR), a recent version of the ad hoc search task discussed here, models IR on a life information feed. Its evolution went hand in hand with a new evaluation track in a competitive evaluation conference series called Text Retrieval Conference (TREC) (see below). Similarly, the TREC web track fostered research into the evaluation of web search engines.

We will look at IR and question answering (QA) in the context of the most successful and largest-scale evaluation conference series to date, TREC (Harman, 1993; TREC, 2004), which has been run by NIST since the early 1990s. Other large-scale evaluation efforts for IR include the Cranfield experiment (Cleverdon, 1967), and recently NTCIR (NTCIR, 2005) in Asia and CLEF (CLEF, 2005) in Europe. Experience has shown that competitive evaluation exercises can act as a driving force for finding an objective and meaningful evaluation, for establishing good practice in evaluation, and thus for measuring and fostering progress. The continuity that these exercises bring has also encouraged research teams to make the investment of participating. When TREC emerged in the early 1990s, it marked a new phase in IR evaluation; 2005 was the 15th annual conference. In contrast, QA, a far newer task than IR, has only been run (as a track within TREC) since 1999. QA systems and evaluation strategies have evolved fast in those few years (e.g., three different evaluation measures have been experimented with since 1999). Nevertheless, those two tasks and their evaluation methods are very naturally connected, and we will now review both tasks in more detail.

1.1 The Tasks

1.1.1 Ad hoc Information Retrieval. *Information Retrieval* (IR) is the task of finding documents from a large document collection which are relevant to a user's query. The query can be expressed as a set of keywords or as a natural language description. Figure 1 shows a sample TREC query, asking for documents about diseases which cause hair loss. Different levels of detail are given by the fields `title`, `desc`, and `narr`.

This task where systems return entire textual documents as their output is technically called *document retrieval*; it is the most well-known and popular subtype of IR. There are other subtasks of IR such as retrieval of speech, video, music, and retrieval on text passages (Callan, 1994); this chapter,

```
<num> Number: 508
<title> hair loss is a symptom of what diseases
<desc> Description:
Find diseases for which hair loss is a symptom.
<narr> Narrative:
A document is relevant if it positively connects the loss of
head hair in humans with a specific disease. In this context,
"thinning hair" and "hair loss" are synonymous. Loss of body
and/or facial hair is irrelevant, as is hair loss caused by drug
therapy.
```

Figure 1. Sample TREC query.

however, only considers document retrieval, and in particular, ad hoc document retrieval. Ad hoc IR is the one-time (batch-based) retrieval of documents that are relevant to a query, which is issued once and can not be refined. This is opposed to interactive search (where a user can iteratively refine queries), or filtering, where the definition of relevance might change over time, e.g., after the first relevant documents have been seen. Ad hoc retrieval also assumes a fixed document collection as opposed to a dynamic one such as the web. It is thus the simplest, most clear-cut definition of the document search task and also the most well-studied form of IR. We concentrate on it here because it allows for the basic concepts of IR to be presented without interactions with more complicated factors introduced by variant tasks.

Most modern IR evaluation set-ups, including TREC, rely on so-called "laboratory-style" performance evaluation: queries are generated by humans and formulated in natural language; subsequently a judge decides which documents (given a fixed, finite document set) are relevant to the query, a task called *relevance decision*. These decisions are used as fixed, a priori gold standard, which each system's result is compared to. Laboratory-style performance evaluations are easier to control, because the factors around the main problem in IR evaluation, the subjectivity of relevance decisions, are kept constant and thus as controllable as possible. The entire test collection consists only of three components: the queries, the relevance decisions, and the document set. During the construction of the collection, the abstract operational set-up is "frozen": practical details such as the kind of presentation of query results, which is known to potentially influence the results considerably, is factored out. This is done by choosing one particular feature (in this case, one particular presentation format), which is then kept constant. The advantage of such a set-up is the repeatability of experiments under different conditions. For instance, TREC produced 50 queries and relevance judgements per year, at considerable cost. But once this frozen document collection is created, any research group can use TREC queries and relevance judgements without having to recreate the exact conditions of the human judges and their interaction with one particular IR system.

However, such a set-up can only test a subset of the parameters that could be changed in an IR system – namely the so-called system parameters. System parameters tested include the following: how the documents are indexed (i.e., assign keywords to them that best describe the contents of the document), which query language is used (a query language is a set of keywords and some rules about how these can be combined), and which retrieval algorithm is used (how are terms in the query matched to index terms – there are different ways of manipulating terms and different mathematical models to calculate the best fit). A system parameter not tested in the classic laboratory model is how to present the results to the user (e.g., as a long unordered list of documents, as a ranked list, or arranged in a graphical way). Sparck Jones and Galliers (1995) list many aspects of the overall information system that could and should be evaluated but lie outside the laboratory-style model, such as the influence of the static document collection containing the universe of documents that can be queried, and of the user who formulates a query in a given query language, thus translating his or her information need into this query language. Such remit aspects (i.e., aspects of motivation, goal, and manner of the evaluation, specifically the environment variables of the influence of the document collection and the user query) are not normally explored; a notable exception to this is the study by Saracevic et al. (1988) which considers environment variables in detail.

1.1.2 Question Answering. *Question Answering* (QA) is the task of returning a short piece of text as an answer to a question which is given in natural language. As search ground, a large collection of documents is used, from which the answer is selected. What counts as an answer is defined below; though it *may* in principle be an entire document, in most cases the answer is a much smaller unit, typically in the order of a sentence or a phrase. In TREC-QA, there are limitations on the type of question which is used: the question has to be factual (not involving any explicit opinion), and it has to be answerable within a short textual piece.

In comparison to the well-established evaluation of IR, the evaluation of QA is still in its infancy. Since 1999, NIST has prepared material for measuring system performance on the task of returning short answer passages (or, recently, exact answers) to a question formulated in natural language (Voorhees and Tice, 2000). This evaluation is run as a track in TREC (TREC-QA, 2004), attracting 20–36 participant groups in the following years. Figure 2 shows example questions from the first three TREC-QAs.

There is a human manual check on all returned system answers to see if they constitute a "correct" answer, under the assumption that there is not just one correct answer in the document collection. This manual check makes QA evaluation even more expensive than IR evaluation. One problem for the evaluation

TREC-8	How many calories are there in a Big Mac? Where is the Taj Mahal?
TREC-9	Who invented the paper clip? Where is Rider college located? What is the best-selling book?
TREC-10	What is an atom? How much does the human adult female brain weigh? When did Hawaii become a state?

