School of Computing and Information Systems
The University of Melbourne
COMP90042 WEB SEARCH AND TEXT ANALYSIS (Semester 1, 2017)

Workshop exercises: Week 12

Discussion

1. Based on the following top-6 retrieval results from a collection of 100 documents, and the accompanying binary relevance judgements

doc	score	relevance
a	0.4	0
b	1.2	0
c	2.2	1
d	0.5	1
e	0.1	1
f	0.8	0

compute the following evaluation metrics:

(a) precision@3

The first step is to form the vector of relevance judgments, ranked by document score

$$\langle c, b, f, d, a, e \rangle = \langle 1, 0, 0, 1, 0, 1 \rangle$$

Now taking the first 3 items, $\langle 1, 0, 0 \rangle$ (c,b,f), one is correct, from 3 total. So we have $P@3 = \frac{1}{3}$.

(b) average precision (do you need to make any assumptions about the document collection?); and

AP is normalised by the number of relevant documents in the collection. We weren't told this in the question, all we know is that of the retrieved documents 3 are relevant (c,d,e). Based on this, we could have anywhere between 3 and 97 relevant documents. Here we'll be optimisic, and use 3:

$$AP = \frac{1}{3} \times (\frac{1}{1} + \frac{2}{4} + \frac{3}{6}) = \frac{1}{3} \times 2 = \frac{2}{3}$$

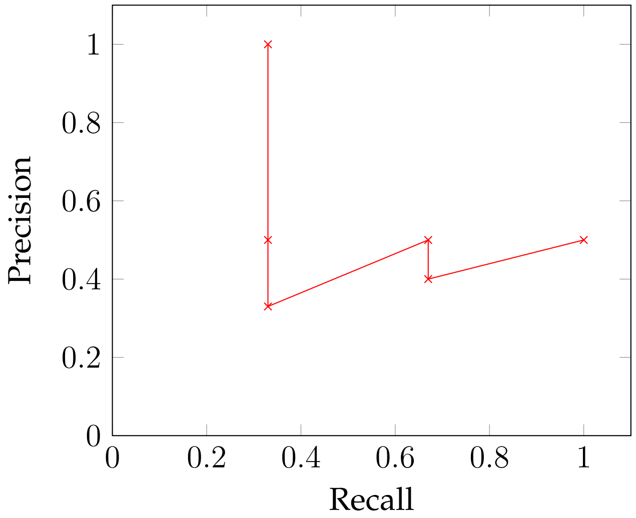
Under the most pessimistic view, the AP is much lower, 2%. Witness the massive difference between the two results, which is worrying as most often we don't know the full relevant set. Why? Too many documents (millions or billions) which we would have to look at to determine if they are relevant.

(c) rank-biased precision (RBP), with $p = 0.5$

$$
\begin{aligned}
RBP &= (1-p) \times \sum_i r_i p^{i-1} && (1)\\
&= (1-0.5) \times \left(0.5^0 + 0.5^3 + 0.5^5\right) && (2)\\
&= \frac{1}{2} \times \left(1 + \frac{1}{8} + \frac{1}{32}\right) && (3)\\
&= \frac{1}{2} \times \frac{1}{32} \times (32 + 4 + 1) && (4)\\
&= \frac{37}{64} && (5)
\end{aligned}
$$

RBP with $p = 0.5$ means the user has a persistence of $p = 0.5$ to move from the fist to the second document and p^2 to the third etc. Higher values of p represent a more patient user model. For impatient users only the top documents are important and the RBP score reflects that. Note for the RBP formula that the factor $(1-p)$ originates from the *expected* number of documents that will be evaluated. This is a normalization factor and for a given p is just a constant.

(d) plot the precision-recall graph, where you plot (precision, recall) point for the top k documents, $k = 1, 2, \ldots 6$.



Note this figure could be drawn in a more disconnected way, as it's questionable connecting the points because you can't return a fractional document.

(e) what are the strengths and weaknesses of the methods above for evaluating IR systems?

precision@k is very easy to evaluate and easily understood, however it doesn't differentiate by rank within k, and doesn't incorporate any adjustment for the size of the relevant set;

avg precision is less easily understood, but adds differentiation by rank, and adjusts for the size of the relevant set, however this requires knowing the relevant set (we typically don't know this); Averaging over multiple queries (mean average precision) with different number of relevant results also distorts the metric and it is very unintuitive of what the MAP score actually means.

rank biased precision has a similar formulation to AP, but it a little more intuitive. It includes differentiation by rank, and avoids the need to know the relevant set beyond the top ranked documents (as the contribution to RBP of lower ranked documents diminishes quickly, and can be upper bounded). A small issues is the need to know the free parameter, “p”, which may differ between users and queries.

