Information Retrieval

Evaluation

Dr. Seán Russell

School of Computer Science,
University College Dublin

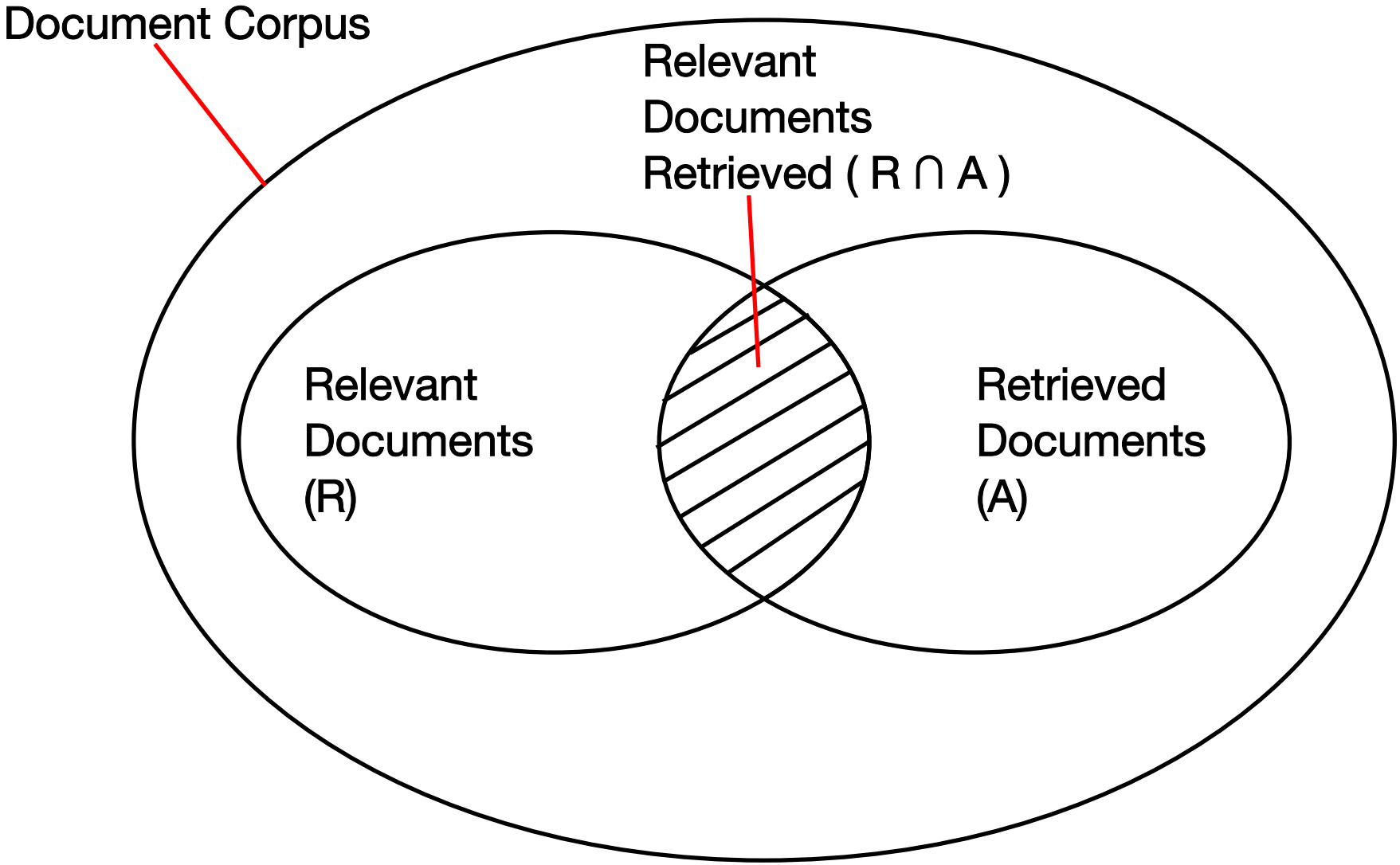
Week 7

Table of Contents

- Below are the commonly used metrics when evaluating IR systems
  - Precision/Recall
  - Precision @ n/R-precision
  - Mean Average Precision (MAP)
  - bPref
  - NDCG