Figure 2. Example TREC questions.

is that it is often unclear what should count as an answer. Answer generality or granularity can pose a problem, for instance, when a question asks for a point in time: while most assessors would judge "the nineties" as too general an answer to the question "When did Princess Diana die?", it is unclear what the right level of generality is in general (in this particular case, the day should probably be given; the time of day is too specific).

There are different question tasks in TREC-QA: factoid questions, list questions, definition questions, context questions, and reformulations of questions in different words.

The simplest and largest part of the questions are factoid questions, which ask for simple, factual information such as "how many calories are there in a Big Mac". Opinion questions, such as "what is the greatest film ever made?" are explicitly excluded.

List questions ask for several instances of one type. Examples from TREC-10 include: 4 US cities that have a "Shubert" theater; 9 novels written by John Updike; 6 names of navigational satellites; 20 countries that produce coffee. In order to qualify as a list question, the answers must not be found in one document; systems should be encouraged to assemble the answers from different documents. In later TRECs, the target number is no longer given; systems are required to find *all* instances of a certain type (see Figure 3).

Definition questions such as "Who is Colin Powell?" and "What is mould?" were used in every TREC-QA apart from TREC-11. They were always controversial, because it is extremely hard to assess what a good answer would be: answers could have more or less detail and be directed at different target users. However, TREC-12 brought definition questions back because they are very prevalent in real search engine logs. Due to space limitations, this chapter

1915: List the names of chewing gums.
Stimorol, Orbit, Winterfresh, Double Bubble, Dirol, Trident, Spearmint, Bazooka, Doublemint, Dentyne, Freedent, Hubba Bubba, Juicy Fruit, Big Red, Chiclets, Nicorette

Figure 3. Answer list for list question 1915 (names of chewing gums found within the AQUAINT corpus).

cannot go into the specifics of their evaluation, which is based on "information nuggets" – however, it is important to note that definition questions were the only type of question in TREC-QAs so far for which the evaluation was not stable (Voorhees, 2003).

The context task in TREC-10 was a pilot evaluation for QA within a particular context. The task was designed to simulate the kind of dialogue processing that a system would need to support an interactive user session. Unfortunately, the results in the pilot were dominated by whether or not a system could answer the particular type of question the context question set started with: the ability to correctly answer questions later in a series was uncorrelated with the ability to correctly answer questions earlier in the series. Thus the task was discontinued.

Apart from question type, there are other dimensions of difficulty. For instance, the source of the questions has a strong impact on the task. In TREC-8, questions were formulated *after* an interesting fact was randomly found in a newspaper text: this meant that the formulation (or re-engineering) of the question was influenced by the actual answer string, making it likely that the answers and the questions are textually similar. Later TRECs used more realistic models of question source, by mining them from web logs. Such "real" questions are harder on assessors and systems, but more representative for those systems which use training; it is easier to find similar questions on the web, but competing systems need to care more about synonymy, polysemy, and other phenomena of superficial (string) differences between question and answers.

Since TREC-10, the organisers no longer guarantee that there is an answer in the document collection; in fact, 10% of the questions on average have no answer (i.e., the assessor did not find the answer and inspection of all system results also did not find an answer). In this case, the correct system return was "NIL". The lack of an answer guarantee makes the task harder, as the systems need an internal measure of their confidence in an answer (only 5 systems in TREC-10 had a NIL precision >0.25; this remained similar in later years).

Document collections for the early TREC-QAs were the same as for the main TREC (979,000 articles). In TREC-11, the collection was changed to the AQUAINT-collection (1,033,000 documents), which covers a more recent time frame (1998–2000). This collection consists of documents from the Associated Press news wire, the New York Times news wire, and the (English portion of the) Xinhua News Agency. The move to the new corpus partially addressed the timeliness problem: TREC rules state that any external sources such as the Internet can be used as long as the answer is reported in connection with a document from the document collection supporting the answer. From TREC-11 onwards systems started using the Internet and projected the answers found in the web back into the (aged) TREC document collection. In one case at TREC-11, a system which returned the objectively correct answer

to a question asking for the current governor of Texas did not receive the credit as only the governor of Texas ruling in the late 1980s (the old TREC collection) could be counted as the correct answer.

The QA task is often thought of as a continuation of IR: even if the task is to pinpoint the right answer rather than finding generally relevant documents, some kind of IR must be performed first to find relevant documents. QA is also considered "harder" than IR at least in one aspect – it needs to perform deeper natural language analysis of the texts, and importantly, more complex query understanding. For instance, in the question "When did Princess Diana die?", an IR search engine would drop question words such as "when", whereas a QA system would use them to determine the correct answer type.

2 Evaluation Criteria

Evaluation criteria define the properties we are looking for in an ideal system output. The ideal IR system will return "relevant" documents, whereas the ideal QA system will return "correct and exact answers". The following sections give an overview of the difficulty in defining these properties operationally.

2.1 Ad Hoc IR: Relevance in Document

Ad hoc IR has been studied since the 1960s. The core problem the field had to solve was the fact that both information needs and perceived relevance are situation-dependent. Firstly, information needs are unique to a particular person at a particular time. The answer of modern evaluations such as TREC is sampling: collecting many different queries from many humans addresses one aspect of this problem, with the aim to sample enough of the entire range of "possible" queries to allow to draw conclusions about how the systems would perform on other, yet unknown queries.

Secondly, there is the relevance problem in IR: even within one information need, relevance of documents to a query, as perceived by a user, is situational and thus inherently subjective. It will differ over time for the same person, and it will differ between different humans even more. One factor influencing relevance is novelty: a document may be perfectly relevant to a query but a user who already knows it might not judge it as relevant because it is not immediately *useful* to him/her. Other factors have to do with the way in which a document is relevant: it may contain the answer itself, it may *point* to a different document that contains the answer, it may remind the user of a source or document that contains the answer, it may provide some relevant background to a question without really containing the information the user was looking for, etc. However, in an operational environment such as TREC, it cannot necessarily be guaranteed that one person will judge all necessary documents, and no particular order can be imposed in which documents are judged.

The challenge for setting up an objective and operationally viable evaluation is thus to find rules about relevance which make relevance decisions as context-independent as possible. In TREC, users were asked to judge whether each given document *in isolation* is relevant to the query, independently of the novelty of that document to the judges. Relevance should thus in principle be an objective property between each individual document and the query. The assumption that judges can mentally cancel previous context is probably not particularly realistic; however, the advantage of creating a (theoretically) situation-independent test collection outweighs these considerations.

