Chapter 11: Probabilistic Information Retrieval

- Estimating the probability of a term t appearing in a relevant document $P(t \mid R = 1)$
- Users' *information needs* $\rightarrow$ translated into *query representations*
- Documents $\rightarrow$ *Document representation*
- In the Boolean or vector space models, matching is done in a formally defined but *semantically imprecise calculus* of index terms: *Uncertain guesses*
- Probability provides a principled foundation for reasoning under uncertainty

11.2 The Probability Ranking Principle

11.2.1 The 1/0 loss case

- $R_{d,q}$: An indicator random variable that says whether d is *relevant* with respect to a given query q.
- **Probability Ranking Principle (PRP)**: Rank documents by their estimated *probabilities* of relevance: $P(R = 1 \mid d, q)$
- *1/0 Loss*: In the simplest case of PRP, you simply lose a point for returning a non-relevant document or failing to return any relevant documents.
- *Bayes Optimal Decision Rule*: d is relevant iff $P(R = 1 \mid d, q) > P(R = 0 \mid d, q)$
  - Theorem: The PRP is optimal, in the sense that it minimizes the expected loss (aka the *Bayes risk*) under 1/0 loss.
  - *Bayes classifier*: Classifiers based on this rule
  - Requires that all the probabilities are known *correctly*, which is rarely the case in practice

11.2.2 The PRP with retrieval costs

- Now consider additional retrieval costs associated with relevant and non-relvant documents.
- C_1: The cost of *not* retrieving a relevant document
- C_0: The cost of retrieving a *non-relevant* document
- According to the PRP, if $C_0 \cdot P(R = 0 \mid d) - C_1 \cdot P(R = 1 \mid d) \leq C_0 \cdot P(R = 0 \mid d') - C_1 \cdot P(R = 1 \mid d')$, for a specific document d and for all documents d' not yet retrieved,
  - Then d should be the next document to be retrieved.
  - This allow us to model differential costs of false positives and false negatives at modelling stage, rather than considering them at *evaluation* stage.

11.3 The Binary Independence Model

- Documents and queries are both represented as binary term incidence vectors.
- A document d is represented by the vector $\vec{x} = (x_1, \cdots, x_M)$ where $x_t = 1$ if term t is present in document d and 0 if not.
  - Many possible documents could have identical representation under this scheme.
- A query is represented by the *incidence vector* $\vec{q}$
- Independence: Terms are occurring in the documents *independently* - not really correct, but often gives satisfactory results in practice
- Assume that the user has a single step information need
- We need to figure out the contribution of terms to the document's relevance
  - How statistics like term frequency, document frequency, document length *influence judgments about document relevance*
  - And how we could reasonably combine them to estimate the probability of relevance
- Another assumption: The relevance of each document is independent of other documents' relevance probabilities
  - This is especially harmful when we allow returning duplicate or near duplicate documents
- Then using Bayes rule, $P(R = 1 \mid \vec{x}, \vec{q}) = \frac{P(\vec{x}|R=1,\vec{q}) \cdot P(R=1|\vec{q})}{P(\vec{x}|\vec{q})}$, and accordingly for $R = 0$.
- We never know the exact probabilities, so we have to use estimates.
  - If we knew the true percentage of relevant documents in the collection, we could use that as priors.

11.3.1 Deriving a ranking function for query terms

- We can rank by just looking at the *odds* of relevance rather than the full probability
  - $O(R \mid \vec{x}, \vec{q}) = \frac{P(R=1|\vec{x},\vec{q})}{P(R=0|\vec{x},\vec{q})}$
  - $= \frac{P(R=1|\vec{q})}{P(R=0|\vec{q})} \cdot \frac{P(\vec{x}|R=1,\vec{q})}{P(\vec{x}|R=0,\vec{q})}$
  - This allows us to ignore $P(\vec{x} \mid \vec{q})$.
- The fraction of prior probabilities are constant for a given query; no need to estimate it
- But how can we estimate the probability of an entire term incidence vector occurring?
- *Naive Bayes Conditional Independence Assumption*: The presence or absence of a word in a document is independent of the presence or absence of any other words, given the query.

