Chapter 1

EVALUATION METHODS IN INFORMATION RETRIEVAL AND QUESTION ANSWERING

Simone Teufel
Computer Laboratory
University of Cambridge, JJ Thomson Avenue, Cambridge CB0 3FD, UK
Simone.Teufel@cl.cam.ac.uk

Abstract This chapter compares the current evaluation strategies and problems in the fields of information retrieval and question answering. These two fields are related because they are at the borderline of traditional IR/information access tasks with NLP (Natural Language Processing). Whereas information retrieval has a long tradition as a task (since the 1950s), question answering is a relatively new task which had to develop its own evaluation metrics fast, based on experiences gained in information retrieval. This chapter will contrast the simple evaluation metrics used in question answering with those of information retrieval, some of which are rather more complicated. 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

Information Retrieval (IR) is the task of finding relevant documents from a large document collection which are relevant to a user's query. The query is typically expressed as a set of keywords. An example query might be "Find me documents talking about chemicals which cause hair loss in humans". The point has been made that the name which is generally accepted for this task is misleading and should really be *document retrieval*, as the systems do nothing more but find relevant documents rather than information (whereas "information" is a vague concept, the

general agreement nevertheless seems to be that it refers to amounts of data below the document threshold, such as paragraphs or even "propositions" – retrieving "information" is thus certainly not something that IR systems currently are capable of). We agree with this point, but for historic reasons we will still use the label "information retrieval".

Question Answering is the task of returning a short piece of text, typically in the order of a sentence or a phrase, as an answer to a question which is given in natural language. Under the currently prevailing definition, there are certain limitations on the type of question which can be used, e.g. the question has to be factual (not involving any explicit opinion), and it has to be answerable within a short textual piece. This task is often thought of as "harder" than IR at least in one aspect – if the task is to pinpoint the right answer rather than finding generally relevant documents, it is necessary to use deeper natural language analysis of the texts.

Finding an objective and meaningful evaluation for these tasks has been seen as a central mechanism for measuring and fostering progress. Competitive evaluation exercises have been held for both task. The continuity that these exercises bring was an important aspect in encouraging research teams to make the investment of participating, and thus in being able to monitor progress in the two fields.

It is indisputable that information retrieval evaluation is by now a mature technology. Ad-hoc information retrieval, that is, the task of finding relevant documents, given a user query and a document set, has been studied since the 1960s. The core problem the field had to solve was the fact that information needs and relevancy are situation dependent. The answer was large-scale sampling across users and across information needs. The NIST-run, competitive TREC evaluations ((Harman, 1993; TREC, 2004), Text Retrieval Conference), which were run since the early 1990s, have provided a large enough, well-balanced test ground. It has been shown that the evaluation strategies are stable, even in the light of annotator disagreement (Voorhees, 2000). 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 information retrieval 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. In the current chapter, we will nevertheless concentrate on ad-hoc IR because it represents the central, simplest IR task in most people's minds, and it illustrates the issues in IR evaluation most clearly.

In comparison to the well-established evaluation of information retrieval, the evaluation of question answering (QA) is still in its infancy. Since 1999, NIST have prepared material for measuring system performance on the task of returning short answer passages (or more 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). One problem for the evaluation is that it is often unclear what should count as an answer: there are problems with answer generality, time-dependency and context-sensitivity. Only relatively few TREC-QA runs have taken place, and the initial rounds were seen as exploratory. However, systems and evaluation strategies have evolved fast in those few years (e.g., three different evaluation measure have been experimented with since 1999), so that by now stable, satisfactory – and surprisingly simple – evaluation metrics have been found.

2. Evaluation of Information Retrieval

The goal of any kind of evaluation is to predict future experience from past experience; in this case, to measure an IR system's performance at the task of finding relevant documents in order to improve that performance in the future. The things that could be changed in an IR system include: how to index the documents (i.e., assign keywords to them that best describe the contents of the document), which query language to use (a query language is a set of keywords and some rules about how these can be combined; different query languages allow for more or less powerful constructs), which retrieval algorithm to use (given the query, how are query terms matched to index terms – in order to do so, there are different ways how the terms could be manipulated, and which mathematical model is used to calculate the best fit), and how to represent the results to the user (e.g. as a long unordered list of documents, as a ranked list or arranged in a different graphical order). The entire abstract information management system, however, is even more complex, because it consists of components outside the IR system proper: there is a large static document collection containing the universe of documents that can be queried, and there is a user who formulates a query in a given query language, thus translating his or her information need into this query language. Based on the system output, the user can then judge the system's answer, deciding which of the returned documents were relevant.