2.2 QA: Relevance in Exact Answer String

While relevance in ad hoc IR is well discussed, defining the notion of a "correct answer" is a new challenge for QA evaluation. One of the problems is that there may be more than one right answer, and even a continuum in quality between answers. Interestingly, there is a correlation between the prescribed answer length and these problematic factors. Early TRECs allowed for 250 bytes or 50 bytes answers, but now exact answers are required. Figure 4 illustrates possible problems with non-exact answer decisions. It gives real example strings submitted to a TREC-9 question, in decreasing order of quality. Human annotators have to make a judgement as to which answers still count as "good enough".

The context of the answer is important. For that reason, systems return [docid, answer string] pairs (five per question in early TRECs, one per question since TREC-11). The document is to "support" the answer in that somewhere in the document the connection between the question and the answer must be present. For systems B, C, and D in Figure 4, the connection is already clear in the answer string. For the other systems, whether or not an answer is supported is judged by the human annotators after reading the submitted document. This makes the answer judgement an especially expensive task.

What river in the US is known as the Big Muddy?

System A:	the Mississippi
System B:	Known as Big Muddy, the Mississippi is the longest
System C:	as Big Muddy , the Mississippi is the longest
System D:	messed with . Known as Big Muddy , the Mississip
System E:	Mississippi is the longest river in the US
System F:	the Mississippi is the longest river in the US
System G:	the Mississippi is the longest river(Mississippi)
System H:	has brought the Mississippi to its lowest
System I:	ipes.In Life on the Mississippi,Mark Twain wrote t
System K:	Southeast;Mississippi;Mark Twain;officials began
System L:	Known; Mississippi; US,; Minnessota; Cult Mexico
System M:	Mud Island,; Mississippi; ``The; history; Memphis

Figure 4. Example answers.

The requirement that answer strings have to be extracts (i.e., they have to be found verbatim in a document), in place since TREC-12, simplifies answer judgement and discourages "answer stuffing" (a practice where several suspected answers are appended into the result string, cf. systems K–M in Figure 4).

Each answer is independently judged as correct, unsupported, or incorrect by two human assessors. When the two judgements differ, an adjudicator makes the final decision. An answer is judged correct if it contains a right answer to the question, if the document from which it is drawn makes it clear that it is a right answer, and if the answer is responsive. Responsive extracts are non-ambiguous (they must not contain multiple entities of the same semantic category as the correct answer), and, for numerical answers which contain units, the answer string must contain that unit. An answer is judged unsupported if it contains a right answer and is responsive, but the document from which it is drawn does not indicate that it is a right answer. Otherwise, an answer is judged as incorrect. Answers supported by a document are accepted even if the answer is "objectively" wrong – in the closed world of the TREC-QA exercise, answers are correct if they are correct according to at least one document in the fixed collection. As mentioned above, another problem concerns answer granularity. Assessor opinion also differs with respect to how much detail is required to answer a question.

Answer judgement is expensive – for instance, in TREC-10, the mean answer pool per question judged was 309 document/answer pairs. This expense could be kept lower with a fixed gold standard agreed before the competition and simple string matching. However, TREC-QA style evaluation (where each returned answer is manually judged) ensures a higher quality of the evaluation, because systems could potentially return answers that are not yet in the gold standard. The decision not to use a gold standard – as is used in IR evaluations – is an important one, which may be partially responsible for the overall higher numerical results in QA in comparison to IR.

An important design criterion of the TREC-QA data-set is that it should be reusable for the training of later systems. Therefore, each year's TREC-QA answers and questions are made available in the form of a set of possible answer patterns, which simulate the manual checking of submitted answers, e.g., Figure 5, which gives known answers to the question "Who was Jane Goddall?". The patterns are expressed as `perl` regular expressions. Evaluation against frozen patterns is a suboptimal solution: false negatives are possible (e.g., some document might exist describing Jane G. which is not covered by these strings, which would be unduly penalised), and false positives are equally possible ("anthropologist" might occur in a non-Jane-G situation, which would be unduly rewarded). The patterns also cannot penalise "answer stuffing", or check for supportedness of answers. Nevertheless, patterns can be useful if

```
naturalist                  chimpanzee\s+ researcher
anthropologist              wife.*van\s* Lawick
ethnologists?               chimpanzee\s* -?\s+ observer
primatologist               animal behaviou?rist
expert\s+ on\s+ chimps      scientist of unquestionable reputation
chimpanzee\s+ specialist    most\s recognizable\s+ living\s+ scientist
                            pioneered\s+ study\s+ of\s primates
```

Figure 5. Example answer patterns for question "Who was Jane Goddall?".

their limitations are understood, as system rankings produced by *lenient* annotator assessment and pattern-based results are highly correlated: Kendall's τ of 0.944 for 250 bytes and 0.894 for 50 bytes. (Lenient assessment, given up after TREC-9, treated unsupported answers as if they were fully correct.)

After discussing the theoretical evaluation problems in the two fields, we will now turn to the actual evaluation metrics used.

3 Evaluation Metrics

3.1 Evaluation Metrics in IR

3.1.1 Recall, precision, and accuracy. Given a test collection, the main metrics used in IR evaluations are precision and recall, and several summary metrics derived from these point-wise metrics.

Consider Figure 6, which defines the categories for precision, recall, and accuracy. What is relevant or non-relevant is decided by the human judge (our definition of "truth", also called a gold standard), whereas what is retrieved or not retrieved is decided by the system.

Recall is defined as the proportion of retrieved items amongst the relevant items ($\frac{A}{A+C}$); *precision* is defined as the proportion of relevant items amongst retrieved items ($\frac{A}{A+B}$); and *accuracy* is defined as the proportion of correctly classified items, either as relevant or as irrelevant ($\frac{A+D}{A+B+C+D}$). Recall, precision, and accuracy all range between 0 and 1.

Even though IR can in principle be seen as a classification task (documents are classified as either relevant or non-relevant), it turns out that accuracy, the evaluation metric of choice for classification tasks, is not a good measure for IR. This is because it conflates performance on relevant items (A) with performance on irrelevant items (D) – which are numerous but less interesting for the task. Due to artificially inflated numbers in any real situation, even systems with a very real quality difference are nearly indistinguishable on account of accuracy.

Figure 7 gives an example of how precision and recall can be used to judge two systems against each other.

Our entire document set in this case is 130 documents (A+B+C+D). For one given query, there are 28 relevant documents (A+C, shaded in light grey). Let

	Relevant	Non-relevant	Total
Retrieved	A	B	A+B
Not retrieved	C	D	C+D
Total	A+C	B+D	A+B+C+D

Figure 6. Categories for precision, recall, and accuracy.