Section Contents

- Precision and Recall are the two most basic (and most widely used) evaluation metrics in IR, upon which many others are based
- Precision is the fraction of the set of retrieved documents (Ret) that are relevant (i.e. that are also in Rel)
- Recall is the fraction of the relevant documents (Rel) that have been retrieved (i.e. they are also in Ret).



$$\text{Precision} = \frac{|R \cap A|}{|A|}$$

$$\text{Recall} = \frac{|R \cap A|}{|R|}$$

- Rel $= \{d_3, d_5, d_9, d_{25}, d_{39}, d_{44}, d_{56}, d_{71}, d_{89}, d_{123}\}$
- Ret:

1. d_{123}
2. d_{84}
3. d_{56}
4. d_6
5. d_8
6. d_9
7. d_{511}
8. d_{129}
9. d_{187}
10. d_{25}
11. d_{38}
12. d_{48}
13. d_{250}
14. d_{113}
15. d_3

- Number of relevant documents: $|Rel| = 10$
- Number of retrieved documents: $|Ret| = 15$
- Number of relevant documents retrieved: $|Rel \cap Ret| = 5$
- Precision $= \frac{|Rel \cap Ret|}{|Ret|} = \frac{5}{15} = 0.33$
- Recall $= \frac{|Rel \cap Ret|}{|Rel|} = \frac{5}{10} = 0.5$

- The two metrics of precision and recall are often **inversely** related: as one increases the other decreases
- Precision will be high whenever a system is good at avoiding non-relevant documents
  - A system can achieve very high precision by retrieving very few documents
- Recall will be high whenever a system finds many relevant documents
  - A system can achieve 100% recall simply by retrieving all the documents in the collection

- Which is most important?
  - That is task-dependent!
- Ideally every IR system would have both recall and precision of 100% (the answer set is equal to the relevant set)

- Users searching the web want **high precision** (i.e. they want the returned documents to be relevant)
  - Because of the size of the web, there are very many documents that will help satisfy the information need, so the user does not need to examine all of them
  - Instead, the user wishes to avoid wasting time looking at non-relevant documents

- A patent lawyer researching a patent must ensure that they get all of the relevant documents and hence they want **high recall**
  - If any relevant documents are missed, this may have serious consequences, so it is essential that all relevant documents are returned
  - They will tolerate lower precision to facilitate this (i.e. they are more likely to be willing to read through some non-relevant documents)

- For older, smaller document collections, the calculation of precision and recall was easy
- Nowadays, because document collections are so huge, it is very difficult to identify all the relevant documents for every query
- This makes it often impossible to calculate recall accurately
- Thus, precision tends to be preferred, in addition to other metrics based on precision
  - Note: precision can be calculated without needing to know all of the relevant documents, only how many of the returned documents are relevant

- Basic Precision is a set-based, unranked retrieval metric (as is recall)
- It is a single-value metric based on the entire list of results that was returned by an IR system
- It does not assess the way the results are ranked
- Most IR systems return ranked lists, not sets of documents, so users are most likely to look at the top of the list first

Rank	IR 1	IR 2
1	N	R
2	N	R
3	N	N
4	R	N
5	R	N

- N is non-relevant, R is relevant
- Precision for both systems is 40%
- But clearly IR system 2 is better!

Section Contents

- The rest of the metrics that we will look at use only a single metric to calculate performance
- These metrics calculate a score for a set of results for just one query
- Of course, when evaluating an IR system, we should use several queries for evaluation
- In this situation, we must first calculate the score for each query, and then get the average over all the queries

Section Contents

- When users make use of an IR system, it pays to consider what they actually want
- Particularly for web search systems where people typically look no further than the first few results
- For example, we may be interested in the precision after 10 documents have been retrieved
  - Known as "Precision at 10", or "@10"
- This is a measure of the quality of the results that a user is likely to look at

- $P@n = \frac{|Relevant\ \ in\ \ top\ \ n\ \ results|}{n}$

1. d_{123}
2. d_{84}
3. d_{56}
4. d_{6}
5. d_{8}
6. d_{9}
7. d_{511}
8. d_{129}
9. d_{187}
10. d_{25}
11. d_{38}
12. d_{48}
13. d_{250}
14. d_{113}
15. d_{3}

- $P@3 = \frac{2}{3} = 0.67$
- $P@6 = \frac{3}{6} = 0.5$
- $P@10 = \frac{4}{10} = 0.4$

- One problem with using P@n is that its performance is affected by the number of relevant documents available
- For a query that has only 10 relevant documents, a good P@10 score is difficult
- If there are 1,000 relevant documents, a good P@10 score is easy