IR evaluation design concerns the hard task to tease the contribution of all these components apart.

2.1 Design of the evaluation

Spärck Jones and Galliers (1996) list many aspects of the overall information system that could and should be evaluated, but observe that today's IR evaluation, useful as it is, have restricted themselves to the "laboratory model", choosing to evaluate design aspects (i.e. performance factors) over 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). A notable exception is the study by Saracevic et al. (1988), which considers environment variables in detail.

"Laboratory-style" performance evaluations are easier to control, because they factor out the main problem in IR evaluation: the human "customer" and the unavoidable variation they bring with them. Obviously, the human is a system too complex to model, and we know that what is perceived as relevant is situational and thus inherently subjective. Information needs are unique to a particular person at a particular time, therefore, relevance judgements are known to differ across judges and even for the same judge at different times. However, there are two ways to deal with the problem of the subjectivity of relevance: a) one can either keep the human constant across a set of experiments, or b) one can sample enough humans/queries such that any idiosyncrasies about one particular user are statistically evened out. Option b) is more practicable for obvious reasons, and indeed is the one that the TREC conference has chosen.

TREC (Harman, 1993), the Text REtrieval conference, which is run by the US National Institute of Standards and Technology (NIST), emerged in the early 1990s and marked a new phase in retrieval evaluation. By defining a common task and dataset, TREC made it attractive for many groups from university and commercial backgrounds (87 in 2002) to participate. It also provides a continuity in the task: 2004 was the 14th annual conference; over the years systems' relative performance can be effectively observed.

TREC uses a large test collection: many queries from many humans (a total of 550 over the years). Figure 1.1 shows a sample TREC query.

