CS 572: Information Retrieval

Lecture 6: BM25+Evaluation

Acknowledgments: Some slides in this lecture were adapted from Manning (Stanford) and Lin (Maryland)
Recap: tf-idf weighting

- The tf-idf weight of a term is the product of its tf weight and its idf weight.

\[ w_{t,d} = (1 + \log \text{tf}_{t,d}) \times \log_{10}(N / \text{df}_t) \]

- Best known weighting scheme in information retrieval
- Increases with the number of occurrences within a document
- Increases with the rarity of the term in the collection
tf-idf weighting has many variants

<table>
<thead>
<tr>
<th>Term frequency</th>
<th>Document frequency</th>
<th>Normalization</th>
</tr>
</thead>
<tbody>
<tr>
<td>n (natural)</td>
<td>tf_{t,d}</td>
<td>n (no)</td>
</tr>
<tr>
<td>l (logarithm)</td>
<td>1 + log(tf_{t,d})</td>
<td>t (idf)</td>
</tr>
<tr>
<td>a (augmented)</td>
<td>0.5 + \frac{0.5 \times tf_{t,d}}{\max_t(tf_{t,d})}</td>
<td>p (prob idf)</td>
</tr>
<tr>
<td>b (boolean)</td>
<td>\begin{cases} 1 &amp; \text{if } tf_{t,d} &gt; 0 \ 0 &amp; \text{otherwise} \end{cases}</td>
<td>\max{0, \log \frac{N - df_t}{df_t}}</td>
</tr>
<tr>
<td>L (log ave)</td>
<td>\frac{1 + \log(tf_{t,d})}{1 + \log(\text{ave}<em>{t \subseteq d}(tf</em>{t,d}))}</td>
<td>\text{(prob unique)}</td>
</tr>
</tbody>
</table>

Columns headed ‘n’ are acronyms for weight schemes.

Why is the base of the log in idf immaterial?
Weighting may differ in queries vs documents

• Many search engines allow for different weightings for queries vs. documents

SMART Notation: denotes the combination in use in an engine, with the notation $ddd\cdot qqq$, using the acronyms from the previous table

• A very standard weighting scheme is: Lnc.Ltc
  – Document: logarithmic $tf$ ($L$ as first character), no idf and yes cosine normalization
  – Query: logarithmic $tf$ ($L$ in leftmost column), $idf$ ($t$ in second column), cosine normalization ...
Problem: Diverse documents

- Ranking books vs. emails vs. news briefs
  - Lengths can be x 1000 of each other
- Need to normalize tf by document length
- How?
- BM (Basic Model): 1...25.
Probability Ranking Principle

- Ranking documents in decreasing order of probability of relevance to the user who submitted the query,
  - \( P( \text{relevant} \mid \text{document} ) \)

- where probabilities are estimated
  - \( P_{\text{est}}(\text{relevant} \mid \text{document} ) \Rightarrow P_{\text{true}}(\text{relevant} \mid \text{document} ) \)

- using all available evidence,
  - \( P_{\text{est}}(\text{relevant} \mid \text{document, query, context, user profile, … } ) \)

- produces the best possible effectiveness
  - So just need estimated probability!
Probability of Relevance

- Probability of an event is the sum of the probabilities of the sample points associated with the event
  - Sample points represent the possible outcomes of a statistical “experiment”
- For a retrieval model, the event space is $Q \times D$, where each sample point is a query-document pair and has an associated relevance judgment
- Because $Q$ and $D$ are representations, there (conceptually) may be many pairs with $Q$ and $D$ the same but with different relevance judgments
- $P(R|Q,D)$ is the proportion of the identical $(Q,D)$ pairs that are judged relevant
- Notationally, we assume a given $Q$ and use $P(R|D)$
Bayes Decision Rule

