The Vector Space Model in Information Retrieval - Term Weighting Problem

Nicola Polettini
Department of Information and Communication Technology,
Via Sommarive 14, 38050 Povo (TN) - Italy
University of Trento, polettini@itc.it

2004

Abstract

Many traditional information retrieval (IR) tasks, such as text search, text clustering or text categorization, have natural language documents as their first-class objects, in the sense that the algorithms that are meant to solve these tasks require explicit internal representations of the documents they need to deal with. In IR documents are usually given as extensional vectorial representation, in which the dimensions (features) of the vector representing a document are the terms occurring in the document. The approach to term representation that the IR community has almost universally adopted is known as *the bag-of-words approach*: a document d_j is represented as a vector of term weights $\overrightarrow{d_j} = \langle \omega_{1j}, ..., \omega_{rj} \rangle$, where r is the cardinality of the dictionary and $0 \leq \omega_{kj} \leq 1$ represents the contribution of term t_k to the specification of the semantics of d_j. This article analyses and compares many different bag-of-words approaches.

1 Introduction

In the vector space model, we represent documents as vectors. The success or failure of the vector space method is based on term weighting. There has been much research on term weighting techniques but little consensus on which method is best [17]. Term weighting is an important aspect of modern text retrieval systems [2]. Terms are words, phrases, or any other indexing units used to identify the contents of a text. Since different terms have different importance in a text, an important indicator - *the term weight* - is associated with every term[11]. The retrieval performance of the information retrieval systems is largely dependent on similarity measures. Furthermore, a term weighting scheme plays an important role for the similarity measure. There are three components in a weighting scheme:

$$a_{ij} = g_i * t_{ij} * d_j$$

Where g_i is the global weight of the i_{th} term, t_{ij} is the local weight of the i_{th} term in the j_{th} document, d_j is the normalization factor for the j_{th} document. Usually the three main components that affect the importance of a term in a text are the term frequency factor (tf), the inverse document frequency factor (idf), and document lenght normalization[12].

2 Local Term-Weighting

These formulas depend only on the frequencies within the document and they not depend on inter-document frequencies.

Formula for t_{ij}	**Description**
$\chi(f_{ij})$	Binary
f_{ij}	Term frequency
$K * \chi(f_{ij}) + (1 - K) * \frac{f_{ij}}{max_k\ (f_{kj})}$	Augmented Normalized Trem Frequency
$\log(f_{ij} + 1)$	Logarithm
$\chi(f_{ij}) * (\log(f_{ij}) + 1)$	Alternate Logarithm

Table 1: Local Term Weight Formulas

2.1 Binary

Binary formula gives every word that appears in a document equal relevance. This can be useful when the number of times a word appears is not considered important.

$$\chi(t) = \begin{cases} 1 & \text{if } t > 0 \\ 0 & \text{if } t = 0 \end{cases}$$

Terms are either present or absent, so the use of binary weights is often too limiting [6], because it doesn't provide consideration for partial matches.

2.2 Term frequency

This formula counts how many times the term occurs in a document. The more times a term t occurs in document d the more likely it is that t is relevant to the document. Used alone, favors common words and long documents. This formula gives more credit to words that appears more frequently, but often too much credit[17]. For instance, a word that appears ten times in a document is not usually ten times more important than a word that only appears one. Binary and term frequency weights are typically used for query weighting, where terms appear only once or twice. For document weighting, these weights are generally not best because binary does not differentiate between terms that appear frequently and terms that appear only once and because term frequency gives too much weight to terms that appear frequently. The logarithm formulas offer a middle ground.

2.3 Augmented normalized term frequency

This formula try to give credit to any word that appears and then give some additional credit to words that appear frequently. The formula gives a value of $K = 0.5$ for appearing in the document plus a bonus (no more than 0.5) that depends on the frequency. This formula was proposed by Croft [4] and parameterized by a value equal to K. Croft suggested that K must be set to something low (0.3) for large documents and to higher values (0.5) for shorter documents [11]. With this formula, the output value varies only between 0,5 and 1 for terms that appear in the document. By restricting the tf factors to a maximum value of 1.0, this tecnique only compensates the problem of the presence of higher term frequencies for normalization. So this tecnique turns out to be a "weak" form of normalization and favors the retrieval of long documents if used alone without another one normalization formula [1].

2.4 Logarithmic term frequency

Logarithms are a way to de-emphasize the effect of frequency. Literature proposes log and alternate log as the most used[17]. Logarithms are used to adjust within-document frequency because a term that appears ten times in a document is not necessarily ten times as important as a term that appears once in that document. Logarithms formulas decrease the effects of large differences in term frequencies.