These queries are considered as "frozen", abstract operational setups (such that different presentation of query results is factored out, for instance). Precision and recall-based evaluation metrics are used as the main measure of success. The advantage of such a setup is its repeatability under different conditions. Any research group can use TREC

```
<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 1.1. Sample TREC Query.

queries and relevance judgements without having to recreate the exact conditions of the human judges and their interaction with one particular IR system.

It is undisputed that there are many ways how a document can be relevant to a query, but the relevance definition that TREC uses has to be as context-independent as possible. Judges (retired information analysts) were asked to judge whether a given document *in isolation* is relevant to the query, independently of the novelty of that document to them. These instructions redefine relevance as a more objective property of each document–query pair, such that each decision can be taken in isolation, and in particular independently of the order in which documents are presented to the user. This assumption is probably neither particularly realistic nor is it easy to judge documents in this way, however the advantage of the use of the test collections as situation-independent instruments outweighed these considerations.

A TREC test collection consists of

- a document set, which needs to be large in order to reflect diversity of subject matter, literary style and noise such as spelling errors; today's TREC test collections include 979,000 newspaper articles and government documents (federal register)
- queries (also called topics), which are short descriptions of information need, but which also come in a longer form, detailing the relevance criteria (cf. Fig. reffig:query, and
- relevance judgements, which are binary and are made by the same user who originally created the query.

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,

	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 1.2. Categories for Precision, Recall and Accuracy.

1967)). 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 slightly non-balanced (it contains newspapers which operated with no copyright restrictions), whereas ideally one would wish for a balanced corpus (reflecting all types of texts typically encountered by a human in their daily life). Additionally, the cost of collecting large amounts of queries and relevance judgements (50 queries per year in TREC) is very high. The effort was worth it: the TREC collection is an extremely valuable collection for the IR community, and many experiments have used them since they have become available.

We have not yet described precisely *which* documents the judges perform their relevance judgements on. We will postpone this discussion until the concepts of precision and recall are encountered in the next section; the problem is of course how to avoid a situation in which the judges would have to consider each document in the entire document collection (which consists of close to one million documents).

We will now look at the metrics to be derived for each system on the basis of the test collection (document collection, queries, relevance judgements).

2.2 Evaluation metrics

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 1.2, 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

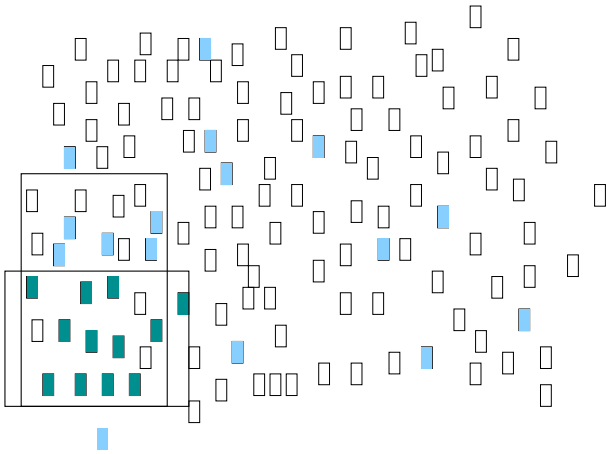

Figure 1.3. Example of the Use of Precision and Recall.

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 be seen as a classification task (documents are classified as either relevant or non-relevant), and accuracy is the main evaluation metric for classification tasks, it turns out that accuracy 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) – systems are then nearly indistinguishable on account of accuracy due to artificially inflated numbers.

Figure 1.3 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 (shaded in light grey). Let us now assume that one fictional system, System 1, retrieves the 25 items given in the upright 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:

$R_1 = \frac{A_1}{A+C} = \frac{16}{28} = .57$

$P_1 = \frac{A_1}{(A+B)_1} = \frac{16}{25} = .64$

$A_1 = \frac{A_1+D_1}{A+B+C+D} = \frac{16+93}{130} = .84$

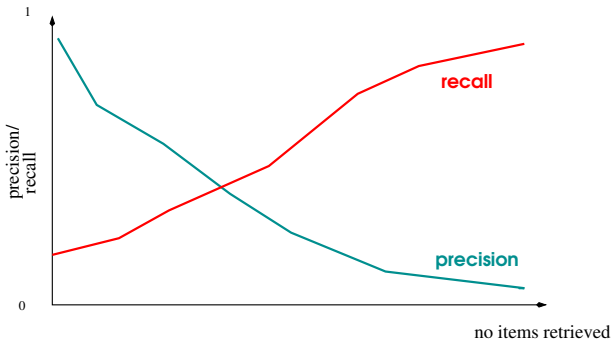

Figure 1.4. Inverse Relationship between Precision and Recall.

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 also relevant ($A_2 = 12$); we can thus calculate the performance of System 2 as follows:
$R_2 = \frac{12}{28} = .43$
$P_2 = \frac{12}{15} = .8$
$A_2 = \frac{12+99}{130} = .85$

System 2 has a higher precision with respect to System 1: it is more "careful" in retrieving items, and as a result, it has managed to retrieve a higher proportion of relevant items amongst its retrieved items (which is measured by precision), but it has missed more of the relevant items which are in the document collection overall (which is measured by recall).

In general, there is a inverse relationship between precision and recall, as Figure 1.4 illustrates. Here, precision and recall of a fictional system is plotted versus the number of items that are retrieved: the more items the system returns, the higher the likelihood that it will retrieve more and more relevant documents from the overall collection (if all documents are retrieved, recall is 1 by definition), but at the same time precision will decrease.

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

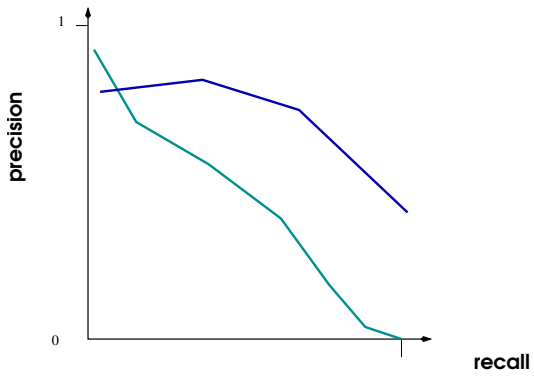

Figure 1.5. A Typical Precision-Recall Curve.

and that the worst-case scenario is having to consider *any* non-relevant documents. 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, whereas time is less of an issue.

