Similarity-Based Text Clustering: A Comparative Study

J. Ghosh and A. Strehl

Summary. Clustering of text documents enables unsupervised categorization and facilitates browsing and search. Any clustering method has to embed the objects to be clustered in a suitable representational space that provides a measure of (dis)similarity between any pair of objects. While several clustering methods and the associated similarity measures have been proposed in the past for text clustering, there is no systematic comparative study of the impact of similarity measures on the quality of document clusters, possibly because most popular cost criteria for evaluating cluster quality do not readily translate across qualitatively different measures. This chapter compares popular similarity measures (Euclidean, cosine, Pearson correlation, extended Jaccard) in conjunction with several clustering techniques (random, self-organizing feature map, hypergraph partitioning, generalized k-means, weighted graph partitioning), on a variety of high dimension sparse vector data sets representing text documents as bags of words. Performance is measured based on mutual information with a human-imposed classification. Our key findings are that in the quasiorthogonal space of word frequencies: (i) Cosine, correlation, and extended Jaccard similarities perform comparably; (ii) Euclidean distances do not work well; (iii) Graph partitioning tends to be superior especially when balanced clusters are desired; (iv) Performance curves generally do *not* cross.

1 Introduction

Document clusters can provide a structure for organizing large bodies of text for efficient browsing and searching. For example, recent advances in Internet search engines (e.g., www.vivisimo.com, www.metacrawler.com) exploit document cluster analysis. For this purpose, a document is commonly represented as a vector consisting of the suitably normalized frequency counts of words or terms. Each document typically contains only a small percentage of all the words ever used. If we consider each document as a multidimensional vector and then try to cluster documents based on their word contents, the problem differs from classic clustering scenarios in several ways: Document data are high dimensional[1], characterized by a very sparse term-document matrix with positive ordinal attribute values and a significant amount of outliers. In such situations, one is truly faced with the "curse of dimensionality" issue [176] since even after feature reduction, one is left with hundreds of dimensions per document.

Since clustering basically involves grouping objects based on their interrelationships or similarities, one can alternatively work in *similarity space* instead of the original feature space. The key insight is that if one can find a similarity measure (derived from the object features) that is appropriate for the problem domain, then a single number can capture the essential "closeness" of a given pair of objects, and any further analysis can be based only on these numbers. Once this is done, the original high-dimensional space is not dealt with at all; we only work in the transformed similarity space, and subsequent processing is independent of the dimensionality of the data [412]. A similar approach can be found in kernel based methods, such as Support Vector Machines (SVMs), for classification problems since the kernel function indicates a similarity measure obtained by a generalized inner product [240,249,430]. It is interesting to note that some very early works on clustering (e.g., [233]) were based on the concept of similarity, but subsequently the focus moved toward working with distances in a suitable embedding space, since typically, n, the number of objects considered, would be much larger than the number of features, d, used to represent each object. With text, d is very high; hence there is a renewal of interest in similarity-based approaches.

A typical pattern clustering activity involves the following five steps according to [242]:

1. Suitable object representation,
2. Definition of proximity between objects,
3. Clustering,
4. Data abstraction,
5. Assessment of output

The choice of similarity or distance in step 2 can have a profound impact on clustering quality. The significant amount of empirical studies in the 1980s and earlier on document clustering largely selected either Euclidean distance or cosine similarity, and emphasized various ways of representing/normalizing

[1] The dimension of a document in vector space representation is the size of the vocabulary, often in the tens of thousands.

documents before this step [377, 443]. Agglomerative clustering approaches were dominant and compared favorably with flat partitional approaches on small-sized or medium-sized collections [367, 443]. But lately, some new partitional methods have emerged (spherical k-means (KM), graph partitioning (GP) based, etc.) that have attractive properties in terms of both quality and scalability and can work with a wider range of similarity measures. In addition, much larger document collections are being generated.[2] This warrants an updated comparative study on text clustering, which is the motivation behind this chapter. Some very recent, notable comparative studies on document clustering [408, 463, 464] also consider some of these newer issues. Our work is distinguished from these efforts mainly by its focus on the key role of the similarity measures involved, emphasis on balancing, and the use of a normalized mutual information based evaluation that we believe has superior properties.

We mainly address steps 2 and 5 and also touch upon steps 3 and 4 in the document clustering domain. We first compare similarity measures analytically and illustrate their semantics geometrically (steps 2 and 4). Second, we propose an experimental methodology to compare high-dimensional clusterings based on mutual information and we argue why this is preferable to the more commonly used purity-based or entropy-based measures (step 5) [75, 408, 463]. Finally, we conduct a series of experiments to evaluate the performance and the cluster quality of four similarity measures (Euclidean, cosine, Pearson correlation, extended Jaccard) in combination with five algorithms (random, self-organizing map (SOM), hypergraph partitioning (HGP), generalized KM, weighted graph partitioning) (steps 2 and 3). Agglomerative clustering algorithms have been deliberately ignored even though they have been traditionally popular in the information retrieval community [367], but are not suitable for very large collections due to their computational complexity of at least $O(n^2 \log n)$ [300]. Indeed, if a hierarchy of documents is required, it is more practical to first partition the collection into an adequately large number (say 100 if finally about ten groups are desired) clusters, and then run an agglomerative algorithm on these partially summarized data.

Section 2 considers previous related work and Sect. 3 discusses various similarity measures.

2 Background and Notation

Clustering has been widely studied in several disciplines, especially since the early 1960s [59, 224, 243]. Some classic approaches include partitional methods such as k-means, hierarchical agglomerative clustering, unsupervised Bayes, and soft[3] techniques, such as those based on fuzzy logic or statistical mechanics

[2] IBM Patent Server has over 20 million patents. Lexis-Nexis contains over 1 billion documents

[3] In *soft* clustering, a record can belong to multiple clusters with different degrees of "association" [299].

[103]. Conceptual clustering [163], which maximizes category utility, a measure of predictability improvement in attribute values given a clustering, is also popular in the machine learning community. In most classical techniques, and even in fairly recent ones proposed in the data mining community (CLARANS, DBSCAN, BIRCH, CLIQUE, CURE, WAVECLUSTER, etc. [217, 368]) the objects to be clustered only have numerical attributes and are represented by low-dimensional feature vectors. The clustering algorithm is then based on distances between the samples in the original vector space [382]. Thus these techniques are faced with the "curse of dimensionality" and the associated sparsity issues, when dealing with very high-dimensional data such as text. Indeed, often, the performance of such clustering algorithms is demonstrated only on illustrative two-dimensional examples.