Section Contents

- A similar metric is R-precision, where we are interested in the precision after the top R documents are returned
  - where R is the number of relevant documents for that query
- This is similar to P@n except that n is not fixed for all queries
  - because the number of relevant documents will be different for each query

- $R - Precision = \frac{|Relevant\ \ in\ \ top\ \ r\ \ results|}{r}$
  - where r is the number of relevant documents that are available for this query

1 d_{123}	4 d_6	7 d_{511}	10 d_{25}	13 d_{250}
2 d_{84}	5 d_8	8 d_{129}	11 d_{38}	14 d_{113}
3 d_{56}	6 d_9	9 d_{187}	12 d_{48}	15 d_3

- $R - Precision = \frac{4}{10} = 0.4$

- $\text{Rel} = \{d_3, d_{56}, d_{129}\}$

1 d_{123}	4 d_6	7 d_{511}	10 d_{25}	13 d_{250}
2 d_{84}	5 d_8	8 d_{129}	11 d_{38}	14 d_{113}
3 d_{56}	6 d_9	9 d_{187}	12 d_{48}	15 d_3

- $R - Precision = \frac{1}{3} = 0.33$
  - There are only 3 relevant documents
  - There is only one relevant document in the first 3 results

Section Contents

- Mean Average Precision (MAP) has for many years been the most commonly used metric in IR literature to evaluate the performance of systems
- It is a single-value metric based on precision
- Unlike simple precision, it rewards systems that rank relevant documents at the beginning of the results returned
- Unlike P@10, it continues to examine the later stages of the ranked list, although with lesser weight

- Calculating Mean Average Precision
  - Firstly, we must calculate the precision at each **recall point** (at each rank where a relevant document is found).
  - The **Average Precision** for this query is found by dividing the sum of these precision calculations by the total number of relevant documents
  - As with most metrics, for multiple queries the same procedure must be performed for each. We must calculate the mean of the queries' average precision values, giving us **Mean Average Precision**.

- Average Precision$= \frac{\sum_{k=1}^{n} P@k \times rel@k}{r}$
  - n is the number of retrieved documents
  - P@k is precision at rank k
  - rel@k is 1 if the document at rank k is relevant, or 0 if not

1. d_{123}
2. d_{84}
3. d_{56}
4. d_{6}
5. d_{8}
6. d_{9}
7. d_{511}
8. d_{129}
9. d_{187}
10. d_{25}
11. d_{38}
12. d_{48}
13. d_{250}
14. d_{113}
15. d_{3}

1. $P@1 = 1.00$
2. $P@3 = 0.67$
3. $P@6 = 0.50$
4. $P@10 = 0.40$
5. $P@15 = 0.33$

- Average Precision (for this query):
$\frac{1.0+0.67+0.5+0.4+0.33}{10} = 0.29$

- Average Precision (for this query): $\frac{1.0+0.67+0.5+0.4+0.33}{10} = 0.29$
- This is the AP for one query
- To get the Mean Average Precision (MAP), we calculate the average over all queries

Section Contents

- All of the evaluation techniques we have mentioned so far are based on the Cranfield Paradigm
- In this, test collections and queries are created that have a known set of relevant documents associated with them
- The point is that for each query, every document in the collection is judged to be "relevant" or "non-relevant"
- Where this occurs, we say that we have "complete relevance judgments"

- With smaller collections, this Cranfield Paradigm is perfect
  - i.e. complete relevance judgments exist
- However, as document collections have become larger, complete judgments have become less common and we have **incomplete judgments**
- This means that some documents have not been judged so they may or may not be relevant to test queries
- With large-scale IR collections (such as those based on the web), this complete judgment is impossible to achieve, with potentially billions of documents to be judged for relevance against hundreds of queries

- When a corpus is very large, it is not feasible to judge the relevance of every document to every query
- The Text REtrieval Conference (TREC) sets IR tasks each year (called "tracks")
- For ad hoc retrieval tasks (the type of IR we are learning about), they release a corpus and some standard queries (called topics)
- Participating groups are invited to use their systems to search the corpus for each topic, and submit a set of results to the organisers
- The organisers use a system called pooling to decide which documents to judge

