Inf Retrieval (2011) 14:466–487
DOI 10.1007/s10791-011-9163-y

Improving document clustering using Okapi BM25 feature weighting

John S. Whissell · Charles L. A. Clarke

Received: 3 May 2010 / Accepted: 5 February 2011 / Published online: 20 February 2011

Abstract We investigate the effect of feature weighting on document clustering, including a novel investigation of Okapi BM25 feature weighting. Using eight document datasets and 17 well-established clustering algorithms we show that the benefit of *tf-idf* weighting over *tf* weighting is heavily dependent on both the dataset being clustered and the algorithm used. In addition, binary weighting is shown to be consistently inferior to both *tf-idf* weighting and *tf* weighting. We investigate clustering using both BM25 term saturation in isolation and BM25 term saturation with *idf*, confirming that both are superior to their non-BM25 counterparts under several common clustering quality measures. Finally, we investigate estimation of the $k1$ BM25 parameter when clustering. Our results indicate that typical values of $k1$ from other IR tasks are not appropriate for clustering; $k1$ needs to be higher.

keywords Document clustering · Feature weighting · Okapi BM25

1 Introduction

Clustering is a popular data mining area, with document clustering in particular receiving much attention (Slonim and Tishby 2000; Steinbach et al. 2000; Zhao and Karypis 2002; Beil et al. 2002; Fung et al. 2003; Xu et al. 2003; Sevillano et al. 2006; Hu et al. 2009 to give just a fraction of the publications in the past decade on this subject). The amount of attention received is likely due to the massive amount of electronic document collections available and the desire (or need) to automatically organize them. Certain assumptions regarding the weighting of text features are nearly ubiquitous in this work. In this paper we explore some of these assumptions, investigating the effect of typical text feature

J. S. Whissell (✉) · C. L. A. Clarke
David R. Cheriton School of Computer Science, University of Waterloo,
Waterloo, ON N2L 3G1, Canada
e-mail: jswhisse@uwaterloo.ca

C. L. A. Clarke
e-mail: claclark@uwaterloo.ca

weighting on document clustering algorithms. We aim to determine whether we can improve on these standard term weighting strategies.

Before we begin our discussion, we give some preliminary notation. Let X be a set of documents we want to cluster, and X_i be the ith document of X. Each X_i is represented using the standard vector space model representation

$$X_i = [x_{i1}, x_{i2}, \ldots, x_{im}]. \tag{1}$$

x_{ij} is the weight of feature j to document i. When raw term counts are used to generate each X_i

$$x_{ij} = tf_{ij}, \tag{2}$$

where tf_{ij} is the *term frequency* of j in i. Typical practice in clustering is to incorporate an *inverse document frequency (idf)* component into the feature weighting:

$$x_{ij} = tf_{ij} \log\left(\frac{n}{n_j}\right), \tag{3}$$

where n is the number of documents in the X, and n_j is the number of documents in X that contain term j. To avoid unfavorable biases based on different document lengths, it is also a standard to make each document unit Euclidean length ($||X_i||_2 = 1$) by setting each x_{ij} as follows:

$$x_{ij} = \frac{x_{ij}}{\sqrt{\sum_{j=1}^{m} x_{ij}^2}} \tag{4}$$

It is common in document clustering literature to use *tf-idf* weighting, mostly with length normalization (Slonim and Tishby 2000; Steinbach et al. 2000; Zhao and Karypis 2002; Xu et al. 2003; Zhao and Karypis 2004; Hu et al. 2009; Aljaber et al. 2010 to name a few such works).

There is some research from fields related to clustering, such as classification, that indicate that *idf* is an important part of feature weighting for those fields, while *tf* is (surprisingly) not as useful (Wilbur and Kim 2009). Such results suggest that the same might be true of clustering. Despite this, we show in this paper that the inclusion of an *idf* component to *tf* is not necessarily beneficial in clustering. While *idf* is shown to generally have a small positive effect on clustering results, for some datasets *idf* is shown to be harmful for a wide range of algorithms. In addition, we show that certain algorithms are biased towards certain evaluation measures, and that the evaluation measures are not entirely consistent with each other.

We also investigate the use of term frequency in feature weighting. Specifically, we compare using *tf* versus binary feature weights:

$$x_{ij} = \begin{cases} 1, & \text{if } tf_{ij} > 0; \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Equation 5 assigns a value of 1 if a term occurs in a document, and 0 otherwise. We show that *tf* outperforms binary weighting, indicating that simple term presence/absence indicators are inferior to using term count information when document clustering.

A novel contribution of this paper is our investigation of Okapi BM25 (*BM25*) feature weighting. Only recently has BM25 been seriously considered in document clustering (de

Vries and Geva 2008; Bashier and Rauber 2009; Whissell et al. 2009; D'hondt et al. 2010; Kutty et al. 2010); with works that do use BM25 still being a small minority. Bashier and Rauber (2009) investigate relevance feedback using clustering. de Vries and Geva (2008) use BM25 weighting when clustering XML documents, but offer no comparison to *tf-idf* weighting using their clustering method. Kutty et al. (2010) also cluster XML documents using BM25 weighting, with the authors showing an improvement over *tf-idf*. D'hondt et al. (2010) use BM25 and *tf-idf*, but there is no direct comparison between their BM25 and *tf-idf* results.

An examination of the literature discussed above suggests that clustering using BM25 feature weighting is promising, but it also reveals a number of areas where research is lacking; we consider a few of these areas specifically in this paper. First; no broad investigation of the suitability of BM25 feature weighting for document clustering has been done, the rational for its use, up to this date, has simply been that it has worked well in other applications. Second, suitable parameter values for BM25 when document clustering have not been investigated, researchers have simply adopted default values for them. Finally, no work has assessed the merits of using just the BM25 term saturation component as a feature weight. Our previous work (Whissell et al. 2009) is the closest to such a task, but it does not offer comparisons, nor focus on the merit of, various weighting functions in a standard clustering experiment.

We show that replacing the *tf* in *tf-idf* weighting (Eq. 3) with the BM25 term saturation component, and changing nothing else about how the clustering is performed, produces results superior to *tf-idf* weighting in an extensive test. We also investigate the use of just the BM25 term saturation component as a feature weight, which we show outperforms *tf*. Parameter estimation for $k1$ in BM25 is also investigated, with our research leading to the conclusion that typical values for $k1$ from other tasks such as ad-hoc retrieval are unsuitable, $k1$ should be higher to achieve better clustering results.

The rest of the paper proceeds as follows. Section 2 describes our *tf, tf-idf*, and binary weighting experiments and the datasets, clustering algorithms, and evaluation measures used in them. Section 3 discusses the results of these experiments, highlighting some key discoveries and analyzing why they occurred. Section 4 demonstrates that BM25-weighted document representations produce superior clusterings when compared to their non-BM25 counterparts. Section 5 gives our conclusion and discussion of future avenues of research.

2 Experimental setup

In this section we describe our *tf, tf-idf*, and binary weighting experiments and the datasets, clustering algorithms, and evaluation measures used in them.

Our goal was to test the effect *tf, tf-idf*, and binary weighting has on document clustering results. To that end, we selected 17 clustering algorithms and eight document datasets, these are described in Sects. 2.1 and 2.2, respectively. For each dataset we generated three representations: one using *tf*, another using *tf-idf*, and a final one using binary weighting. The definitions used for the weighting functions, while creating the representations, were exactly as detailed in Sect. 1 All three weightings were length normalized. Each clustering algorithm was run on each representation with two to 30 clusters. This gave a total of 11,832 (8*3*17*29) clusterings. These clusterings were evaluated using the four evaluation measures described in Sect. 2.3 The results of the evaluation measures on the clusterings are used in Sect. 3 to analyze the effect of the weightings. The following

Table 1 The datasets used in our experiments

Dataset	# of Doc	# of Terms	# of Classes
fbis	2,463	2,000	17
new3	9,556	36,306	44
tr31	927	10,128	7
tr41	878	7,454	10
tr45	690	8,261	10
re0	1,504	2,886	13
re1	1,657	3,758	25
wap	1,560	8,460	20

subsections detail the specific datasets, clustering algorithms, and evaluation measures we used.