Clustering algorithms may take an alternative view based on a notion of *similarity* or dissimilarity. Similarity is often derived from the inner product between vector representations, a popular way to quantify document similarity. In [136], the authors present a spherical KM algorithm for document clustering using this similarity measure. Graph-based clustering approaches, which attempt to avoid the "curse of dimensionality" by transforming the problem formulation into a similarity space, include [75, 411, 461]. Finally, when only pairwise similarities are available, techniques such as Multi-Dimensional Scaling (MDS) [422] have been used to embed such points into a low-dimensional space such that the stress (relative difference between embedded point distances and actual distances) is minimized. Clustering can then be done in the embedding space. However, in document clustering this is not commonly used since for acceptable stress levels the dimensionality of the embedding space is too high.

Note that similarity-based methods take a *discriminative* approach to clustering. An alternative would be to take a *generative* viewpoint, starting from an underlying probabilistic model of the data and then finding suitable parameters typically through a maximum likelihood procedure. Cluster locations and properties are then derived as a by-product of this procedure. A detailed discussion of the pros and cons of discriminative approaches as compared to generative ones is given in [187]. Often discriminative approaches give better results, but any approach that required all-pairs similarity calculation is inherently at least $\mathrm{O}(N^2)$ in both computational and storage requirements. In contrast, model-based generative approaches can be linear in N. A detailed empirical comparison of different model-based approaches to document clustering is available in [464] and hence we do not revisit these models in this chapter. Clustering has also been studied for the purpose of *browsing.* A two-dimensional SOM [284] has been applied to produce a map of, e.g., Usenet postings in WEBSOM [285]. The emphasis in WEBSOM is not to maximize cluster quality but to produce a human interpretable two-dimensional spatial map of known categories (e.g., newsgroups). In the Scatter/Gather approach [120] document clustering is used for improved interactive browsing of large query results. The focus on this work is mostly on speed/scalability and not necessary maximum cluster quality. In [451], the effectiveness of clustering for organizing web documents was studied.

There is also substantial work on *categorizing* documents. Here, since at least some of the documents have labels, a variety of supervised or semi-supervised techniques can be used [342, 350]. A technique using the support vector machine is discussed in [249]. There are several comparative studies on document classification [447, 448].

Dimensionality reduction for text classification/clustering has been studied as well. Often, the data are projected onto a small number of dimensions corresponding to principal components or a scalable approximation thereof (e.g., Fastmap [156]). In latent semantic indexing (LSI) [124] the term-document matrix is modeled by a rank-K approximation using the top K singular values. While LSI was originally used for improved query processing in information retrieval, the base idea can be employed for clustering as well.

In *bag-of-words* approaches the term-frequency matrix contains occurrence counts of terms in documents. Often, the matrix is preprocessed in order to enhance discrimination between documents. There are many schemes for selecting term, and global, normalization components. One popular preprocessing is normalized term frequency, inverse document frequency (TF-IDF), which also comes in several variants [40, 377]. However, this chapter does not discuss the properties of feature extraction, see, e.g., [312, 459] instead. In [447, 448] classification performance of several other preprocessing schemes is compared.

Following Occam's Razor, we do *not* use any weighting but use the raw frequency matrix of selected words for our comparison. Hence, appropriate normalization has to be encoded by the similarity measure.

Let n be the number of objects (documents, samples) in the data and d the number of features (words, terms) for each object $\mathbf{x}_j$ with $j \in \{1, \ldots, n\}$. Let k be the desired number of clusters. The input data can be represented by a $d \times n$ data matrix $\mathbf{X}$ with the jth column vector representing the sample $\mathbf{x}_j$. $\mathbf{x}_j^{\mathrm{T}}$ denotes the transpose of $\mathbf{x}_j$. Hard clustering assigns a label λ_j to each d-dimensional sample $\mathbf{x}_j$, such that similar samples tend to get the same label. In general the labels are treated as nominals with no inherent order, though in some cases, such as one-dimensional SOM or GP approaches based on swapping of vertices with neighboring partitions the labeling contains extra ordering information. Let $\mathcal{C}_\ell$ denote the set of all objects in the ℓth cluster ($\ell \in \{1, \ldots, k\}$), with $\mathbf{x}_j \in \mathcal{C}_\ell \Leftrightarrow \lambda_j = \ell$ and $n_\ell = |\mathcal{C}_\ell|$. The number of distinct labels is k, the desired number of clusters. We treat the labels as nominals with no order, though in some cases, such as the SOM or graph partitioning, the labeling may contain extra ordering information. Batch clustering proceeds from a set of raw object descriptions $\mathcal{X}$ via the vector space description $\mathbf{X}$ to the cluster labels λ ($\mathcal{X} \rightarrow \mathbf{X} \rightarrow \lambda$). Section 3 briefly describes the compared similarity measures.

3 Similarity Measures

In this section, we introduce several similarity measures, illustrate some of their properties, and show why we are interested in some but not others. In Sect. 4, the algorithms using these similarity measures are discussed.

3.1 Conversion from a Distance Metric

The Minkowski distances $L_p(\mathbf{x}_a, \mathbf{x}_b) = \left(\sum_{i=1}^{d} |\mathbf{x}_{i,a} - \mathbf{x}_{i,b}|^p \right)^{1/p}$ are commonly used when objects are represented in a vector space. For $p = 2$ we obtain the Euclidean distance. There are several possibilities for converting such a distance metric (in $[0, \inf)$) into a similarity measure (in $[0, 1]$; usually similarity of 1 corresponds to a distance of 0) by a monotonic decreasing function. For Euclidean space, a good choice is: similarity $= \exp(-(\text{distance})^2)$, as it relates the squared error loss function to the negative log-likelihood for a Gaussian model for each cluster. In this chapter, we use the Euclidean $[0, 1]$-normalized similarity expressed by

$$s^{(\mathrm{E})}(\mathbf{x}_a, \mathbf{x}_b) = \mathrm{e}^{-\|\mathbf{x}_a - \mathbf{x}_b\|_2^2} \tag{1}$$

rather than alternatives such as $s(\mathbf{x}_a, \mathbf{x}_b) = 1/(1 + \|\mathbf{x}_a - \mathbf{x}_b\|_2)$.

3.2 Cosine Measure

A popular measure of similarity for text clustering is the cosine of the angle between two vectors. The cosine measure is given by

$$s^{(\mathrm{C})}(\mathbf{x}_a, \mathbf{x}_b) = \frac{\mathbf{x}_a^{\mathrm{T}} \mathbf{x}_b}{\|\mathbf{x}_a\|_2 \cdot \|\mathbf{x}_b\|_2} \tag{2}$$

and captures a scale invariant understanding of similarity. The cosine similarity does not depend on the length of the vectors, only their direction. This allows documents with the same relative distribution of terms to be treated identically. Being insensitive to the size of the documents makes this a very popular measure for text documents. Also, due to this property, document vectors can be normalized to the unit sphere for more efficient processing [136].