- In its basic form, the top n documents returned in each set of results for every query are gathered into a pool
- Human assessors then judge the relevance of only the documents in the pool for each query
- This is based on the assumption that different groups using a variety of state-of-the-art retrieval techniques should, collectively, find all the relevant documents that were available in the corpus
- In practice, this is not guaranteed – there may still be relevant documents that were available, but that no system actually retrieved in response to the query

Section Contents

- For the evaluation metrics we have seen so far, they are simplified by **assuming** that unjudged documents are non-relevant
- It was noticed that many unjudged documents had the effect of lowering evaluation score
  - **NOTE:** This does not mean that the retrieval was worse: whether a document is relevant or not is not affected by whether somebody has judged it
- This is a problem because evaluation scores no longer accurately reflect the effectiveness of retrieval
- For example, what would the effect on the Average Precision score be in our example of nobody had judged document d_9?

Rank	Document	d_9 judged	d_9 unjudged
1	d_{123}	$P = \frac{1}{1} = 1.00$	$P = \frac{1}{1} = 1.00$
2	d_{84}		
3	d_{56}	$P = \frac{2}{3} = 0.67$	$P = \frac{2}{3} = 0.67$
4	d_6		
5	d_8		
6	d_9	$P = \frac{3}{6} = 0.5$	
7	d_{511}		
8	d_{129}		
9	d_{187}		
10	d_{25}	$P = \frac{4}{10} = 0.4$	$P = \frac{3}{10} = 0.3$
11	d_{38}		
12	d_{48}		
13	d_{250}		
14	d_{113}		
15	d_3	$P = \frac{5}{15} = 0.33$	$P = \frac{4}{15} = 0.27$

	d_9 judged	d_9 unjudged
Average Precision	$\frac{1+0.67+0.5+0.4+0.33}{10} = 0.29$	$\frac{1+0.67+0.3+0.27}{9} = 0.25$

- The idea behind bpref is that these unjudged documents should not impact so largely on the evaluation score
  - From the paper: Buckley & Voorhees, "Retrieval Evaluation with Incomplete Information", SIGIR 2004
- bpref ignores documents that have not been judged
- The only documents considered are those that were judged relevant or judged non-relevant
- This is an accepted metric for large-scale IR systems where complete relevance judgments are impossible to achieve