- Retrieve if $P(R|D) > P(NR|D)$
- Minimizes the average probability of error:

$$P(\text{error} | D) = \begin{cases} P(R | D) & \text{if we decide NR} \\ P(NR | D) & \text{if we decide R} \end{cases}$$
Review: Conditional Probability

\[ P(A \mid B) \equiv \frac{P(A \text{ and } B)}{P(B)} \]

\[ \text{Event Space} \]

\[ P(A) = \text{prob. of } A \text{ relative to entire event space} \]

\[ P(A \mid B) = \text{prob. of } A \text{ considering that we know } B \text{ is true} \]
Probabilistic Inference

• Suppose there’s a horrible, but very rare disease
  The probability that you contracted it is 0.01%

• But there’s a very accurate test for it
  The test is 99% accurate

• Unfortunately, you tested positive...

Should you panic?
Bayes’ Theorem

• You want to find

\[ P(\text{“have disease”} \mid \text{“test positive”}) \]

• But you only know
  – How rare the disease is
  – How accurate the test is

• Use Bayes’ Theorem (hence Bayesian Inference)

\[
P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}
\]

Prior probability

Posterior probability
Applying Bayes’ Theorem

- \( P(“\text{have disease”}) = 0.0001 \) (0.01%)
- \( P(“\text{test positive”} \mid “\text{have disease”}) = 0.99 \) (99%)
- \( P(“\text{test positive”}) = 0.010098 \)

\[
P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}
\]

Solution: board

Don’t worry!
Let $x$ be a document in the collection.
Let $R$ represent \textbf{relevance} of a document w.r.t. given (fixed) query and let $NR$ represent \textbf{non-relevance}.

Need to find $p(R|x)$ - probability that a document $x$ is relevant.

\begin{align*}
p(R \mid x) &= \frac{p(x \mid R)p(R)}{p(x)} \\
p(NR \mid x) &= \frac{p(x \mid NR)p(NR)}{p(x)}
\end{align*}

$R=\{0,1\}$ vs. NR/R

$p(R), p(NR)$ - prior probability of retrieving a (non) relevant document

$p(R \mid x) + p(NR \mid x) = 1$

$p(x \mid R), p(x \mid NR)$ - probability that if a relevant (non-relevant) document is retrieved, it is $x$. 
Basic Model overview

\[ P(\text{Rel}|d, q) \propto_q \frac{P(\text{Rel}|d, q)}{P(d|\text{Rel}, q)} \]

\[ \propto_q \frac{\prod_{v} P(t_{f_i}|\text{Rel}, q)}{\prod_{q_t} P(t_{f_i}|\text{Rel})} \]

\[ \propto_q \sum_{q_t} \log \frac{P(t_{f_i}|\text{Rel})}{P(t_{f_i}|\text{Rel})} \]

\[ = \sum_{q_t} U_i(t_{f_i}) \]

\[ = \sum_{q_t, t_{f_i} > 0} U_i(t_{f_i}) + \sum_{q_t, t_{f_i} = 0} U_i(0) \]

\[ - \sum_{q_t, t_{f_i} > 0} U_i(0) + \sum_{q_t, t_{f_i} > 0} U_i(0) \]

\[ \propto_q \sum_{t_{f_i} > 0} W_i(t_{f_i}) \]
Basic Model 1 (BM1)

\[
P(\text{Rel}|d, q) \propto_q \frac{P(\text{Rel}|d, q)}{P(\text{Rel}|d, q)}
\]

Probability ranking principle; transform to odds

\[
\propto_q \frac{P(d|\text{Rel}, q)}{P(d|\text{Rel}, q)}
\]

Bayesian inversions; drop odds of relevance

\[
\prod_v \frac{P(tf_i|\text{Rel}, q)}{P(tf_i|\text{Rel}, q)}
\]

Independence assumptions

\[
\prod_{qt} \frac{P(tf_i|\text{Rel})}{P(tf_i|\text{Rel})}
\]

Restrict to query terms

\[
\sum_{qt} \log \frac{P(tf_i|\text{Rel})}{P(tf_i|\text{Rel})}
\]