3.3 Pearson Correlation

In collaborative filtering, correlation is often used to predict a feature from a highly similar mentor group of objects whose features are known. The $[0, 1]$ normalized Pearson correlation is defined as

$$s^{(\mathrm{P})}(\mathbf{x}_a, \mathbf{x}_b) = \frac{1}{2} \left(\frac{(\mathbf{x}_a - \bar{x}_a)^{\mathrm{T}} (\mathbf{x}_b - \bar{x}_b)}{\|\mathbf{x}_a - \bar{x}_a\|_2 \cdot \|\mathbf{x}_b - \bar{x}_b\|_2} + 1 \right), \tag{3}$$

where $\bar{x}$ denotes the average feature values of $\mathbf{x}$. Note that this definition of Pearson correlation tends to give a full matrix. Other important correlations have been proposed, such as Spearman correlation [406], which works well on rank orders.

3.4 Extended Jaccard Similarity

The binary Jaccard coefficient[4] measures the degree of overlap between two sets and is computed as the ratio of the number of shared attributes (words) of $\mathbf{x}_a$ AND $\mathbf{x}_b$ to the number possessed by $\mathbf{x}_a$ OR $\mathbf{x}_b$. For example, given two sets' binary indicator vectors $\mathbf{x}_a = (0,1,1,0)^{\mathrm{T}}$ and $\mathbf{x}_b = (1,1,0,0)^{\mathrm{T}}$, the cardinality of their intersect is 1 and the cardinality of their union is 3, rendering their Jaccard coefficient 1/3. The binary Jaccard coefficient is often used in retail market-basket applications. The binary definition of Jaccard coefficient can be extended to continuous or discrete non-negative features as:

$$s^{(\mathrm{J})}(\mathbf{x}_a, \mathbf{x}_b) = \frac{\mathbf{x}_a^{\mathrm{T}}\mathbf{x}_b}{\|\mathbf{x}_a\|_2^2 + \|\mathbf{x}_b\|_2^2 - \mathbf{x}_a^{\mathrm{T}}\mathbf{x}_b}, \tag{4}$$

which is equivalent to the binary version when the feature vector entries are binary. Extended Jaccard similarity retains the sparsity property of the cosine while allowing discrimination of collinear vectors as we show in Sect. 3.6. Another similarity measure highly related to the extended Jaccard is the Dice coefficient

$$s^{(\mathrm{D})}(\mathbf{x}_a, \mathbf{x}_b) = \frac{2\mathbf{x}_a^{\mathrm{T}}\mathbf{x}_b}{\|\mathbf{x}_a\|_2^2 + \|\mathbf{x}_b\|_2^2}.$$

The Dice coefficient can be obtained from the extended Jaccard coefficient by adding $\mathbf{x}_a^{\mathrm{T}}\mathbf{x}_b$ to both the numerator and the denominator. It is omitted here since it behaves very similar to the extended Jaccard coefficient.

3.5 Other (Dis-)Similarity Measures

Many other (dis-)similarity measures, such as shared nearest neighbor [247] or the edit distance, are possible [243]. In fact, the ugly duckling theorem states [442] the somewhat "unintuitive" fact that there is no way to distinguish between two different classes of objects, when they are compared over all possible features. As a consequence, any two arbitrary objects are equally similar unless we use domain knowledge. The similarity measures discussed in Sects. 3.1–3.4 are some of the popular ones that have been previously applied to text documents [170, 377].

[4] Also called the Tanimoto coefficient in the vision community.

3.6 Discussion

Clearly, if clusters are to be meaningful, the similarity measure should be invariant to transformations natural to the problem domain. Also, normalization may strongly affect clustering in a positive or a negative way. The features have to be chosen carefully to be on comparable scales and similarity has to reflect the underlying semantics for the given task.

Euclidean similarity is translation invariant but scale sensitive while cosine is translation sensitive but scale invariant. The extended Jaccard has aspects of both properties as illustrated in Fig. 1. Iso-similarity lines at $s = 0.25$, 0.5, and 0.75 for points $\mathbf{x}_1 = (3, 1)^{\mathrm{T}}$ and $\mathbf{x}_2 = (1, 2)^{\mathrm{T}}$ are shown for Euclidean, cosine, and the extended Jaccard. For cosine similarity only the 4 (out of 12) lines that are in the positive quadrant are plotted: The two lines in the lower right part are one of two lines from $\mathbf{x}_1$ at 0.5 and 0.75. The two lines in the upper left are for $\mathbf{x}_2$ at $s = 0.5$ and 0.75. The dashed line marks the locus of equal similarity to $\mathbf{x}_1$ and $\mathbf{x}_2$, which always passes through the origin for cosine and the extended Jaccard similarity.

Using Euclidean similarity $s^{(\mathrm{E})}$, isosimilarities are concentric hyperspheres around the considered point. Due to the finite range of similarity, the radius decreases hyperbolically as $s^{(\mathrm{E})}$ increases linearly. The radius does not depend on the center point. The only location with similarity of 1 is the considered point itself and all finite locations have a similarity greater than 0. This last property tends to generate nonsparse similarity matrices. Using the cosine measure $s^{(\mathrm{C})}$ renders the isosimilarities to be hypercones all having their apex at the origin and the axis aligned with the considered point. Locations with similarity 1 are on the one-dimensional subspace defined by this axis. The locus of points with similarity 0 is the hyperplane through the origin and perpendicular to this axis. For the extended Jaccard similarity $s^{(\mathrm{J})}$, the isosimilarities are nonconcentric hyperspheres. The only location with similarity 1

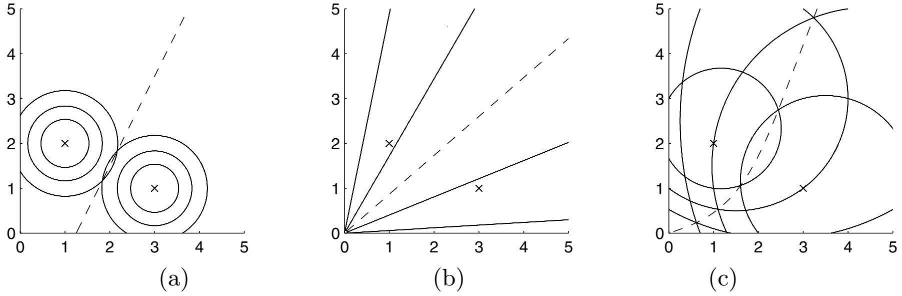


Fig. 1. Properties of **(a)** Euclidean-based, **(b)** cosine, and **(c)** extended Jaccard similarity measures illustrated in two dimensions. Two points $(1, 2)^{\mathrm{T}}$ and $(3, 1)^{\mathrm{T}}$ are marked with ×. For each point isosimilarity surfaces for $s = 0.25$, 0.5, and 0.75 are shown with solid lines. The surface that is equisimilar to the two points is marked with a dashed line

is the point itself. The hypersphere radius increases with the distance of the considered point from the origin so that longer vectors turn out to be more tolerant in terms of similarity than smaller vectors. Sphere radius also increases with similarity, and as $s^{(\mathrm{J})}$ approaches 0 the radius becomes infinite, rendering the sphere to the same hyperplane as obtained for cosine similarity. Thus, for $s^{(\mathrm{J})} \to 0$, the extended Jaccard behaves like the cosine measure, and for $s^{(\mathrm{J})} \to 1$, it behaves like the Euclidean distance.