3 Global Term-Weighting

These formulas are used to place emphasis on terms that are discriminating and they are based on the dispersion of a particular term throughout the documents. Global weighting tries to give a discrimination value to each term. Many schemes are based on the idea that the less frequently a term appears in the whole collection, the more discriminating it is. Global weighting is in general very successful. The use of global weighting can, in theory, eliminate the need for stop word removal since stop words should have very small global weights. In practice, however, it is easier to remove the stop words in the preprocessing phase so that there are fewer terms to handle.

Formula for g_i	**Description**
1	No changes
$log\left(\frac{n}{\sum_{k=1}^{n}\chi(f_{ik})}\right)$	Inverse Document Frequency (IDF)
$log\left(\frac{n}{\sum_{k=1}^{n}\chi(f_{ik})}\right)^2$	Squared Inverse Document Frequency
$log\left(\frac{n-\sum_{k=1}^{n}\chi(f_{ik})}{\sum_{k=1}^{n}\chi(f_{ik})}\right)$	Probabilistic Inverse Document Frequency
$\frac{\sum_{k=1}^{n} f_{ik}}{\sum_{k=1}^{n}\chi(f_{ik})}$	GFIDF
$1+\sum_{j=1}^{n}\frac{p_{ij}*\log(p_{ij})}{log(n)}$, $p_{ij}=\frac{f_{ij}}{\sum_{k=1}^{n} f_{ik}}$	Entropy

Table 2: Global Term Weight Formulas

3.1 No changes

Sometimes is useful to consider only term frequency terms, when term frequencies are very small or when we are interested to emphasize the term frequencies in a document.

3.2 Inverse document frequency (IDF)

Inverse Document Frequency (IDF) is a popular measure of a word's importance. It's defined as the logarithm of the ratio of number of documents in a collection to the number of documents containing the given word[16]. This means rare words have high IDF and common words have low IDF. For example we obtain an output value eqaul to 0 if the given term appears in every document. The weight increases as the number of documents in wich the term appears decreases. High value indicates that the word occurs more often in this document than average. Examples for a collection of 10000 documents:

$$\log\left(\frac{10000}{10000}\right) = 0 \ ; \ \log\left(\frac{10000}{20}\right) = 2.698 \ ; \ \log\left(\frac{10000}{1}\right) = 4 \ ;$$

It's the most used global term weighting formula[10]. Sometimes is used alone without local term weight formula.

3.3 Other IDF Schemes

- **Squared Inverse Document Frequency:** Used rarely as a variant of IDF scheme[8].
- **Probabilistic Inverse Document Frequency:** Another IDF weight. It assigns weights ranging from $-\infty$ for a term that appears in every document to $log(n-1)$ for a term that appears in only one document. It differs from IDF because probabilistic inverse actually awards negative weight for terms appearing in more than half of the documents in the collection, and the lowest weight gives, is one[17].
- **GFIDF:** It computes the ratio of the total number of times the term appears in the collection to the number of documents it appears in. Here, if a term appears once in every document or once in one document, it is given a weight of one, the smallest possible weight. A term that is frequent relative to the number of documents in which it appears gets a large weight. This weight often works best when combined with a different global weight on the query vector[5].

3.4 Entropy

Entropy is based on information theoretic ideas and is the most sophisticated weighting scheme. It assigns weights between 0 and 1 for a term that appears in only one document. If a term appears once in every document, then that term is given a weight of zero. If a term appears once in one document, then that term is given a weight of one. Any other combination of frequencies will yield a weight somewhere between zero and one. Entropy is a useful weight because it gives higher weight for terms that appear fewer times in a small number of documents. So this formula takes into account the distribution of terms over documents [5].

4 Normalization

The third component of the weighting scheme is the normalization factor, which is used to correct discrepancies in document lengths. It is useful to normalize the document vectors so that documents are retrieved independent of their lengths. It's important. If we do not, short documents may not be recognized as relevant. Automatic information retrieval systems have to deal with documents of varying lenghts in text collection. Document lenght normalization is used to fairly retrieve documents of all lenghts [14] and it's used to remove the advantage that the long documents have in retrieval over the short documents. Two main reasons that necessitate the use of normalization in term weights are:

- **Higher term frequencies:** long documents usually use the same terms repeatedly. As a result, the term frequency factors may be large for long documents.

- **Number of terms:** long documents also have different numerous terms. This increases the number of matches between a query and a long document, increasing the chances of retrieval of long documents in preference over shorter documents.

It's also possible no normalization but the 2-norm (cosine normalization) is the most popular.