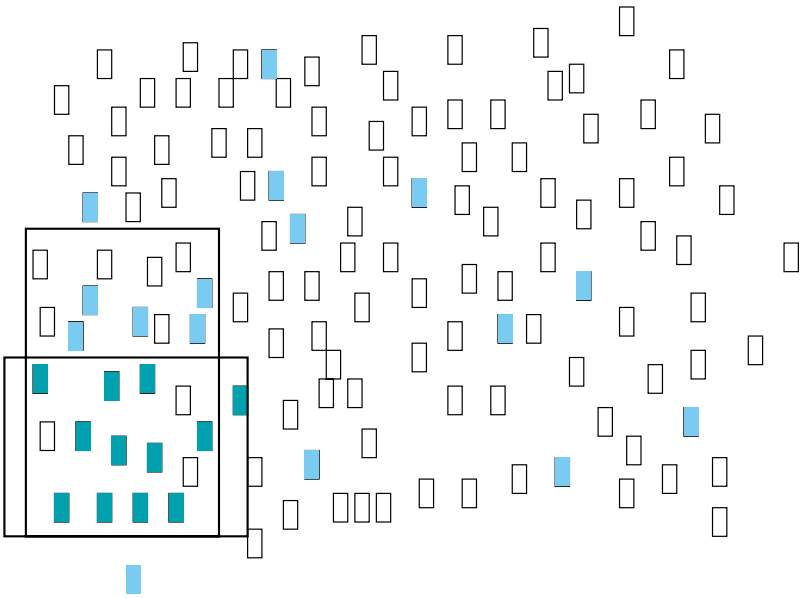

Figure 7. Example of the use of precision and recall.

us now assume that one fictional system, System 1, retrieves the 25 items given in the upper rectangle ($(A+B)_1$). Of these retrieved items, 16 are relevant (A_1). Precision, recall, and accuracy of System 1 can thus be calculated as follows (consider that with realistically large collections, accuracy would be close to 100% for all systems):

$$R_1 = \frac{A_1}{A+C} = \frac{16}{28} = 0.57 \tag{6.1}$$

$$P_1 = \frac{A_1}{(A+B)_1} = \frac{16}{25} = 0.64 \tag{6.2}$$

$$A_1 = \frac{A_1 + D_1}{A+B+C+D} = \frac{16+93}{130} = 0.84 \tag{6.3}$$

Another system, System 2, might retrieve the 15 items given in the lower rectangle $(A+B)_2$; out of the retrieved items of System 2, 12 happen to be relevant ($A_2 = 12$); we can thus calculate the performance of System 2 as follows:

$$R_2 = \frac{12}{28} = 0.43 \tag{6.4}$$

$$P_2 = \frac{12}{15} = 0.8 \tag{6.5}$$

$$A_2 = \frac{12 + 99}{130} = 0.85 \tag{6.6}$$

System 2 has a higher precision than System 1: it is more "careful" in retrieving items, and as a result, its return set contains a higher proportion of relevant items (which is measured by precision), but it has missed more of the relevant items in the document collection at large (which is measured by recall).

3.1.2 Relation between recall and precision; F-measure.

In general, there is an inverse relationship between precision and recall, as Figure 8 illustrates. Here, precision and recall of a fictional system are plotted versus the number of items that are retrieved: the more items the system returns, the higher the likelihood that it will retrieve relevant documents from the overall collection – if all documents are retrieved, recall is 1 by definition. This comes at the cost of also retrieving many irrelevant documents, so the more documents are retrieved, the more precision will decrease.

The inverse relationship between precision and recall forces systems to compromise between them. But there are tasks which particularly need good precision whereas others need good recall. An example of a precision-critical task

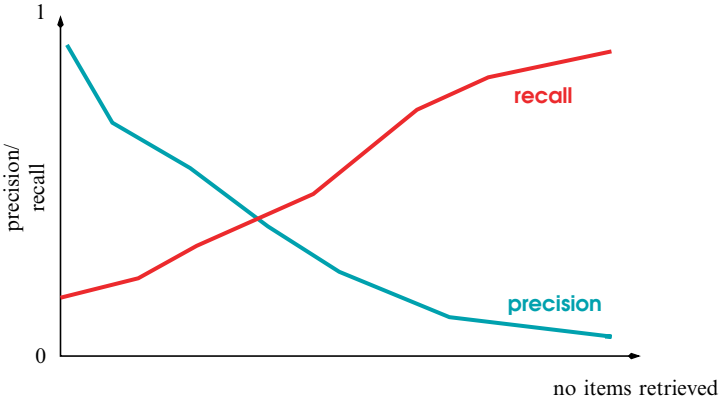

Figure 8. Inverse relationship between precision and recall.

are quick web searches, where little time is available, and where more than one relevant document exists which can answer the information need, due to the redundancy inherent in the web. That means that a full recall of *all* relevant documents is not required, and that the user would not want to consider non-relevant documents – full recall of all documents is not required, as at least one relevant document is expected to come up in a high rank anyway. An example of a recall-critical task is a patent search, where the worst-case scenario (with costly consequences) would be to miss even one single relevant document; time is less of an issue in this scenario.

Figure 9 shows the relationship between precision and recall in a more standard form, namely as precision plotted against recall (the so-called precision–recall curve). Data points are gained by manipulating the number of items retrieved, as in Figure 8 (but in contrast to Figure 8, this number cannot be directly read off here). Ideal systems, which combine high precision with high recall, will show curves that stretch as far as possible into the upper right corner. The precision–recall graph is related to the so-called receiver operating characteristic (ROC) graph known from the life sciences, which plots the hit rate (A in Figure 6) versus the false alarm rate (B) – in this graph, ideal curves stretch in the upper left corner.

Because of the inverse relationship between precision and recall, it is not obvious how the overall performance of a given system can be estimated. One could consider many possible precision/recall data points for any query, as arbitrarily many different cut-offs could be used with relevance-weighted IR engines – this is in opposition to Boolean systems, which always return

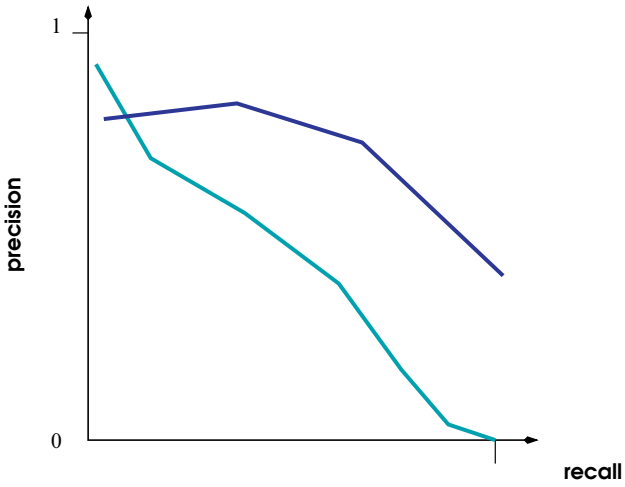

Figure 9. Two precision–recall curves.

a fixed set of documents. One possible answer is to consider the area under the precision–recall curve as an estimate of system performance. However, in a practical setting, one does not want to manipulate a system's retrieval set, plot a precision-recall curve, and then estimate the area under the curve. Two simpler estimations are to empirically determine the crossing-over of precision and recall, or to calculate the F-measure.