In traditional Euclidean k-means clustering, the optimal cluster representative $\mathbf{c}_\ell$ minimizes the sum of squared error criterion, i.e.,

$$\mathbf{c}_\ell = \arg\min_{\mathbf{z}\in\mathcal{F}} \sum_{\mathbf{x}_j\in\mathcal{C}_\ell} \|\mathbf{x}_j - \mathbf{z}\|_2^2. \tag{5}$$

In the following, we show how this convex distance-based objective can be translated and extended to similarity space. Consider the generalized objective function $f(\mathcal{C}_\ell, \mathbf{z})$ given a cluster $\mathcal{C}_\ell$ and a representative $\mathbf{z}$:

$$f(\mathcal{C}_\ell, \mathbf{z}) = \sum_{\mathbf{x}_j\in\mathcal{C}_\ell} d(\mathbf{x}_j, \mathbf{z})^2 = \sum_{\mathbf{x}_j\in\mathcal{C}_\ell} \|\mathbf{x}_j - \mathbf{z}\|_2^2. \tag{6}$$

We use the transformation from (1) to express the objective in terms of similarity rather than distance:

$$f(\mathcal{C}_\ell, \mathbf{z}) = \sum_{\mathbf{x}_j\in\mathcal{C}_\ell} -\log(s(\mathbf{x}_j, \mathbf{z})). \tag{7}$$

Finally, we simplify and transform the objective using a strictly monotonic decreasing function: Instead of minimizing $f(\mathcal{C}_\ell, \mathbf{z})$, we maximize $f'(\mathcal{C}_\ell, \mathbf{z}) = \mathrm{e}^{-f(\mathcal{C}_\ell, \mathbf{z})}$. Thus, in similarity space, the least squared error representative $\mathbf{c}_\ell \in \mathcal{F}$ for a cluster $\mathcal{C}_\ell$ satisfies

$$\mathbf{c}_\ell = \arg\max_{\mathbf{z}\in\mathcal{F}} \prod_{\mathbf{x}_j\in\mathcal{C}_\ell} s(\mathbf{x}_j, \mathbf{z}). \tag{8}$$

Using the concave evaluation function f', we can obtain optimal representatives for non-Euclidean similarity spaces.

To illustrate the values of the evaluation function $f'(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{z})$ are used to shade the background in Fig. 2. The maximum likelihood representative of $\mathbf{x}_1$ and $\mathbf{x}_2$ is marked with an $*$ in Fig. 2. For cosine similarity all points on the equi-similarity are optimal representatives. In a maximum likelihood interpretation, we constructed the distance similarity transformation such that $p(\mathbf{z}|\mathbf{c}_\ell) \sim s(\mathbf{z}, \mathbf{c}_\ell)$. Consequently, we can use the dual interpretations of probabilities in similarity space and errors in distance space.

4 Algorithms

In this section, we briefly summarize the algorithms used in our comparison. A random algorithm is used as a baseline to compare the result quality of KM, GP, HGP, and SOM.

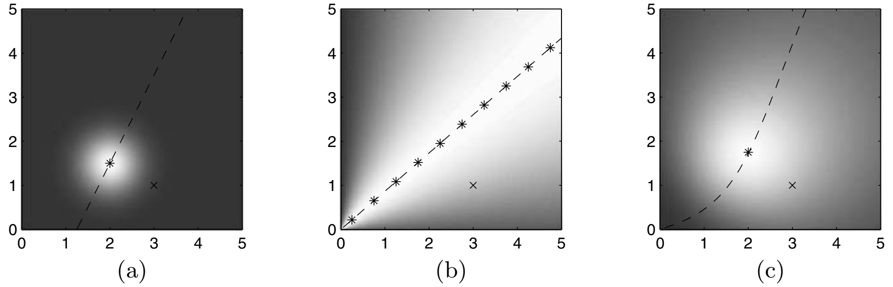


Fig. 2. More similarity properties shown on the two-dimensional example of Fig. 1. The goodness of a location as the common representative of the two points is indicated with brightness. The best representative is marked with an $*$. **(c)** The extended Jaccard adopts the middle ground between **(a)** Euclidean and **(b)** cosine-based similarity

4.1 Random Baseline

As a baseline for comparing algorithms, we use clustering labels drawn from a uniform random distribution over the integers from 1 to k. The complexity of this algorithm is $O(n)$.

4.2 Weighted Graph Partitioning

Clustering can be posed as a GP problem. The objects are viewed as the set of vertices $\mathcal{V}$. Two documents $\mathbf{x}_a$ and $\mathbf{x}_b$ (or vertices v_a and v_b) are connected with an undirected edge of positive weight $s(\mathbf{x}_a, \mathbf{x}_b)$, or $(a, b, s(\mathbf{x}_a, \mathbf{x}_b)) \in \mathcal{E}$. The cardinality of the set of edges $|\mathcal{E}|$ equals the number of *nonzero* similarities between all pairs of samples. A set of edges whose removal partitions a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ into k pairwise disjoint subgraphs $\mathcal{G}_\ell = (\mathcal{V}_\ell, \mathcal{E}_\ell)$ is called an edge separator. The objective in GP is to find such a separator with a minimum sum of edge weights. While striving for the minimum cut objective, the number of objects in each cluster has to be kept approximately equal. We produce balanced (equal sized) clusters from the similarity matrix using the multilevel graph partitioner Metis [262]. The most expensive step in this $O(n^2 \cdot d)$ technique is the computation of the $n \times n$ similarity matrix. In document clustering, sparsity can be induced by looking only at the v strongest edges or at the subgraph induced by pruning all edges except the v nearest neighbors for each vertex. Sparsity makes this approach feasible for large data sets. Sparsity is induced by particular similarities definitions based, for example, on the cosine of document vectors.

4.3 Hypergraph Partitioning

A hypergraph is a graph whose edges can connect more than two vertices (hyperedges). The clustering problem is then formulated as a finding the minimum cut of a hypergraph. A minimum cut is the removal of the set of hyperedges (with minimum edge weight) that separates the hypergraph into k unconnected components. Again, an object $\mathbf{x}_j$ maps to a vertex v_j. Each word (feature) maps to a hyperedge connecting all vertices with nonzero frequency count of this word. The weight of this hyperedge is chosen to be the total number of occurrences in the data set. Hence, the importance of a hyperedge during partitioning is proportional to the occurrence of the corresponding word. The minimum cut of this hypergraph into k unconnected components gives the desired clustering. We employ the hMetis package [263] for partitioning. An advantage of this approach is that the clustering problem can be mapped to a graph problem without the explicit computation of similarity, which makes this approach computationally efficient with $O(n \cdot d \cdot k)$ assuming a (close to) linear performing hypergraph partitioner. Note that samplewise frequency information gets lost in this formulation since there is only a single weight associated with a hyperedge.