Figure 1.5 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 1.4 (but in contrast to Figure 1.4, 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.

Because of the inverse relationship between precision and recall, it is not obvious how overall performance of a given system can be estimated. Another factor in defining precision and recall is the advance in IR system technology from Boolean systems (which returned a fixed set of documents it considered relevant) to a non-binary definition of relevance, such as used in today's search engines, which return a very large number of documents, but sort them by their relevance to the query. In the latter case, many more data points can be collected, by artificially considering increasingly larger return sets.

The area under the precision-recall curve is an estimate of the system's performance. Because in a practical setting, it is not typically possible to manipulate a system's retrieval set, to plot a precision-recall curve and then estimate the area under the curve, simpler estimations considered are the crossing-over of precision and recall; the F-measure (cf. below) which provides a summary measure conflating precision and recall; and more complex evaluation metrics such as the 11-point average precision (cf. below).

We are now in a position to be able to discuss the recall-problem of collecting 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 (A quick calculation shows that it would take an estimated 6500 hours for one single query (at a very 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 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 there is a large increase in pool quality if humans can additionally do manual recall-oriented searches, using a normal search engine, e.g., by spelling out all possible synonyms the searcher can think of. Fortunately and expectedly, there is considerable overlap in returned documents: the pool is smaller than theoretical maximum of $N \cdot n$ systems (around $\frac{1}{3}$ the maximum size).

The result found by Voorhees (2002) is also very relevant at this point: 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. mean precision by seen documents) differs 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 .9). 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 enough systems that the rank vector is large enough to be meaningful. Voorhees' results show that no interaction occurs (no annotator's relevance decisions favours any system over any other). This property is called *stability* of judgements.

We now consider composite evaluation measures for IR. *F-measure* (van Rijsbergen, 1979) is defined as the weighted harmonic mean of precision and recall:

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

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

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

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 (e.g., 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: P(R=0.2) is the Precision at that point when a recall of 0.2 has been reached).

Another dimension of difficulty becomes apparent when we consider IR measures which can average over more than one query. The problem is that no single cutoff (as discussed above) can be set for all queries, as queries can have different numbers of relevant documents. The use of fixed thresholds would not allow systems to achieve the theoretically possible maximal values in all conditions. For instance, if we set the cutoff at 10 documents, then by definition all queries with more than 10 relevant documents could never achieve full recall, and all queries with less than 10 relevant documents could never achieve full precision. Thus, more complicated joint measures are required.

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

$$P_{11\_pt} = \frac{1}{11}\sum_{j=0}^{10}\frac{1}{N}\sum_{i=1}^{N}P_{ip,i}(r_j)$$

with $P_{ip,i}(r_j)$ being the jth interpolated recall point in the ith query (out of N queries):

$$P_{ip}(r_i) = max(r_i < r \leq r_{i+1})P(r)$$

$P(R=n)$ is precision at that point where recall has first reached n
$P(r_n) = P(R=\frac{n}{10})$
$P(r_0), P(r_1), ...P(r_{10})$ are standard recall points.
$P(r_2)$: precision at the point where $R = 0.2$.

The $P(r_n)$ do not in the general case coincide with a measured data point, which is why the value may have to be interpolated, for instance by the following formula:

$$P_{ip}(r_i) = max(r_i < r \leq r_{i-1})$$

Query 1			$P_1(r_i)$	$\mathbf{P_2(r_i)}$
#		R		
1	X	0.20	$P_{ip,1}(r_0) = 1.00$	$P_{ip,2}(r_0) = 1.00$
2			$P_{ip,1}(r_1) = 1.00$	$P_{ip,2}(r_1) = 1.00$
3	X	0.40	$P_1(r_2) = 1.00$	$P_{ip,2}(r_2) = 1.00$
4			$P_{ip,1}(r_3) = 0.67$	$P_{ip,2}(r_3) = 1.00$
5			$P_1(r_4) = 0.67$	
6	X	0.60		$P_{ip,2}(r_4) = 0.67$
7			$P_{ip,1}(r_5) = 0.50$	$P_{ip,2}(r_5) = 0.67$
8			$P_1(r_6) = 0.50$	$P_{ip,2}(r_6) = 0.67$
9				
10	X	0.80	$P_{ip,1}(r_7) = 0.40$	$P_{ip,2}(r_7) = 0.20$
11			$P_1(r_8) = 0.40$	$P_{ip,2}(r_8) = 0.20$
12				
13				
14			$P_{ip,1}(r_9) = 0.25$	$P_{ip,2}(r_9) = 0.20$
15				
16				
17				$P_2(r_{10}) = 0.20$
18				
19				
20	X	1.00	$P_1(r_{10}) = 0.25$	

Query 2		
R		#
0.33	X	1
		2
0.67	X	3
		4
		5
		6
		7
		8
		9
		10
		11
		12
		13
		14
1.00	X	15

Figure 1.6. Example Calculation: 11-Point Average Precision, for two Queries.

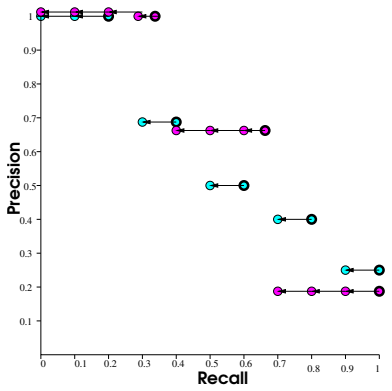

Figure 1.7. Precision-Recall Curve for the Example Calculation.

In Figure 1.6, $P_{ip}(r_i)$ values have been interpolated, and $P(r_i)$ values have been exactly measured. Figure 1.7 gives the precision-recall curve for this example (dark for Query 1; light for Query 2; bold circles for measured data points and thin circles for interpolated data points).

Note that it does not matter that the two queries have different numbers of relevant queries (and that the last relevant query occurs at differ-