Formula for d_j	**Description**
1	No changes
$\sqrt{\sum_{k=1}^{n}(g_k * t_{kj})^2}$	Cosine Normalization
$\sum_{k=1}^{n}(g_k * t_{kj})$	Sum of weights
$\sum_{k=1}^{n}(g_k * t_{kj})^4$	Fourth normalization
$max_{k=1}^{n}(g_k * t_{kj})$	Max weight normalization
$\frac{1}{(1-slope)*pivot+(slope*l_j)}$	Pivoted unique normalization

Table 3: Normalization Formulas

4.1 No changes

It's used when we want to emphasize long documents over the short documents. Sometimes no normalization it's used when we use *Augmented Normalized Term Frequency* as local term weight formula. In fact in this formula there is a kind of normalization of individual tf weights using the maximum tf in the document. Normalization of tf weights by maximum tf in a document can possibly be used, but Singal et al.[14] believe that max_{tf} is not an optimal normalization scheme to fix the higher term frequencies problem. For example, if a query term occurs five times in a document D_1 in which all other terms occur just once, then D_1 is possibly more interesting than another document D_2 in which the same query term occurs five times as well, but all other terms also occurr five times each. If max_{tf} is used for normalization, D_1 has no advantage over D_2 since the query term will have the same weight in both the documents.

4.2 Cosine Normalization

Cosine Normalization resolves both the reasons for normalization (Higher term frequencies, number of terms) in one step. Higher individual term frequencies increase individual w_i values, increasing the penalty on the term weights. Also, if a document has more terms, the number of individual weights in the cosine factor increases, yielding a higher normalization factor [14] [13]. So longer documents have smaller individual term weights and smaller documents are favored over longer ones in retrieval. It's the most used and popular normalization.

4.3 Other Cosine Normalization Schemes

- **Sum of weights:** It's rarely used as a variant of cosine normalization[8].
- **Fourth normalization:** It's rarely used as a variant of cosine normalization[8].

4.4 Max weight normalization

It's not a real normalization. It assigns weights between 0 and 1, but this formula doesn't take into account the distribution of terms over documents. It's useful when we want to give high importance to the most relevant weighted terms within a document [8].

4.5 Pivoted unique normalization

The problem using cosine normalization is that often we have high values in the normalization factor. The higher the value of the normalization factor for a document is, the lower are the chances of retrieval for that document. Pivoted unique normalization, a relatively new normalization method, tries to correct this problem and also try to solve the problem of favoring short documents [3]. In the formula, l_j is the number of distinct terms in document j. Thanks to the suggestion of Singal et al.[14], slope is set to 0,2 and pivot is set to the average number of distinct terms per document in the entire collection. The basic principle behind pivoted normalization methods is to correct for discrepancies based on document length between the probability that a document is relevant and the probability that the document will be retrieved. Using another normalization factor, such as $\frac{1}{l_j}$, a set of documents is retrieved, and the retrieval and the relevance curves are plotted against document length. The point at which these curves intersect is the *pivot*. The documents on the left side of the pivot generally have a higher probability of being retrieved than they have of being relevant, and the documents on the right side of the pivot generally have a higher probability of being relevant than they have of being retrieved. The normalization factor can now be pivoted at the pivot and "tilted" so that the normalization factor can be increased or decreased to better match the probabilities of relevance and retrieval. The amount of "tilting" needed becomes a parameter of the weighting scheme and is called *the slope* [15].

4.6 Normalization problems

1. **Document Length Normalization Problems:** Long documents have an unfair advantage:

   - They use a lot of terms so they get more matches than short documents.
   - They use the same words repeatedly so they have much higher term frequencies.
   - Normalization seeks to remove these effects:
     - Related somehow to maximum term frequency.
     - But also sensitive to the of number of terms.
     - If you don't normalize short documents may not be recognized as relevant.

2. **Vantages - Disadvantages:** One of the problems is solved: shorter (and presumably) more focused documents receive a higher normalized score than longer documents with the same matching terms. On the other hand, we've got a new problem: now shorter documents are generally preferred over longer ones.

3. Some of this problems are solved using pivoted unique normalization but it depends a lot from the dataset used.

In conclusion normalize the term weights (so longer vectors are not unfairly given more weight) has vantages and disadvantages. It's often used to force all values to fall within a certain range, usually between 0 and 1, inclusive.