F-measure (van Rijsbergen, 1979) is defined as the weighted harmonic mean of precision and recall:

$$F_\alpha = \frac{PR}{(1-\alpha)P + \alpha R} \tag{6.7}$$

α is a parameter that allows to vary the relative importance of recall versus precision (a high *alpha* means that precision is more important and vice versa). The F-measure is most commonly used with $\alpha = 0.5$:

$$F_{0.5} = \frac{2PR}{P+R} \tag{6.8}$$

The maximum value of $F_{0.5}$-measure (or F-measure for short) for a system is a good indication of the best compromise between precision and recall.

TREC also uses precision at a certain document rank (cut-off; e.g., $\mathrm{P(r} = 200)$ is the precision at rank 200, i.e., after the top 200 documents are considered), and precision at a certain level of recall (example: $\mathrm{P(R} = 0.2)$ is the precision at that point when a recall of 0.2 has been reached).

All these measures, however, assume that we can always determine recall exactly, but this is not so.

3.1.3 The recall problem. The IR recall problem concerns the impossibility of collecting exhaustive relevance judgements in a realistically large document set. In order to be absolutely sure that no potentially relevant documents have been missed when making relevance judgements, judges would have to go through the entire document set of nearly a million documents, which is infeasible (for each individual query, it would take an estimated 6,500 hours to judge all documents in the TREC collection – and this would be at an unrealistically high speed of only 30 seconds per document). Therefore, methods are desirable which can determine a smaller set of documents to be judged manually in such a way that it is unlikely that other relevant documents exist outside this set. *Pooling* (van Rijsbergen and Sparck Jones, 1976) is one such method: the document pool to be manually judged is constructed by putting together the top N retrieval results from a set of n systems (in TREC N = 100). Humans judge all the documents in the pool, and documents outside the pool are automatically considered to be irrelevant. Pooling works best if the systems used are maximally different, and if many systems are available, as is

the case in TREC. It was also found that there is a large increase in pool quality if humans additionally do manual recall-oriented searches, using an IR system and the best queries they can think of, e.g., involving synonyms. Fortunately, there is considerable overlap in returned documents: the pool is smaller than the theoretical maximum of $N \cdot n$ systems (around one-third the maximum size).

3.1.4 11-point average precision. Another dimension of difficulty becomes apparent when we consider IR measures which can average over more than one query. The problem is that queries can have different numbers of relevant documents and that it is not possible to set a fixed threshold such that systems can achieve the theoretically possible maximal values under all conditions. For instance, for any query with more than 10 relevant documents, full recall could by definition never be achieved if the cut-off were set at 10 documents; for any query with less than 10 relevant documents, full precision could never be reached. Thus, more complicated joint measures are required.

11-point average precision is one of these, defined as:

$$P_{11} = \frac{1}{11} \sum_{j=0}^{10} \frac{1}{N} \sum_{i=1}^{N} \tilde{P}_i(r_j) \tag{6.9}$$

with $\tilde{P}_i(r_j)$ being the precision (interpolated or measured) at the jth recall point for the ith query (out of N queries). $r_0, r_1, ...r_{10}$ are the 11 standard recall points ($r_j = \frac{j}{10}$). The precision we can *measure* is $P_i(R = r)$: the precision at the point where recall has first reached r. The reason why the $\tilde{P}_i(r_j)$ often has to be interpolated is that the measured recall points r do not in general fall onto a standard recall point (only when $r = \frac{j}{10}$)). There are many interpolation methods such as the following one:

$$\tilde{P}_i(r_j) = \begin{cases} max(r_j \leq r < r_{j+1}) P_i(R = r) & \text{if such } r \text{ exists} \\ \tilde{P}_i(r_{j+1}) & \text{otherwise} \end{cases} \tag{6.10}$$

Note that with interpolation it does not matter that the two queries have different numbers of relevant queries (and that the last relevant document occurs at different ranks): we still get exactly 11 precision – recall points per query, as required. The final calculation is the average of all $\tilde{P}_i(r_j)$ (1.00, 1.00, 1.00, 0.84, 0.67, 0.59, 0.59, 0.30, 0.23, 0.23) over 11, resulting in a $\tilde{P}_{11}$ of 0.61.

In Figure 10, $\tilde{P}_i(r_j)$ values have been interpolated, and $P_i(R = r)$ values have been exactly measured. Figure 11 gives the precision – recall curve for

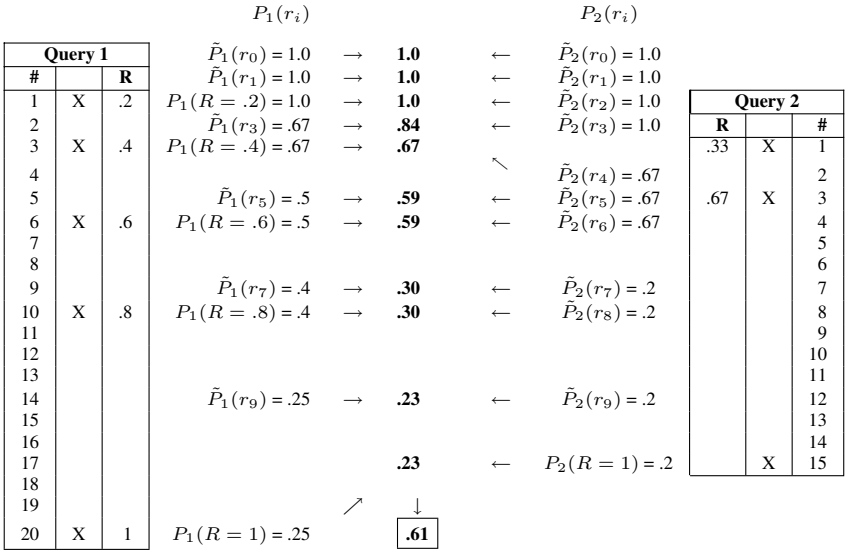

Figure 10. Example calculation: 11-point average precision, for two queries.

this example (bold circles for measured data points; thin circles for interpolated data points; dark for Query 1; light for Query 2).