2.1 Datasets

We used a total of eight datasets in our experiment, all of which are available at the Karypis Lab[1] in the form of preprocessed term frequency count vectors. The vectors for each dataset could be created from base documents using a simple script called doc2mat.[2] Conceptually, doc2mat performs the following operations to generate the vectors for the datasets: (1) all non-alphanumeric characters are converted to whitespace; (2) documents are tokenized using whitespace as a separator; (3) a simple stopword list (built in to the program) filters out all stopwords; (4) a Porter stemmer is applied to the tokens; (5) tokens containing any non-alphabetic characters are discarded and all other tokens are case-normalized (lower case); (6) the remaining tokens are used to generate the terms for the dataset; and (7) the final term count vectors are created using the results of steps (5) and (6). We did not apply doc2mat ourselves, instead we simply used the preprocessed vectors provided by the site owners (it should be further noted that the default parameter settings of doc2mat do not match those discussed here, although doc2mat is easily configured to match them).

The eight datasets have been used in numerous publications (Zhao and Karypis 2002; Xu et al. 2003; Fung et al. 2003; Steinbach et al. 2000; Beil et al. 2002) and may thus be considered as standard test sets for document clustering. Table 1 summarizes their characteristics.

The document collections *new3, tr51, tr41*, and *tr31* are derived from collections used at TREC (Text REtrieval Conference[3]). The *fbis* collection is from the Foreign Broadcast Information Service dataset in TREC-5. The first 2000 of the fbis documents were used in our tests. This allowed us to use a standard Java matrix package (*JAMA*[4]) which required that the number of dimensions be greater than or equal to the number of objects when applying singular value decomposition (fbis has 2000 dimensions). *Re0* and *re1* are from the Reuters-21578 text categorization test collection distribution 1.0.[5] The *wap* collection

[1] http://glaros.dtc.umn.edu/gkhome/views/cluto/download

[2] http://glaros.dtc.umn.edu/gkhome/files/fs/sw/cluto/doc2mat.html

[3] http://trec.nist.gov

[4] http://math.nist.gov/javanumerics/jama/

[5] http://www.daviddlewis.com/resources/testcollections/reuters21578/

Table 2 The clustering algorithms used in our experiments

Algorithm	Short	Reference
K-means	Kmeans	Lloyd (1982)
Partition around medoids	PAM	Kaufman and Rousseeuw (1990)
Repeated bisecting *k*-means	RB-Kmeans	Steinbach et al. (2000)
Unnormalized spectral	Spect-Un	von Luxberg (2007)
Random walk spectral	Spect-RW	Shi and Malik (2000)
Symmetric spectral	Spect-Sy	Ng et al. (2001)
Principle component analysis+Kmeans	PCA-Kmeans	Pearson (1901)
Non-negative matrix factorization	NC-NMF	Xu et al. (2003)
Unweighted pair group method	UPGMA	Kaufman and Rousseeuw (1990)
Single linkage	Slink	Jain et al. (1999)
Complete linkage	Clink	Jain et al. (1999)
Repeated bisecting I2	RB-I2	Zhao and Karypis (2002)
Repeated bisecting H1	RB-H1	Zhao and Karypis (2002)
Direct I2	Direct-I2	Zhao and Karypis (2002)
Direct H1	Direct-H1	Zhao and Karypis (2002)
Agglomerative I2	Agglo-I2	Zhao and Karypis (2002)
Agglomerative H1	Agglo-H1	Zhao and Karypis (2002)

is the from the WebACE project (Boley et al. 1999). Each document of the *wap* dataset was a single web page in the Yahoo! directory.

2.2 Clustering algorithms

Of the 17 clustering algorithms used in our experiments, all but the six based on Zhao and Karypis (2001, 2002, 2004) were independently implemented by one of the authors. Where possible, the implementations were validated through comparisons to previously published results. The algorithms using the objective functions from Zhao and Karypis (2001, 2002, 2004) were performed using the authors' own clustering toolkit.[6]

Selection of the clustering algorithms was based on the following criterion: (1) Were they well-established algorithms? (2) Did they take a pre-specified number of clusters as a parameter and produce hard clusters? (3) Together, did the set of algorithms cover a breadth of the well-established techniques for document clustering? (4) Together, did the set of algorithms include those algorithms reported to produce good results in previous research? This last requirement was especially important for a meaningful analysis of the effects of *tf-idf* and other term weightings. Table 2 lists the 17 clustering algorithms we used. Below we give a brief description of each method.

Our *Kmeans* algorithm uses Lloyd's method (Lloyd 1982) with the initial centroids being selected randomly from the vectors of the dataset. We ran the algorithm 20 times for each value of *k* and kept only best result according to the *Kmeans* internal objective function. *PAM* (Kaufman and Rousseeuw 1990) was included in our study as a different

[6] http://glaros.dtc.umn.edu/gkhome/views/cluto/download

approach to optimizing the same objective function as our *Kmeans* algorithm. It uses medoid swapping. *RB-Kmeans* repeatedly splits the dataset using *Kmeans*. Binary splitting is used, with the largest remaining cluster split at each iteration (Steinbach et al. 2000).

We selected three varieties of spectral clustering. For each of these three methods the weighted adjacency matrix W was generated through an r-nearest neighbor scheme with cosine similarity using $r = 20$. *Spect-Un* clusters on the k eigenvectors of the Laplacian $L = D - W$, where D is the degree matrix of W. For details on the Laplacian for *Spect-RW* see Shi and Malik (2000), and for *Spect-Sy* see Ng et al. (2001). The clustering of the eigenvectors for all three methods was done using our *Kmeans* algorithm.

PCA (Pearson 1901) is a common dimensionality reduction technique based on singular value decomposition. For our PCA algorithm, we first applied PCA to produce an $n \times 20$ reduced document feature space. Dimensionality reduction was followed by the application of our *Kmeans* algorithm.

Non-negative matrix factorization is a relatively recent document clustering technique that has been shown to be highly effective, the variety we implemented is discussed by Xu et al. (2003). Note that we used *NC-NMF* as the authors showed it to be more effective than simple *NMF*.

UPGMA (Kaufman and Rousseeuw 1990), Slink (Jain et al. 1999), and Clink (Jain et al. 1999) are from the same family of agglomerative clustering algorithms. In each of these methods every document begins as a singleton cluster and clusters are progressively merged with the best similarity. In Slink similarity between clusters is equal to the similarity of their closest documents. In Clink the farthest documents are used. In UPGMA the average similarity between all documents is used. Cosine similarity was used for all three of these methods.

Finally, we selected two of the objective functions from Zhao and Karypis (2001); Zhao and Karypis (2002): *I2* and *H1*. For each of these, we used three distinct optimization methods: repeated bisection, direct (partitional), and agglomerative. This gave us a total of six algorithms. For details on their exact implementations, readers can consult Zhao and Karypis (2001); Zhao and Karypis (2002); Zhao and Karypis (2004) as we used the authors' own clustering toolkit to perform these algorithms. The *I2* function is essentially the *Kmeans* objective function except any similarity metric may be used in the calculation. The *H1* function is *I*1/*E*1, where *E1* is an objective function based around minimizing the weighted similarity of cluster centroids from the centroid of the whole dataset.

The above algorithms are by no means a full list of document clustering algorithms. Beyond just the variants of what we used there are some entirely different methods such as those based on frequent itemsets (Fung et al. 2003; Beil et al. 2002), the information bottleneck (Slonim and Tishby 2000), numerous model based approaches (*PLSA* Hofmann 1999, etc.), and so on. Still, we believe we implemented a sufficient set of algorithms to perform our weighting experiments.