4.4 Self-organizing Map

The SOM [70, 284] is a popular topology preserving clustering algorithm with nice visualization properties. For simplicity, we only use a one-dimensional line topology. Also, two-dimensional or higher dimensional topologies can be used. To generate k clusters we use k cells in a line topology and train the network for $m = 5,000$ epochs or 10 min (whichever comes first). All experiments are run on a dual processor 450 MHz Pentium using the SOM implementation in the Matlab neural network toolbox. The resulting network is subsequently used to generate the label vector λ from the index of the most activated neuron for each sample. The complexity of this incremental algorithm is $O(n \cdot d \cdot k \cdot m)$ and mostly determined by the number of epochs m and samples n.

4.5 Generalized *k*-means

The KM algorithm using the squared Euclidean or Mahalonobis distances as a measure of divergence, is perhaps the most popular partitional approach to clustering. This is really a generative approach, being a limiting case of soft clustering achieved by fitting a mixture of Gaussians to the data via the EM algorithm [266]. It has been recently shown that the scope of this framework is very broad, the essential properties of KM carry over to all regular Bregman divergences (and only to this class of divergence measures), and a similar generalization is also possible for the soft version [46]. The complexity of this set of algorithms is $O(n \cdot d \cdot k \cdot m)$, where m is the number of iterations needed for convergence.

Given the popularity of KM, we decided to convert cosine, Jaccard, and Pearson similarity measures into the corresponding divergences using (1), in addition to retaining the squared Euclidean distance to obtain four versions of KM. However we have not considered the use of KL-divergence, which has a natural correspondence with multinomial mixture modeling, as extensive work using this information theoretic measure is already available [463].

4.6 Other Clustering Methods

Several other clustering methods have also been considered but have not been used in our experimental comparison. Agglomerative models (single link, average link, complete link) [143] are computationally expensive (at least $O(n^2 \log n)$) and often result in highly skewed trees, indicating domination by one very large cluster. A detailed comparative study of generative, mixture model-based approaches to text, is available from [464]. Certain clustering algorithms from the data mining community (e.g., CLARANS, DBSCAN, BIRCH, CLIQUE, CURE, WAVECLUSTER [217, 368]) have been omitted since they are mostly scalable versions designed for low-dimensional data. Partitioning approaches based on principal directions have not been shown here since they perform comparably to hierachical agglomerative clustering [75]. Other GP approaches such as spectral bisectioning [227] are not included since they are already represented by the multilevel partitioner Metis.

5 Evaluation Methodology

We conducted experiments with all five algorithms, using four variants (involving different similarity measures) each for KM and GP, yielding 11 techniques in total. This section gives an overview of ways to evaluate clustering results. A good recent survey on clustering evaluation can be found in [463], where the emphasis is on determining the impact of a variety of cost functions, built using distance or cosine similarity measures, on the quality of two generic clustering approaches.

There are two fundamentally different ways of evaluating the quality of results delivered by a clustering algorithm. *Internal* criteria formulate quality as a function of the given data and/or similarities. For example, the mean squared error criterion is a popular evaluation criterion. Hence, the clusterer can evaluate its own performance and tune its results accordingly. When using internal criteria, clustering becomes an optimization problem. *External* criteria impose quality by additional, external information not given to the clusterer, such as class labels. While this makes the problem ill-defined, it is sometimes more appropriate since groupings are ultimately evaluated externally by humans.

5.1 Internal (Model-Based, Unsupervised) Quality

Internal quality measures, such as the sum of squared errors, have traditionally been used extensively. Given an internal quality measure, clustering can be posed as an optimization problem that is typically solved via greedy search. For example, KM has been shown to greedily optimize the sum of squared errors.

- Error (mean/sum-of-squared error, scatter matrices)
  The most popular cost function is the scatter of the points in each cluster. Cost is measured as the mean square error of data points compared to their respective cluster centroid. The well-known KM algorithm has been shown to heuristically minimize the squared error objective. Let n_ℓ be the number of objects in cluster $\mathcal{C}_\ell$ according to λ. Then, the cluster centroids are
  $$\mathbf{c}_\ell = \frac{1}{n_\ell} \sum_{\lambda_j = \ell} \mathbf{x}_j . \tag{9}$$
  The sum of squared errors (SSE) is
  $$\mathrm{SSE}(\mathbf{X}, \lambda) == \sum_{\ell=1}^{k} \sum_{\mathbf{x} \in \mathcal{C}_\ell} \|\mathbf{x} - \mathbf{c}_\ell\|_2^2 . \tag{10}$$
  Note that the SSE formulation can be extended to other similarities by using $\mathrm{SSE}(\mathbf{X}, \lambda) = \sum_{\ell=1}^{k} \sum_{\mathbf{x} \in \mathcal{C}_\ell} - \log s(\mathbf{x}, \mathbf{c}_\ell)$. Since we are interested in a quality measure ranging from 0 to 1, where 1 indicates a perfect clustering, we define quality as
  $$\phi^{(\mathrm{S})}(\mathbf{X}, \lambda) = e^{-\mathrm{SSE}(\mathbf{X}, \lambda)} . \tag{11}$$
  This objective can also be viewed from a probability density estimation perspective using EM [126]. Assuming the data are generated by a mixture of multivariate Gaussians with identical, diagonal covariance matrices, the SSE objective is equivalent to maximizing the likelihood of observing the data by adjusting the centers and minimizing weights of the Gaussian mixture.
- Edge cut
  When clustering is posed as a GP problem, the objective is to minimize edge cut. Formulated as a [0, 1]-quality maximization problem, the objective is the ratio of remaining edge weights to total precut edge weights:
  $$\phi^{(\mathrm{C})}(\mathbf{X}, \lambda) = \frac{\sum_{\ell=1}^{k} \sum_{a \in \mathcal{C}_\ell} \sum_{b \in \mathcal{C}_\ell, b > a} s(\mathbf{x}_a, \mathbf{x}_b)}{\sum_{a=1}^{n} \sum_{b=a+1}^{n} s(\mathbf{x}_a, \mathbf{x}_b)} \tag{12}$$
  Note that this quality measure can be trivially maximized when there are no restrictions on the sizes of clusters. In other words, edge cut quality evaluation is only fair when the compared clusterings are well balanced. Let us define the balance of a clustering λ as
  $$\phi^{(\mathrm{BAL})}(\lambda) = \frac{n/k}{\max_{\ell \in \{1,\ldots,k\}} n_\ell} . \tag{13}$$