5 Conclusion

Salton and Buckley [11] confirms that the most used document term weighting is obtained by the inner product operation of the within document term frequency and the inverse document frequency, all normalized by the lenght of the document. So the most used term weighting is $tf * idf$ normalized by cosine:

$$\frac{f_{ij} * log\left(\frac{n}{\sum_{k=1}^{n} \chi(f_{ik})}\right)}{\sqrt{\sum_{k=1}^{n} \left(f_{ij} * log\left(\frac{n}{\sum_{k=1}^{n} \chi(f_{ik})}\right)\right)^2}}$$

Salton and Buckley proposed (augmented normalized term frequency * idf) normalized by cosine as the best term weighting scheme [7]:

$$\frac{\left(0,5 * \chi(f_{ij}) + 0,5 * \frac{f_{ij}}{max_k\ f_{kj}}\right) * log\left(\frac{n}{\sum_{k=1}^{n} \chi(f_{ik})}\right)}{\sqrt{\sum_{k=1}^{n} \left((0,5 * \chi(f_{ij}) + 0,5 * \frac{f_{ij}}{max_k\ (f_{kj})}) * log\left(\frac{n}{\sum_{k=1}^{n} \chi(f_{ik})}\right)\right)^2}}$$

Very interesting is also the formula proposed by Singal et al. [14]. They use $tf * idf$ weights normalized using the pivoted unique normalization:

$$\frac{f_{ij} * log\left(\frac{n}{\sum_{k=1}^{n} \chi(f_{ik})}\right)}{(1-0,2) * pivot + (0,2 * l_j)}$$

where l_j is the number of distinct terms in document j, *pivot* is set to the average number of distinct terms per document in the entire collection, *slope* is set at the value 0,20. Infact Singal et al. [14] found with this type of normalization substantial improvements over cosine normalization for all the collections, fixing a constant slope value of 0,20, effective across collections. They showed the weakness of the cosine function for very long documents and proposed the pivoted normalization that can be used, with these values, in general applications.

References

[1] J. Broglio, J.P.Callan, W.B.Croft, and D.W.Nachbar. *Document retrieval and routing using the INQUERY system.* In D.K. Harman, editor, Proceedings of the Third Text Retrieval Conference (TREC-3), pages 29-38. NIST Special Publication 500-225, April 1995.

[2] Chris Buckley. *The importance of proper weighting methods.* In M. Bates, editor, Human Language Technology. Morgan Kaufman, 1993.

[3] Erica Chisholm and Tamara G. Kolda. *New Term Weighting formulas for the Vector Space Method in Information Retrieval.* Computer Science and Mathematics Division. Oak Ridge National Laboratory, 1999.

[4] W. B. Croft. *Experiments with representation in a document retrieval system.* Information Technology: Research and Development, 2:1-21, 1983.

[5] Susan T. Dumais. *Enhancing Performance in Latent Semantic Indexing (LSI) Retrieval.* Bellcore, 21236, 1992.

[6] Zachary G. Ives. *Information Retrieval.* University of Pennsylvania, CSE 455 - Internet and Web Systems, 2004.

[7] Yunjae Jung, Haesun Park, and Ding-zhu Du. *An effective Term-Weighting Scheme for Information Retrieval.* Technical Report TR 00-008, Department of Computer Science and Engineering, University of Minnesonta, Minneapolis, Usa, 2000.

[8] Ray Larson, Marc Davis. *SIMS 202: Information Organization and Retrieval.* UC Berkeley SIMS, Lecture 18: Vector Representation, 2002.

[9] Lavelli, Sebastiani, Zanoli. *Distributional Term Representations: An experimental Comparison.* CIKM 2004, November 8-13, Washington, DC, USA.

[10] Kishore Papineni. *Why Inverse Document Frequency?.* IBM T.J. Watson Research Center Yorktown Heights, New York, Usa, 2001.

[11] Gerald Salton and Chris Buckley. *Term weighting approaches in automatic text retrieval.* Information Processing and Management, 24(5):513-523, Issue 5. 1988.

[12] Gerard Salton and M.J.McGill. *Introduction to Modern Information Retrieval.* McGraw Hill Book Co., New York, 1983.

[13] Gerard Salton, A. Wong, and C.S. Yang. *A vector space model for Information Retrieval.* Journal of the American Society for Information Science, 18(11):613-620, November 1975.

[14] A. Singal, C. Buckley, M. Mitra, and G. Salton. *Pivoted document length normalization.* Technical Report TR95-1560, Department of Computer Science, Cornell University, Ithaca, New York, 1995.

[15] A.Singal, Gerard Salton, Mandar Mitra, and Chris Buckley. *Document Lenght Normalization.* Information Processing and Management. Technical Report TR95-1529, Department of Computer Science, Cornell University, Ithaca, New York, July 1995.

[16] K. Sparck Jones. *A statistical interpretation of term specificity and its application in retrieval.* J. Documentation, 28(1):1-21, 1972.

[17] Tamara Gibson Kolda. *Limited-Memory Matrix Methods with Applications.* Applied Mathematics Program. University of Maryland at College Park, 59-68, 1997.