ent ranks), because we still receive exactly 11 precision-recall points as required. The final calculation is the average of all $P_{ip,i}(r_i)$ (1.00, 1.00, 1.00, .84, .67, .59, .59, .30, .23, 23)), resulting in 0.61.

There is a second, simpler composite measurement which generalises over different queries, called *mean precision at seen relevant documents* (often also called "mean average precision"). Precision is calculated at each point when a new relevant document is retrieved (using 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.

$$P_{srd} = \frac{1}{N} \sum_{j=1}^{N} \frac{1}{Q_j} \sum_{i=1}^{Q_j} P(rel = i)$$

with:

Q_j	number of relevant documents for query j
N	number of queries
$P(rel = i)$	precision at ith relevant document

Again, an example calculation for the same two queries will show the details:

Query 1		
#		P
1	X	1.00
2		
3	X	0.67
4		
5		
6	X	0.50
7		
8		
9		
10	X	0.40
11		
12		
13		
14		
15		
16		
17		
18		
19		
20	X	0.25
AVG:		0.564

Query 2		
#	.	P
1	X	1.00
2		
3	X	0.67
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15	X	0.2
AVG:		0.623

In our example calculation, mean precision at seen documents is as follows: $P_{srd} = \frac{0.564+0.623}{2} = 0.594$.

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 overall is doing, it is necessary to look at more than one of these. TREC has been criticised for putting too much emphasis on recall, given that most of today's IR requirements are precision-based. This is one of the reasons why many researchers prefer mean precision over 11-point-average precision as an overall summary IR measure.

So how well do systems typically perform? Spärck-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 8 (the last two 'ad-hoc' TREC conferences in 1997 and 1998), the best results were in the range of .40 to .45 (in terms of precision at document cutoff 30). These systems mostly achieve these results only when using long queries and narratives, but one team in TREC-7 managed results in the .40 to .45 range even for the short queries. The systems were clearly optimised for long queries; all performed worse for shorter, more realistic queries. Generally, the best systems are statistically not significantly different, which points to the fact that an apparent plateau that has been reached. Manual searches resulted in best results in the range between .55 and .60. Comparison of the TREC results over the years shows that the .40 to .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 .55 to .60 range – however this was achieved only for long queries. At cutoff 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 more superior systems, as the manual performance has remained on a plateau.

In summary, IR evaluation as done by TREC only covers one small part of the spectrum of IR evaluations: Firstly, only *system* performance is measured, and only in batch mode, ignoring interactive search. The

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 1.8. Example TREC Questions.

evaluation is set in laboratory conditions, not directly involving real users in natural interaction with their IR system. The method uses precision and recall measured from large, fixed test collections; a host of sophisticated performance metrics is available, e.g. 11-point average precision and mean-precision at seen relevant documents. In principle there is a problem with the subjectivity of relevance, but it turns out that this problem is solvable by extensive sampling. There also is a recall problem, but it too is solvable – with pooling. Most importantly, there is proof that the evaluation methodology used is stable towards human judgement differences. Therefore, we can consider IR evaluation as mature.

3. Evaluation of Question Answering

Since 1999, a TREC track for open-domain factual question answering has been in existence (Voorhees, 1999), attracting 20–36 participant groups in the following years. The task description is different to traditional IR: the input questions are in natural language, not in keywords, and the system output is not documents but smaller units, "answers" (formally defined below). Figure 1.8 shows example questions from the first three TREC-QAs.

The TREC evaluation procedure is as follows: Participants in TREC have access to prior questions and to the document collection. NIST assessors prepare about 500 questions + appropriate responses. System runs are performed at the participants' sites; once the questions are received, the systems are to be kept frozen, and answers are to be sent back to TREC within one week. Answers may be extracted (in earlier TRECs, they could be generated in any other automatic way) from the material in the document collection only. The use of external resources (dictionaries, ontologies, WWW) is explicitly allowed. Each returned answer is checked manually by TREC-QA for correctness (i.e., there is no fixed gold standard to compare to). NIST then calculates the results

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 1.9. Answer List for List Question 1915 (TREC-12): Names of Chewing Gums Found Within the AQUAINT Corpus.

and sends these to the participants, who in turn present their systems at the conference in November. The proceedings are published in February on the web.

3.1 Question types

There are different question types in TREC-QA: factoid questions (i.e., questions which can be answered with a single fact), list questions (where more than one instance of an answer type has to be found) and definition questions. Amongst the question types which TREC-QA experimented with, but gave up on, are 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". These questions make up the largest part of the QA question set. Opinion questions, such as "what is the greatest film ever made?" are explicitly excluded.