A balance of 1 indicates that all clusters have the same size. In certain applications, balanced clusters are desirable because each cluster represents an equally important share of the data. Balancing has application-driven advantages, e.g., for distribution, navigation, summarization of the clustered objects. In [409] retail customer clusters are balanced, so they represent an equal share of revenue. Balanced clustering for browsing text documents has also been proposed [44]. However, some natural classes may not be of equal size, so relaxed balancing may become necessary. A middle ground between no constraints on balancing (e.g., k-means) and tight balancing (e.g., GP) can be achieved by overclustering using a balanced algorithm and then merging clusters subsequently [461]

- Category Utility [162, 193]
  The category utility function measures quality as the increase in predictability of attributes given a clustering. Category utility is defined as the *increase* in the expected number of attribute values that can be correctly guessed given a partitioning, *over* the expected number of correct guesses with no such knowledge. A weighted average over categories allows comparison of different sized partitions. Recently, it has been shown that category utility is related to squared error criterion for a particular standard encoding [338], whose formulation is used here. For binary features (i.e., attributes) the probability of the ith attribute being 1 is the mean of the ith row of the data matrix $\mathbf{X}$:

$$\bar{x}_i = \frac{1}{n} \sum_{j=1}^{n} x_{i,j} . \tag{14}$$

The conditional probability of the ith attribute to be 1 given that the data point is in cluster ℓ is

$$\bar{x}_{i,\ell} = \frac{1}{n_\ell} \sum_{\lambda_j = \ell} x_{i,j} . \tag{15}$$

Hence, category utility can be written as

$$\phi^{(\mathrm{CU})}(\mathbf{X}, \lambda) = \frac{4}{d} \sum_{\ell=1}^{k} \frac{n_\ell}{n} \left[\left(\sum_{i=1}^{d} \left(\bar{x}_{i,\ell}^2 - \bar{x}_{i,\ell} \right) \right) - \left(\sum_{i=1}^{d} \left(\bar{x}_i^2 - \bar{x}_i \right) \right) \right] . \tag{16}$$

Note that this definition divides the standard category by d so that $\phi^{(\mathrm{CU})}$ never exceeds 1. Category utility is defined to maximize predictability of attributes for a clustering. This limits the scope of this quality measure to low-dimensional clustering problems (preferably with each dimension being a categorical variable with small cardinality). In high-dimensional problems, such as text clustering, the objective is *not* to be able to predict the appearance of any possible word in a document from a particular cluster. In fact, there might be more unique words/terms/phrases than documents in a small data set. In preliminary experiments, category utility did not succeed in differentiating among the compared approaches (including random partitioning).

Using internal quality measures, fair comparisons can only be made amongst clusterings with the same choices of vector representation and similarity/ distance measure. For example, using edge cut in cosine-based similarity would not be meaningful for an evaluation of Euclidean KM. So, in many applications a consensus on the internal quality measure for clustering is not found. However, in situations where the pages are categorized (labeled) by an external source, there is a plausible way out!

5.2 External (Model-Free, Semisupervised) Quality

External quality measures require an external grouping, for example as indicated by category labels, that is assumed to be "correct." However, unlike in classification such ground truth is not available to the clustering algorithm. This class of evaluation measures can be used to compare start-to-end performance of any kind of clustering regardless of the models or the similarities used. However, since clustering is an unsupervised problem, the performance cannot be judged with the same certitude as for a classification problem. The external categorization might not be optimal at all. For example, the way Web pages are organized in the Yahoo! taxonomy is certainly not the best structure possible. However, achieving a grouping similar to the Yahoo! taxonomy is certainly indicative of successful clustering.

Given g categories (or classes) $\mathcal{K}_h$ ($h \in \{1, \ldots, g\}$), we denote the categorization label vector κ, where $x_a \in \mathcal{K}_h \Leftrightarrow \kappa_a = h$. Let $n^{(h)}$ be the number of objects in category $\mathcal{K}_h$ according to κ, and n_ℓ the number of objects in cluster $\mathcal{C}_\ell$ according to λ. Let $n_\ell^{(h)}$ denote the number of objects that are in cluster ℓ according to λ as well as in category h given by κ. There are several ways of comparing the class labels with cluster labels.

- Purity
  Purity can be interpreted as classification accuracy under the assumption that all objects of a cluster are classified to be members of the dominant class for that cluster. For a single cluster, $\mathcal{C}_\ell$, purity is defined as the ratio of the number of objects in the *dominant* category to the total number of objects:

$$\phi^{(\mathrm{A})}(\mathcal{C}_\ell, \kappa) = \frac{1}{n_\ell} \max_h (n_\ell^{(h)}). \tag{17}$$

  To evaluate an entire clustering, one computes the average of the cluster-wise purities weighted by cluster size:

$$\phi^{(\mathrm{A})}(\lambda, \kappa) = \frac{1}{n} \sum_{\ell=1}^{k} \max_h (n_\ell^{(h)}). \tag{18}$$

- Entropy [115]
  Entropy is a more comprehensive measure than purity since rather than just considering the number of objects "in" and "not in" the dominant class, it takes the entire distribution into account. Since a cluster with all objects from the same category has an entropy of 0, we define entropy-based quality as 1 minus the [0,1]-normalized entropy. We define entropy-based quality for each cluster as:

$$\phi^{(\mathrm{E})}(\mathcal{C}_\ell, \kappa) = 1 - \sum_{h=1}^{g} -\frac{n_\ell^{(h)}}{n_\ell} \log_g \left(\frac{n_\ell^{(h)}}{n_\ell} \right). \tag{19}$$

  And through weighted averaging, the total entropy quality measure falls out to be:

$$\phi^{(\mathrm{E})}(\lambda, \kappa) = 1 + \frac{1}{n} \sum_{\ell=1}^{k} \sum_{h=1}^{g} n_\ell^{(h)} \log_g \left(\frac{n_\ell^{(h)}}{n_\ell} \right). \tag{20}$$

  Both purity and entropy are biased to favor a large number of clusters. In fact, for both these criteria, the globally optimal value is trivially reached when each cluster is a single object!
- Precision, recall, and F-measure [429]
  Precision and recall are standard measures in the information retrieval community. If a cluster is viewed as the results of a query for a particular category, then precision is the fraction of correctly retrieved objects:

$$\phi^{(\mathrm{P})}(\mathcal{C}_\ell, \mathcal{K}_h) = n_\ell^{(h)} / n_\ell. \tag{21}$$

  Recall is the fraction of correctly retrieved objects out of all matching objects in the database:

$$\phi^{(\mathrm{R})}(\mathcal{C}_\ell, \mathcal{K}_h) = n_\ell^{(h)} / n^{(h)}. \tag{22}$$

