8 *Evaluation in information retrieval*

We have seen in the preceding chapters many alternatives in designing an IR system. How do we know which of these techniques are effective in which applications? Should we use stop lists? Should we stem? Should we use inverse document frequency weighting? Information retrieval has developed as a highly empirical discipline, requiring careful and thorough evaluation to demonstrate the superior performance of novel techniques on representative document collections.

In this chapter we begin with a discussion of measuring the effectiveness of IR systems (Section 8.1) and the test collections that are most often used for this purpose (Section 8.2). We then present the straightforward notion of relevant and nonrelevant documents and the formal evaluation methodology that has been developed for evaluating unranked retrieval results (Section 8.3). This includes explaining the kinds of evaluation measures that are standardly used for document retrieval and related tasks like text classification and why they are appropriate. We then extend these notions and develop further measures for evaluating ranked retrieval results (Section 8.4) and discuss developing reliable and informative test collections (Section 8.5).

We then step back to introduce the notion of user utility, and how it is approximated by the use of document relevance (Section 8.6). The key utility measure is user happiness. Speed of response and the size of the index are factors in user happiness. It seems reasonable to assume that relevance of results is the most important factor: blindingly fast, useless answers do not make a user happy. However, user perceptions do not always coincide with system designers' notions of quality. For example, user happiness commonly depends very strongly on user interface design issues, including the layout, clarity, and responsiveness of the user interface, which are independent of the quality of the results returned. We touch on other measures of the quality of a system, in particular the generation of high-quality result summary snippets, which strongly influence user utility, but are not measured in the basic relevance ranking paradigm (Section 8.7).

8.1 Information retrieval system evaluation

To measure ad hoc information retrieval effectiveness in the standard way, we need a test collection consisting of three things:

1. A document collection
2. A test suite of information needs, expressible as queries
3. A set of relevance judgments, standardly a binary assessment of either *relevant* or *nonrelevant* for each query-document pair.

RELEVANCE

The standard approach to information retrieval system evaluation revolves around the notion of *relevant* and *nonrelevant* documents. With respect to a user information need, a document in the test collection is given a binary classification as either relevant or nonrelevant. This decision is referred to as the *gold standard* or *ground truth* judgment of relevance. The test document collection and suite of information needs have to be of a reasonable size: you need to average performance over fairly large test sets, as results are highly variable over different documents and information needs. As a rule of thumb, 50 information needs has usually been found to be a sufficient minimum.

GOLD STANDARD
GROUND TRUTH

INFORMATION NEED

Relevance is assessed relative to an information need, *not* a query. For example, an information need might be:

> Information on whether drinking red wine is more effective at reducing your risk of heart attacks than white wine.

This might be translated into a query such as:

wine AND red AND white AND heart AND attack AND effective

A document is relevant if it addresses the stated information need, not because it just happens to contain all the words in the query. This distinction is often misunderstood in practice, because the information need is not overt. But, nevertheless, an information need is present. If a user types python into a web search engine, they might be wanting to know where they can purchase a pet python. Or they might be wanting information on the programming language Python. From a one word query, it is very difficult for a system to know what the information need is. But, nevertheless, the user has one, and can judge the returned results on the basis of their relevance to it. To evaluate a system, we require an overt expression of an information need, which can be used for judging returned documents as relevant or nonrelevant. At this point, we make a simplification: relevance can reasonably be thought of as a scale, with some documents highly relevant and others marginally so. But for the moment, we will use just a binary decision of relevance. We

discuss the reasons for using binary relevance judgments and alternatives in Section 8.5.1.

Many systems contain various weights (often known as parameters) that can be adjusted to tune system performance. It is wrong to report results on a test collection which were obtained by tuning these parameters to maximize performance on that collection. That is because such tuning overstates the expected performance of the system, because the weights will be set to maximize performance on one particular set of queries rather than for a random sample of queries. In such cases, the correct procedure is to have one or more *development test collections*, and to tune the parameters on the development test collection. The tester then runs the system with those weights on the test collection and reports the results on that collection as an unbiased estimate of performance.

DEVELOPMENT TEST COLLECTION

8.2 Standard test collections

Here is a list of the most standard test collections and evaluation series. We focus particularly on test collections for ad hoc information retrieval system evaluation, but also mention a couple of similar test collections for text classification.

CRANFIELD

The *Cranfield* collection. This was the pioneering test collection in allowing precise quantitative measures of information retrieval effectiveness, but is nowadays too small for anything but the most elementary pilot experiments. Collected in the United Kingdom starting in the late 1950s, it contains 1398 abstracts of aerodynamics journal articles, a set of 225 queries, and exhaustive relevance judgments of all (query, document) pairs.

TREC

Text Retrieval Conference (TREC). The U.S. National Institute of Standards and Technology (NIST) has run a large IR test bed evaluation series since 1992. Within this framework, there have been many tracks over a range of different test collections, but the best known test collections are the ones used for the TREC Ad Hoc track during the first 8 TREC evaluations between 1992 and 1999. In total, these test collections comprise 6 CDs containing 1.89 million documents (mainly, but not exclusively, newswire articles) and relevance judgments for 450 information needs, which are called *topics* and specified in detailed text passages. Individual test collections are defined over different subsets of this data. The early TRECs each consisted of 50 information needs, evaluated over different but overlapping sets of documents. TRECs 6–8 provide 150 information needs over about 528,000 newswire and Foreign Broadcast Information Service articles. This is probably the best subcollection to use in future work, because it is the largest and the topics are more consistent. Because the test

document collections are so large, there are no exhaustive relevance judgments. Rather, NIST assessors' relevance judgments are available only for the documents that were among the top k returned for some system which was entered in the TREC evaluation for which the information need was developed.

GOV2

In more recent years, NIST has done evaluations on larger document collections, including the 25 million page *GOV2* web page collection. From the beginning, the NIST test document collections were orders of magnitude larger than anything available to researchers previously and GOV2 is now the largest Web collection easily available for research purposes. Nevertheless, the size of GOV2 is still more than 2 orders of magnitude smaller than the current size of the document collections indexed by the large web search companies.

NTCIR

CROSS-LANGUAGE INFORMATION RETRIEVAL

NII Test Collections for IR Systems (*NTCIR*). The NTCIR project has built various test collections of similar sizes to the TREC collections, focusing on East Asian language and *cross-language information retrieval*, where queries are made in one language over a document collection containing documents in one or more other languages. See: http://research.nii.ac.jp/ntcir/data/data-en.html

CLEF

Cross Language Evaluation Forum (*CLEF*). This evaluation series has concentrated on European languages and cross-language information retrieval. See: http://www.clef-campaign.org/

REUTERS

Reuters-21578 and Reuters-RCV1. For text classification, the most used test collection has been the Reuters-21578 collection of 21578 newswire articles; see Chapter 13, page 279. More recently, Reuters released the much larger Reuters Corpus Volume 1 (RCV1), consisting of 806,791 documents; see Chapter 4, page 69. Its scale and rich annotation makes it a better basis for future research.