2.3 Evaluation measures

We used four evaluation measures in our experiments: *normalized mutual information (NMI)*; *F-measure (FQ)*; a *purity measure (PQ)*; and an *entropy measure (EQ)*. Let A be a clustering of a X, and let B be the true labeling of X. Let a be a cluster of A, and let b be a class of B. We define $p(a) = \frac{|a|}{n}$, $p(b) = \frac{|b|}{n}$, and $p(a, b) = \frac{|a \cap b|}{n}$.

The variant of NMI we use was presented by Strehl and Ghosh (2002):

$$\mathrm{NMI}(A;B) = \frac{\mathrm{I}(A;B)}{\sqrt{\mathrm{H}(A)\mathrm{H}(B)}}, \tag{6}$$

where $\mathrm{I}(A; B)$ is the mutual information between A and B, defined as:

$$\mathrm{I}(A;B) = \sum_{b\in B}\sum_{a\in A} p(a,b)\log\left(\frac{p(a,b)}{p(a)p(b)}\right), \tag{7}$$

and where $\mathrm{H}(A)$ is the entropy of A, defined as:

$$\mathrm{H}(A) = \sum_{a\in A} p(a)\log(p(a)). \tag{8}$$

NMI is symmetric, with a value between zero and one. In a sense, it measures how much A tells us about B and vice versa, with higher NMIs indicating A is a better clustering.

The form of F-measure we used is based on F1 (van Rijsbergen 1979):

$$\mathrm{F1}(a,b) = \frac{2 * \mathrm{Precision}(a,b) * \mathrm{Recall}(a,b)}{\mathrm{Precision}(a,b) + \mathrm{Recall}(a,b)} \tag{9}$$

where

$$\mathrm{Precision}(a,b) = \frac{p(a,b)}{p(a)} \tag{10}$$

$$\mathrm{Recall}(a,b) = \frac{p(a,b)}{p(b)}. \tag{11}$$

F1 is defined on an individual cluster and class. Our full F-measure quality is

$$\mathrm{FQ}(A,B) = \sum_{b\in B} p(b)\max_{a\in A}\mathrm{F1}(a,b). \tag{12}$$

FQ was applied, exactly as explained here (although using different notion), to hierarchical clusterings in Zhao and Karypis (2002). FQ looks for the best cluster to represent each class in B. The individual F1 scores are weighted by their true label class sizes in the final scoring. As with NMI, FQ is between zero and one with higher scores indicating a better clustering. One potential issue with this formula is that, as many classes may map to a single cluster, but each individual class only maps to one cluster, there may be 'classless' clusters. A classless cluster does not effect the score of this measure, leading us to question how good of an evaluation of a clustering's overall quality this measure really is. Nevertheless, measures of this form have been used in previous comparisons of document clustering algorithms (Steinbach et al. 2000; Beil et al. 2002; Zhao and Karypis 2002), so we believe it is appropriate to apply it here.

The final two evaluation measures we used were a purity measure (Zhao and Karypis 2001), defined as:

$$\mathrm{PQ}(A,B) = \sum_{a\in A} p(a)\max_{b\in B}\frac{p(a,b)}{p(a)}, \tag{13}$$

and an entropy measure (Zhao and Karypis 2004), defined as:

$$\mathrm{EQ}(A,B) = 1 - \sum_{a \in A} p(a) \frac{1}{-log(q)} \mathrm{E}(a,b), \tag{14}$$

where

$$\mathrm{E}(a,b) = \sum_{b \in B} \frac{p(a,b)}{p(a)} \log\left(\frac{p(a,b)}{p(a)}\right). \tag{15}$$

In the equation for the entropy measure, q is the number of classes in B.

PQ treats each cluster as representing a single class; the class of which it contains the most documents. This mapping is done irrespective of how other clusters are mapped to classes, so many clusters may be assigned to the same class. PQ has a maximum of one (perfect).

EQ assesses the quality of each cluster by examining the distribution of class labels it contains. The notion behind this measure is that good clusters are ones that contain mostly one class, while a cluster containing classes in equal proportions is the worst possible. The exact quality value of each $a \in A$ is simply the entropy of the distribution of its class labels (Eq. 15, for clarity we note that $p(a, b)/p(a)$ is just the fraction of cluster a that is class b), weighted by the size of the cluster ($p(a)$). The $\log(q)$ component is included to normalize Eq. 8 between zero and one. As a final step, the weighted total entropy measure is subtracted from one. This makes EQ consistent with the other three measures: higher is better.

3 Effects of document feature weightings

Rather than compare *tf, tf-idf*, and binary weightings together at once we chose to examine two questions we believe are key with respect to document feature weighting; (1) Is the idf component of tf-idf weighting needed for document clustering?; and (2) Is term frequency more useful than simple term presence/absence for document clustering? Question (1) is evaluated in Sect. 3.1 by comparing our *tf* and *tf-idf* results. We show the benefit of *idf* is heavily dependent both on the dataset and clustering algorithm. We examine question (2) in Sect. 3.2 by comparing our *tf* and binary results. We show that *tf* is substantially superior to binary weighting.

3.1 Effect of *tf-idf* on document clustering

To determine if *tf-idf* was having a positive effect when compared to *tf*, we first took the evaluation measures on the 7888 tuples of (weighting, dataset, algorithm, #clusters) for the *tf* and *tf-idf* weightings and collapsed them by averaging each evaluation measure over the number of clusters. This gave 272 tuples of (weighting, dataset, algorithm) with averaged evaluation measures. From these tuples we derived three tables comparing *tf* and *tf-idf* weighting; (1) By dataset and average over all clustering algorithm for that dataset (Table 3); (2). By dataset and the best clustering algorithm for that dataset (Table 4); and (3). By algorithm (Table 5).

The overall row in Table 3 indicates that, on average, *tf-idf* offers improved results over *tf*. However, *tf-idf* is actually substantially worse than *tf* for re0 and somewhat worse for fbis and new3. Table 4 shows the best clustering algorithm for each dataset and weighting, by each of the four evaluation measures. One can see that Table 4 is mostly consistent with

Table 3 The percentage change in evaluation measures when using *tf-idf* document representations over *tf*, by dataset and overall

Dataset	NMI (%)	FQ (%)	PQ (%)	EQ (%)
fbis	−1.6	−0.4	−1.0	−1.3
new3	0.1	−2.3	−2.0	0.3
re0	−11.0	−4.1	−5.3	−5.4
re1	14.8	4.9	4.9	8.6
tr31	15.9	8.2	7.5	10.4
tr41	12.5	7.8	8.3	8.8
tr45	20.0	17.5	12.5	15.6
wap	6.6	7.6	6.4	7.2
Overall	7.2	4.9	3.9	5.5

Table 3 in terms of the when *tf-idf* or *tf* is better, with the best *tf* and *tf-idf* results being closer, in general, than average *tf* and *tf-idf* results.

A possible reason for *idf*'s harmful effect on some datasets is apparent from a nearest neighbor analysis. For each dataset and weighting we calculated the percentage of *r*-nearest neighbors, per document, that share the same label as that document. Figure 1 presents the results of this analysis for $r = 1$–30. Considering just the *tf* and *tf-idf* lines for the moment, we see definite trends. For re0, where *idf* is harmful, we see *tf-idf* yielding a consistently worse nearest neighborhood than *tf*. For the datasets where *idf* is beneficial (*re1, tr31, tr41, tr45*, and *wap*), we see *tf* yielding better small neighborhoods, but as *r* increases *tf-idf* reduces less than *tf*, yielding substantially better neighborhoods than *tf* at higher *r*s. For new3 and fbis, where *idf* is only somewhat harmful, we see that *tf* again begins with better neighborhoods, but as *r* increases *tf* and *tf-idf* approach the same quality of neighborhood (as opposed to *tf-idf* becoming better). As all clustering algorithms are more or less dependent on the quality of nearest neighborhoods, this provides a reasonable explanation for our different by-dataset results.