  The F-measure combines precision and recall into a single number given a weighting factor. The F_1-measure combines precision and recall with equal weights. The following equation gives the F_1-measure when querying for a particular category $\mathcal{K}_h$

$$\phi^{(F_1)}(\mathcal{K}_h) = \max_\ell \frac{2\ \phi^{(\mathrm{P})}(\mathcal{C}_\ell, \mathcal{K}_h)\ \phi^{(\mathrm{R})}(\mathcal{C}_\ell, \mathcal{K}_h)}{\phi^{(\mathrm{P})}(\mathcal{C}_\ell, \mathcal{K}_h) + \phi^{(\mathrm{R})}(\mathcal{C}_\ell, \mathcal{K}_h)} = \max_\ell \frac{2 n_\ell^{(h)}}{n_\ell + n^{(h)}}. \tag{23}$$

  Hence, for the entire clustering the total F_1-measure is:

$$\phi^{(F_1)}(\lambda, \kappa) = \frac{1}{n} \sum_{h=1}^{g} n^{(h)} \phi^{(\mathrm{F})}(\mathcal{K}_h) = \frac{1}{n} \sum_{h=1}^{g} n^{(h)} \max_\ell \frac{2 n_\ell^{(h)}}{n_\ell + n^{(h)}}. \tag{24}$$

  Unlike purity and entropy, the F_1-measure is not biased toward a larger number of clusters. In fact, it favors coarser clusterings. Another issue is that random clustering tends not to be evaluated at 0.

- Mutual information [115]
  Mutual information is the most theoretically well founded among the considered external quality measures [140]. It is symmetric in terms of κ and λ. Let X and Y be the random variables described by the cluster labeling λ and category labeling κ, respectively. Let $H(X)$ denote the entropy of a random variable X. Mutual information between two random variables is defined as

$$I(X,Y) = H(X) + H(Y) - H(X,Y). \tag{25}$$

Also,

$$I(X,Y) \leq \min(H(X), H(Y)). \tag{26}$$

Since $\min(H(X), H(Y)) \leq (H(X) + H(Y))/2$, a tight upper bound on $I(X,Y)$ is given by $(H(X) + H(Y))/2$. Thus, a worst-case upper bound for all possible labelings and categorizations is given by

$$I(X,Y) \leq \max_{X,Y} \left(\frac{H(X) + H(Y)}{2} \right). \tag{27}$$

Hence, we define [0,1]-normalized mutual information-based quality as

$$NI(X,Y) = \frac{2 \cdot I(X,Y)}{\max_X(H(X)) + \max_Y(H(Y))}. \tag{28}$$

Using

$$I(X,Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log \frac{p(x,y)}{p(x) \cdot p(y)}. \tag{29}$$

Note that normalizing by the geometric mean of $H(X)$ and $H(Y)$ instead of the arithmetic mean will also work [410].
Now, approximating probabilities with frequency counts yields our quality measure $\phi^{(\mathrm{NMI})}$:

$$\phi^{(\mathrm{NMI})}(\lambda, \kappa) = \frac{2 \cdot \sum_{\ell=1}^{k} \sum_{h=1}^{g} \frac{n_l^{(h)}}{n} \log \frac{n_l^{(h)}/n}{n^{(h)}/n n_l/n}}{\log(k) + \log(g)} \tag{30}$$

Basic simplifications yield:

$$\phi^{(\mathrm{NMI})}(\lambda, \kappa) = \frac{2}{n} \sum_{\ell=1}^{k} \sum_{h=1}^{g} n_\ell^{(h)} \log_{k \cdot g} \left(\frac{n_\ell^{(h)} n}{n^{(h)} n_\ell} \right) \tag{31}$$

Mutual information is less prone to biases than purity, entropy, and the F_1-measure. Singletons are not evaluated as perfect. Random clustering has mutual information of 0 in the limit. However, the best possible labeling evaluates to less than 1, unless classes are balanced, i.e., of equal size.

Note that our normalization penalizes over-refinements unlike the standard mutual information.[5]

External criteria enable us to compare different clustering methods fairly provided the external ground truth is of good quality. One could argue against external criteria that clustering does not have to perform as well as classification. However, in many cases clustering is an interim step to better understand and characterize a complex data set before further analysis and modeling.

Normalized mutual information is our preferred choice of evaluation in Sect. 6, because it is a relatively unbiased measure for the usefulness of the knowledge captured in the clustering in predicting category labels. Another promising evaluation method based on PAC-MDL bounds is given in [45].

6 Experiments

6.1 Data Sets and Preprocessing

We chose four text data sets for comparison. Here we briefly describe them:

- `YAH`. These data were parsed from Yahoo! news web pages [75]. The 20 original categories for the pages are `Business`, `Entertainment` (`no sub-category`, `art`, `cable`, `culture`, `film`, `industry`, `media`, `multimedia`, `music`, `online`, `people`, `review`, `stage`, `television`, `variety`), `Health`, `Politics`, `Sports`, `Technology`. The data can be downloaded from `ftp://ftp.cs.umn.edu/ dept/users/boley/` (K1 series).
- `N20`. The data contain roughly 1,000 postings each from the following 20 newsgroup topics [302][6]:
  1. `alt.atheism`,
  2. `comp.graphics`,
  3. `comp.os.ms-windows.misc`,
  4. `comp.sys.ibm.pc.hardware`,
  5. `comp.sys.mac.hardware`,
  6. `comp.windows.x`,
  7. `misc.forsale`,
  8. `rec.autos`,
  9. `rec.motorcycles`,
  10. `rec.sport.baseball`,
  11. `rec.sport.hockey`,

[5] Let $\kappa = (1,1,2,2)^{\mathrm{T}}$, $\lambda^{(1)} = (1,1,2,2)^{\mathrm{T}}$, and $\lambda^{(2)} = (1,2,3,4)^{\mathrm{T}}$. $\lambda^{(2)}$ is an over-refinement of correct clustering $\lambda^{(1)}$. The mutual information between κ and $\lambda^{(1)}$ is 2 and the mutual information between κ and $\lambda^{(2)}$ is also 2. Our [0,1]-normalized mutual information measure $\phi^{(\mathrm{NMI})}$ penalizes the useless refinement: $\phi^{(\mathrm{NMI})}(\lambda^{(2)},\kappa) = 2/3$ which is less than $\phi^{(\mathrm{NMI})}(\lambda^{(1)},\kappa) = 1$.

[6] The data can be found at `http://www.at.mit.edu/~jrennie/20Newsgroups/`.

12. sci.crypt,
13. sci.med,
14. sci.electronics,
15. sci.space,
16. soc.religion.christian,
17. talk.politics.guns,
18. talk.politics.mideast,
19. talk.politics.misc,
20. talk.religion.misc.

- WKB. From the CMU Web KB Project [116], web pages from the following 10 industry sectors according to Yahoo! were selected: **airline**, **computer hardware**, **electronic instruments and controls**, **forestry and wood products**, **gold and silver**, **mobile homes and rvs**, **oil well services and equipment**, **railroad**, **software and programming**, **trucking**. Each industry contributes about 10% of the pages.
- REU. The Reuters-21578, Distribution 1.0.[7] We use the primary topic keyword as the category. There are 82 unique primary topics in the data. The categories are highly imbalanced.