20 NEWSGROUPS

20 Newsgroups. This is another widely used text classification collection, collected by Ken Lang. It consists of 1000 articles from each of 20 Usenet newsgroups (the newsgroup name being regarded as the category). After the removal of duplicate articles, as it is usually used, it contains 18941 articles.

8.3 Evaluation of unranked retrieval sets

Given these ingredients, how is system effectiveness measured? The two most frequent and basic measures for information retrieval effectiveness are precision and recall. These are first defined for the simple case where an

IR system returns a set of documents for a query. We will see later how to extend these notions to ranked retrieval situations.

PRECISION *Precision* (P) is the fraction of retrieved documents that are relevant

$$\text{Precision} = \frac{\#(\text{relevant items retrieved})}{\#(\text{retrieved items})} = P(\text{relevant}|\text{retrieved}) \tag{8.1}$$

RECALL *Recall* (R) is the fraction of relevant documents that are retrieved

$$\text{Recall} = \frac{\#(\text{relevant items retrieved})}{\#(\text{relevant items})} = P(\text{retrieved}|\text{relevant}) \tag{8.2}$$

These notions can be made clear by examining the following contingency table:

(8.3)

	Relevant	Nonrelevant
Retrieved	true positives (tp)	false positives (fp)
Not retrieved	false negatives (fn)	true negatives (tn)

Then:

$$\begin{aligned} P &= tp/(tp+fp) \\ R &= tp/(tp+fn) \end{aligned} \tag{8.4}$$

ACCURACY An obvious alternative that may occur to the reader is to judge an information retrieval system by its *accuracy*, that is, the fraction of its classifications that are correct. In terms of the contingency table above, accuracy $= (tp+tn)/(tp+fp+fn+tn)$. This seems plausible, since there are two actual classes, relevant and nonrelevant, and an information retrieval system can be thought of as a two-class classifier which attempts to label them as such (it retrieves the subset of documents which it believes to be relevant). This is precisely the effectiveness measure often used for evaluating machine learning classification problems.

There is a good reason why accuracy is not an appropriate measure for information retrieval problems. In almost all circumstances, the data is extremely skewed: normally over 99.9% of the documents are in the nonrelevant category. A system tuned to maximize accuracy can appear to perform well by simply deeming all documents nonrelevant to all queries. Even if the system is quite good, trying to label some documents as relevant will almost always lead to a high rate of false positives. However, labeling all documents as nonrelevant is completely unsatisfying to an information retrieval system user. Users are always going to want to see some documents, and can be

assumed to have a certain tolerance for seeing some false positives providing that they get some useful information. The measures of precision and recall concentrate the evaluation on the return of true positives, asking what percentage of the relevant documents have been found and how many false positives have also been returned.

The advantage of having the two numbers for precision and recall is that one is more important than the other in many circumstances. Typical web surfers would like every result on the first page to be relevant (high precision) but have not the slightest interest in knowing let alone looking at every document that is relevant. In contrast, various professional searchers such as paralegals and intelligence analysts are very concerned with trying to get as high recall as possible, and will tolerate fairly low precision results in order to get it. Individuals searching their hard disks are also often interested in high recall searches. Nevertheless, the two quantities clearly trade off against one another: you can always get a recall of 1 (but very low precision) by retrieving all documents for all queries! Recall is a non-decreasing function of the number of documents retrieved. On the other hand, in a good system, precision usually decreases as the number of documents retrieved is increased. In general we want to get some amount of recall while tolerating only a certain percentage of false positives.

F MEASURE

A single measure that trades off precision versus recall is the *F measure*, which is the weighted harmonic mean of precision and recall:

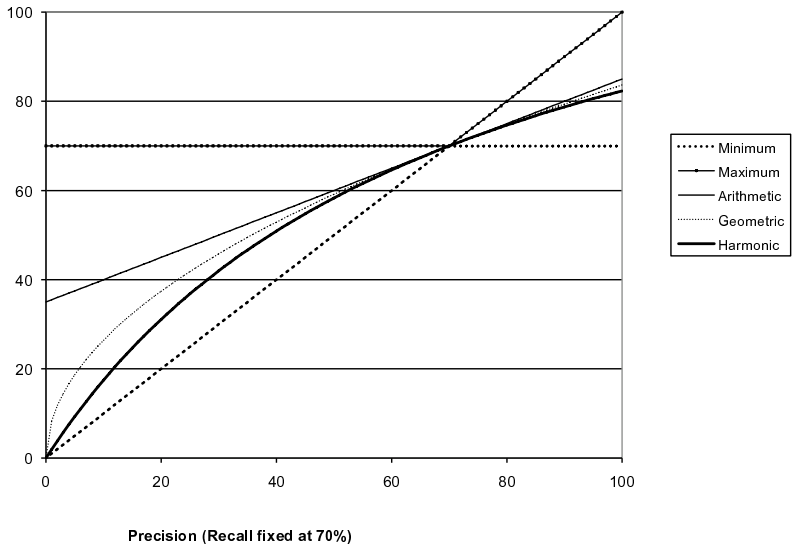
$$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}} = \frac{(\beta^2+1)PR}{\beta^2 P + R} \quad \text{where} \quad \beta^2 = \frac{1-\alpha}{\alpha} \tag{8.5}$$

where $\alpha \in [0,1]$ and thus $\beta^2 \in [0,\infty]$. The default *balanced F measure* equally weights precision and recall, which means making $\alpha = 1/2$ or $\beta = 1$. It is commonly written as F_1, which is short for $F_{\beta=1}$, even though the formulation in terms of α more transparently exhibits the F measure as a weighted harmonic mean. When using $\beta = 1$, the formula on the right simplifies to:

$$F_{\beta=1} = \frac{2PR}{P+R} \tag{8.6}$$

However, using an even weighting is not the only choice. Values of $\beta < 1$ emphasize precision, while values of $\beta > 1$ emphasize recall. For example, a value of $\beta = 3$ or $\beta = 5$ might be used if recall is to be emphasized. Recall, precision, and the F measure are inherently measures between 0 and 1, but they are also very commonly written as percentages, on a scale between 0 and 100.