Log for linear combination
Basic model 1 – Discussion

- Assume probability ranking principle – we want to rank documents in order of probability of relevance to the query
- Use odds instead of probability
- Assume independence of terms given relevance
  - (implies correlation between some terms)
  - can replace with linked-dependence assumptions
Basic model 1 – Discussion

• Restrict to query terms
  – Assume that
    \[
    \frac{P(tf_i | \text{Rel})}{P(tf_i | \text{Rel})} = 1 \text{ for non-query terms}
    \]
  – ... or rather that, in the absence of any other information, this is the best vague-prior point-estimate

• Take logs to give us a formula that is linear in the terms
  – Scale is now log-odds, i.e. covers the whole real line
Basic model 2

\[ P(\text{Rel}|d, q) \propto_q \sum_{qt} \log \frac{P(tf_i|\text{Rel})}{P(tf_i|\text{Rel})} \]

\[ = \sum_{qt} U_i(tf_i) \]

\[ = \sum_{qt, tf_i > 0} U_i(tf_i) + \sum_{qt, tf_i = 0} U_i(0) - \sum_{qt, tf_i > 0} U_i(0) + \sum_{qt, tf_i > 0} U_i(0) \]

\[ \propto_q \sum_{qt, tf_i > 0} W_i(tf_i) \]

(from earlier slide)

(both only)

Separate terms present/absent in document

Subtract and add the same thing

Ignore document-independent bit
Basic model 2

where

\[ U_i(tf_i) = \log \frac{P(tf_i|\text{Rel})}{P(tf_i|\overline{\text{Rel}})} \]

\[ W_i(tf_i) = U_i(tf_i) - U_i(0) \]

\[ = \log \frac{P(tf_i|\text{Rel})P(tf_i = 0|\overline{\text{Rel}})}{P(tf_i|\text{Rel})P(tf_i = 0|\text{Rel})} \]

• Very general model:
  - Take the \( tf_i \) as the values of any discrete attribute
  - assuming only a ‘natural’ zero
  - (otherwise we can revert to the \( U_i \) form)
  - For continuous attributes, need probability density functions
Basic model 2 – Discussion

• ‘Natural’ zero = term absence
  – Document containing none of the query terms has score zero
  ... therefore we never need to calculate it!
  – Fits very well with the sparse nature of the document-term matrix
  ... and more practically with inverted files
  – The maximal list of documents for which we have to calculate scores is the OR set
    = union of inverted lists over query terms

• Model is linear over attributes

• Select attributes which we think might relate to the query
  – Query terms
  – ... and other attributes? (e.g. query-independent features)
The binary independence model

\[ W_i^{\text{BIM}} = \log \frac{P(t_i | \text{Rel}) (1 - P(t_i | \overline{\text{Rel}}))}{(1 - P(t_i | \text{Rel})) P(t_i | \overline{\text{Rel}})} \]

Binary attribute \( t_i \) (e.g. term presence)

- collection size \( N \)
- document frequency \( n_i \)
- relevant set size \( R \)
- relevant doc frequency \( r_i \)

Counts (assuming that we know which documents are relevant)

Reasonable estimators give standard Robertson / Sparck Jones weighting

\[ W_i^{\text{RSJ}} = \log \frac{(r_i + 0.5)(N - R - n_i + r_i + 0.5)}{(n_i - r_i + 0.5)(R - r_i + 0.5)} \]
Now assume no known relevant

- Gives us an approximation to classical IDF

\[
W_{i, \text{IDF}} = \log \frac{N - n_i + 0.5}{n_i + 0.5}
\]

- This is equivalent to assuming

\[
P(t_i | \text{Rel}) = 0.5
\]

(another vague-prior point estimate)

- Can use more sophisticated estimates

... and get closer to classical IDF
Saturation functions

We call these *saturation* functions for $tf$ (need to multiply by something for the limiting value)

We could imagine various functions ...