- Then $O(R \mid \vec{x}, \vec{q}) = O(R \mid \vec{q}) \cdot \prod_{t=1} \frac{P(x_t|R=1,\vec{q})}{P(x_t|R=0,\vec{q})}$

- Shorter names for the conditional probabilities
  - $p_t = P(x_t = 1 \mid R = 1, \vec{q})$
  - $u_t = P(x_t = 1 \mid R = 0, \vec{q})$
- Additional assumption: Terms *not* occurring in the query are *equally likely* to occur in relevant and nonrelevant documents
  - If $q_t = 0$ Then $p_t = u_t$.
  - Then we only need to consider terms that appear in the query
  - And divide the product term into the product over the query terms *found in the document* and the ones *not* found in the document.
  - Under this assumption, terms not in query simply do not affect the outcome at all.
- **Retrieval Status Value (RSV)**: The only thing we need to estimate eventually to rank documents
  - $\text{RSV}_d = \sum_{t:x_t=q_t=1} \log \frac{p_t \cdot (1-u_t)}{u_t \cdot (1-p_t)}$
  - Define $c_t = \log \frac{p_t}{1-p_t} + \log \frac{1-u_t}{u_t}$
    - This term is log odds ratios for each term in the query.
    - The value will be 0 if a term has equal odds of appearing in relevant and non-relevant documents
    - Positive if it is more likely to appear in *relevant* documents.
    - RSV_d is the document score for a query.

11.3.2 Probability estimates in theory

- If we say that there are S relevant documents in total, and x_t is present in s documents, then
  - $p_t = \frac{s}{S}$ and $u_t = \frac{\text{df}_t - s}{N-S}$.
- Relative Frequency: One way to estimate the probability of an event from data is simply to count the number of times an event occurred divided by the total number of trials.
- Maximum Likelihood Estimate (MLE): Estimating the probability as the relative frequency.
  - The probability given to events we *happened to see* is usually too high
  - While probabilities given to unseen events become 0 and breaks our models
- *Smoothing*: Simultaneously decreasing the estimated probability of *seen* events and increasing the probability of *unseen* events
- Maximum a Posteriori (MAP): Choose the most likely point value for probabilities based on the prior and the observed evidence
  - Pseudocounts: Add a number α to each of the observed counts.
    - Equivalent to using a uniform distribution over the vocabulary as a *Bayesian Prior*
    - α denotes the strength of our belief in uniformity

11.3.3 Probability estimates in practice

- Under the assumption that relevant documents are a *very small percentage* of the collection
- Plausible to approximate statistics for non-relevant documents by using statistics for the *whole collection*
  - Then $u_t = \text{df}_t/N$ and $\log[(1-u_t)/u_t] \approx \log \frac{N}{\text{df}_t}$.
  - This gives a justifcation for idf weighting used in Chapter 6.
- But we can't estimate p_t using the idea like this. Instead:
  - *Relevance feedback*: use the frequency of term occurrence in *known* relevant documents
  - Croft and Harper (1979): Using a constant for p_t
    - If $p_t = 0.5$: Cancels out the first term in RSV. Weak estimate, but doesn't disagree violently with our hopes for the search terms appearing in many but *not all* relevant documents
    - Combined with our u_t estimation, the document ranking is determined simply by which query terms occur in documents, scaled by their idf weighting.
      - Works well enough in short documents (titles or abstracts), but we want to do better
  - Greiff (1988): Empirically, p_t rises with df_t. He estimates it to be $p_t = 1/3 + 2/3 \cdot \frac{\text{df}_t}{N}$.

11.3.4 Probabilistic approaches to relevance feedback

- Use relevance feedback to get a better estimate of p_t
  1. Guess initial estimates of p_t and u_t.
  2. Use the current estimate of p_t and u_t to make the best guess about what the true set of relevant documents $R = \{d : R_{d,q} = 1\}$ should be.
  3. Make the user provide judgments about those documents presented and we put them in the set `V`. We can partition this V into two parts:
     - $VR = \{d \in V, R_{d,q} = 1\} \subset R$
     - $VNR = \{d \in V, R_{d,q} = 0\}$, *disjoint* from R