Why do we use a harmonic mean rather than the simpler average (arithmetic mean)? Recall that we can always get 100% recall by just returning all documents, and therefore we can always get a 50% arithmetic mean by the



► **Figure 8.1** Graph comparing the harmonic mean to other means. The graph shows a slice through the calculation of various means of precision and recall for the fixed recall value of 70%. The harmonic mean is always less than either the arithmetic or geometric mean, and often quite close to the minimum of the two numbers. When the precision is also 70%, all the measures coincide.

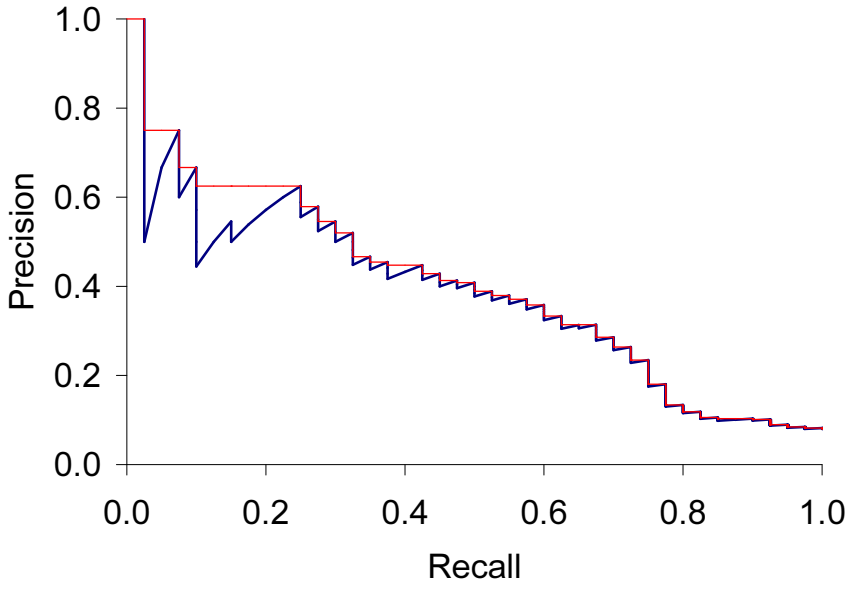
same process. This strongly suggests that the arithmetic mean is an unsuitable measure to use. In contrast, if we assume that 1 document in 10,000 is relevant to the query, the harmonic mean score of this strategy is 0.02%. The harmonic mean is always less than or equal to the arithmetic mean and the geometric mean. When the values of two numbers differ greatly, the harmonic mean is closer to their minimum than to their arithmetic mean; see Figure 8.1.

? **Exercise 8.1** [⋆]

An IR system returns 8 relevant documents, and 10 nonrelevant documents. There are a total of 20 relevant documents in the collection. What is the precision of the system on this search, and what is its recall?

Exercise 8.2 [⋆]

The balanced F measure (a.k.a. F_1) is defined as the harmonic mean of precision and recall. What is the advantage of using the harmonic mean rather than "averaging" (using the arithmetic mean)?



► **Figure 8.2** Precision/recall graph.

Exercise 8.3 [⋆⋆]

Derive the equivalence between the two formulas for F measure shown in Equation (8.5), given that $\alpha = 1/(\beta^2 + 1)$.

8.4 Evaluation of ranked retrieval results

Precision, recall, and the F measure are set-based measures. They are computed using unordered sets of documents. We need to extend these measures (or to define new measures) if we are to evaluate the ranked retrieval results that are now standard with search engines. In a ranked retrieval context, appropriate sets of retrieved documents are naturally given by the top k retrieved documents. For each such set, precision and recall values can be plotted to give a *precision-recall curve*, such as the one shown in Figure 8.2. Precision-recall curves have a distinctive saw-tooth shape: if the $(k+1)^{\text{th}}$ document retrieved is nonrelevant then recall is the same as for the top k documents, but precision has dropped. If it is relevant, then both precision and recall increase, and the curve jags up and to the right. It is often useful to remove these jiggles and the standard way to do this is with an interpolated precision: the *interpolated precision* p_{interp} at a certain recall level r is defined

PRECISION-RECALL CURVE

INTERPOLATED PRECISION

Recall	Interp. Precision
0.0	1.00
0.1	0.67
0.2	0.63
0.3	0.55
0.4	0.45
0.5	0.41
0.6	0.36
0.7	0.29
0.8	0.13
0.9	0.10
1.0	0.08

▶ **Table 8.1** Calculation of 11-point Interpolated Average Precision. This is for the precision-recall curve shown in Figure 8.2.

as the highest precision found for any recall level $r' \geq r$:

$$p_{interp}(r) = \max_{r' \geq r} p(r') \tag{8.7}$$

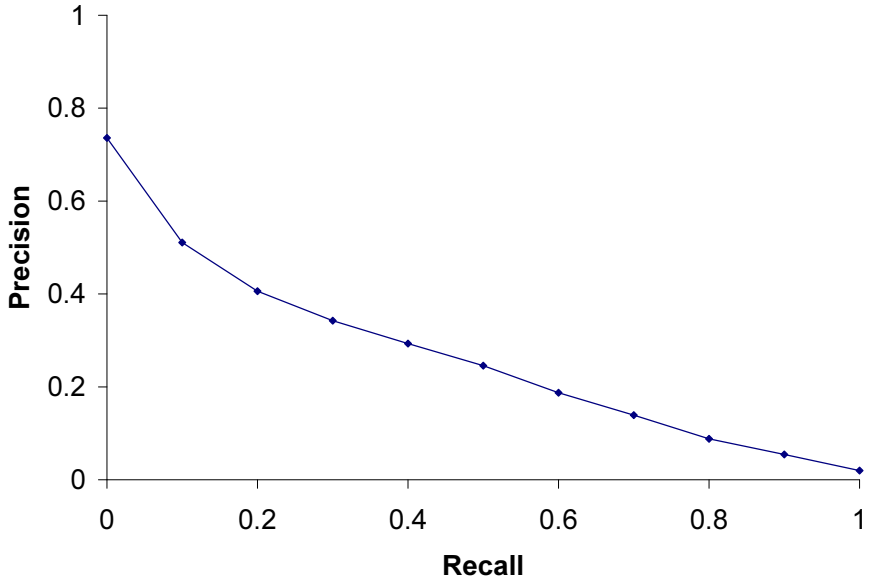
The justification is that almost anyone would be prepared to look at a few more documents if it would increase the percentage of the viewed set that were relevant (that is, if the precision of the larger set is higher). Interpolated precision is shown by a thinner line in Figure 8.2. With this definition, the interpolated precision at a recall of 0 is well-defined (Exercise 8.4).