The average improvement by algorithms presented in Table 5 are split in to hierarchical and partitional groups (note that RB-Kmeans and other repeated bisection methods are placed in the hierarchical section as they generate hierarchies of clusters, even though the splitting decision at each level is based around partitioning). It is immediately notable from Table 5 the benefit of *idf* is not divisible along partitional versus hierarchical lines. For example, UPGMA and NC-NMF gain the largest benefit from using *tf-idf*, the former being hierarchical and the latter partitional. Another notable aspect is that the better clustering algorithms (from Table 4, 6 and 7) benefit less from *tf-idf* than most of the other algorithms (except Slink and Clink).

A noteworthy side point uncovered from our investigation of *tf* versus *tf-idf* weighting is that certain algorithms appear to favor certain evaluation methods. To show this, we used the dataset collapsed by number of clusters again. For each weighting, dataset, and evaluation measure, the clustering algorithms were ranked by their evaluation measure, from one (best) to 17 (worst). We then computed each algorithm's average rank by weighting and evaluation measure. Table 6 shows our algorithms, ordered by this average ranking (from best to worst) when *tf* is used, for each evaluation measure. Table 7 shows similar results for *tf-idf* weighting.

The most striking inconsistency in Tables 6 and 7 is the behavior of FQ. We notice that RB-H1 and RB-I2, which are overall the best algorithms, rank much lower by FQ for both *tf* and *tf-idf* weighting. Also, for both *tf* and *tf-idf*, UPGMA fairs much better with FQ than

Table 4 The best clustering algorithm for *tf* and *tf-idf* weighting on each dataset by each evaluation measure

	tf		*tf-idf*		Diff (%)
NMI					
fbis	RB-H1	0.566	RB-H1	0.558	−1.9
new3	RB-H1	0.577	RB-I2	0.590	2.3
re0	RB-I2	0.420	RB-H1	0.417	−0.7
re1	RB-I2	0.492	RB-I2	0.556	12.9
tr31	RB-H1	0.533	Agglo-I2	0.591	10.8
tr41	Agglo-I2	0.630	Agglo-I2	0.657	4.4
tr45	RB-I2	0.622	Agglo-I2	0.667	7.2
wap	RB-I2	0.571	Agglo-H1	0.573	0.3
FQ					
fbis	Agglo-H1	0.560	Agglo-H1	0.549	−2.0
new3	RB-H1	0.324	RB-I2	0.323	−0.3
re0	Direct-H1	0.478	Direct-I2	0.431	−9.8
re1	Agglo-I2	0.470	Agglo-I2	0.479	1.8
tr31	Clink	0.585	UPGMA	0.688	17.6
tr41	Agglo-I2	0.611	Direct-I2	0.655	7.2
tr45	Agglo-I2	0.590	Agglo-H1	0.672	14.0
wap	Agglo-I2	0.507	Agglo-H1	0.539	6.4
EQ					
fbis	RB-H1	0.704	RB-H1	0.692	−1.8
new3	RB-H1	0.601	RB-H1	0.594	−1.1
re0	RB-H1	0.703	RB-H1	0.689	−2.1
re1	RB-I2	0.632	RB-I2	0.663	5.0
tr31	RB-H1	0.859	RB-I2	0.895	4.3
tr41	RB-I2	0.860	RB-I2	0.884	2.8
tr45	RB-I2	0.831	RB-H1	0.854	2.8
wap	RB-I2	0.686	RB-H1	0.689	0.5
PQ					
fbis	RB-H1	0.687	RB-H1	0.676	−1.6
new3	RB-H1	0.666	RB-I2	0.679	1.9
re0	RB-H1	0.681	RB-H1	0.679	−0.2
re1	RB-I2	0.626	RB-I2	0.685	9.4
tr31	RB-H1	0.810	RB-I2	0.850	4.9
tr41	RB-I2	0.829	RB-I2	0.859	3.7
tr45	RB-I2	0.783	RB-H1	0.816	4.3
wap	RB-H1	0.670	RB-H1	0.677	1.1

Diff is the improvement in using the best *tf-idf* over the best *tf* algorithm

with other measures, as do Direct-H1 and Direct-I2. A Kendall's τ test for correlation between pairs of the eight rankings in Tables 6 and 7 is presented in Table 8. One may note a relatively strong measure of agreement between *tf* NMI, *tf* PQ, *tf* EQ, *tf-idf* NMI, *tf-idf* PQ, and *tf-idf* EQ rankings, with the minimum τ between any pair of those being

Table 5 Improvements by clustering algorithms when using *tf-idf* over *tf* weighting

Main type	Algorithm	NMI (%)	FQ (%)	PQ (%)	EQ (%)
Hierarchical	UPGMA	22.1	16.4	15.8	22.9
	RB-Kmeans	14.0	10.5	7.7	9.7
	Agglo-H1	5.6	4.3	2.9	4.4
	Agglo-I2	5.4	3.1	2.8	4.5
	RB-H1	3.1	2.1	1.1	2.6
	RB-I2	2.8	1.2	1.1	2.6
	Slink	6.6	0.0	0.2	0.8
	Clink	−5.4	−1.2	−0.1	0.0
Partitional	NMF-NC	19.4	13.5	12.0	13.2
	Direct-I2	14.7	7.4	7.7	10.5
	PAM	10.4	8.3	7.0	8.0
	Direct-H1	11.9	6.0	6.1	8.0
	Spect-Un	6.5	5.4	4.3	5.8
	Spect-RW	5.3	5.7	3.3	5.0
	KMeans	7.6	4.4	2.5	4.1
	Spect-Sy	4.1	3.7	2.4	3.8
	PCA-Kmeans	2.1	1.2	−1.2	−1.2

0.618. Perhaps unsurprisingly, the *tf* FQ and *tf-idf* FQ rankings have a τ of 0.5, with much lower (even negative in one case) τs with the other six rankings.

A potential source for FQ's large disagreement is visible from its formula. NMI, FQ, PQ, and EQ may be decomposed to contain 'by cluster' contributions to their overall values. For example, in PQ the contribution of a single cluster to the overall PQ score is, using the notation from the evaluation measure section, $p(a)\max_{b\in\mathcal{B}}\frac{p(a,b)}{p(a)}$. The by-cluster contributions of NMI, PQ, and EQ reference two things; (1) the cluster itself; and (2) the true labeling. None of their by cluster contributions directly reference the other clusters in the clustering. However, this is not the case of FQ. If a cluster has any by cluster contribution in FQ can only be determined by examining if its F1 score for some b exceeds all other clusters' F1s for that b. This reference to other clusters within the clustering makes FQ distinct from the other measures. While it is difficult to say whether this distinctness makes FQ a poor evaluation measure in general, it at least makes FQ problematic from the standpoint of it being inconsistent with the other quality notions we examined.

With respect to which algorithms are better, several algorithms have generally high ranks in Tables 6 and 7, including RB-H1, RB-h2, Agglo-H1, Agglo-I2, and Spect-Sy. RB-Kmeans and NC-NMF perform well with *tf-idf* evaluation measures only. Interestingly, Kmeans performs reasonably well by all measures. On the other hand, we note that Slink and Clink provide uniformly poor performance.

3.2 Effect of *tf* and binary weighting on document clustering

To determine if term frequency was more beneficial than binary term weights we compared our *tf* results to our binary results. The procedure for performing this experiment was the same as in the previous subsection, except our *tf-idf* results were replaced with our binary results. Table 9 shows the difference in the best algorithm results of binary and *tf* weightings.