2. How can a retrieval method be learned using supervised machine learning methods? Consider how to frame the learning problem, what data will be required for supervision, and what features are likely to be useful.

   Learning to rank deals with the problem of taking the top k results from a heuristic IR system, and reordering these to improve the user acceptability of the results. This can be framed as a learning in several ways, with the simplest being a “pointwise” objective whereby a learning algorithm is trained to give a high score to documents that are relevant, and a low score to irrelevant documents. This could be training a binary classifier, or a multi-class classifier (or a regression system or ordinal regression). The supervision for this are relevance judgments over the top k results, or, more commonly, user click data from query logs, or other related extrinsic data such as whether the user reformulates the query, makes a purchase, etc. There are a plethora of features that can be useful, deriving from the user behaviour, the document url and its text, and the query.

3. What aspects of human language make automatic translation difficult?

   The whole gamut of linguistics, from lexical complexity, morphology, syntax, semantics etc. In particular if the two languages have very different word forms (e.g., consider translating from an morphologically light language like English into Turkish, which has very complex morphology), or very different syntax, leading to different word order. These raise difficult learning problems for a translation system, which needs to capture these differences in order to learn translations from bi-texts, and produce these for test examples.

4. For the following “bi-text”:

Language A	Language B
`green house`	`casa verde`
`the house`	`la casa`

   (a) What is the logic behind **IBM Model 1** for deriving word alignments?

   - The core idea is that we are going to have a translation table which stores the probability of translating a word from the target language into every possible word in the source language (again, this is the wrong direction due to the noisy channel model).

- The probability of a sentence can then be treated as a **uni-gram** probability, conditioned on how the tokens in the two sentences are aligned. Or, essentially, the product of all of the corresponding probabilities from the translation table.

(b) Work through the first few iterations of using the **Expectation Maximisation** algorithm to build a translation table for this collection. Check your work by comparing to the `WSTA_N21_machine_translation.ipynb` output.

- We need to establish the direction of translation before we begin (although it isn't important in this particular example): let's say, we're trying to translate language `B` into language `A`. Consequently, we want the alignments where we're translating `A` into `B`. (Again, the opposite direction, due to the "noisy channel" model.)
- We're going to initialise our translation table T with **uniform** values: every word from `A` is equally likely to be translated as every word from `B`:

T	`casa`	`la`	`verde`	Total
`green`	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	1
`house`	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	1
`the`	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	1

- The other thing we'll want to establish is the set of possible alignments. This can be done exhaustively by hand, because the "sentences" under consideration are so short; this is not practical for longer sentences, however.
- I'm going to follow the notebook in ignoring the possibility of alignment words from `B` with the null element of `A`. We can deal with the converse — where words in `A` align with the null element of `B` by simply not aligning them to anything. (Proper models also deal with the former.)
- Consequently, each of the two sentences (called I and II below) has four (2^2) possible alignments (where every token of `B` is accounted for), namely:
  - Ia: `green` aligns with `casa` and `house` aligns with `verde`
  - Ib: `green` aligns with `verde` and `house` aligns with `casa`
  - Ic: `green` aligns with `casa` and `verde` (`house` implicitly aligns to null)
  - Id: `house` aligns with `casa` and `verde`
  - IIa: `the` aligns with `la` and `house` aligns with `casa`
  - IIb: `the` aligns with `casa` and `house` aligns with `la`
  - IIc: `the` aligns with `la` and `casa`
  - IId: `house` aligns with `la` and `casa`
- Now, we're going to calculate the **expected** likelihood of each of these possible alignments, according to the following formula:

$$\hat{P}(F, A|E) \quad = \quad \frac{\epsilon}{(I+1)^J} \prod_{j=1}^{J} t(f_j|e_{a_j})$$

- A close inspection might lead us to say that the $(+1)$ should be excluded, because we're neglecting the null term from the A tokens, but it doesn't actually matter, as we'll see in a moment.
- For Ia, we observe the following:

$$\begin{aligned}\hat{P}(F,A|E) &= \frac{\epsilon}{(I+1)^J} t(\texttt{casa}|\texttt{green}) t(\texttt{verde}|\texttt{house}) \\ &= \frac{\epsilon}{(2+1)^2}(\frac{1}{3})(\frac{1}{3}) = \frac{\epsilon}{9}\frac{1}{9}\end{aligned}$$