Examining the entire precision-recall curve is very informative, but there is often a desire to boil this information down to a few numbers, or perhaps even a single number. The traditional way of doing this (used for instance in the first 8 TREC Ad Hoc evaluations) is the *11-point interpolated average precision*. For each information need, the interpolated precision is measured at the 11 recall levels of 0.0, 0.1, 0.2, ..., 1.0. For the precision-recall curve in Figure 8.2, these 11 values are shown in Table 8.1. For each recall level, we then calculate the arithmetic mean of the interpolated precision at that recall level for each information need in the test collection. A composite precision-recall curve showing 11 points can then be graphed. Figure 8.3 shows an example graph of such results from a representative good system at TREC 8.

11-POINT INTERPOLATED AVERAGE PRECISION

In recent years, other measures have become more common. Most standard among the TREC community is *Mean Average Precision* (MAP), which provides a single-figure measure of quality across recall levels. Among evaluation measures, MAP has been shown to have especially good discrimination and stability. For a single information need, Average Precision is the

MEAN AVERAGE PRECISION



► **Figure 8.3** Averaged 11-point precision/recall graph across 50 queries for a representative TREC system. The Mean Average Precision for this system is 0.2553.

average of the precision value obtained for the set of top k documents existing after each relevant document is retrieved, and this value is then averaged over information needs. That is, if the set of relevant documents for an information need $q_j \in Q$ is $\{d_1, \ldots d_{m_j}\}$ and R_{jk} is the set of ranked retrieval results from the top result until you get to document d_k, then

$$\text{MAP}(Q) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_{k=1}^{m_j} \text{Precision}(R_{jk}) \tag{8.8}$$

When a relevant document is not retrieved at all,[1] the precision value in the above equation is taken to be 0. For a single information need, the average precision approximates the area under the uninterpolated precision-recall curve, and so the MAP is roughly the average area under the precision-recall curve for a set of queries.

Using MAP, fixed recall levels are not chosen, and there is no interpolation. The MAP value for a test collection is the arithmetic mean of average

1. A system may not fully order all documents in the collection in response to a query or at any rate an evaluation exercise may be based on submitting only the top k results for each information need.

precision values for individual information needs. (This has the effect of weighting each information need equally in the final reported number, even if many documents are relevant to some queries whereas very few are relevant to other queries.) Calculated MAP scores normally vary widely across information needs when measured within a single system, for instance, between 0.1 and 0.7. Indeed, there is normally more agreement in MAP for an individual information need across systems than for MAP scores for different information needs for the same system. This means that a set of test information needs must be large and diverse enough to be representative of system effectiveness across different queries.

PRECISION AT k

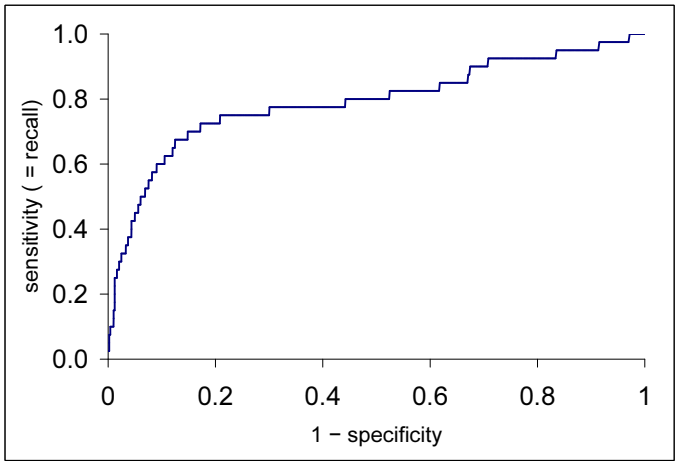
The above measures factor in precision at all recall levels. For many prominent applications, particularly web search, this may not be germane to users. What matters is rather how many good results there are on the first page or the first three pages. This leads to measuring precision at fixed low levels of retrieved results, such as 10 or 30 documents. This is referred to as "Precision at k", for example "Precision at 10". It has the advantage of not requiring any estimate of the size of the set of relevant documents but the disadvantages that it is the least stable of the commonly used evaluation measures and that it does not average well, since the total number of relevant documents for a query has a strong influence on precision at k.

R-PRECISION

An alternative, which alleviates this problem, is *R-precision*. It requires having a set of known relevant documents *Rel*, from which we calculate the precision of the top *Rel* documents returned. (The set *Rel* may be incomplete, such as when *Rel* is formed by creating relevance judgments for the pooled top k results of particular systems in a set of experiments.) R-precision adjusts for the size of the set of relevant documents: A perfect system could score 1 on this metric for each query, whereas, even a perfect system could only achieve a precision at 20 of 0.4 if there were only 8 documents in the collection relevant to an information need. Averaging this measure across queries thus makes more sense. This measure is harder to explain to naive users than Precision at k but easier to explain than MAP. If there are $|Rel|$ relevant documents for a query, we examine the top $|Rel|$ results of a system, and find that r are relevant, then by definition, not only is the precision (and hence R-precision) $r/|Rel|$, but the recall of this result set is also $r/|Rel|$.

BREAK-EVEN POINT

Thus, R-precision turns out to be identical to the *break-even point*, another measure which is sometimes used, defined in terms of this equality relationship holding. Like Precision at k, R-precision describes only one point on the precision-recall curve, rather than attempting to summarize effectiveness across the curve, and it is somewhat unclear why you should be interested in the break-even point rather than either the best point on the curve (the point with maximal F-measure) or a retrieval level of interest to a particular application (Precision at k). Nevertheless, R-precision turns out to be highly correlated with MAP empirically, despite measuring only a single point on



► **Figure 8.4** The ROC curve corresponding to the precision-recall curve in Figure 8.2.

the curve.

ROC CURVE

Another concept sometimes used in evaluation is an *ROC curve*. ("ROC" stands for "Receiver Operating Characteristics", but knowing that doesn't help most people.) An ROC curve plots the true positive rate or sensitivity against the false positive rate or (1 − specificity). Here, *sensitivity* is just another term for recall. The false positive rate is given by $fp/(fp+tn)$. Figure 8.4 shows the ROC curve corresponding to the precision-recall curve in Figure 8.2. An ROC curve always goes from the bottom left to the top right of the graph. For a good system, the graph climbs steeply on the left side. For unranked result sets, *specificity*, given by $tn/(fp+tn)$, was not seen as a very useful notion. Because the set of true negatives is always so large, its value would be almost 1 for all information needs (and, correspondingly, the value of the false positive rate would be almost 0). That is, the "interesting" part of Figure 8.2 is $0 <$ recall < 0.4, a part which is compressed to a small corner of Figure 8.4. But an ROC curve could make sense when looking over the full retrieval spectrum, and it provides another way of looking at the data. In many fields, a common aggregate measure is to report the area under the ROC curve, which is the ROC analog of MAP. Precision-recall curves are sometimes loosely referred to as ROC curves. This is understandable, but not accurate.