3.1.5 Mean average precision (MAP). There is a second, simpler composite measurement which generalises over different queries, called *Mean Average Precision* (MAP), which is sometimes also referred to as *mean precision at seen relevant documents*. Precision is calculated at each point when a new relevant document is retrieved (using $\mathrm{P} = 0$ for each relevant document that was not retrieved). The average is then determined for each query. Finally, an average over all queries is calculated.

$$MAP = \frac{1}{N}\sum_{j=1}^{N}\frac{1}{Q_j}\sum_{i=1}^{Q_j}P(rel = i) \tag{6.11}$$

with Q_j being the number of relevant documents for query j; N the number of queries, and $P(rel = i)$ the precision at ith relevant document.

Again, an example MAP calculation for the two queries in Figure 10 would return 5 data points for Query 1: 1.00 (at rank 1), 0.67 (at rank 3), 0.5 (at rank 6), 0.4 (at rank 10), and 0.25 (at rank 20), averaging to 0.564 for Query 1; and 3 data points for Query 2: 1.0 at rank 1, 0.67 at rank 3, and 0.2 at rank 15, averaging to 0.623 for Query 2. This results in an overall MAP of $\frac{0.564+0.623}{2} = 0.594$.

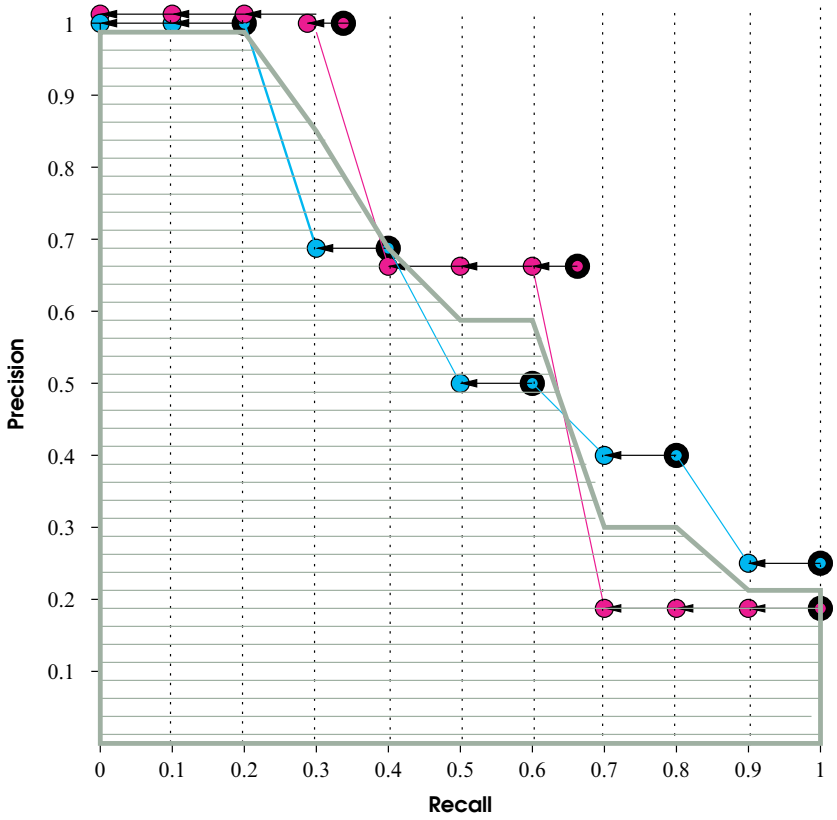

Figure 11. Precision – recall curve for the example calculation.

Mean precision at seen relevant documents favours systems which return relevant documents *fast*; it is therefore precision-biased. TREC publishes many composite precision/recall-based performance measures per run because in order to gain insight into what the system is doing overall, it is necessary to look at more than one metric. TREC has been criticised for putting too much emphasis on recall, given that most of today's IR requirements are precision-based and given that recall-oriented test collection preparation is very time- intensive (see above). This is one of the reasons why many researchers prefer mean precision over 11-point average precision as an overall summary IR measure.

3.2 Evaluation Metrics in QA

There has been experimentation in TREC with three different evaluation metrics: mean reciprocal rank (MRR), weighted confidence, and average accuracy. Average accuracy is the simplest evaluation metric and has been the

official evaluation metric since 2003 for the main task. For list questions, precision and recall had to be adapted to deal with the difficulty of different numbers of return items for different list questions. Additionally, NIL accuracy is calculated.

In the case of MRR, each system returned five answers, in rank of confidence of their correctness, for each question. It was decided to look at the top five answers only, as the task is precision-oriented and lower ranks are assumed not to be of interest. The MRR is defined as the mean of the inverse rank of the first correct answer, taken over all n questions:

$$MRR = \frac{1}{n}\sum_{i=1}^{n} RR_i \tag{6.12}$$

The score for an individual question i is the reciprocal rank r_i where the first correct answer appeared (0 if no correct answer in top five returns). Thus, there are only six possible reciprocal ranks per question: 0, 0.2, 0.25, 0.33, 0.5, and 1.

$$RR_i = \frac{1}{r_i} \tag{6.13}$$

MRR is bounded between 0 and 1 and averages well, and while systems are penalised for not retrieving an answer, they are not penalised unduly so. However, there is no credit for systems which know that they do not know, and there is no credit for multiple (potentially different) correct answers.

The list task uses precision and recall as evaluation measures. The instance precision (IP) and instance recall (IR) for a list question can be computed from the final answer list and the assessor's judgement. Let S be the size of the final answer list (i.e., the number of known answers), D the number of correct, distinct responses returned by the system, and N the total number of responses returned by the system. Then $IP = \frac{D}{N}$ and $IR = \frac{D}{S}$. Precision and recall were then combined using the F-measure with equal weight given to recall and precision ($F = \frac{2*IP*IR}{IP+IR}$), and the average F-score over all list questions is reported.

In TREC-11, the evaluation metric was changed to *confidence-weighted score*, designed to reward systems for their confidence in their answers, and only one answer per question was returned. Within the submission file, systems had to rank their answers to the 500 questions according to their confidence in that answer, with the answer they were most confident with ranked highest in the file. Confidence-weighted score is defined as $\frac{1}{Q}\sum_{1}^{Q}\frac{\text{\# correct in first } i}{i}$ (Q being the number of questions). With this measurement, it is possible for two systems with the same answer strings for every question to score considerably differently, if the answers are ranked (confidence scored) differently.

In TREC-12, evaluation was changed once more (Voorhees, 2003). The new main evaluation score for a passages task run is surprisingly simple:

accuracy, the fraction of questions judged correct (with one answer per question). In comparison to MRR, this measure assumes that the user is only really interested in the first answer, which must be correct for any score to be given to that question.