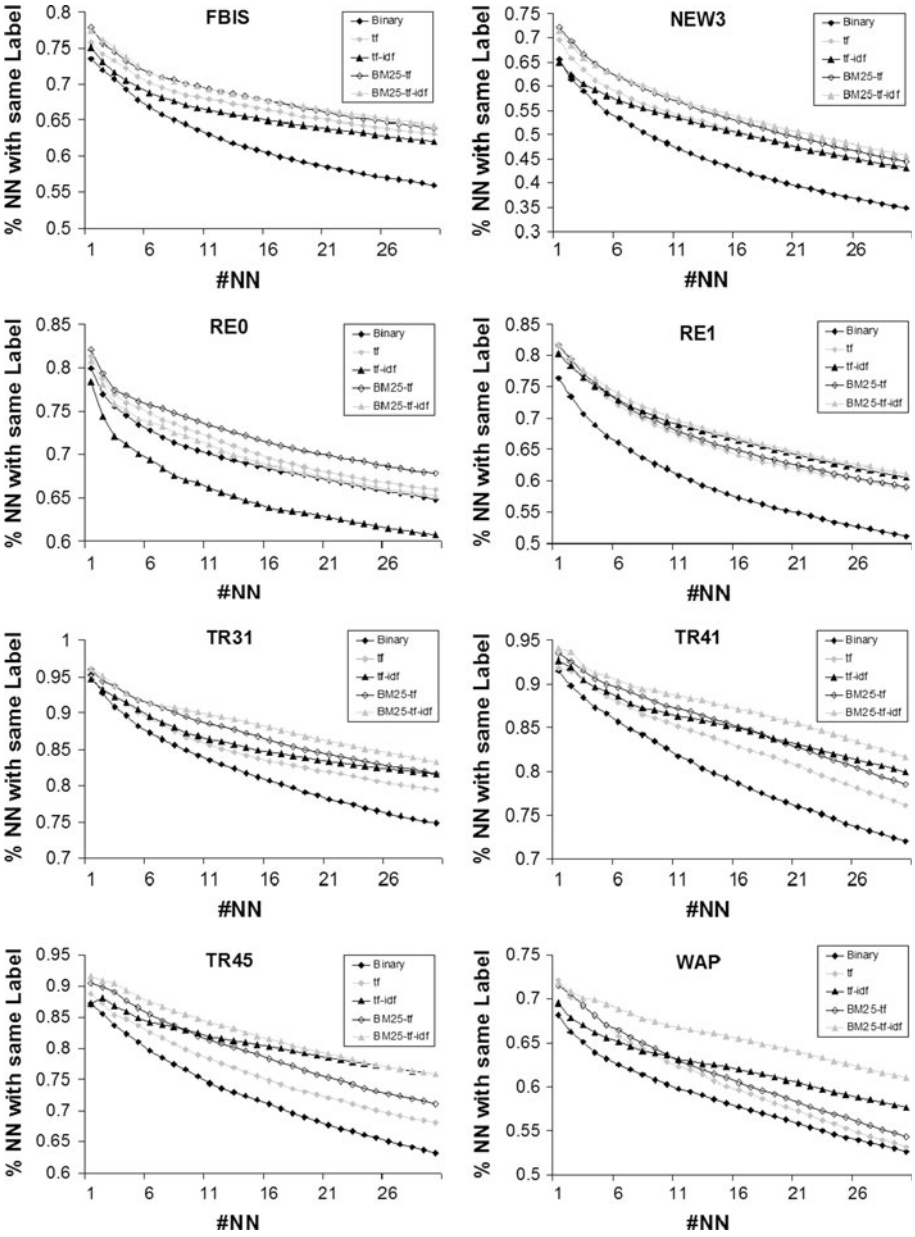

Fig. 1 Percentages of r-nearest neighbors (using cosine) that share the same label for each dataset

It is clear from Table 9 that binary weighting is notably worse than standard *tf* weighting. In a few cases the best binary results are better than the *tf* results, but they are often dramatically worse. When examining the average behavior of binary weighting, both by dataset and by clustering algorithm, we likewise found it to be notably worse than *tf*. A simple explanation for binary weighting's poor performance can be found by examining

Table 6 Algorithmic rankings by our evaluation measures when using *tf*

NMI	FQ	PQ	EQ
tf			
RB-I2	Agglo-I2	RB-I2	RB-H1
RB-H1	Agglo-H1	RB-H1	RB-I2
Agglo-I2	Kmeans	Agglo-H1	Agglo-I2
Agglo-H1	Direct-H1	Agglo-I2	Agglo-H1
Spect-Sy	Direct-I2	Spect-Sy	Spect-Sy
Kmeans	Spect-Sy	Kmeans	Kmeans
PCA-Kmeans	Spect-Un	PCA-Kmeans	RB-Kmeans
Spect-Un	RB-I2	RB-Kmeans	PCA-Kmeans
Spect-RW	RB-H1	Spect-RW	Spect-Un
RB-Kmeans	UPGMA	Spect-Un	Spect-RW
Direct-I2	NC-NMF	PAM	NC-NMF
Direct-H1	Clink	NC-NMF	PAM
NC-NMF	PCA-Kmeans	Clink	Clink
Clink	Spect-RW	Direct-H1	Direct-H1
PAM	RB-Kmeans	Direct-I2	Direct-I2
UPGMA	PAM	UPGMA	UPGMA
Slink	Slink	Slink	Slink

Table 7 Algorithmic rankings by our evaluation measures when using *tf-idf*

NMI	FQ	PQ	EQ
tf-idf			
Agglo-H1	Agglo-I2	RB-H1	RB-I2
RB-I2	NC-NMF	RB-I2	RB-H1
Agglo-I2	Agglo-H1	Agglo-H1	Agglo-I2
RB-H1	UPGMA	Agglo-I2	Agglo-H1
NC-NMF	Direct-I2	RB-Kmeans	RB-Kmeans
Spect-Sy	Direct-H1	NC-NMF	Spect-Sy
RB-Kmeans	Kmeans	Spect-Sy	NC-NMF
Spect-Un	Spect-Un	Kmeans	Kmeans
Kmeans	RB-Kmeans	Spect-Un	Spect-Un
Spect-RW	Spect-RW	Spect-RW	Spect-RW
Direct-I2	Spect-Sy	PAM	PCA-Kmeans
UPGMA	RB-H1	PCA-Kmeans	PAM
Direct-H1	Clink	UPGMA	Clink
PCA-Kmeans	RB-I2	Clink	UPGMA
PAM	PAM	Direct-I2	Direct-I2
Clink	PCA-Kmeans	Direct-H1	Direct-H1
Slink	Slink	Slink	Slink

Fig. 1. One can see that in every case except re0, the nearest neighborhoods of binary weightings are greatly inferior to those of *tf*. For re0, binary weighting produces only slightly worse nearest neighborhoods (with its clustering results being correspondingly

Table 8 Kendall's τ correlation between all the rankings in Tables 6 and 7

		tf				*tf-idf*			
		NMI	FQ	PQ	EQ	NMI	FQ	PQ	EQ
tf	NMI	1.000	0.426	0.809	0.824	0.618	0.044	0.632	0.706
	FQ	0.426	1.000	0.265	0.279	0.397	0.500	0.176	0.221
	PQ	0.809	0.265	1.000	0.926	0.603	−0.059	0.735	0.779
	EQ	0.824	0.279	0.926	1.000	0.618	0.015	0.779	0.824
tf-idf	NMI	0.618	0.397	0.603	0.618	1.000	0.279	0.750	0.735
	FQ	0.044	0.500	-0.059	0.015	0.279	1.000	0.176	0.132
	PQ	0.632	0.176	0.735	0.779	0.750	0.176	1.000	0.926
	EQ	0.706	0.221	0.779	0.824	0.735	0.132	0.926	1.000

closer to *tf* in quality). From this we conclude that term frequency counts are important in clustering, it is not sufficient to cluster on simple binary term presence/absences.

4 BM25 based feature weighting

The superiority of *tf* over binary weighting, which we demonstrated in the previous section, indicates that term counts are an important aspect of document clustering. A natural next question is if we can perform some other modification to *tf* which will yield superior clustering results. To that end, we apply BM25 (Robertson et al. 1994), which contains a term frequency dampening component, as a basis for feature weighting. If Q is a query consisting of a set of terms, then X_i's BM25 score with respect to that query (using our *idf* formulation) is

$$\text{Score}(Q, X_i) = \sum_{j \in Q} \frac{tf_{ij}(k1+1)}{tf_{ij} + k1\left((1-b) + b\frac{dl_i}{avgdl}\right)} \log\left(\frac{n}{n_j}\right), \tag{16}$$

where dl_i is a count of the tokens in document X_i:

$$dl_i = \sum_{j=1}^{m} tf_{ij} \tag{17}$$

avgdl is the average document length for documents in the collection, and b and $k1$ are parameters that are tuned, with $k1 \geq 0$ and $0 \leq b \leq 1$. Accounting for document length is handled by the b parameter. The term component in Eq. 16 *saturates* at a maximum of $k1 + 1$ as $tf_{ij} \to \infty$, with the gain from increasing *tf* diminishing as the *tf* value increases.