SENSITIVITY

SPECIFICITY

A final approach that has seen increasing adoption, especially when employed with machine learning approaches to ranking (see Section 15.4, page 341) is measures of *cumulative gain*, and in particular *normalized discounted cumu-*

CUMULATIVE GAIN

NORMALIZED DISCOUNTED CUMULATIVE GAIN

Zkj - how to compute ? : Take the reference (train/test) data set, compute the sum in the right(= St) then Zkj = 1/ St. The idea is to have a score 1 for reference set !

NDCG

lative gain (*NDCG*). NDCG is designed for situations of non-binary notions of relevance (cf. Section 8.5.1). Like precision at k, it is evaluated over some number k of top search results. For a set of queries Q, let $R(j,d)$ be the relevance score assessors gave to document d for query j. Then,

(8.9)
$$\text{NDCG}(Q,k) = \frac{1}{|Q|}\sum_{j=1}^{|Q|} Z_{kj} \sum_{m=1}^{k} \frac{2^{R(j,m)}-1}{\log_2(1+m)},$$

where Z_{kj} is a normalization factor calculated to make it so that a perfect ranking's NDCG at k for query j is 1. For queries for which $k' < k$ documents are retrieved, the last summation is done up to k'.

?

Exercise 8.4 [⋆]

What are the possible values for interpolated precision at a recall level of 0?

Exercise 8.5 [⋆⋆]

Must there always be a break-even point between precision and recall? Either show there must be or give a counter-example.

Exercise 8.6 [⋆⋆]

What is the relationship between the value of F_1 and the break-even point?

Exercise 8.7 [⋆⋆]

DICE COEFFICIENT

The *Dice coefficient* of two sets is a measure of their intersection scaled by their size (giving a value in the range 0 to 1):

$$\text{Dice}(X,Y) = \frac{2|X \cap Y|}{|X| + |Y|}$$

Show that the balanced F-measure (F_1) is equal to the Dice coefficient of the retrieved and relevant document sets.

Exercise 8.8 [⋆]

Consider an information need for which there are 4 relevant documents in the collection. Contrast two systems run on this collection. Their top 10 results are judged for relevance as follows (the leftmost item is the top ranked search result):

System 1	R	N	R	N	N	N	N	N	R	R
System 2	N	R	N	N	R	R	R	N	N	N

a. What is the MAP of each system? Which has a higher MAP?

b. Does this result intuitively make sense? What does it say about what is important in getting a good MAP score?

c. What is the R-precision of each system? (Does it rank the systems the same as MAP?)

Exercise 8.9 [⋆⋆]

The following list of Rs and Ns represents relevant (R) and nonrelevant (N) returned documents in a ranked list of 20 documents retrieved in response to a query from a collection of 10,000 documents. The top of the ranked list (the document the system thinks is most likely to be relevant) is on the left of the list. This list shows 6 relevant documents. Assume that there are 8 relevant documents in total in the collection.

R R N N N N N N N R N R N N N N R N N N N R

a. What is the precision of the system on the top 20?
b. What is the F_1 on the top 20?
c. What is the uninterpolated precision of the system at 25% recall?
d. What is the interpolated precision at 33% recall?
e. Assume that these 20 documents are the complete result set of the system. What is the MAP for the query?

Assume, now, instead, that the system returned the entire 10,000 documents in a ranked list, and these are the first 20 results returned.

f. What is the largest possible MAP that this system could have?
g. What is the smallest possible MAP that this system could have?
h. In a set of experiments, only the top 20 results are evaluated by hand. The result in (e) is used to approximate the range (f)–(g). For this example, how large (in absolute terms) can the error for the MAP be by calculating (e) instead of (f) and (g) for this query?

8.5 Assessing relevance

To properly evaluate a system, your test information needs must be germane to the documents in the test document collection, and appropriate for predicted usage of the system. These information needs are best designed by domain experts. Using random combinations of query terms as an information need is generally not a good idea because typically they will not resemble the actual distribution of information needs.

Given information needs and documents, you need to collect relevance assessments. This is a time-consuming and expensive process involving human beings. For tiny collections like Cranfield, exhaustive judgments of relevance for each query and document pair were obtained. For large modern collections, it is usual for relevance to be assessed only for a subset of the documents for each query. The most standard approach is *pooling*, where relevance is assessed over a subset of the collection that is formed from the top k documents returned by a number of different IR systems (usually the ones to be evaluated), and perhaps other sources such as the results of Boolean keyword searches or documents found by expert searchers in an interactive process.

POOLING

		Judge 2 Relevance		
		Yes	No	Total
Judge 1 Relevance	Yes	300	20	320
	No	10	70	80
	Total	310	90	400

Observed proportion of the times the judges agreed
$P(A) = (300 + 70)/400 = 370/400 = 0.925$
Pooled marginals
$P(nonrelevant) = (80 + 90)/(400 + 400) = 170/800 = 0.2125$
$P(relevant) = (320 + 310)/(400 + 400) = 630/800 = 0.7878$
Probability that the two judges agreed by chance
$P(E) = P(nonrelevant)^2 + P(relevant)^2 = 0.2125^2 + 0.7878^2 = 0.665$
Kappa statistic
$\kappa = (P(A) - P(E))/(1 - P(E)) = (0.925 - 0.665)/(1 - 0.665) = 0.776$

► **Table 8.2** Calculating the kappa statistic.

A human is not a device that reliably reports a gold standard judgment of relevance of a document to a query. Rather, humans and their relevance judgments are quite idiosyncratic and variable. But this is not a problem to be solved: in the final analysis, the success of an IR system depends on how good it is at satisfying the needs of these idiosyncratic humans, one information need at a time.

Nevertheless, it is interesting to consider and measure how much agreement between judges there is on relevance judgments. In the social sciences, a common measure for agreement between judges is the *kappa statistic*. It is designed for categorical judgments and corrects a simple agreement rate for the rate of chance agreement.

