CS124 Information Retrieval Group Work Extra Handout

Krishna Patel, TA Winter 2020

Cosine Similarity

As we've seen in class, if we have any two vectors, $\vec{a}$ and $\vec{b}$, we can find the cosine similarity between the two by using the following formula:

$$\frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|}$$

Here $|\vec{a}|$ is the magnitude of $\vec{a}$, calculated using: $\sqrt{a_1^2 + a_2^2 ... a_n^2}$, where a_1 is first element in the vector $\vec{a}$, and a_i is the i-th element in $\vec{a}$.

In other words, to find the cosine similarity between two vectors, you take the dot product of the two vectors and then normalize by (divide by) their magnitudes.

Ranked Retrieval

Now, in the context of information retrieval, let's say that our two vectors are a query vector $\vec{q}$ and a document vector $\vec{d}$. Given these two vectors, we *can* calculate the cosine similarity between them by using the formula above.

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)	$\frac{1}{\sqrt{w_1^2+w_2^2+...+w_M^2}}$
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)	$\frac{1+\log(\text{tf}_{t,d})}{1+\log(\text{ave}_{t \in d}(\text{tf}_{t,d}))}$				

However, often in the context of **ranked retrieval**, we utilize the *ltc.lnn* weighting and normalization pattern (described in the table above), where we apply *ltc* to the document vector, $\vec{d}$, and apply *lnn* to the query vector $\vec{q}$. Then, in order to calculate the

similarity between the two vectors, we simply take the dot product between the two:

$$\vec{v} \cdot \vec{d}$$

Let's investigate this a little further! The *c* in the *ltc* weighting applied to the document vector refers to "cosine normalization." If we compare the formula for cosine similarity and what we're doing here, we see that we're still taking the dot product between the two vectors, and we are still normalizing by the magnitude of $\vec{d}$. *But*, we're not normalizing by the query vector $\vec{q}$ anymore! Does this matter?

It turns out that it doesn't! This is because of the context in which we're calculating these scores. We're calculating these similarity scores for the purpose of ranked retrieval, where we have *1 query* and multiple documents. Since we're comparing all of the documents to the same query, the value of $|\vec{q}|$ (the magnitude of the query vector) is the same for every comparison. So, if we were to normalize by $|\vec{q}|$ as well, we'd be dividing all of the scores by the same number, which doesn't impact the overall ranking of documents at all (since when you divide a set of numbers by the same number, all you end up doing is scaling them).