CS824 Information Retrieval

Assignment 3

Name: Chengjun Shi
Student#: 200380163

Q1 Explain the ideas of a perfect ranking and an acceptable ranking. Why it is more meaningful to use an acceptable ranking?

Solution:

In information retrieval system, we have the following: $\succ_u$ denotes the user preference relation, $\succ_s$ denotes the system ranking, and u is the retrieval function.

We say a perfect ranking must satisfy:

$$\forall d_1, d_2 \in D,\ d_1 \succ_u d_2 \iff u(d_1) > u(d_2),$$

which means

$$d_1 \succ_u d_2 \iff d_1 \succ_s d_2.$$

A perfect ranking requires that the set of pairs $\succ_s$ is strictly equal to the set of pairs $\succ_u$.

We say an acceptable ranking must satisfy:

$$\begin{aligned} \forall d_1, d_2 \in D, d_1 \succ_u d_2 &\implies u(d_1) > u(d_2) \\ \equiv d_1 \succ_u d_2 &\implies d_1 \succ_s d_2, \end{aligned}$$

which means if a pair $(d_1, d_2) \in \succ_u$, then $(d_1, d_2) \in \succ_s$. The system ranking never contradicts user preference.

An acceptable ranking is practically more usefully and realistic. It is not that strict, so it is possible to find a real-valued retrieval function. It allows re-arrangement in equivalence classes in $D/\sim_u$.

Q2 Derive the ndpm formula.

Solution:

The evaluation of an IR system can be considered as how close is the user ranking to the system ranking, how close is $\succ_u$ to $\succ_s$. The distance of two rankings may be viewed as a distance between two sets. By comparing user ranking and system ranking, the distance between two rankings on any one pair (d_1, d_2) can be classified into three cases:

$$\delta(d_1, d_2) = \begin{cases} 0, \text{ if user and system agree on } (d_1, d_2); \\ 1, \text{ if one is } \succ \text{ and the other one is } \sim; \\ 2, \text{ contradict each other.} \end{cases}$$

Then the total distance between $\succ_u$ and $\succ_s$ may be calculated by:

$$\begin{aligned}\beta(\succ_u, \succ_s) &= \sum_{d_1, d_2 \in D} \delta(d_1, d_2) \\ &= 2 * \text{the number of contradiction pairs} + \text{the number of compatible pairs} \\ &= 2 * |\succ_u{}^c \cap \succ_s| + |\succ_u \cap \sim_s|.\end{aligned}$$

Furthermore, we have normalized distance as:

$$\begin{aligned}n\beta(\succ_u, \succ_s) &= \frac{\beta(\succ_u, \succ_s)}{\beta(\succ_u, \succ_u{}^c)} \\ &= \frac{2 * |\succ_u \cap \succ_s{}^c| + |\succ_u \cap \sim_s|}{2 * |\succ_u|}.\end{aligned}$$

Q3 Discuss the main ideas of vector space models.

* document representation
* query representation
* matching

Solution:

In VSM, we have some basic assumptions:

1. a document is represented as a m-dimensional vector, where m is the number of terms;
2. a query is represented as m-dimensional vector;
3. matching function is defined as
   a distance, or
   b similarity

   between two vectors.

In a binary vector space model, we consider the binary matrix B. Each row in B is a document d, where $\vec{d} = (B(d, t_1), B(d, t_2), \ldots, B(d, t_m))$. The query $\vec{q}$ is also a binary vector. We have the following method to measure the similarity between $\vec{d}$ and $\vec{q}$:

1. coordination level match: $sim(d, q) = \#\, of\ common\ terms$;
2. $sim(d, q) = \#\, of\ common\ terms/m$;

3. a. precision-based $sim_P(d,q) = \frac{\#\ of\ common\ terms}{\#\ of\ terms\ in\ d} = \frac{|d \cap q|}{|d|}$;

   b. recall-based $sim_R(d,q) = \frac{|d \cap q|}{|q|}$;

   c. balanced $sim_B(d,q) = \frac{|d \cap q|}{|d \cup q|}$.

In a non-binary vector space model, we can use different weighting formulas to represent a document such as TF matrix, normalized B, normalized TF, $TF \times IDF$, ... A document d and a query q can be represented by $\vec{d} = (d_1, d_2, \ldots, d_m)$ and $\vec{q} = (q_1, q_2, \ldots, q_m)$, which works for any weighting model. Measures are formulated as the following:

1. $\cos(\vec{d}, \vec{q}) = \frac{\vec{d} \cdot \vec{q}}{||\vec{d}|| \cdot ||\vec{q}||} = \sum_{i=1}^{m} \frac{d_i}{\sqrt{\sum_{j=1}^{m} d_j^2}} \cdot \frac{q_i}{\sqrt{\sum_{j=1}^{m} q_j^2}}$;

2. linear model $ret(\vec{d}, \vec{q}) = \vec{nd} \cdot \vec{q}$, where $\vec{nd}$ is a normalized vector;

3. a. Precision-oriented $\frac{\sum_{i=1}^{m} d_i \cdot q_i}{\sum_{i=1}^{m} d_i}$;

   b. Recall-oriented $\frac{\sum_{i=1}^{m} d_i \cdot q_i}{\sum_{i=1}^{m} q_i}$;

   c. Balanced $\frac{\sum_{i=1}^{m} d_i \cdot q_i}{\sum_{i=1}^{m} d_i + \sum_{i=1}^{m} q_i}$.