KAPPA STATISTIC

$$kappa = \frac{P(A) - P(E)}{1 - P(E)} \tag{8.10}$$

where $P(A)$ is the proportion of the times the judges agreed, and $P(E)$ is the proportion of the times they would be expected to agree by chance. There are choices in how the latter is estimated: if we simply say we are making a two-class decision and assume nothing more, then the expected chance agreement rate is 0.5. However, normally the class distribution assigned is skewed, and it is usual to use *marginal* statistics to calculate expected agreement.[2] There are still two ways to do it depending on whether one pools

MARGINAL

2. For a contingency table, as in Table 8.2, a marginal statistic is formed by summing a row or column. The marginal $a_{i.k} = \sum_j a_{ijk}$.

the marginal distribution across judges or uses the marginals for each judge separately; both forms have been used, but we present the pooled version because it is more conservative in the presence of systematic differences in assessments across judges. The calculations are shown in Table 8.2. The kappa value will be 1 if two judges always agree, 0 if they agree only at the rate given by chance, and negative if they are worse than random. If there are more than two judges, it is normal to calculate an average pairwise kappa value. As a rule of thumb, a kappa value above 0.8 is taken as good agreement, a kappa value between 0.67 and 0.8 is taken as fair agreement, and agreement below 0.67 is seen as data providing a dubious basis for an evaluation, though the precise cutoffs depend on the purposes for which the data will be used.

Interjudge agreement of relevance has been measured within the TREC evaluations and for medical IR collections. Using the above rules of thumb, the level of agreement normally falls in the range of "fair" (0.67–0.8). The fact that human agreement on a binary relevance judgment is quite modest is one reason for not requiring more fine-grained relevance labeling from the test set creator. To answer the question of whether IR evaluation results are valid despite the variation of individual assessors' judgments, people have experimented with evaluations taking one or the other of two judges' opinions as the gold standard. The choice can make a considerable *absolute* difference to reported scores, but has in general been found to have little impact on the *relative* effectiveness ranking of either different systems or variants of a single system which are being compared for effectiveness.

8.5.1 Critiques and justifications of the concept of relevance

The advantage of system evaluation, as enabled by the standard model of relevant and nonrelevant documents, is that we have a fixed setting in which we can vary IR systems and system parameters to carry out comparative experiments. Such formal testing is much less expensive and allows clearer diagnosis of the effect of changing system parameters than doing user studies of retrieval effectiveness. Indeed, once we have a formal measure that we have confidence in, we can proceed to optimize effectiveness by machine learning methods, rather than tuning parameters by hand. Of course, if the formal measure poorly describes what users actually want, doing this will not be effective in improving user satisfaction. Our perspective is that, in practice, the standard formal measures for IR evaluation, although a simplification, are good enough, and recent work in optimizing formal evaluation measures in IR has succeeded brilliantly. There are numerous examples of techniques developed in formal evaluation settings, which improve effectiveness in operational settings, such as the development of document length normalization methods within the context of TREC (Sections 6.4.4 and 11.4.3)

and machine learning methods for adjusting parameter weights in scoring (Section 6.1.2).

That is not to say that there are not problems latent within the abstractions used. The relevance of one document is treated as independent of the relevance of other documents in the collection. (This assumption is actually built into most retrieval systems – documents are scored against queries, not against each other – as well as being assumed in the evaluation methods.) Assessments are binary: there aren't any nuanced assessments of relevance. Relevance of a document to an information need is treated as an absolute, objective decision. But judgments of relevance are subjective, varying across people, as we discussed above. In practice, human assessors are also imperfect measuring instruments, susceptible to failures of understanding and attention. We also have to assume that users' information needs do not change as they start looking at retrieval results. Any results based on one collection are heavily skewed by the choice of collection, queries, and relevance judgment set: the results may not translate from one domain to another or to a different user population.

Some of these problems may be fixable. A number of recent evaluations, including INEX, some TREC tracks, and NTCIR have adopted an ordinal notion of relevance with documents divided into 3 or 4 classes, distinguishing slightly relevant documents from highly relevant documents. See Section 10.4 (page 210) for a detailed discussion of how this is implemented in the INEX evaluations.

MARGINAL RELEVANCE

One clear problem with the relevance-based assessment that we have presented is the distinction between relevance and *marginal relevance*: whether a document still has distinctive usefulness after the user has looked at certain other documents (Carbonell and Goldstein 1998). Even if a document is highly relevant, its information can be completely redundant with other documents which have already been examined. The most extreme case of this is documents that are duplicates – a phenomenon that is actually very common on the World Wide Web – but it can also easily occur when several documents provide a similar precis of an event. In such circumstances, marginal relevance is clearly a better measure of utility to the user. Maximizing marginal relevance requires returning documents that exhibit diversity and novelty. One way to approach measuring this is by using distinct facts or entities as evaluation units. This perhaps more directly measures true utility to the user but doing this makes it harder to create a test collection.

? **Exercise 8.10** [⋆⋆]

Below is a table showing how two human judges rated the relevance of a set of 12 documents to a particular information need (0 = nonrelevant, 1 = relevant). Let us assume that you've written an IR system that for this query returns the set of documents {4, 5, 6, 7, 8}.

docID	Judge 1	Judge 2
1	0	0
2	0	0
3	1	1
4	1	1
5	1	0
6	1	0
7	1	0
8	1	0
9	0	1
10	0	1
11	0	1
12	0	1

a. Calculate the kappa measure between the two judges.

b. Calculate precision, recall, and F_1 of your system if a document is considered relevant only if the two judges agree.

c. Calculate precision, recall, and F_1 of your system if a document is considered relevant if either judge thinks it is relevant.

8.6 A broader perspective: System quality and user utility

Formal evaluation measures are at some distance from our ultimate interest in measures of human utility: how satisfied is each user with the results the system gives for each information need that they pose? The standard way to measure human satisfaction is by various kinds of user studies. These might include quantitative measures, both objective, such as time to complete a task, as well as subjective, such as a score for satisfaction with the search engine, and qualitative measures, such as user comments on the search interface. In this section we will touch on other system aspects that allow quantitative evaluation and the issue of user utility.

8.6.1 System issues

There are many practical benchmarks on which to rate an information retrieval system beyond its retrieval quality. These include:

- How fast does it index, that is, how many documents per hour does it index for a certain distribution over document lengths? (cf. Chapter 4)
- How fast does it search, that is, what is its latency as a function of index size?
- How expressive is its query language? How fast is it on complex queries?

- How large is its document collection, in terms of the number of documents or the collection having information distributed across a broad range of topics?

All these criteria apart from query language expressiveness are straightforwardly *measurable*: we can quantify the speed or size. Various kinds of feature checklists can make query language expressiveness semi-precise.