List questions ask for several instances of one type. 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 TREC-10, examples for list questions included: 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 later TRECs, the target number is no longer given; systems are required to find *all* instances of a certain type (cf. Figure 1.9). The 37 list questions in TREC-12 had different numbers of distinct answers, ranging from a low of 3 (What Chinese provinces have a McDonald's restaurant?) to a high of 44 (Which countries were visited by first lady Hillary Clinton?).

Definition questions such as "Who is Colin Powell?" and "What is mold"? 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 cannot go into the specifics of their

exploratory evaluation 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 question answering 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 not repeated after TREC-10.

Apart from question type, there are other dimensions of difficulty as defined in TREC-QA, and we will quickly discuss these here.

3.2 Other factors of question difficulty in TREC

The source of the question is an important factor. While in initial TRECs, the questions were formulated by first finding interesting facts in newspapers, and then formulating questions about those facts, later TRECs used more realistic models of question sources, by mining them from web logs (Encarta, Excite). The source of the questions has a strong impact on the task, as more realistic questions are harder on assessors and systems, but more representative for training.

The fact that in TREC-8 questions were formulated *after* the answer has already been located meant that the formulation of the question was influenced by the actual answer string. It is therefore likely that the answers and the questions are more similar to each other than they were to become in later TREC-QA tracks. This made it more important for systems competing in later TREC-QAs to deal with synonymy, polysemy and other phenomena of superficial (string) differences between question and answers.

There is also variation in the type of acceptable answer. Early TRECs allowed for 250 Bytes or 50 Bytes (required in TREC-10). Since TREC-11, exact answers are required. Figure 1.10 illustrates the reasons behind this decision. It gives real example strings submitted to a TREC-9 question, in decreasing order of quality. In order to foster better QA systems, it was felt that a mechanism should be in place to reward more precise answers – something unqualified 50 Byte answers do not afford. Since TREC-12, 250 Byte answers are allowable again in a specialised task (the *passage* 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 1.10. Example Answers.

Since TREC-10 there is no guarantee of the existence of an answer in the document collection anymore, and on average 10% of the questions 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 answer guarantee makes the task harder, as the systems now need an internal measure of their confidence in an answer (only 5 systems in TREC-10 had a NIL-precision $> .25$; this remained similar in later years).

In early TRECs, any automatically generated string could be submitted as an answer. Since TREC-12, these strings have to be extracts (ie they have to be found verbatim in a document). This simplifies answer judgement and discourages "answer stuffing" (a practise where several suspected answers are stuffed into the result string).

Document collections for TREC-8 to TREC-10 were the same as for the main TREC (disks 1-5, 979,000 articles), but in TREC-11, the collection was changed to the roughly same-sized AQUAINT-collection which covers a more recent time frame (1998-2000). This collection consists of documents from three different sources: the AP newswire from 1998 – 2000, the New York Times newswire from 1998 – 2000, and the (English portion of the) Xinhua News Agency from 1996 – 2000. There are approximately 1,033,000 documents and 3 gigabytes of text in the collection.

Timeliness of the collection was felt to be an important factor, as from TREC-11 onwards systems started using the Internet as additional information source and projected the answers found in the web back into the document collection. There were several cases where the web

search returned the right answer, but the back-projection into the aged document collection failed or brought out an incorrect answer. The move to the AQUAINT corpus ameliorated this situation considerably.

3.3 Human judgement of answers

Systems return [docid, answer-string] pairs (five per question in early TRECs, one per question since TREC-11). Each answer is independently judged as correct, unsupported, or incorrect by two human assessors. When the two judgements differed, an adjudicator made the final decision. An answer was judged correct if it contained a right answer to the question, if the document from which it was drawn made it clear that it was a right answer, and if the answer was 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 was judged unsupported if it contained a right answer and was responsive, but the document from which it was drawn does not indicate that it is a right answer. Otherwise, an answer was judged as incorrect. Answers supported by a document are accepted even if 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.

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.

Assessor opinions differed, for instance with respect to how much detail was required to answer a question. Therefore, unsurprisingly, inter-assessor agreement was not very high, but at least the *relative* MMRs (the ranks) were stable even if the absolute MMRs were not (Voorhees and Tice, 2000), mirroring the situation in IR. The 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.

It was considered important that the TREC-QA dataset would be reusable for 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

Who was Jane Goddall?