The 2-Poisson model generates smooth functions
Shape depends on parameters (3)
  - mostly convex
  - but some combinations give an initial concavity
BM25 (1)

- Approximate the shape of the saturation function

\[
\frac{tf}{k + tf} \quad \text{for some } k > 0
\]

Asymptotically approaches 1

Middle line is \( k=1 \)
Upper line is lower \( k \)
Lower line is higher \( k \)

Note: since this function is applied to all terms, absolute height does not matter: what matters is the relative increments with \( tf \).
BM25 (2)

- Approximate $W_i^{\text{BIM}}$ based on binary eliteness with $W_i^{\text{RSJ}}$ based on term presence/absence
- First version:
  \[ W_i^{\text{BM25}} = \frac{tf_i}{k + tf_i} W_i^{\text{RSJ}} \]
- But we need to think about document length:
  - May affect $tf$
  - Therefore we may need to normalize $tf$ in some way
- Why are some documents longer than others?
  - Verbosity: some authors are more verbose
  - Scope: some authors have more to say
- First suggests normalization, second doesn't
  - So maybe soft normalization?
BM25 (3)

document length $dl := \sum_V tf_i$

average doclength $avdl$ average over collection

- Soft normalization factor:

$$B = \left( (1 - b) + b \frac{dl}{avdl} \right), \quad 0 \leq b \leq 1$$

Now divide $tf$ by $B$ before applying the saturation function

$$tf'_i = \frac{tf_i}{B}$$

$$W_{i}^{BM25} = \frac{tf'_i}{k_1 + tf'_i} W_{i}^{RSJ}$$

$$= \frac{tf_i}{k_1 \left( (1 - b) + b \frac{dl}{avdl} \right) + tf_i} W_{i}^{RSJ}$$
Putting it all together: BM0 → BM25

(BM0) \( w = 1 \)

(BM1) \( w = \log \frac{N - n + 0.5}{n + 0.5} \times \frac{qtf}{(k_3 + qtf)} \)

(BM15) \( w = \frac{tf}{(k_1 + tf)} \times \log \frac{N - n + 0.5}{n + 0.5} \times \frac{qtf}{(k_3 + qtf)} + k_2 \times n_q \frac{(\Delta - d)}{\Delta + d} \)

(BM11) \( w = \frac{tf}{(k_1 \times d + tf)} \times \log \frac{N - n + 0.5}{n + 0.5} \times \frac{qtf}{(k_3 + qtf)} + k_2 \times n_q \frac{(\Delta - d)}{\Delta + d}. \)

BM25

\[
\left[ \log \frac{N}{n} \right] \cdot \frac{(k_1 + 1)tf_{ij}}{k_1((1 - b) + b \times (dl_j/avdl)) + tf_{ij}} \cdot \frac{(k_3 + 1)tf_{iq}}{k_3 + tf_{iq}}
\]

\( b = 0.6 - 0.75; \quad k_1 = 1.0 - 2.0; \quad k_3 = 8 \)
BM25 (4)

- As in the binary case, this can be used with or without relevance feedback information.

- Without relevance feedback information, $W_i^{RSJ}$ can be replaced by $W_i^{IDF}$ as previously defined, or by classical idf.

- ... which makes it look like a tf*idf model.
What about document structure?

• Questions:
  – How to weigh terms within Title, Abstract, Body?
  – How to weigh each field?

• BM25F (to be continued)

http://portal.acm.org/citation.cfm?id=1031171.1031181
Measures for a search engine

• How fast does it index
  – Number of documents/hour
  – (Average document size)

• How fast does it search
  – Latency as a function of index size

• Expressiveness of query language
  – Ability to express complex information needs
  – Speed on complex queries

• UI (more later!)