8.6.2 User utility

What we would really like is a way of quantifying aggregate user happiness, based on the relevance, speed, and user interface of a system. One part of this is understanding the distribution of people we wish to make happy, and this depends entirely on the setting. For a web search engine, happy search users are those who find what they want. One indirect measure of such users is that they tend to return to the same engine. Measuring the rate of return of users is thus an effective metric, which would of course be more effective if you could also measure how much these users used other search engines. But advertisers are also users of modern web search engines. They are happy if customers click through to their sites and then make purchases. On an eCommerce web site, a user is likely to be wanting to purchase something. Thus, we can measure the time to purchase, or the fraction of searchers who become buyers. On a shopfront web site, perhaps both the user's and the store owner's needs are satisfied if a purchase is made. Nevertheless, in general, we need to decide whether it is the end user's or the eCommerce site owner's happiness that we are trying to optimize. Usually, it is the store owner who is paying us.

For an "enterprise" (company, government, or academic) intranet search engine, the relevant metric is more likely to be user productivity: how much time do users spend looking for information that they need. There are also many other practical criteria concerning such matters as information security, which we mentioned in Section 4.6 (page 80).

User happiness is elusive to measure, and this is part of why the standard methodology uses the proxy of relevance of search results. The standard direct way to get at user satisfaction is to run user studies, where people engage in tasks, and usually various metrics are measured, the participants are observed, and ethnographic interview techniques are used to get qualitative information on satisfaction. User studies are very useful in system design, but they are time consuming and expensive to do. They are also difficult to do well, and expertise is required to design the studies and to interpret the results. We will not discuss the details of human usability testing here.

8.6.3 Refining a deployed system

If an IR system has been built and is being used by a large number of users, the system's builders can evaluate possible changes by deploying variant versions of the system and recording measures that are indicative of user satisfaction with one variant vs. others as they are being used. This method is frequently used by web search engines.

A/B TEST

The most common version of this is *A/B testing*, a term borrowed from the advertising industry. For such a test, precisely one thing is changed between the current system and a proposed system, and a small proportion of traffic (say, 1–10% of users) is randomly directed to the variant system, while most users use the current system. For example, if we wish to investigate a change to the ranking algorithm, we redirect a random sample of users to a variant system and evaluate measures such as the frequency with which people click on the top result, or any result on the first page. (This particular analysis method is referred to as *clickthrough log analysis* or *clickstream mining*. It is further discussed as a method of implicit feedback in Section 9.1.7 (page 187).)

CLICKTHROUGH LOG ANALYSIS
CLICKSTREAM MINING

The basis of A/B testing is running a bunch of single variable tests (either in sequence or in parallel): for each test only one parameter is varied from the control (the current live system). It is therefore easy to see whether varying each parameter has a positive or negative effect. Such testing of a live system can easily and cheaply gauge the effect of a change on users, and, with a large enough user base, it is practical to measure even very small positive and negative effects. In principle, more analytic power can be achieved by varying multiple things at once in an uncorrelated (random) way, and doing standard multivariate statistical analysis, such as multiple linear regression. In practice, though, A/B testing is widely used, because A/B tests are easy to deploy, easy to understand, and easy to explain to management.

8.7 Results snippets

Having chosen or ranked the documents matching a query, we wish to present a results list that will be informative to the user. In many cases the user will not want to examine all the returned documents and so we want to make the results list informative enough that the user can do a final ranking of the documents for themselves based on relevance to their information need.[3] The standard way of doing this is to provide a *snippet*, a short summary of the document, which is designed so as to allow the user to decide its relevance. Typically, the snippet consists of the document title and a short

SNIPPET

3. There are exceptions, in domains where recall is emphasized. For instance, in many legal disclosure cases, a legal associate will review *every* document that matches a keyword search.

summary, which is automatically extracted. The question is how to design the summary so as to maximize its usefulness to the user.

STATIC SUMMARY
DYNAMIC SUMMARY

The two basic kinds of summaries are *static*, which are always the same regardless of the query, and *dynamic* (or query-dependent), which are customized according to the user's information need as deduced from a query. Dynamic summaries attempt to explain why a particular document was retrieved for the query at hand.

A static summary is generally comprised of either or both a subset of the document and metadata associated with the document. The simplest form of summary takes the first two sentences or 50 words of a document, or extracts particular zones of a document, such as the title and author. Instead of zones of a document, the summary can instead use metadata associated with the document. This may be an alternative way to provide an author or date, or may include elements which are designed to give a summary, such as the `description` metadata which can appear in the `meta` element of a web HTML page. This summary is typically extracted and cached at indexing time, in such a way that it can be retrieved and presented quickly when displaying search results, whereas having to access the actual document content might be a relatively expensive operation.

TEXT SUMMARIZATION

There has been extensive work within natural language processing (NLP) on better ways to do *text summarization*. Most such work still aims only to choose sentences from the original document to present and concentrates on how to select good sentences. The models typically combine positional factors, favoring the first and last paragraphs of documents and the first and last sentences of paragraphs, with content factors, emphasizing sentences with key terms, which have low document frequency in the collection as a whole, but high frequency and good distribution across the particular document being returned. In sophisticated NLP approaches, the system synthesizes sentences for a summary, either by doing full text generation or by editing and perhaps combining sentences used in the document. For example, it might delete a relative clause or replace a pronoun with the noun phrase that it refers to. This last class of methods remains in the realm of research and is seldom used for search results: it is easier, safer, and often even better to just use sentences from the original document.

KEYWORD-IN-CONTEXT

Dynamic summaries display one or more "windows" on the document, aiming to present the pieces that have the most utility to the user in evaluating the document with respect to their information need. Usually these windows contain one or several of the query terms, and so are often referred to as *keyword-in-context* (KWIC) snippets, though sometimes they may still be pieces of the text such as the title that are selected for their query-independent information value just as in the case of static summarization. Dynamic summaries are generated in conjunction with scoring. If the query is found as a phrase, occurrences of the phrase in the document will be

... ***In recent years, Papua New Guinea has faced severe economic difficulties and*** economic growth has slowed, partly as a result of weak governance and civil war, and partly as a result of external factors such as the Bougainville civil war which led to the closure in 1989 of the Panguna mine (at that time the most important foreign exchange earner and contributor to Government finances), the Asian financial crisis, a decline in the prices of gold and copper, and a fall in the production of oil. ***PNG's economic development record over the past few years is evidence that*** governance issues underly many of the country's problems. Good governance, which may be defined as the transparent and accountable management of human, natural, economic and financial resources for the purposes of equitable and sustainable development, flows from proper public sector management, efficient fiscal and accounting mechanisms, and a willingness to make service delivery a priority in practice. ...