```
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 1.11. Example Answer Patterns in Perl.

answers, e.g. Figure 1.11. This is only a partial solution, as there is no guarantee that a later system would not return a new, better answer, which was not known at the time when the patterns were fixed (i.e., immediately after the the competition). Evaluation against frozen patterns is therefore to be considered as inferior, as 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". Nevertheless, patterns can be useful if their limitations are understood, as system rankings produced by *lenient* annotator assessment and pattern-based results are highly correlated: Kendall τ of .944 for 250 bytes and .894 for 50 bytes. (Lenient assessment, given up on after TREC-9, treated unsupported answers as if they were fully correct.)

3.4 QA Evaluation metrics

There has been some experimentation in TREC with three different evaluation metrics: mean reciprocal rank, weighted confidence, and average accuracy. Average accuracy is the simplest evaluation metric and is 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 has been calculated since 2001, when the guarantee of a answer existence was given up and the first questions requiring NIL answers appeared.

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 5 answers only, as the task is precision-oriented and lower ranks are assumed not to be of interest. The mean reciprocal rank

(MRR) is defined as the mean of the inverse rank of the first correct answer, taken over all n questions:

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

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 5 returns). Thus, there are only 6 possible reciprocal ranks per question: 0, 0.2, 0.25, 0.33, 0.5, and 1.

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

There are advantages and disadvantages to MRR. 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 not credit for system which know that they don't know, and there is no credit for multiple but 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 judgements. Let S be the size of the final answer list (i.e., the number of known answers), D be the number of correct, distinct responses returned by the system, and N be 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 changed to *confidence-weighted score*, which was 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 the confidence in that answer, with the answer they were most confident with ranked highest. 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). Under this measurement, it was possible for two systems with the same returned answers to score considerably differently, if they ranked their answers differently. As was the case in previous evaluations, relative confidence-weighted score is stable at Kendall's tau of above 0.9 (Voorhees, 2002).

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).

	50 Bytes		
	TREC-8, 50B	TREC-9, 50B	TREC-10 (50B)
Best MRR/No Answer	.66/27%	.58/32%	.68/31%
2nd best MRR/No Answer	.56/32%	.32/58%	.59/36%
Best system	Cymphony	SMU	Insight
	250 Bytes		
	TREC-8	TREC-9	
Best MRR/No Answer	65%/22%	.75/16%	
2nd best MRR/No Answer	55%/32%	.46/49%	
Best system	SMU	SMU	

Figure 1.12. Results of Top-Scoring Systems in MRRs, TREC-8 to TREC-10.