Also reported are the recall and precision of recognising when no answer exists in the document collection (called NIL recall and NIL precision). Precision of recognising no answer is the ratio of the number of times NIL was returned and correct to the number of times it was returned; recall is the ratio of the number of times NIL was returned and correct to the number of times it was correct.

3.3 Scalability and Stability of the Measures

In the early days of IR, there was little concern about scalability of IR performance, and thus small document collections and small numbers of queries were generally accepted (cf. the Cranfield 2 experiments which operated on only 1400 documents, albeit with 279 queries; Cleverdon:67). As discussed above, the current consensus is that a large number of queries (as well as a large document collection) is necessary in order to capture user variation, to support claims of statistical significance in results, and to demonstrate (for commercial credibility) that performance levels and differences hold as document collection sizes grow. There are practical difficulties in obtaining large document collections, which is why the current TREC collection is not a balanced corpus: it contains those newspapers which were easy to incorporate for unrelated reasons, e.g., because they operated without copyright restrictions. This is acceptable for the experiment, but ideally, one would wish for a balanced corpus (reflecting all types of texts typically encountered by humans in their daily life). Additionally, the cost of collecting large amounts of queries and relevance judgements is very high. But over the years, the TREC collection has proved to be an extremely valuable and frequently used collection for the IR community, supporting many experiments since it became available.

There have also been experiments to test the effect of IR ad hoc query size (long, medium, short queries). The best results are achieved for long queries, for which the systems were clearly optimised; all performed worse for shorter, more realistic queries. This showed that, not surprisingly, automatic query expansion in systems deployed today is not perfect yet (the additional information in the long queries lists additional relevance conditions). In QA, this is correlated to the difference in performance between factoid questions (which systems are generally better at) and definition questions (which are harder to do, and which are also harder to evaluate).

Scalability also concerns the number of participants. The fact that TREC has many participants is important for two practical reasons: firstly, the quality of the document pool in pooling is dependent on the number of systems,

particularly those using different technologies. Secondly, one needs a reasonable number of systems (i.e., data points) to prove stability of the evaluation (inter-annotator agreement), because correlation measures between results by different judges (such as Kendall's tau) require a reasonable number of data points to be statistically significant.

Voorhees (2002) found that the comparative effectiveness of different retrieval methods is stable in the face of changes to the relevance judgements. This means that even though TREC annotators show relatively low agreement in their relevance judgements, and even though, as a result, the numerical performance measures of a system (e.g., MAP) differ considerably when relevance judgements by a different annotator are used, the *ranks* given to systems according to different annotators did not substantially differ (Kendall's tau was over 0.9). Voorhees' results show that no interaction occurs (no annotator's relevance decision favours any system over any other). This property is called *stability* of judgements. This is a very positive result for TREC evaluations, where the relative ranking of systems matters more than any absolute system results, and where there are many systems such that the rank vector is large enough to be meaningful. However, it is worth keeping in mind that the subjectivity of relevance judgements remains a problem – if we wanted to make statements about the absolute results of competing systems (and absolute differences between them), current TREC judgements are not stable enough to support these.

Results about stability in QA are similar: Low inter-assessor agreement on question correctness meant that absolute MRRs were not stable, but *relative* MRRs (i.e., their ranks) were (Voorhees and Tice, 2000). Voorhees (2002) showed that relative confidence-weighted score is stable at Kendall's tau of above 0.9, but definition questions are not stable. Concerning assessor disagreement, the TREC-QA organisers decided that it was not only impossible to force agreement among TREC assessors, but also undesirable because it would not address the problem of measuring success rate of deployed systems.

4 Real-World Performance

4.1 TREC Ad Hoc IR

Sparck Jones (1995, 2000) provides insightful detailed summaries of the results of TRECs over the years. The most important lessons from these observations can be summarised as follows: of those systems which performed fully automatic searches in TREC-7 and TREC-8 (the last two "ad hoc" TREC conferences in 1997 and 1998), the best results were in the range of 0.40–0.45 (in terms of precision at document cut-off 30). These systems mostly achieve these results only when using long queries and narratives, but one team in TREC-7 managed results in the 0.40–0.45 range even for the short queries.

All systems performed worse for shorter queries. Generally, the best systems are statistically not significantly different, which points to the fact that an apparent plateau has been reached (at least if measured with the current, stringent and gold standard – based evaluation). Manual searches resulted in best results in the range of 0.55–0.60. Comparison of the TREC results over the years shows that the 0.40–0.45 result achieved in TREC-7 and TREC-8 was the highest fully automatic IR result ever measured for short and medium-length queries (apart from one occurrence at TREC-4). TREC-3 was exceptional in that its highest automatic results were in the 0.55–0.60 range – however, this was achieved only for long queries. At cut-off 10, several systems achieved almost 50% precision in automatic searching even with very short queries, and several exceeded 50% with the medium-length queries. (Manual searching under these conditions can lead to 70%, but with additional time and effort required.) Performance in the "middle" TRECs (4, 5, and 6) declined under much less favourable data conditions (less relevant documents available, less information on topics given). The better performance in TREC-7 and TREC-8 must be attributed to superior systems, as the manual performance has remained on a plateau.

4.2 TREC QA

Results of the top-scoring systems in the early TRECs are: 0.66 (27% questions unanswered) for TREC-8, 0.58 for TREC-9, 0.68 for TREC-10 (for 50 bytes); for the 250 bytes the results were 0.65% in TREC-8 and 0.75% in TREC-9. Interestingly, in 55% of cases where the answer was found in the first five answers, this answer was in rank 1 (TREC-9, average over all systems). This was part of the reason of the TREC-QA organisers for moving to evaluation metrics, which consider only one answer per question.

The highest scores for the TREC-11 factoid task (confidence-weighted score) were 0.86, 0.69, and 0.61, for the list task the results were 0.65, 0.15, and 0.11. For the TREC-12 list task the numbers were 0.396, 0.319, and 0.134 (all in average F-scores). The list task can therefore be seen as a task which is much harder than the factoid task.

Highest scorers for the TREC-12 passage task achieved 0.685 accuracy (with the second best system at 0.419), whereas the highest entry in the exact (factoid) main task was as high as 0.700 (with the second best system at 0.622).

5 Conclusion

Numerically, the results for TREC-QA are consistently higher than those for adhoc IR. While the evaluation measures are not directly comparable, we can still observe that the best systems in QA are close to the maximal possible

value, but the best systems in IR, a much maturer field, are relatively poor compared to the maximal value. This could be a side effect of the evaluation design, as well as an indication that QA is somehow a "simpler" (or better-defined) task than IR, due to IR's problems with relevance. In TREC-QA system output is not compared against a fixed gold standard; instead judges assess the actual system output. In this setting, humans are well-known to be more lenient judging something that is suggested as correct than choosing answers/documents for a fixed gold standard. Thus, it is possible that gold standard–based evaluations are inherently harsher.