► **Figure 8.5** An example of selecting text for a dynamic snippet. This snippet was generated for a document in response to the query new guinea economic development. The figure shows in bold italic where the selected snippet text occurred in the original document.

shown as the summary. If not, windows within the document that contain multiple query terms will be selected. Commonly these windows may just stretch some number of words to the left and right of the query terms. This is a place where NLP techniques can usefully be employed: users prefer snippets that read well because they contain complete phrases.

Dynamic summaries are generally regarded as greatly improving the usability of IR systems, but they present a complication for IR system design. A dynamic summary cannot be precomputed, but, on the other hand, if a system has only a positional index, then it cannot easily reconstruct the context surrounding search engine hits in order to generate such a dynamic summary. This is one reason for using static summaries. The standard solution to this in a world of large and cheap disk drives is to locally cache all the documents at index time (notwithstanding that this approach raises various legal, information security and control issues that are far from resolved) as shown in Figure 7.5 (page 147). Then, a system can simply scan a document which is about to appear in a displayed results list to find snippets containing the query words. Beyond simply access to the text, producing a good KWIC snippet requires some care. Given a variety of keyword occurrences in a document, the goal is to choose fragments which are: (i) maximally informative about the discussion of those terms in the document, (ii) self-contained enough to be easy to read, and (iii) short enough to fit within the normally strict constraints on the space available for summaries.

Generating snippets must be fast since the system is typically generating many snippets for each query that it handles. Rather than caching an entire document, it is common to cache only a generous but fixed size prefix of the document, such as perhaps 10,000 characters. For most common, short documents, the entire document is thus cached, but huge amounts of local storage will not be wasted on potentially vast documents. Summaries of documents whose length exceeds the prefix size will be based on material in the prefix only, which is in general a useful zone in which to look for a document summary anyway.

If a document has been updated since it was last processed by a crawler and indexer, these changes will be neither in the cache nor in the index. In these circumstances, neither the index nor the summary will accurately reflect the current contents of the document, but it is the differences between the summary and the actual document content that will be more glaringly obvious to the end user.

8.8 References and further reading

Definition and implementation of the notion of relevance to a query got off to a rocky start in 1953. Swanson (1988) reports that in an evaluation in that year between two teams, they agreed that 1390 documents were variously relevant to a set of 98 questions, but disagreed on a further 1577 documents, and the disagreements were never resolved.

Rigorous formal testing of IR systems was first completed in the Cranfield experiments, beginning in the late 1950s. A retrospective discussion of the Cranfield test collection and experimentation with it can be found in (Cleverdon 1991). The other seminal series of early IR experiments were those on the SMART system by Gerard Salton and colleagues (Salton 1971b; 1991). The TREC evaluations are described in detail by Voorhees and Harman (2005). Online information is available at http://trec.nist.gov/. Initially, few researchers computed the statistical significance of their experimental results, but the IR community increasingly demands this (Hull 1993). User studies of IR system effectiveness began more recently (Saracevic and Kantor 1988; 1996).

F MEASURE

The notions of recall and precision were first used by Kent et al. (1955), although the term *precision* did not appear until later. The F measure (or, rather its complement $E = 1 - F$) was introduced by van Rijsbergen (1979). He provides an extensive theoretical discussion, which shows how adopting a principle of decreasing marginal relevance (at some point a user will be unwilling to sacrifice a unit of precision for an added unit of recall) leads to the harmonic mean being the appropriate method for combining precision and recall (and hence to its adoption rather than the minimum or geometric mean).

R-PRECISION

Buckley and Voorhees (2000) compare several evaluation measures, including precision at k, MAP, and R-precision, and evaluate the error rate of each measure. R-precision was adopted as the official evaluation metric in the TREC HARD track (Allan 2005). Aslam and Yilmaz (2005) examine its surprisingly close correlation to MAP, which had been noted in earlier studies (Tague-Sutcliffe and Blustein 1995, Buckley and Voorhees 2000). A standard program for evaluating IR systems which computes many measures of ranked retrieval effectiveness is Chris Buckley's `trec_eval` program used in the TREC evaluations. It can be downloaded from: http://trec.nist.gov/trec_eval/.

Kekäläinen and Järvelin (2002) argue for the superiority of graded relevance judgments when dealing with very large document collections, and Järvelin and Kekäläinen (2002) introduce cumulated gain-based methods for IR system evaluation in this context. Sakai (2007) does a study of the stability and sensitivity of evaluation measures based on graded relevance judgments from NTCIR tasks, and concludes that NDCG is best for evaluating document ranking.

Schamber et al. (1990) examine the concept of relevance, stressing its multidimensional and context-specific nature, but also arguing that it can be measured effectively. (Voorhees 2000) is the standard article for examining variation in relevance judgments and their effects on retrieval system scores and ranking for the TREC Ad Hoc task. Voorhees concludes that although the numbers change, the rankings are quite stable. Hersh et al. (1994) present similar analysis for a medical IR collection. In contrast, Kekäläinen (2005) analyze some of the later TRECs, exploring a 4-way relevance judgment and the notion of cumulative gain, arguing that the relevance measure used does substantially affect system rankings. See also Harter (1998). Zobel (1998) studies whether the pooling method used by TREC to collect a subset of documents that will be evaluated for relevance is reliable and fair, and concludes that it is.

KAPPA STATISTIC

The kappa statistic and its use for language-related purposes is discussed by Carletta (1996). Many standard sources (e.g., Siegel and Castellan 1988) present pooled calculation of the expected agreement, but Di Eugenio and Glass (2004) argue for preferring the unpooled agreement (though perhaps presenting multiple measures). For further discussion of alternative measures of agreement, which may in fact be better, see Lombard et al. (2002) and Krippendorff (2003).

Text summarization has been actively explored for many years. Modern work on sentence selection was initiated by Kupiec et al. (1995). More recent work includes (Barzilay and Elhadad 1997) and (Jing 2000), together with a broad selection of work appearing at the yearly DUC conferences and at other NLP venues. Tombros and Sanderson (1998) demonstrate the advantages of dynamic summaries in the IR context. Turpin et al. (2007) address how to generate snippets efficiently.

Clickthrough log analysis is studied in (Joachims 2002b, Joachims et al. 2005).

In a series of papers, Hersh, Turpin and colleagues show how improvements in formal retrieval effectiveness, as evaluated in batch experiments, do not always translate into an improved system for users (Hersh et al. 2000a;b; 2001, Turpin and Hersh 2001; 2002).

User interfaces for IR and human factors such as models of human information seeking and usability testing are outside the scope of what we cover in this book. More information on these topics can be found in other textbooks, including (Baeza-Yates and Ribeiro-Neto 1999, ch. 10) and (Korfhage 1997), and collections focused on cognitive aspects (Spink and Cole 2005).