	Conf. weighted score	Correct answers	Number inexact	NIL prec	NIL recall
LCCmain2002	.856	415 (75%)	8	.578	.804
exactanswer	.691	271 (54%)	12	.222	.848
pris2002	.610	290 (58%)	17	.241	.891

Figure 1.13. Best 3 Systems at TREC-11.

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.

Figure 1.12 lists results of the top-scoring systems in the early TRECs, in terms of MRRs and percentages of unanswered questions. Interestingly, in 55% of cases where answer was found in the first 5 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.

Figure 1.13 lists the best systems in TREC-11. The highest scores for the TREC-11 list task 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 in the passage task (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).

	TREC-8	**TREC-9**	**TREC-10**	**TREC-11**	**TREC-12**
Year	1999	2000	2001	2002	2003
# quest.	200	500+193 variants	500	500	413
Excluded	2	11	8		
Source of questions	Assessors, FAQ finder logs, participants	Encarta, Excite, MSNSearch, AskJeeves	MSNSearch, AskJeeves	MSNSearch, AskJeeves	AOL and MSNSearch logs
Corpus	TREC 4/5	TREC 4/5	TREC 4/5	AQUAINT	AQUAINT
# docum.	528,000	979,000	979,000	1,033,461	1,033,461
Answer length	50B or 250B	50B or 250B	50B	exact	exact or 250B
Tasks	main	main, variants	main, list, context	main, list	main, definition, list
# NIL questions	—	—	49	46	30
New aspect	—	unsupport.	no answer (NIL)	exact answers	passage task
# of participants	20	28	36	34	13 passage/25 main
Eval measure	MRR	MRR	MRR	weighted confidence	accuracy

Figure 1.14. Overview of TREC-QA Tracks.

Figure 1.14 gives an overview of aspects of the last five TREC-QA tracks discussed in the text.

In summary, QA is a task that has only appeared in the last five 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. There has been an enormous human effort in question creation and answer judgement.

The questions currently used in TREC-QA are factual and simple, and there has been some doubt about whether the field are ready to move to harder tasks. Undoubtedly, deep NLP (e.g., comparison on the basis of logical form) helps for QA: those systems which consistently perform very well in all TREC-QAs (Harabagiu et al., 2003) have shown this to be the case. However, there have been competitor systems which used 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" system. 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 only 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, then the task may have to be made much harder now – for instance, by using harder question types and requiring reasoning behind the answers.

However, overall the evaluation of question answering can be considered as a task which has managed to find its defining coordinates in a short time. Factors of the evaluation have constantly been adjusted, following system developments and factors which could not be known beforehand. In only five 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. The systems were encouraged to answer succinctly and unambiguously, and to indicate when they were not certain about the answer returned. Overall, QA evaluation can be seen as a success story, partially due to the fact that lessons from IR evaluation could be taken into account directly.

References

Cleverdon, C. W. (1967). The Cranfield tests on index language devices. *Aslib Proceedings*, 19:173–192.

Harabagiu, S., Moldovan, D., Clark, C., Bowden, M., Williams, J., and Bensley, J. (2003). Answer mining by combining extraction techniques with abductive reasoning. In *Proceedings of TREC-12*, page 375.

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

Saracevic, T., Kantor, P. B., Chamis, 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*, 31:291–314.

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

Sparck Jones, K. and Galliers, J. (1996). *Evaluating Natural Language Systems*. Springer Verlag.

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 Jones, K. S. (1976). Information retrieval test collections. *Journal of Documentation*, 32:59–75.

Voorhees, E. (1999). The TREC-8 question answering track report. In *Proceedings of TREC*. `http://trec.nist.gov/pubs/trec8/t8_proceedings.html`.

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 *TREC proceedings*.

Voorhees, E. (2003). Overview of the TREC 2003 question answering track. In *TREC proceedings*.

Voorhees, E. and Tice, D. (2000). Building a question answering test collection. In *Proceedings of SIGIR-2000*, pages 200–207.