In summary, ad hoc IR evaluation as done by TREC only covers one small part of the spectrum of IR evaluations, using laboratory conditions, precision and recall measured from large, fixed test collections. Real users in natural interaction with their IR system are not involved. A host of sophisticated performance metrics is available, e.g., 11-point average precision and mean average precision. There is a theoretical problem with the subjectivity of relevance, but it turns out that this problem is partially solvable by extensive sampling (cf. Section 2.1). There is also a recall problem, but it too is solvable, by pooling (cf. Section 3.1.3). The evaluation methodology used is provably stable towards human judgement differences. Thus, we can consider IR evaluation as mature. However, one of the big problems is the high cost of these evaluations, in terms of preparing collections and relevance judgements. There is current research into how this cost could be reduced. For instance, one could interpret the observable actions of Internet users as implicit relevance decisions: how long is a document looked at after search; is it saved; which action follows? These actions, however, are psychologically and practically very complex and not very well researched at this point.

Over the years and across systems, TREC performance in ad hoc retrieval has reached a plateau, possibly quite close to the best performance currently reachable with word-based statistical methods. Thus, interest in ad hoc IR has waned, and recently, new and related IR tasks have been tested in TREC "tracks" (such as filtering and web-based retrieval) – these new tasks are harder, but more realistic, and some of them require new evaluation measures. Future developments could concern evaluation metrics that mirror user satisfaction in a dynamic IR set-up – where the definition of the task, the information need, and even the collection are changing during a run, where extremely large-scale document collections are used, and where success and failure of personalised searches are evaluated.

QA, on the other hand, is a task that has only appeared in the last 5 years but that has been the object of a major evaluation effort in the framework of TREC. The main evaluation metrics changed from mean reciprocal rank, via a weighted confidence measure, to simple accuracy of answer return on a single answer, which is currently considered as fully appropriate to assess system

performance for this task. Answers are individually judged, rather than compared to a gold standard. There has been an enormous human effort in question creation and answer judgement.

The questions used in current TREC-QA are factual and simple, and there has been some debate about whether the field is ready to move to harder tasks. Undoubtedly, Deep Natural Language Processing (NLP) (e.g., comparison on the basis of logical form) helps for QA: those systems which use deeper processing (Harabagiu et al., 2003) consistently performed very well in all TREC-QAs. However, there have been competitor systems which use very little deep processing, and which rely on heuristics and redundant data on the web instead. While they could not rival the NLP-based systems, they nevertheless performed in mid-field, which was seen as a surprising success, given the very short development time when compared to the years of effort going into "deeper" systems. However, this also brought up questions about the goal of QA evaluation in general. Is the aim of QA evaluation to measure performance of the mundane task of answering simple, factual questions? Then today's QA systems have (almost) reached that goal; once redundancy/data-based systems can perform this task as well as deep NLP systems, a plateau will be reached. If, however, the aim of the QA task is to act as a diagnostic tool for how far the field has advanced in the overall goal of "intelligent" text understanding, the task may have to be made much harder, for instance, by using harder question types and requiring more detailed reasoning behind the answers.

However, overall the evaluation of QA can be considered a task which has managed to find its defining coordinates in a short time. The evaluation has constantly been adjusted, following system developments and factors which could not be known beforehand. In only 5 years of evolution, a satisfactory solution has been found, leading to a generally accepted evaluation methodology. The history of QA evaluation also shows how research in a certain direction can be fostered by directly manipulating the evaluation rules and metrics to encourage desirable properties of systems.

References

Callan, J. P. (1994). Passage-Level Evidence in Document Retrieval. In *Proceedings of the Annual International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 302–310, Dublin, Ireland.

CLEF (2005). Cross Language Evaluation Framework, http://www.clef-campaign.org/.

Cleverdon, C. W. (1967). The Cranfield Tests on Index Language Devices. *Aslib Proceedings*, 19(6):173–194.

Harabagiu, S., Moldovan, D., Clark, C., Bowden, M., Williams, J., and Bensley, J. (2003). Answer Mining by Combining Extraction Techniques with Abductive Reasoning. In Voorhees, E. M. and Buckland, L. P., editors,

Proceedings of the Twelfth Text REtrieval Conference (TREC-12), pages 375–382, Department of Commerce, National Institute of Standards and Technology.

Harman, D. (1993). The First Text REtrieval Conference (TREC-1). *Information Processing and Management*, 29(4):411–414.

NTCIR (2005). NII Test Collection for IR Systems, http://research.nii.ac.jp/ntcir/.

Saracevic, T., Kantor, P. B., Chamis, M. A. Y., and Trivison, D. (1988). A Study of Information Seeking and Retrieving. Parts I–III. *Journal of the American Society for Information Science*, 39(3):161–216.

Sparck Jones, K. (1995). Reflections on TREC. *Information Processing and Management*, 31(3):291–314.

Sparck Jones, K. (2000). Further Reflections on TREC. *Information Processing and Management*, 36(1):37–85.

Sparck Jones, K. and Galliers, J. (1995). *Evaluating Natural Language Processing Systems*. Springer Verlag, Berlin, Heidelberg, Germany.

TREC (2004). The Text REtrieval Conference, official website, http:// trec.nist.gov/.

TREC-QA (2004). The Text REtrieval Conference, Question Answering Track, http://trec.nist.gov/data/qa.html.

van Rijsbergen, C. J. (1979). *Information Retrieval*. Butterworth, London, UK, 2nd edition.

van Rijsbergen, C. J. and Sparck Jones, K. (1976). Information Retrieval Test Collections. *Journal of Documentation*, 32:59–75.

Voorhees, E. (2000). Variations in Relevance Judgements and the Measurement of Retrieval Effectiveness. *Information Processing and Management*, 36:697–716.

Voorhees, E. (2002). Overview of the TREC 2002 Question Answering Track. In Voorhees, E. M. and Buckland, L. P., editors, *Proceedings of the Eleventh Text REtrieval Conference (TREC)*, Department of Commerce, National Institute of Standards and Technology.

Voorhees, E. (2003). Overview of the TREC 2003 Question Answering Track. In Voorhees, E. M. and Buckland, L. P., editors, *Proceedings of the Twelfth Text REtrieval Conference (TREC)*, pages 54–68, Department of Commerce, National Institute of Standards and Technology.

Voorhees, E. and Tice, D. (2000). Building a Question Answering Test Collection. In *Proceedings of the Annual International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 200–207, Athens, Greece.