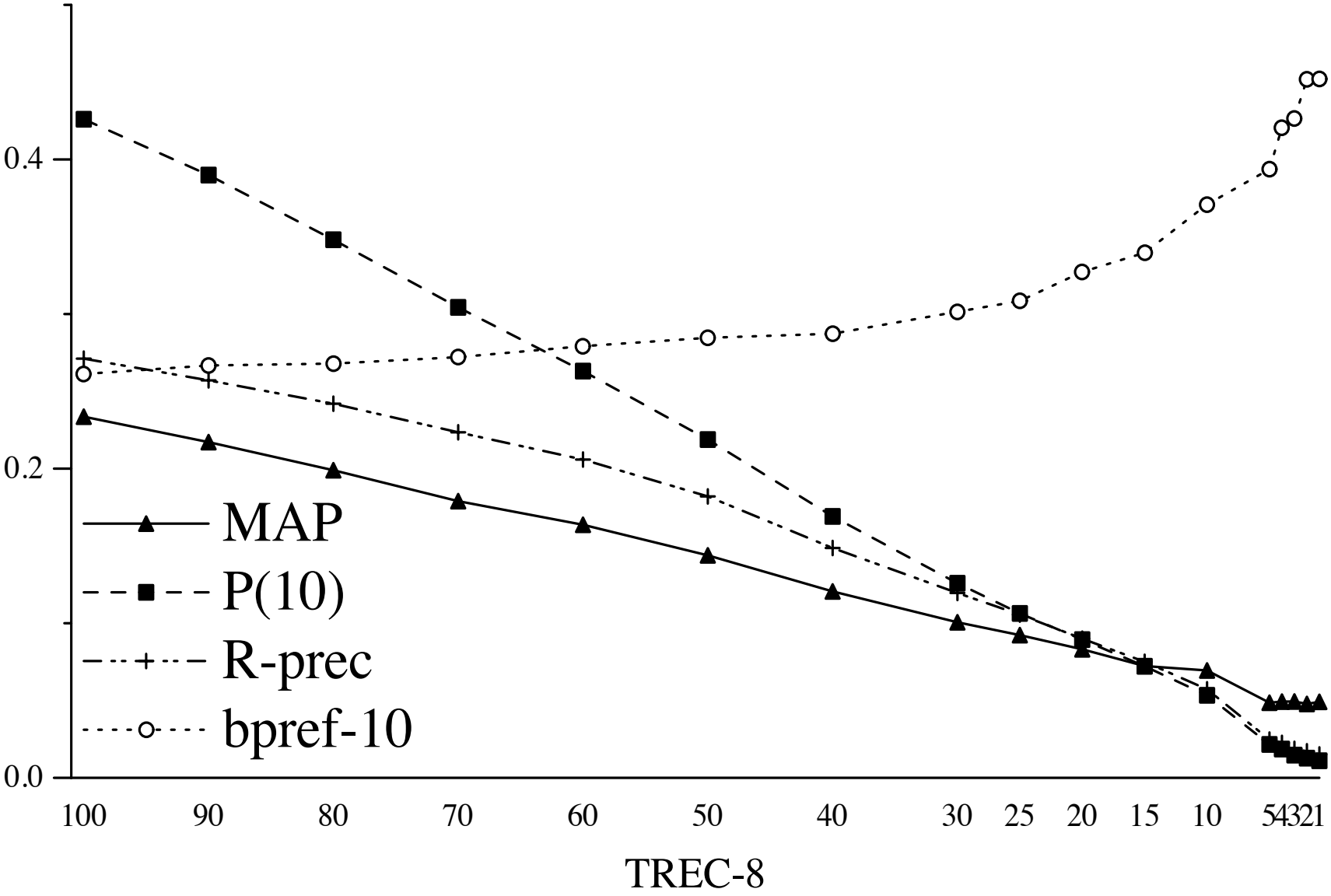
- bpref, for a query with R relevant documents is calculated as
  - $B = \frac{1}{R} \sum_{r \in R} 1 - \frac{|n \ ranked \ higher \ than \ r|}{R}$
    - Where n is a member of the first R judged non-relevant documents
- In other words:
  - For each relevant document in that was retrieved in the answer set:
    - Count the number of non-relevant documents above it in the result set (this is $|n \ ranked \ higher \ than \ r|$)
    - This cannot be greater than the total number of relevant documents (R)
    - The score for that document is $1 - \frac{|n \ ranked \ higher \ than \ r|}{R}$
  - Average this score over all the judged relevant documents

- Key:
  - R: Relevant
  - N: Non-Relevant
  - U: Unjudged
- Assume a total of 4 available judged relevant documents (R=4)
- bpref = (0.75+0.75+0+0)/4 = 0.375

Document	bPref Contribution
N	
R	$1 - \frac{1}{4} = 0.75$
U	
R	$1 - \frac{1}{4} = 0.75$
U	
N	
N	
N	
R	$1 - \frac{4}{4} = 0$
N	
R	$1 - \frac{4}{4} = 0$

- This works well in most situations
- However, when R is very small (i.e. there are only one or two relevant documents) it fails
- To overcome this problem, we can instead use bpref-10, which is given by
  - $bpref\,10 = \frac{1}{R}\sum_{r \in R} 1 - \frac{|n \ \ ranked \ \ higher \ \ than \ \ r|}{10+R}$
    - where n is a member of the first 10+R judged non-relevant documents

- From Buckley and Voorhees 2004 it can be seen that when complete judgments are available, there is no noticeable difference between MAP and bpref-10
- As the judgments become less complete, bpref is more stable than the others
  - y-axis: evaluation metric score
  - x-axis: percentage of judgments used (they kept removing judged documents and re-evaluating)


0.4
0.2
0.0
MAP
P(10)
R-prec
bpref-10
100
90
80
70
60
50
40
30
25
20
15
10
54321
TREC-8

Section Contents

- The metrics so far make use of **binary** relevance judgments
  - Documents are judged to be **relevant** or **non-relevant**
  - Any document that helps to satisfy an information need in any way is considered to be **relevant**
- **BUT**: In reality, some documents are more relevant than others
  - Normalised Discounted Cumulated Gain (NDCG) supports graded relevance judgments

- NDCG rewards systems that
  - Rank highly-relevant documents ahead of mildly relevant ones
  - Position relevant documents in early positions in the ranking

- The first thing that is required is a set of graded relevance judgments
  - Let us assume that a 0-3 scale where 3 is a highly relevant document and 0 is a non-relevant document
- $R = \{[d_3, 3], [d_5, 3], [d_9, 3], [d_{25}, 2], [d_{39}, 2], [d_{44}, 2], [d_{56}, 1], [d_{71}, 1], [d_{89}, 1], [d_{123}, 1]\}$
  - d_3, d_5 and d_9 are highly relevant documents
  - d_{25}, d_{39}, and d_{44} are relevant documents
  - d_{56}, d_{71}, d_{89}, and d_{123} are somewhat relevant documents
  - Everything else is non-relevant.