• Is it free?
Measures for a search engine

- All of the preceding criteria are *measurable*: we can quantify speed/size
  - we can make expressiveness precise
- The key measure: user happiness
  - What is this?
  - Speed of response/size of index are factors
  - But blindingly fast, useless answers won’t make a user happy
- Need a way of quantifying user happiness
Measuring user happiness

- Issue: who is the user we are trying to make happy?
  - Depends on the setting

- **Web engine:**
  - User finds what they want and return to the engine
    - Can measure rate of return users
  - User completes their task – search as a means, not end

- **eCommerce site:** user finds what they want and buy
  - Is it the end-user, or the eCommerce site, whose happiness we measure?
  - Measure time to purchase, or fraction of searchers who become buyers?
Measuring user happiness

• **Enterprise (company/govt/academic):** Care about “user productivity”
  – How much time do my users save when looking for information?
  – Many other criteria having to do with breadth of access, secure access, etc.
Happiness: elusive to measure

- Most common proxy: relevance of search results
- But how do you measure relevance?
- We will detail a methodology here, then examine its issues
- Relevance measurement requires 3 elements:
  1. A benchmark document collection
  2. A benchmark suite of queries
  3. A usually binary assessment of either Relevant or Nonrelevant for each query and each document
    - Some work on more-than-binary, but not the standard
Evaluating an IR system

• Note: the **information need** is translated into a query

• Relevance is assessed relative to the **information need** *not* the query

• E.g., **Information need**: *I'm looking for information on whether drinking red wine is more effective at reducing your risk of heart attacks than white wine.*

• **Query**: *wine red white heart attack effective*

• You evaluate whether the doc addresses the information need, not whether it has these words
Standard relevance benchmarks

• TREC - National Institute of Standards and Technology (NIST) has run a large IR test bed for many years
• Reuters and other benchmark doc collections used
• “Retrieval tasks” specified
  – sometimes as queries
• Human experts mark, for each query and for each doc, Relevant or Nonrelevant
  – or at least for subset of docs that some system returned for that query
Unranked retrieval evaluation: 
Precision and Recall

- **Precision**: fraction of retrieved docs that are relevant = $P(\text{relevant} | \text{retrieved})$

- **Recall**: fraction of relevant docs that are retrieved = $P(\text{retrieved} | \text{relevant})$

<table>
<thead>
<tr>
<th></th>
<th>Relevant</th>
<th>Nonrelevant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieved</td>
<td>tp</td>
<td>fp</td>
</tr>
<tr>
<td>Not Retrieved</td>
<td>fn</td>
<td>tn</td>
</tr>
</tbody>
</table>

- Precision $P = \frac{tp}{tp + fp}$
- Recall $R = \frac{tp}{tp + fn}$
Should we instead use the accuracy measure for evaluation?

- Given a query, an engine classifies each doc as “Relevant” or “Nonrelevant”
- The **accuracy** of an engine: the fraction of these classifications that are correct
  \[- \frac{(tp + tn)}{(tp + fp + fn + tn)}\]
- **Accuracy** is a commonly used evaluation measure in machine learning classification work
- Why is this not a very useful evaluation measure in IR?
Why not just use accuracy?

• How to build a 99.9999% accurate search engine on a low budget....

• People doing information retrieval want to find something and have a certain tolerance for junk.
You can get high recall (but low precision) by retrieving all docs for all queries!

Recall is a non-decreasing function of the number of docs retrieved

In a good system, precision decreases as either the number of docs retrieved or recall increases

- This is not a theorem, but a result with strong empirical confirmation
Difficulties in using precision/recall

• Should average over large document collection/query ensembles
• Need human relevance assessments
  – People aren’t reliable assessors
• Assessments have to be binary
  – Nuanced assessments?
• Heavily skewed by collection/authorship
  – Results may not translate from one domain to another
A combined measure: $F$

- Combined measure that assesses precision/recall tradeoff is **$F$ measure** (weighted harmonic mean):

$$F = \frac{1}{\frac{1}{P} \alpha + (1-\alpha) \frac{1}{R}} = \frac{(\beta^2 + 1)PR}{\beta^2P + R}$$

- People usually use balanced $F_1$ measure
  - i.e., with $\beta = 1$ or $\alpha = \frac{1}{2}$

- Harmonic mean is a conservative average
  - See CJ van Rijsbergen, *Information Retrieval*
$F_1$ and other averages
Evaluating ranked results

• Evaluation of ranked results:
  – The system can return any number of results
  – By taking various numbers of the top returned documents (levels of recall), the evaluator can produce a *precision-recall curve*
A precision-recall curve
Averaging over queries

• A precision-recall graph for one query isn’t a very sensible thing to look at

• You need to average performance over a whole bunch of queries.

