CS 124 / LING 180 From Languages to Information

Dan Jurafsky, Winter 2020

Part 1: Group Exercise

1. An IR system returns eight relevant documents and ten non-relevant documents. There are a total of twenty relevant documents in the collection. What is the precision of the system on this search, and what is its recall?

   Precision: 8/18
   Recall: 8/20

2. Draw the inverted index that would be built for the following document collection.

   Doc 1: new home sales top forecasts
   Doc 2: home sales rise in july
   Doc 3: increase in home sales in july
   Doc 4: july new home sales rise

   forecasts→1
   home→1→2→3→4
   in→2→3
   increase→3
   july→2→3→4
   new→1→4
   rise→2→4
   sales→1→2→3→4
   top→1

3. Compute cosines to find out whether Doc1, Doc2, and Doc3 will be ranked higher for the two-word query "Linus pumpkin", given these counts for the (only) 3 documents in the corpus:

term	Doc1	Doc2	Doc3
Linus	10	0	1
Snoopy	1	4	0
pumpkin	4	100	10

   Do this by computing the tf-idf cosine between the query and Doc1 and the cosine between the query and Doc2, and choose the higher value. You should the ltc.lnn weighting variation (remember that's ddd.qqq), using the following table:

Term frequency		Document frequency		Normalization	
n (natural)	$\text{tf}_{t,d}$	n (no)	1	n (none)	1
l (logarithm)	$1 + \log(\text{tf}_{t,d})$	t (idf)	$\log \frac{N}{\text{df}_t}$	c (cosine)	$\frac{1}{\sqrt{w_1^2+w_2^2+\ldots+w_M^2}}$
a (augmented)	$0.5 + \frac{0.5 \times \text{tf}_{t,d}}{\max_t(\text{tf}_{t,d})}$	p (prob idf)	$\max\{0, \log \frac{N-\text{df}_t}{\text{df}_t}\}$	u (pivoted unique)	$1/u$
b (boolean)	$\begin{cases} 1 & \text{if } \text{tf}_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$			b (byte size)	$1/CharLength^{\alpha}$, $\alpha < 1$
L (log ave)	$\frac{1+\log(\text{tf}_{t,d})}{1+\log(\text{ave}_{t \in d}(\text{tf}_{t,d}))}$				

term	Doc1	Doc2	Doc3
Linus	10	0	1
Snoopy	1	4	0
pumpkin	4	100	10

Let's start with our query "Linus pumpkin".

term	tf-raw	tf-wt
Linus	1	$1 + log(1) = 1$
pumpkin	1	$1 + log(1) = 1$

We don't need to do anything else for the query vector because we are simply using lnn weights.

Now let's do our calculations for Doc 1.

term	tf-raw	tf-wt	df	idf	wt	normalized
Linus	10	$1 + log(10) = 2$	2	$log(\frac{3}{2}) = 0.176$	$2 \cdot 0.176 = 0.352$	$\frac{0.352}{\sqrt{0.352^2+0.176^2+0^2}} = 0.894$
Snoopy	1	$1 + log(1) = 1$	2	$log(\frac{3}{2}) = 0.176$	$1 \cdot 0.176 = 0.176$	$\frac{0.176}{\sqrt{0.352^2+0.176^2+0^2}} = 0.447$
pumpkin	4	$1 + log(4) = 1.602$	3	$log(\frac{3}{3}) = 0$	$1.602 \cdot 0 = 0$	$\frac{0}{\sqrt{0.352^2+0.176^2+0^2}} = 0$

Now calculations for Doc 2.

term	tf-raw	tf-wt	df	idf	wt	normalized
Linus	0	0	0	0	0	0
Snoopy	4	$1 + log(4) = 1.602$	2	$log(\frac{3}{2}) = 0.176$	$1.602 \cdot 0.176 = 0.282$	$\frac{0.282}{\sqrt{0.282^2+0^2}} = 1$
pumpkin	100	$1 + log(100) = 3$	3	$log(\frac{3}{3}) = 0$	$3 \cdot 0 = 0$	$\frac{0}{\sqrt{0.282^2+0^2}} = 0$

Now calculations for Doc 3.

term	tf-raw	tf-wt	df	idf	wt	normalized
Linus	1	$1 + log(1) = 1$	2	$log(\frac{3}{2}) = 0.176$	$1 \cdot 0.176 = 0.176$	$\frac{0.176}{\sqrt{0.176^2+0^2}} = 1$
Snoopy	0	0	0	0	0	0
pumpkin	10	$1 + log(10) = 2$	3	$log(\frac{3}{3}) = 0$	$2 \cdot 0 = 0$	$\frac{0}{\sqrt{0.176^2+0^2}} = 0$

Now for the cosine similarity of each document, we take the dot product of the query vector and the normalized vector for that document. We can disregard the Snoopy element in each of the documents vectors because the query vector does not contain the word Snoopy. As such, we get the following:

$$\text{Doc1: } 1 \cdot 0.894 + 1 \cdot 0 = 0.894$$
$$\text{Doc2: } 1 \cdot 0 + 1 \cdot 0 = 0$$
$$\text{Doc3: } 1 \cdot 1 + 1 \cdot 0 = 1$$

Now we simply choose the highest score and that belongs to Doc3.

Doc3 will be ranked highest for the query "Linus pumpkin" , because between Doc1, Doc2, and Doc3, we see that Doc3 has a greater cosine similarity.

Part 2: Ethics Problems
(for discussion)

Part 3: Challenge Problems

1. Do modern web search engines use stemming? If so, are all suffixes removed or just some of them? How to search engines deal with Boolean terms like OR or AND? Do some experimenting with Google, Bing, DuckDuckGo, or your favorite search engines.

   Google and Bing do not use exactly use stemming, since searches for words with different suffixes return different results. But they clearly do some sort of lemmatization, since searches for words do return pages that only have the words in different forms. These search engines also distinguish between OR and AND.

2. Consider two documents A and B whose Euclidean distance is d and cosine similarity is c (using no normalization other than raw term frequencies). If we create a new document A' by appending A to itself and another document B' by appending $B$to itself, then:

   (a) What is the Euclidean distance between A' and B' (using raw term frequency)?

   $2d$

   (b) What is the cosine similarity between A' and B' (using raw term frequency)?

c

(c) What does this say about using cosine similarity as opposed to Euclidean distance in information retrieval?

Cosine similarity may be a better way to measure similarity of documents of different lengths.

3. Is it important to remove stop words in a system that uses idf in its weighting scheme?

No. Stop words tend to appear in many documents and therefore will receive a low idf score already.