- Let's revisit the example from before
- This time, relevant documents are also shown with the degree of relevance attached
- We create a gain vector that records the relevance level at each rank
- G = (1, 0, 1, 0, 0, 3, 0, 0, 0, 2, 0, 0, 0, 0, 3)
- This is the starting point for our calculations

Rank	Doc	G
1	d_{123} **(r: 1)**	1
2	d_{84}	0
3	d_{56} **(r: 1)**	1
4	d_6	0
5	d_8	0
6	d_9 **(r: 3)**	3
7	d_{511}	0
8	d_{129}	0
9	d_{187}	0
10	d_{25} **(r : 2)**	2
11	d_{38}	0
12	d_{48}	0
13	d_{250}	0
14	d_{113}	0
15	d_3 **(r: 3)**	3

- Next, we calculated a cumulated gain vector (also sometimes called a "cumulative gain vector").
- Each rank i has its own CG value

$$CG[i] = \begin{cases} G[1] & i = 1 \\ G[i] + CG[i-1] & i < 1 \end{cases}$$

- e.g. at rank 10:
  - $CG[10] = G[10] + CG[9] = 2 + 5 = 7$

Rank	Doc	G	CG
1	d_{123} **(r: 1)**	1	1
2	d_{84}	0	1
3	d_{56} **(r: 1)**	1	2
4	d_6	0	2
5	d_8	0	2
6	d_9 **(r: 3)**	3	5
7	d_{511}	0	5
8	d_{129}	0	5
9	d_{187}	0	5
10	d_{25} **(r : 2)**	2	7
11	d_{38}	0	7
12	d_{48}	0	7
13	d_{250}	0	7
14	d_{113}	0	7
15	d_3 **(r: 3)**	3	10

- The idea is that each relevant document we find should add to the gain (i.e. the overall usefulness of the list)
- More highly relevant documents add more to the gain than less relevant documents
- BUT: documents late in the list can add as much to the gain as documents early in the list
  - e.g. at rank 15, a highly-relevant document adds 3 to the cumulated gain: the same as at rank 6

- This motivates us to create a vector for Discounted Cumulated Gain (DCG)
- Here, later documents add less to the gain than earlier ones
- It is calculated in the same way as CG, except that the gain at each rank is discounted by dividing it by the log of the rank (except for the first rank, which is unaffected).
- Each rank i has its own DCG value

$$DCG[i] = \begin{cases} G[1] & i = 1 \\ \frac{G[i]}{log_2(i)} + CG[i-1] & i < 1 \end{cases}$$

Rank	Doc	G	CG	Calculation	DCG
1	d_{123} **(r: 1)**	1	1	1	1
2	d_{84}	0	1		1
3	d_{56} **(r: 1)**	1	2	$\frac{1}{log_2(3)} + 1$	1.6
4	d_6	0	2		1.6
5	d_8	0	2		1.6
6	d_9 **(r: 3)**	3	5	$\frac{3}{log_2(6)} + 1.6$	2.8
7	d_{511}	0	5		2.8
8	d_{129}	0	5		2.8
9	d_{187}	0	5		2.8
10	d_{25} **(r : 2)**	2	7	$\frac{2}{log_2(10)} + 2.8$	3.4
11	d_{38}	0	7		3.4
12	d_{48}	0	7		3.4
13	d_{250}	0	7		3.4
14	d_{113}	0	7		3.4
15	d_3 **(r: 3)**	3	10	$\frac{3}{log_2(15)} + 3.4$	4.2

DCG=(1.0, 1.0, 1.6, 1.6, 1.6, 2.8, 2.8, 2.8, 2.8, 3.4, 3.4, 3.4, 3.4, 3.4, 4.2)

- We have now calculated a Discounted Cumulated Gain vector for a query:
  DCG=(1.0, 1.0, 1.6, 1.6, 1.6, 2.8, 2.8, 2.8, 2.8, 3.4, 3.4, 3.4, 3.4, 3.4, 4.2)
- This shows how finding relevant documents increases the quality of the results to that point
  - More relevant documents contribute more to gain
  - Relevant documents found earlier in the ranked list also contribute more to gain