• But there’s a technical issue:
  – Precision-recall calculations place some points on the graph
  – How do you determine a value (interpolate) between the points?
Interpolated precision

• Idea: If locally precision increases with increasing recall, then you should get to count that...

• So you max of precisions to right of value
Evaluation

• Graphs are good, but people want summary measures!
  – Precision at fixed retrieval level
    • Precision-at-\(k\): Precision of top \(k\) results
    • Perhaps appropriate for most of web search: all people want are good matches on the first one or two results pages
    • But: averages badly and has an arbitrary parameter of \(k\)
  – 11-point interpolated average precision
    • The standard measure in the early TREC competitions: you take the precision at 11 levels of recall varying from 0 to 1 by tenths of the documents, using interpolation (the value for 0 is always interpolated!), and average them
    • Evaluates performance at all recall levels
Typical (good) 11 point precisions

- SabIR/Cornell 8A1 11pt precision from TREC 8 (1999)
Yet more evaluation measures...

• Mean average precision (MAP)
  – Average of the precision value obtained for the top \( k \) documents, each time a relevant doc is retrieved
  – Avoids interpolation, use of fixed recall levels
  – MAP for query collection is arithmetic ave.
    • Macro-averaging: each query counts equally

• R-precision
  – If have known (though perhaps incomplete) set of relevant documents of size \( Rel \), then calculate precision of top \( Rel \) docs returned
  – Perfect system could score 1.0.
Variance

• For a test collection, it is usual that a system does badly on some information needs (e.g., MAP = 0.1) and excellently on others (e.g., MAP = 0.7)

• Usually, variance in performance of the same system across queries is greater than the variance of different systems on the same query.

• That is, there are easy information needs and hard ones!
CREATING TEST COLLECTIONS FOR IR EVALUATION
### TABLE 4.3 Common Test Corpora

<table>
<thead>
<tr>
<th>Collection</th>
<th>NDocs</th>
<th>NQrys</th>
<th>Size (MB)</th>
<th>Term/Doc</th>
<th>Q-D RelAss</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADI</td>
<td>82</td>
<td>35</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AIT</td>
<td>2109</td>
<td>14</td>
<td>2</td>
<td>400</td>
<td>&gt;10,000</td>
</tr>
<tr>
<td>CACM</td>
<td>3204</td>
<td>64</td>
<td>2</td>
<td>24.5</td>
<td></td>
</tr>
<tr>
<td>CISI</td>
<td>1460</td>
<td>112</td>
<td>2</td>
<td>46.5</td>
<td></td>
</tr>
<tr>
<td>Cranfield</td>
<td>1400</td>
<td>225</td>
<td>2</td>
<td>53.1</td>
<td></td>
</tr>
<tr>
<td>LISA</td>
<td>5872</td>
<td>35</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Medline</td>
<td>1033</td>
<td>30</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NPL</td>
<td>11,429</td>
<td>93</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OSHMED</td>
<td>34,8566</td>
<td>106</td>
<td>400</td>
<td>250</td>
<td>16,140</td>
</tr>
<tr>
<td>Reuters</td>
<td>21,578</td>
<td>672</td>
<td>28</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td>TREC</td>
<td>740,000</td>
<td>200</td>
<td>2000</td>
<td>89-3543</td>
<td>» 100,000</td>
</tr>
</tbody>
</table>
From document collections to test collections

• Still need
  – Test queries
  – Relevance assessments

• Test queries
  – Must be germane to docs available
  – Best designed by domain experts
  – Random query terms generally not a good idea

• Relevance assessments
  – Human judges, time-consuming
  – Are human panels perfect?
Unit of Evaluation

• We can compute precision, recall, F, and ROC curve for different units.