4. We re-estimate p_t and u_t. If the sets VR and VNR are large enough, we may be able to estimate this quantities directly from these documents as MLEs:
   - $p_t = \frac{|VR_t|}{|VR|}$
   - Since we will need some smoothing, we can try $p_t = \frac{|VR_t|+1/2}{|VR|+1}$, by adding 1/2 to both the counts of the relevant documents containing and *not* containing the term.
   - However, the user would give very small V, so the estimate would be unreliable even with smoothing.
   - Hence it is often better to combine the new information using *Bayesian updating*
     * $p_t^{(k+1)} = \frac{|VR_t|+\kappa \cdot p_t^{(k)}}{|VR|+\kappa}$.
     * $p_t^{(k)}$ is the k-th estimate for p_t in an iterative updating process.
     * We use $p_t^{(k)}$ as a Bayesian prior with the weight of κ.
     * This allow us to distribute κ pseudocounts according to the previous estimate, instead of uniformly distributing them.
     * In the absence of further evidence, and assuming the users indicating relevance/non-relevance of 5 documents), $\kappa = 5$ might be appropriate. In other words, prior is weighted strongly enough in order to prevent estimates changing too much due to evidence coming from too small number of documents.
5. Repeat the process from step 2 until the user is satisfied.

- We can also do a pseudo-relevance version, by assuming that $VR = V$.

11.4 An appraisal and some extensions

11.4.1 An appraisal of probabilistic models

- Major assumptions of probabilistic IR models:
  - A Boolean representation of documents, queries, and relevance
  - Term independence
  - Terms not in the query don't affect the outcome
  - Document relevance values are independent
- The severity of the modelling assumptions had made achieving good performance difficult
- The difference between vector space model and probabilistic models are not that great
  - In prob. IR, you score queries not by cosine similarity and tf-idf in a vector space, but instead with a formula derived from probability theory
  - Some people have changed an existing vector space system by adopting term weighting formulas from prob. models

11.4.2 Tree-structured dependencies between terms

- Let's remove the assumption that terms are independent, which is very far from true in practice.
- Tree structure of term dependencies: Each term can be directly dependent on only one other term
- Tree Augmented Naive Bayes Model

11.4.3 Okapi BM25: a non-binary model

- The BIM was originally designed for short catalog records and abstracts of fairly consistent length
- For more modern full-text searches, we need to consider term frequencies and document length
- *The BM25 ("Best Match 25") weighting scheme (Okapi weighting)*: Sensitive to those quantities while not introducing too many parameters into the model
- From `opensourceconnections.com`, "BM25 The Next Generation of Lucene Relevance":
  - In the classic Lucene similarity, IDF score $\cdot$ TF score $\cdot$ Field Norm $= \log \frac{N}{\mathrm{df}_t+1} \cdot \sqrt{\mathrm{tf}} \cdot \frac{1}{\sqrt{\mathrm{length}}}$
  - BM25 ***IDF*** is similar to classic IDF, but has a potential to give a negative value (Lucene solved this by adding 1 to the value before taking the log.)
  - BM25 ***TF*** dampens the impact of term frequency even further than traditional tf-idf: $\frac{(k+1)\cdot tf}{k+tf}$
    * Approaches $k+1$ asymptotically
    * Quickly hit diminishing returns
    * Higher k causes TF to take longer to reach saturation
  - BM25 ***Document Length***: Adjust TF score by whether the document is above or below *the average length* of a dcoument in the corpus.
    * $\frac{(k+1)\cdot \mathrm{tf}}{k\cdot(1-b+b\cdot L)+\mathrm{tf}}$, where L is how long a document is relative to the average document length. The constant b will allow us to finely tune how much influence L has on scoring.
    * Shorter documents hit the asymptote much faster. The more matches in the short docs, the more certain you can feel confident in the relevance.

    * A lengthy book, on the other hand, takes many more matches to get a point where we can feel confident. So reaching "max relevance" takes longer.

11.4.4 Bayesian network approaches to IR

- Turtle and Croft: A document collection network and a query network
- The document collection network can be pre-computed
  - Maps from documents to terms to *concepts*
  - The concepts are a theasaurus-based expansion of the terms appearing in the document
- The query network is small but needs to be built for each new queries
  - Maps from query terms, to query expressions (built using probabilistic or noisy version of `AND` and `OR` operators), to the user's information need