As discussed in the introduction, the previous rationale for the use of BM25 in document clustering was its performance at other tasks. Since its introduction in the early 1990s, the BM25 formula has been widely adopted, and it has repeatedly proved its value across a variety of search domains. The saturation characteristics of the BM25 term weighting function have been identified as a key element in the success of the formula. Unlike other proposed modifications to *tf*, growth of the BM25 term weighting function is relatively rapid when *tf* is small. However, the function quickly approaches an asymptote, limiting the impact of a single term.

Table 9 The best clustering algorithm for *tf* and binary weighting on each dataset by each evaluation measure

	tf		Binary		Diff (%)
NMI					
fbis	RB-H1	0.569	RB-H1	0.498	−12.4
new3	RB-H1	0.577	RB-H1	0.527	−8.6
re0	RB-I2	0.420	RB-I2	0.437	4.1
re1	RB-I2	0.493	RB-H1	0.431	−12.4
tr31	RB-H1	0.533	RB-I2	0.505	−5.3
tr41	Agglo-I2	0.630	RB-H1	0.603	−4.2
tr45	RB-I2	0.622	RB-H1	0.567	−8.8
wap	RB-I2	0.571	Agglo-I2	0.598	4.6
FQ					
fbis	Agglo-H1	0.560	Clink	0.476	−15.0
new3	RB-H1	0.324	RB-H1	0.271	−16.4
re0	Direct-H1	0.478	Direct-H1	0.455	−4.7
re1	Agglo-I2	0.470	Direct-H1	0.379	−19.5
tr31	Clink	0.585	UPGMA	0.558	−4.6
tr41	Agglo-I2	0.611	Agglo-H1	0.550	−10.0
tr45	Agglo-I2	0.590	PCA-Kmeans	0.537	−9.0
wap	Agglo-I2	0.507	Agglo-I2	0.589	16.1
EQ					
fbis	RB-H1	0.704	RB-H1	0.641	−9.0
new3	RB-H1	0.601	RB-H1	0.554	−7.9
re0	RB-H1	0.703	RB-I2	0.716	1.7
re1	RB-I2	0.632	RB-H1	0.560	−11.4
tr31	RB-H1	0.859	RB-I2	0.833	−3.0
tr41	RB-I2	0.860	RB-H1	0.843	−2.0
tr45	RB-I2	0.831	RB-H1	0.765	−7.9
wap	RB-I2	0.686	RB-I2	0.710	3.4
PQ					
fbis	RB-H1	0.687	RB-H1	0.618	−10.0
new3	RB-H1	0.666	RB-H1	0.622	−6.6
re0	RB-H1	0.681	RB-I2	0.697	2.3
re1	RB-I2	0.626	RB-H1	0.570	−8.9
tr31	RB-H1	0.810	RB-I2	0.781	−3.7
tr41	RB-I2	0.829	RB-H1	0.807	−2.6
tr45	RB-I2	0.783	RB-H1	0.722	−7.7
wap	RB-H1	0.670	RB-I2	0.691	3.2

Diff is the improvement in using the best binary algorithm over the best *tf* algorithm

Although document retrieval and clustering are not identical tasks, there is now enough clustering research to suggest BM25 might aid in document clustering (Bashier and Rauber 2009; de Vries and Geva 2008; Whissell et al. 2009; D'hondt et al. 2010; Kutty et al., 2010). This, coupled with the fact that no thorough analysis on the specific benefits of

BM25 in document clustering exists, led us to use BM25 in a clustering experiment similar to our initial experiment discussed Sect. 2.

We altered the document representations of Eqs. 2 and 3 to use the term saturation component of BM25. Specifically, Eq. 2 became

$$x_{ij} = \frac{tf_{ij}(k1+1)}{tf_{ij} + k1\left((1-b) + b\frac{dl_i}{avgdl}\right)}, \tag{18}$$

and Eq. 3 became

$$x_{ij} = \frac{tf_{ij}(k1+1)}{tf_{ij} + k1\left((1-b) + b\frac{dl_i}{avgdl}\right)} \log\left(\frac{n}{n_j}\right). \tag{19}$$

We refer to Eq. 18 as *BM25-tf* and Eq. 19 as *BM25-tf-idf*. The selection of values for the parameters b and $k1$ is discussed in the next subsection. After creating the BM25 document representations for the various datasets, our experiment followed the procedure described in Sect. 2 exactly, including the length normalization process. Section 4.2 discusses the results of the experiment, comparing our BM25 weightings with their non-BM25 counterparts and with each other.

4.1 Parameter estimation

Typical values for the BM25 parameters in document retrieval are $b = 0.75$ and $k1 = 1.2$ (or 2.0), previous BM25 clustering papers have all used these default values. For our experiments, we felt it did not make sense to experiment with different b values while still doing Euclidean length normalization on top of BM25. We further did not feel it was appropriate to simply drop the Euclidean length normalization, even with BM providing some length normalization (on a term by term basis), as just BM might have yielded document vectors of significantly different length. We therefore selected a fixed b = 1.0 rather than experimenting, and applied Euclidean length normalization on top of BM25, focusing instead on selecting the $k1$ parameter.

We set aside two of our datasets, fbis and tr31, to use in selecting $k1$, while the other six were kept for testing. On each of these two datasets, we ran several of our algorithms using document representations based on both Eqs. 18 and 19 with $b = 1.0$ and $k1 = 0\ldots 100$ in increments of one, various numbers of clusters were used as well. We applied our four evaluation measures to the resulting clusterings, Fig. 2 presents the trend in our evaluation measures when varying $k1$ with UPGMA clustering and *BM25-tf-idf* document representations (results for other clustering algorithms and the other weighting are mostly consistent with these results).

While the plots in Fig. 2 fluctuate, it is clear that a low value of $k1$ such as the typical document retrieval value of 1.2 or 2.0 is not appropriate. Further, setting $k1$ too high diminishes performance, although this is much less pronounced. While a more complex analysis of which $k1$ is best would be appropriate, we selected a value of $k1 = 20$ for use in our experiments based on these plots.

4.2 Results

From our previous experiments, we saw that *tf* and *tf-idf* behaved differently based on the dataset and algorithm, thus is made sense to compare *BM25-tf* versus *tf*, and *BM25-tf-idf*