• Possible units
  – Documents (most common)
  – Facts (used in some TREC evaluations)
  – Entities (e.g., car companies)

• May produce different results. Why?
Kappa measure for inter-judge (dis)agreement

- Kappa measure
  - Agreement measure among judges
  - Designed for categorical judgments
  - Corrects for chance agreement
- Kappa = \[ \frac{P(A) - P(E)}{1 - P(E)} \]
- $P(A)$ – proportion of time judges agree
- $P(E)$ – what agreement would be by chance
- Kappa = 0 for chance agreement, 1 for total agreement.
### Kappa Measure: Example

<table>
<thead>
<tr>
<th>Number of docs</th>
<th>Judge 1</th>
<th>Judge 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td>Relevant</td>
<td>Relevant</td>
</tr>
<tr>
<td>70</td>
<td>Nonrelevant</td>
<td>Nonrelevant</td>
</tr>
<tr>
<td>20</td>
<td>Relevant</td>
<td>Nonrelevant</td>
</tr>
<tr>
<td>10</td>
<td>Nonrelevant</td>
<td>Relevant</td>
</tr>
</tbody>
</table>
Kappa Example (cont'd)

- $P(A) = \frac{370}{400} = 0.925$
- $P(\text{nonrelevant}) = \frac{(10+20+70+70)}{800} = 0.2125$
- $P(\text{relevant}) = \frac{(10+20+300+300)}{800} = 0.7878$
- $P(E) = 0.2125^2 + 0.7878^2 = 0.665$
- $\text{Kappa} = \frac{0.925 - 0.665}{1-0.665} = 0.776$

- Kappa $> 0.8$ = good agreement
- $0.67 < \text{Kappa} < 0.8$ -> “tentative conclusions” (Carletta ’96)
- Depends on purpose of study
- For $>2$ judges: average pairwise kappas
• TREC Ad Hoc task from first 8 TREC5s is standard IR task
  – 50 detailed information needs a year
  – Human evaluation of pooled results returned
  – More recently other related things: Web track, HARD

• A TREC query (TREC 5)
  <top>
  <num> Number:  225
  <desc> Description:
  What is the main function of the Federal Emergency Management Agency (FEMA) and the funding level provided to meet emergencies? Also, what resources are available to FEMA such as people, equipment, facilities?
  </top>
Standard relevance benchmarks: Others

• GOV2
  – Another TREC/NIST collection
  – 25 million web pages
  – Largest collection that is easily available
  – But still 3 orders of magnitude smaller than what Google/Yahoo/MSN index

• NTCIR
  – East Asian language and cross-language information retrieval

• Cross Language Evaluation Forum (CLEF)
  – This evaluation series has concentrated on European languages and cross-language information retrieval.

• Many others
Critique of pure relevance

- Relevance vs Marginal Relevance
  - A document can be redundant even if it is highly relevant
  - Duplicates
  - The same information from different sources
  - Marginal relevance is a better measure of utility for the user.

- Using facts/entities as evaluation units more directly measures true relevance.

- But harder to create evaluation set

- See Carbonell reference
Can we avoid human judgment?

- No
- Makes experimental work hard
  - Especially on a large scale
- In some very specific settings, can use proxies
  - E.g.: for approximate vector space retrieval, we can compare the cosine distance closeness of the closest docs to those found by an approximate retrieval algorithm
- But once we have test collections, we can reuse them (so long as we don’t overtrain too badly)
Evaluation at Web search engines

• Search engines have test collections of queries and hand-ranked results
• Recall is difficult to measure on the web
• Search engines often use precision at top k, e.g., k = 10
• ... or measures that reward you more for getting rank 1 right than for getting rank 10 right.
  – NDCG (Normalized Cumulative Discounted Gain)
• Search engines also use non-relevance-based measures.
  – Clickthrough on first result
    • Not very reliable if you look at a single clickthrough ... but pretty reliable in the aggregate.
  – Studies of user behavior in the lab
  – A/B testing