Q4 Discuss the main ideas of two probability distribution models.

* document representation

* query representation

* matching

Solution:

In a probability distribution model, each document is a probability distribution of terms. A document vector $\vec{d}$ is constructed by row-wise normalization, $\vec{d} = (\ldots, d_i = \frac{TF(t_i)}{\sum_{j=1}^{m} TF(t_j)}, \ldots)$. Based on utility theory, a query vector $\vec{q}$ can be considered as a utility function of terms based on user need, $\vec{q} = (\ldots, q_i = u(t_i), \ldots)$. The retrieval function is stated as $\vec{d} \cdot \vec{q} = \sum_{i=1}^{m} d_i \cdot q_i$. The meaning of it is the expectation of $\vec{q}$ based on $\vec{d}$. Thus, we could rank documents based on added utilities.

As discussed above, a document can be represented as a probability distribution P_d. We may also view a query vector $\vec{q}$ as a simple probability distribution P_q on the term set T. So, the query is consistent with the representation of documents as probability distributions. With this interpretation of a query, the potential relevance of a document can be determined by comparing its term probability distribution with the query's representation.

Suppose $\vec{d}_i = (d_{i1}, d_{i_2}, \ldots, d_{im})$ and $\vec{q} = (q_1, q_2, \ldots, q_m)$. A divergence measure can be introduced to compare similarity of two distributions and is defined by:

$$I(\vec{d}_i, \vec{q}) = \sum_{j=1}^{m} d_{ij} \cdot \log \frac{d_{ij}}{q_j}.$$

Based on Shannon's entropy function, we define an entropy function for a probability distribution:

$$H(P) = -\sum_{i=1}^{m} p_i \log p_i.$$

Then, we have a function β to measure the increase of entropy for combining the two discrete probability distributions, P_d and P_q.

$$\beta(P_d, P_q, \lambda) = H[\lambda \cdot P_d + (1-\lambda) \cdot P_q] - [\lambda \cdot H(P_d) + (1-\lambda) \cdot H(P_q)].$$

We know that $0 \leq \beta(P_d, P_q, \lambda) \leq 1$. The similarity between two probability distributions is defined by:

$$sim(P_d, P_q) = 1 - \beta(P_d, P_q, \lambda).$$

Q5 Discuss the main ideas of relevance feedback with respect to the following two methods for representing your judgments on a sample set of documents.

a. binary relevance

b. user preference

Solution:
1. Binary relevance:

Based on binary judgment, the set of documents D is divided into two disjoint subsets S and $\overline{S}$ according to a query q. S denotes the set of documents which are considered relevant to q by the IR system. $\overline{S}$ contains the remaining documents. In the set S, we have S^+ and S^-. Documents in S^+ are actually relevant, and documents in S^- are not actually relevant. The main task of feedback is to make q' to be more similar to documents in S^+ and to be less similar to document in S^-.

A new query is computed by:

$$\vec{q'} = \alpha \cdot \vec{q} + \beta \cdot \frac{1}{|S^+|} \cdot \sum_{d \in S^+} \vec{d} - \gamma \cdot \frac{1}{|S^-|} \cdot \sum_{d \in S^-} \vec{d},$$

a special case is $\alpha + \beta + \gamma = 1$.

2. User preference:

In an acceptable ranking, suppose we have two documents $d_1, d_2 \in D$. If $d_1 \succ_u d_2$, then the system gives $f(\vec{d_1}, \vec{q}) > f(\vec{d_2}, \vec{q})$, where $f(\vec{d}, \vec{q}) = \vec{d} \cdot \vec{q}$. We say the query q did a good job. If $f(\vec{d_1}, \vec{q}) \leq f(\vec{d_2}, \vec{q})$, then we say q did a bad job.

The preference relation induces a set of different vectors $B(\succ) = \{\vec{b} = \vec{d} - \vec{d'} \mid \vec{d} \succ \vec{d'}\}$. Let $\Gamma(\vec{q_0})$ denote the set of different vectors that $\vec{q_0}$ made an error. Repeat computing $\vec{q_k}$ and $\Gamma(\vec{q_k})$, until the set $\Gamma(\vec{q_k})$ is empty.

The algorithm is stated below:

Algorithm 1 Preference relevance feedback

1: $k = 0$
2: $\vec{q_0}$ is given by a user
3: Compute $\Gamma(\vec{q_k})$
4: **while** $\Gamma(\vec{q_k}) \neq \emptyset$ **do**
5: $\quad \vec{q_{k+1}} = \vec{q_k} + \sum_{\vec{b} \in \Gamma(\vec{q_k})} \vec{b}$
6: $\quad$ Compute $\Gamma(\vec{q_{k+1}})$
7: $\quad k = k + 1$
8: **end while**
9: output $\vec{q_k}$