- Because our translation table is uniform, every calculation will look the same.
- Now, we're going to make a **maximum** likelihood estimate of each entry in our translation table. We do this by summing the expected probability of the alignment for each possible translation.
- For green:
  - It aligns with casa in Ia ($\frac{\epsilon}{9}\frac{1}{9}$) and Ic (same), to give a total of $\frac{\epsilon}{9}\frac{2}{9}$.
  - It aligns with verde in Ib ($\frac{\epsilon}{9}\frac{1}{9}$) and Ic (same), to give a total of $\frac{\epsilon}{9}\frac{2}{9}$.
  - It never aligns with la, because they don't appear in a sentence together.
- Let's summarise our likelihoods in the (un-normalised) translation table.

T	casa	la	verde	Total
green	$\frac{\epsilon}{9}\frac{2}{9}$	0	$\frac{\epsilon}{9}\frac{2}{9}$	$\frac{\epsilon}{9}\frac{4}{9}$
house	$\frac{\epsilon}{9}\frac{4}{9}$	$\frac{\epsilon}{9}\frac{2}{9}$	$\frac{\epsilon}{9}\frac{2}{9}$	$\frac{\epsilon}{9}\frac{8}{9}$
the	$\frac{\epsilon}{9}\frac{2}{9}$	$\frac{\epsilon}{9}\frac{2}{9}$	0	$\frac{\epsilon}{9}\frac{4}{9}$

- We will now normalise the rows so that they look like probabilities. Doing this causes all of the $\frac{\epsilon}{9}$ terms to vanish; consequently, we will just ignore them for the rest of the steps below.
- After simplifying, here is the new translation table:

T	casa	la	verde	Total
green	$\frac{1}{2}$	0	$\frac{1}{2}$	1
house	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$	1
the	$\frac{1}{2}$	$\frac{1}{2}$	0	1

- At this point, it perhaps isn't obvious that this table will give us better alignment estimates, but it does:
- For Ia, we observe the following (ignoring the ϵ term):

$$\begin{aligned}\hat{P}(F,A|E) &= t(\texttt{casa}|\texttt{green}) t(\texttt{verde}|\texttt{house}) \\ &= (\frac{1}{2})(\frac{1}{4}) = \frac{1}{8}\end{aligned}$$

- For Ib:

$$\begin{aligned}\hat{P}(F,A|E) &= t(\texttt{verde}|\texttt{green}) t(\texttt{casa}|\texttt{house}) \\ &= (\frac{1}{2})(\frac{1}{2}) = \frac{1}{4}\end{aligned}$$

- For Ic:

$$\begin{aligned}\hat{P}(F,A|E) &= t(\texttt{casa}|\texttt{green})t(\texttt{verde}|\texttt{green}) \\ &= (\frac{1}{2})(\frac{1}{2}) = \frac{1}{4}\end{aligned}$$

- For Id:

$$\begin{aligned}\hat{P}(F,A|E) &= t(\texttt{casa}|\texttt{house})t(\texttt{verde}|\texttt{house}) \\ &= (\frac{1}{2})(\frac{1}{4}) = \frac{1}{8}\end{aligned}$$

- The calculations for II are similar.
- Updating the alignment counts for `green` gives us:
  - It aligns with `casa` in Ia ($\frac{1}{8}$) and Ic ($\frac{1}{4}$), to give a total of $\frac{3}{8}$.
  - It aligns with `verde` in Ib ($\frac{1}{4}$) and Ic (same), to give a total of $\frac{1}{2}$.
  - It never aligns with `la`.
- Our un-normalised counts (neglecting the ϵ terms) are now:

T	`casa`	`la`	`verde`	Total
`green`	$\frac{3}{8}$	0	$\frac{1}{2}$	$\frac{7}{8}$
`house`	$\frac{3}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{5}{4}$
`the`	$\frac{3}{8}$	$\frac{1}{2}$	0	$\frac{7}{8}$

- We can see that we have correctly observed that `green` is most likely to be `verde`, `house` to be `casa`, and `the` to be `la`. Summarising the normalised probabilities:

T	`casa`	`la`	`verde`	Total
`green`	$\frac{3}{7}$	0	$\frac{4}{7}$	1
`house`	$\frac{3}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	1
`the`	$\frac{3}{7}$	$\frac{4}{7}$	0	1

- Further iterations will continue to improve these counts, and to observe that Ib and IIa are the most likely alignments.