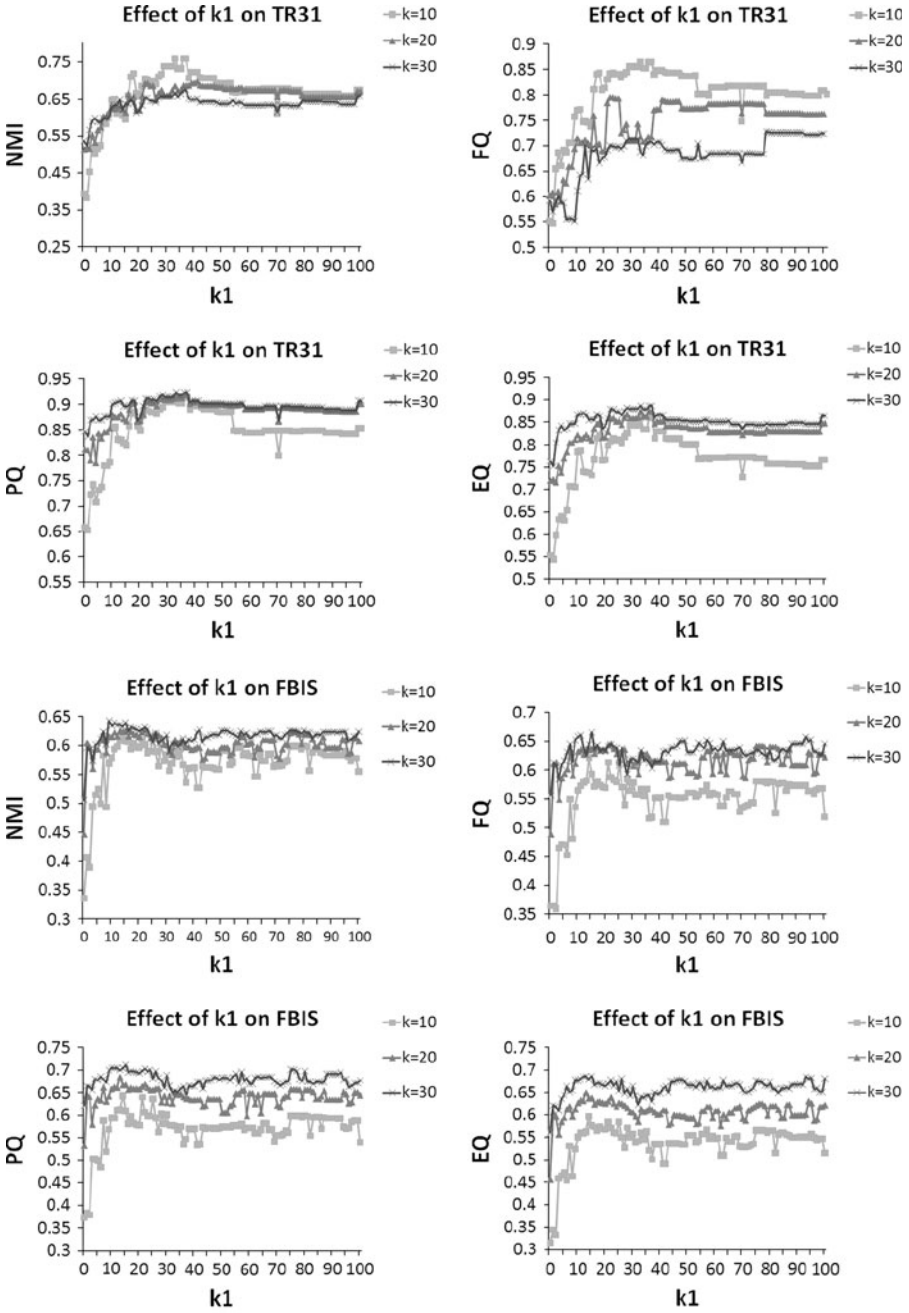

Fig. 2 The effect on our evaluation measures when using UPGMA clustering on the *BM25-tf-idf* document representation while varying $k1$ from 0 to 100

Table 10 Improvement by *BM25-tf* over *tf*

Dataset	NMI (%)	FQ (%)	PQ (%)	EQ (%)
New	−0.8	−4.5	−1.5	−0.7
re0	4.2	2.5	2.6	2.3
re1	2.2	−0.5	1.2	1.4
tr41	2.5	0.5	1.9	2.5
tr45	3.7	3.2	2.8	3.6
wap	4.2	3.3	2.5	3.7
Overall	2.7	0.8	1.6	2.1

Table 11 Improvement by *BM25-tf-idf* over *tf-idf*

Dataset	NMI (%)	FQ (%)	PQ (%)	EQ (%)
New3	2.3	3.4	2.9	1.2
re0	13.1	4.6	6.0	6.4
re1	−1.0	0.5	0.7	−0.7
tr41	1.6	−0.8	1.0	2.1
tr45	−3.1	−3.9	−1.9	−1.8
wap	4.4	4.1	3.5	3.5
Overall	2.9	1.3	2.0	1.8

versus *tf-idf*. Table 10 shows the by dataset improvement of *BM25-tf* over *tf*, and Table 11 shows the by dataset improvement of *BM25-tf-idf* over *tf-idf*. Tables 12 and 13 show the by-algorithm improvement for *BM25-tf* over *tf* and *BM25-tf-idf* over *tf-idf*, respectively.

One can see from Tables 10 and 11 that using BM25 weightings improves the average clustering quality results for all evaluation measures. Both weightings offer approximately the same improvement over their non-BM25 counterparts. The by-algorithm results in Tables 12 and 13 reveal that the large majority of algorithms benefit from BM25 weighting. It is worth noting that the four best performing algorithms for either *tf* and *tf-idf* from our previous experiment (specifically Agglo-H1, Agglo-I2, RB-I2, and RB-i1) all improve when BM25 term saturation is used. The benefit of BM25 term saturation is likely due to its effect on nearest neighborhoods, this is visible in Fig. 1. From Fig. 1, we can see that *BM25-tf* always has an equal or better neighborhood than *tf*, likewise, *BM25-tf-idf* has better neighborhoods than *tf-idf*.

With respect to comparing *BM25-tf* and *BM-tf-idf*, they follow a pattern similar to that of *tf* and *tf-idf*. For example, Table 14 shows the by algorithm improvement from using *BM25-tf-idf* over *BM25-tf*. The algorithms that benefit most from *tf-idf* can be seen to benefit most from *BM25-tf-idf*. When we analyzed the by dataset behavior of *BM25-tf-idf* versus *BM-tf* we found it to be similar to the behavior of *tf-idf* versus *tf* as well. Additionally, the relative nearest neighborhood behaviors of *BM25-tf-idf* versus *BM25-tf* in Fig. 1 follow the same pattern as *tf-idf* versus *tf*.

On average (by algorithm, dataset, best algorithm, and nearest neighborhoods), *BM25-tf-idf* is somewhat better than *BM25-tf*, and notably better than any of the other three weightings. From our BM25 weighting experiments in this section we conclude that BM25 term saturation is superior to raw term count information when used as a component of

Table 12 Improvements by clustering algorithms when using *BM25-tf* over *tf* weighting

Main type	Algorithm	NMI (%)	FQ (%)	PQ (%)	EQ (%)
Hierarchical	Clink	6.6	3.5	5.7	8.0
	RB-Kmeans	4.5	2.0	2.3	3.3
	Agglo-H1	2.4	0.5	1.3	1.7
	RB-H1	1.6	0.0	0.4	1.1
	UPGMA	2.6	−0.5	1.1	2.2
	RB-I2	1.7	0.4	0.8	1.2
	Agglo-I2	1.6	0.2	0.8	1.0
	Slink	−1.8	0.0	0.0	−0.2
Partitional	PCA-Kmeans	7.3	5.5	4.5	5.0
	NMF-NC	3.7	2.4	3.2	3.4
	PAM	3.9	2.9	2.4	2.5
	Spect-Un	3.8	1.3	2.9	3.1
	Spect-RW	3.2	1.8	1.9	2.7
	Direct-H1	2.9	1.2	2.1	2.3
	KMeans	2.9	1.6	1.4	2.2
	Spect-Sy	3.0	0.5	1.5	2.4
	Direct-I2	1.9	1.1	1.6	1.5

Table 13 Improvements by clustering algorithms when using *BM25-tf-idf* over *tf-idf* weighting

Main type	Algorithm	NMI (%)	FQ (%)	PQ (%)	EQ (%)
Hierarchical	Clink	18.0	11.8	8.2	7.5
	RB-I2	3.2	2.6	2.5	1.9
	UPGMA	3.3	0.3	1.8	2.4
	RB-H1	2.4	1.2	1.8	1.6
	Agglo-H1	2.1	0.9	1.3	1.3
	Agglo-I2	1.5	0.5	1.7	0.7
	Slink	−3.0	0.2	−0.2	−0.3
	RB-Kmeans	−3.3	−4.3	−2.3	−2.3
Partitional	PCA-Kmeans	10.0	6.1	7.8	9.0
	Spect-Un	5.1	3.1	3.7	3.3
	PAM	5.4	1.9	2.6	3.0
	Spect-RW	5.0	1.6	2.4	3.0
	KMeans	2.6	2.0	3.3	3.0
	Spect-Sy	4.4	0.9	2.1	2.4
	Direct-H1	0.5	−0.7	0.0	0.0
	NMF-NC	−0.6	−0.2	0.0	0.0
	Direct-I2	−1.2	−1.1	−0.3	−1.6

feature weighting in document clustering. Further, the behavior of our BM25 weightings are very similar to their non-BM25 counterparts with respect to which clustering algorithms and datasets benefit the most from their application.

