Distance Function δ

- let (t, d) be a joint categorical random variable over terms and documents (td-matrix)

- range of t is a set of terms T with $m = |T|$

- range of d is a set of documents D with $n = |D|$. if all documents shall be equally important, d must be equally distributed, which demands the sum over each column to be the same. So absolute frequencies may not suffice...

- let's call δ the (normalized) "semantic diameter" of D:

  $$0 \leq \delta(D) = \frac{H(t) - H(t|d)}{ld(n)} \leq 1$$

  main part is: "absolute entropy minus conditional entropy"

  update: $H(t) - H(t|d)$ is actually the "mutual entropy" $I(t; d)$

  ... not to be confused with the "joint entropy" $H(t, d) = H(t) + H(d|t)$

- if the columns of the td-matrix are already term probability vectors $v_1, ..., v_n$ (values of each vector add up to 1), we can formulate δ more easily:

  $$\delta(v_1, ..., v_n) = \frac{1}{ld(n)} \left(H\left(\frac{1}{n}\sum_{i=1}^{n} v_i\right) - \frac{1}{n}\sum_{i=1}^{n} H(v_i) \right)$$

  note that $H(v)$ is the entropy of one vector $v = (t_1, ..., t_m)$:

  $$H(v) = -\sum_{i=1}^{m} t_i \; ld(t_i)$$

  main part is now: "entropy of average minus average of entropies" which is similar to variance: "average square minus square of average"

- ... for $n = 2$ and term probability vectors v, w it gets even simpler:

  $$\delta(v, w) = H\left(\frac{v + w}{2}\right) - \frac{H(v) + H(w)}{2}$$

  this is how the distance between 2 documents would be calculated. it is automatically in [0...1] because we need no more than 1 bit to distinguish 2 things.

Properties of δ

- interpretable:
  - the entropy of a term distribution roughly reflects how **un**specific that combination of terms (concepts) is....
  - "absolute entropy minus conditional entropy" reflects how much my uncertainty about the term distribution probably decreases, if i suddenly know, which of the n documents i have
  - "bit" seems to be the appropriate measuring unit to quantify semantic distance
- may be normalized. but if we want to compare diameters of a large and a small set (cluster) of documents, we just don't divide by $ld(n)$, so we get pure values in bit
- applicable to all objects that can be represented as probability vectors
- efficient:
  - in general as cheap as eucleadian distance
  - agglomerative clustering is very possible: if we save n, $H(t|d)$ and t (as probability vector) with each set (cluster) of documents, we can calculate the distance between arbitrary sets (clusters) in constant time (meaning that the sizes of both sets don't matter)
- matches intuitive notion of distance:
  - it's a **metric**
  - term co-occurence has strong impact, meaning that distance values are smaller than those of other functions
  - none saturation: more of the matching terms must lead to lower distance, example:

    $$u = (\frac{1}{2}, \frac{1}{2}, 0) \qquad v = (1, 0, 0) \qquad w = (\frac{1}{2}, 0, \frac{1}{2})$$

    obviously u and v are closer than u and w which is not the case with eucleadian distance

What should (could) be done with δ

- make shure no distance function like that was introduced in the literature or find that
- evaluate the effectiveness of δ itself or of clustering results on some benchmark test data
- show mathematically, that δ is a metric (it is. but that still needs to be proofed)
- show mathematically, that δ is in [0 ... 1]
- find mathematical formulation of properties implied by intuitive notion of distance and compare distance functions (especially cosine distance) with respect to these properties
- biggest challenge: develop an index structure that exploits the way δ is defined, just like octrees exploit eucleadian distance. with insight of the distance function and access to the feature vector, tighter bounds can be used and the search space shrinks faster... otherwise, only index structures for pure metric spaces can be used...

Applying δ to positioning by optimizing treemaps?

i implemented an algorithm that optimizes the order of a binary treemap/clustering in linear time for distance preservation. in the end, close leafs (data objects) are close to each other on the screen as well... an additional condition was to have square shaped spaces for the objects without wasting screen space.

this might not be new at all. i just never heard of using treemap algorithms for positioning hierarchical high dimensional data or of distance preservation as a criteria for treemaps... and i like the idea/topic...

References

[1] J. A. Aslam and M. Frost. An information-theoretic measure for document similarity. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 449–450, New York, NY, USA, 2003. ACM.

[2] J. Calmet and A. Daemi. From entropy to ontology. In *CYBERNETICS AND SYSTEMS 2004 - AT2AI-4: FROM AGENT THEORY TO AGENT IMPLEMENTATION*, 2004.

[3] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98, New York, NY, USA, 2003. ACM.

[4] D. Lin. An information-theoretic definition of similarity. In *ICML '98: Proceedings of the Fifteenth International Conference on Machine Learning*, pages 296–304, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.