- We have now calculated a Discounted Cumulated Gain vector for a query:

  DCG=(1.0, 1.0, 1.6, 1.6, 1.6, 2.8, 2.8, 2.8, 2.8, 3.4, 3.4, 3.4, 3.4, 3.4, 4.2)

- BUT: By itself, this is not easy to compare with others
  - Is 4.2 a good score for this query?
  - Which figure(s) do we choose to compare?
- Most evaluation metrics give a single value that is in the range between 0 and 1.
- **Normalised** DCG allows us to achieve this

- Normalised Discounted Cumulated Gain is calculated by comparing the DCG vector against an Ideal DCG vector
- The Ideal DCG vector is the DCG vector that we would see if the IR system had perfect retrieval
  - i.e. it begins with all the documents of relevance level 3
  - then it includes all the documents of relevance level 2
  - then it includes all the documents of relevance level 1
- We calculate an Ideal DCG vector to be the same length as the DCG vector (with 0 relevance values inserted at the end if there are not enough relevant documents)

Rank	IG	Calculation	IDCG
1	3	3	3
2	3	$\frac{3}{log_2(2)} + 3.0$	6.0
3	3	$\frac{3}{log_2(3)} + 6.0$	7.9
4	2	$\frac{2}{log_2(4)} + 7.9$	8.9
5	2	$\frac{2}{log_2(5)} + 8.9$	9.8
6	2	$\frac{2}{log_2(6)} + 9.8$	10.5
7	1	$\frac{1}{log_2(7)} + 10.5$	10.9
8	1	$\frac{1}{log_2(8)} + 10.9$	11.2
9	1	$\frac{1}{log_2(9)} + 11.2$	11.5
10	1	$\frac{1}{log_2(10)} + 11.5$	11.8
11	0		11.8
12	0		11.8
13	0		11.8
14	0		11.8
15	0		11.8

- The IDCG vector represents the best possible DCG scores that a perfect IR system would achieve
- IDCG=(3.0, 6.0, 7.9, 8.9, 9.8, 10.5, 10.9, 11.2, 11.5, 11.8, 11.8, 11.8, 11.8, 11.8, 11.8)
- We can normalise the DCG by dividing the score that was actually achieved at each rank by the ideal score, to yield a score between 0 and 1

Rank	1	2	3	4	5	6	7	8	9	10
DCG	1.0	1.0	1.6	1.6	1.6	2.8	2.8	2.8	2.8	3.4
IDGC	3.0	6.0	7.9	8.9	9.8	10.5	10.9	11.2	11.5	11.8
NDGC	0.33	0.17	0.20	0.18	0.16	0.27	0.26	0.25	0.24	0.29

Rank	11	12	13	14	15
DCG	3.4	3.4	3.4	3.4	4.2
IDGC	11.8	11.8	11.8	11.8	11.8
NDGC	0.29	0.29	0.29	0.29	0.36

- We still have the problem that we have 15 different scores for the evaluation
- This is solved in a similar way to Precision@n: we choose a rank to measure NDCG at
- NDCG@10 is a very commonly used metric: here it is 0.29

- Combines document ranks and graded relevance judgments
- Single measure of quality at any rank, without needing to know recall
- NDCG@n only considers relevant documents found to that point: not affected by many relevant documents being found very late
- Gives less weight to relevant documents found late in the ranking

Section Contents

- We have looked at numerous different metrics for IR evaluation
- All have their advantages and disadvantages
- We need to use an appropriate metric in order to evaluate an IR system's performance
- How "performance" is defined is dependent on the final use of the system:
  - web search;
  - intranet search;
  - research environment;
  - desktop search;
  - legal search;
  - etc...

- A general rule is that if complete judgments are available any metric can be used
- For each of these, a single value is normally reported, which is the average value from all the queries used in the evaluation
- MAP gives a good indication of performance within a single metric as it is averaging the results over multiple queries
- For collections that do not have complete judgments, bPref is a more suitable metric

- For tasks such as web search (due to user behaviour), metrics like P@10 might be used
- If graded relevance judgments are available, NDCG is preferred: this has continually gained in popularity in the last few years
- In reality, most evaluations use multiple metrics