Table 14 Improvements by clustering algorithms when using *BM25-tf-idf* over *BM25-tf* weighting

Main type	Algorithm	NMI (%)	FQ (%)	PQ (%)	EQ (%)
Hierarchical	UPGMA	20.0	18.3	17.3	22.8
	Agglo-H1	4.7	4.5	3.0	3.8
	Agglo-I2	4.4	3.4	3.3	3.9
	RB-Kmeans	5.0	3.5	2.6	3.7
	RB-I2	3.7	2.3	2.3	3.1
	RB-H1	2.4	1.2	1.8	1.6
	Clink	−0.6	4.9	0.5	−2.4
	Slink	0.7	−0.2	−0.2	0.0
Partitional	NMF-NC	14.6	8.5	9.4	10.2
	PAM	13.2	9.1	9.3	10.2
	Spect-Un	8.3	7.6	6.4	7.3
	Spect-RW	7.4	6.3	4.8	6.4
	Direct-I2	8.4	5.1	5.0	6.1
	Direct-H1	8.6	4.6	4.0	5.8
	KMeans	7.8	4.6	4.7	5.7
	Spect-Sy	4.9	4.2	3.5	4.1
	PCA-Kmeans	6.0	2.7	1.9	3.0

5 Concluding discussion

We examined the merits of applying *tf-idf* term weighting to document clustering through an experiment involving a variety of clustering algorithms, datasets, and evaluation measures. We found that the *idf* component of *tf-idf* weighting does influence clustering results, but this result can be either positive or negative when compared against *tf* weighting alone. On average, *tf-idf* produces better results than *tf*, but the benefit of using *tf-idf* depends very heavily on the exact dataset and the clustering algorithm used. Binary weighting was also examined, and it was determined that it is noticeably inferior to both *tf* weighting and *tf-idf* weighting.

An interesting point to come out of these experiments was that certain algorithms favor certain evaluation measures. While some small bias might be expected, the large amount observed was surprising. For example, UPGMA is very heavily biased towards F-measure. We found that the evaluation measures we used, all of which are well established in the literature, are far from perfectly correlated. However, NMI, our purity measure, and our entropy measure are well correlated. F-measure is poorly correlated with the other three measures. It is worth noting that the algorithmic preferences of our evaluation measures are relatively consistent across different weighting functions, but there are some notable exceptions such as NC-NMF, RB-Kmeans, and UPGMA ranking notably better when *tf-idf* weighting is used. Our findings suggest that caution be used when reporting the relative performance of clustering algorithms using a single evaluation measure. Using a range of accepted evaluation measures might be an appropriate approach to address the inconsistencies between individual measures.

We proposed and evaluated the use of the BM25 term weighting function in clustering. This function is noted for its term saturation characteristics. We showed that using it in place of the standard *tf* component in both *tf* and *tf-idf* leads to a noticeable improvement in clustering results.

With respect to future research, automatic estimation of the $k1$ parameter is of particular interest to us, this could greatly enhance the quality of clusterings when using BM25 term saturation. Another aspect we wish to examine is the apparent inconsistencies in evaluation measures used in clustering.

References

Aljaber, B., Stokes, N., Bailey, J., & Pei, J. (2010). Document clustering of scientific texts using citation contexts. *Information Retrieval, 13*, 101–131.

Bashier, S., & Rauber, A. (2009). Improving retrievability of patents with cluster-based pseudo-relevance feedback documents selection. In *CIKM* (pp. 1863–1866).

Beil, F., Ester, M., & Xu, X. (2002). Frequent term-based text clustering. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 436–442).

Boley, D., Gini, M., Gross, R., Han, E. H., Hastings, K., Karypis, G., et al. (1999). Document categorization and query generation on the World Wide Web using WebACE. *AI Review, 11*, 365–391.

de Vries, C. M., & Geva, S. (2008). Document clustering with K-tree. In *INEX* (pp. 420–431).

D'hondt, J., Vertommena, J., Verhaegena, P., Cattryssea, D., & Dufloua, J. R. (2010). Pairwise-adaptive dissimilarity measure for document clustering. *Information Sciences, 180*, 2341–2358.

Fung, B. C. M., Wangy, K., & Ester, M. (2003). Hierarchical document clustering using frequent itemsets. In *SDM '03: Proceedings of the SIAM international conference on data mining* (pp. 59–70).

Hofmann, T. (1999). Probabilistic latent semantic analysis. In *UAI '99: Uncertainty in Artificial Intelligence* (pp. 289–296).

Hu, X., Zhang, X., Lu, C., Park, E. K., & Zhou, X. (2009). Exploiting wikipedia as external knowledge for document clustering. In *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 389–396).

Jain, A. K., Murthy, M. N., & Flynn, P. J. (1999). Data clustering: A review. *ACM Computing Reviews, 31*, 264–323.

Kaufman, L., & Rousseeuw, P. (1990). *Finding groups in data: An introduction to cluster analysis*. Wiley: New York.

Kutty, S., Nayak, R., & Li, Y. (2010). Utilising semantic tags in XML clustering. In *Focused retrieval and evaluation* (pp. 416–425).

Lloyd, S. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory, 28*, 129–137.

Ng, A. Y., Jordan, M. I., & Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems 14* (pp. 849–856). Cambridge: MIT Press.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical magazine, 2*, 559–572.

Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M., & Gatford, M. (1994). Okapi at TREC-3. In *TREC '94: The third text retrieval conference*.

Sevillano, X., Cobo, G., Alías, F., & Socoró, J. C. (2006). Feature diversity in cluster ensembles for robust document clustering. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 697–698).

Shi, J., & Malik J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 22*(8), 888–905.

Slonim, N., & Tishby, N. (2000). Document clustering using word clusters via the information bottleneck method. In *SIGIR '00: Proceedings of the 26th annual international ACM SIGIR conference on research and development in informaion retrieval* (pp. 208–215).

Steinbach, M., Karypis, G., & Kumar, V. (2000). A comparison of document clustering techniques. In *KDD 00' text mining workshop*.

Strehl, A., & Ghosh, J. (2002). Cluster ensembles – a knowledge reuse framework for combining multipe partitions. *Journal of Machine Learning Research, 3*, 583–617.

van Rijsbergen, C. J. (1979). *Information retrieval*. Butterworth, 2nd ed.

von Luxberg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing, 17*, 395–416.

Whissell, J. S., Clarke, C. L. A., & Ashkan, A. (2009). Clustering web queries. In *CIKM 09: Proceedings of the 18th ACM conference on information and knowledge management* (pp. 899–908).

Wilbur, W. J., & Kim, W. (2009). The ineffectiveness of within-document term frequency in text classification. *Information Retrieval, 12*(5), 509–525.

Xu, W., Liu, X., & Gong, Y. (2003). Document clustering based on non-negative matrix factorization. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on research and development in informaion retrieval* (pp. 267–273).

Zhao, Y., & Karypis, G. (2001). Criterion functions for document clustering: Experiments and analysis. Technical Report 01-40, University of Minnesota, Department of Computer Science/Army HPC Research Center.

Zhao, Y., & Karypis, G. (2002). Evaluation of hierarchical clustering algorithms for document datasets. In *Data mining and knowledge discovery* (pp. 515–524).

Zhao, Y., & Karypis, G. (2004). Empirical and theoretical comparisons of selected criterion functions for document clustering. *Machine Learning, 55*, 311–331.