The data sets encompass several text styles. For example, WKB documents vary significantly in length: some are in the wrong category, some are dead links or have little content (e.g., are mostly images). Also, the hub pages that Yahoo! refers to are usually top-level branch pages. These tend to have more similar bag-of-words content across different classes (e.g., contact information, search windows, welcome messages) than news content-oriented pages. In contrast, the content of REU is well-written news agency messages. However, they often belong to more than one category.

Words were stemmed using Porter's suffix stripping algorithm [170] in YAH and REU. For all data sets, words occurring on average between 0.01 and 0.1 times per document were counted to yield the term-document matrix. This excludes stop words such as a, and very generic words such as new, as well as too rare words such as haruspex.

6.2 Summary of Results

In this section, we present and compare the results of the 11 approaches on the four document data sets. Clustering quality is understood in terms of mutual information and balance. For each data set we set the number of clusters k to be twice the number of categories g, except for the REU data set where we used $k = 40$ since there are many small categories. Using a greater number of clusters than classes allows multimodal distributions for each class. For example, in an XOR like problem, there are two classes, but four clusters.

[7] Available from Lewis at www.research.att.com/∼lewis.

Let us first look at a representative result to illustrate the behavior of some algorithms and our evaluation methodology. In Fig. 3, confusion matrices illustrating quality differences of RND, KM E, KM C, and GP C approaches on a sample of 800 documents from N20 are shown. The horizontal and the vertical axes correspond to the categories and the clusters, respectively. Clusters are sorted in increasing order of dominant category. Entries indicate the number $n_\ell^{(h)}$ of documents in cluster ℓ and category h by darkness. Expectedly, random partitioning RND results in indiscriminating clusters with a mutual information score $\phi^{(\mathrm{NMI})} = 0.16$. The purity score $\phi^{(\mathrm{P})} = 0.16$ indicates that on average the dominant category contributes 16% of the objects in a cluster. However, since labels are drawn from a uniform distribution, cluster sizes are somewhat balanced with $\phi^{(\mathrm{BAL})} = 0.63$. KM E delivers one large cluster (cluster 15) and many small clusters with $\phi^{(\mathrm{BAL})} = 0.03$. This strongly imbalanced clustering is characteristic of KM E on high-dimensional sparse data and is problematic because it usually defeats certain application specific purposes such as browsing. It also results in subrandom quality $\phi^{(\mathrm{NMI})} = 0.11$ ($\phi^{(\mathrm{P})} = 0.17$). KM C results are good. A "diagonal" can be clearly seen in the confusion matrix. This indicates that the clusters align with the ground truth categorization, which is reflected by an overall mutual information $\phi^{(\mathrm{NMI})} = 0.35$ ($\phi^{(\mathrm{P})} = 0.38$). Balancing is good as well with $\phi^{(\mathrm{BAL})} = 0.45$. GP C exceeds KM C in both aspects with $\phi^{(\mathrm{NMI})} = 0.47$ ($\phi^{(\mathrm{P})} = 0.48$) as well as balance $\phi^{(\mathrm{BAL})} = 0.95$. The "diagonal" is stronger and clusters are very balanced.

The rest of the results are given in a summarized form instead of the more detailed treatment in the example mentioned earlier, since the comparative

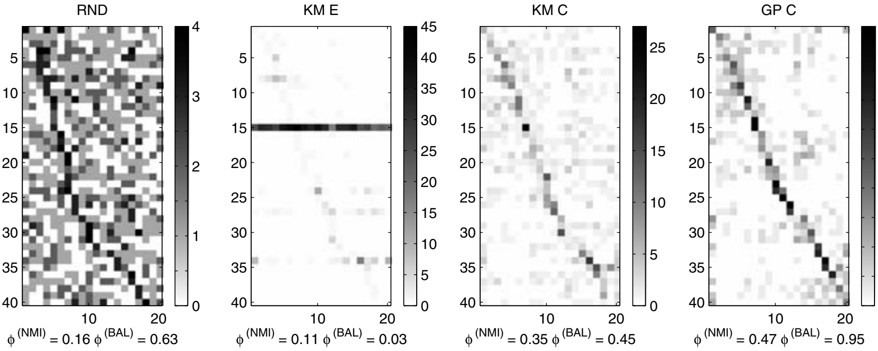


Fig. 3. Confusion matrices illustrating quality differences of RND, KM E, KM C, and GP C approaches on a sample of 800 documents from N20. Matrix entries indicate the number $n_\ell^{(h)}$ of documents in cluster ℓ (row) and category h (column) by darkness. Clusters are sorted in ascending order of their dominant category. KM E delivers one large cluster and shows subrandom quality $\phi^{(\mathrm{NMI})}$. KM C results are good, but are exceeded by GP C in terms of mutual information $\phi^{(\mathrm{NMI})}$ as well as balance $\phi^{(\mathrm{BAL})}$

trends are very clear even at this macrolevel. Some examples of detailed confusion matrices and pairwise t-tests can be found in our earlier work [413].

For a systematic comparison, ten experiments were performed for each of the random samples of sizes 50, 100, 200, 400, and 800. Figure 4 shows performance curves in terms of (relative) mutual information comparing ten algorithms on four data sets. Each curve shows the *difference* $\Delta\phi^{(\mathrm{NMI})}$ in mutual information-based quality $\phi^{(\mathrm{NMI})}$ compared to random partitioning for five sample sizes (at 50, 100, 200, 400, and 800). Error bars indicate ±1 standard deviations over ten experiments. Figure 5 shows quality in terms of balance for four data sets in combination with ten algorithms. Each curve shows the cluster balance $\phi^{(\mathrm{BAL})}$ for five sample sizes (again at 50, 100, 200, 400, and 800). Error bars indicate ±1 standard deviations over ten experiments. Figure 6 summarizes the results on all four data sets at the highest sample size level ($n = 800$). We also conducted pairwise t-tests at $n = 800$ to ensure differences in average performance are significant. For illustration and brevity, we chose to show mean performance with standard variation bars rather than the t-test results (see our previous work [413]).

First, we look at quality in terms of mutual information (Figs. 4 and 6a). With increasing sample size n, the quality of clusterings tends to improve. Nonmetric (cosine, correlation, Jaccard) GP approaches work best on text data followed by nonmetric KM approaches. Clearly, a nonmetric, e.g., dot-product based similarity measure is necessary for good quality. Due to the conservative normalization, depending on the given data set the maximum obtainable mutual information (for a perfect *classifier*!) tends to be around 0.8–0.9. A mutual information-based quality around 0.4 and 0.5 (which is approximately 0.3–0.4 better than random at $n = 800$) is an excellent result.[8] HP constitutes the third tier. Euclidean techniques including SOM perform rather poorly. Surprisingly, the SOM still delivers significantly better than random results despite the limited expressiveness of the implicitly used Euclidean distances. The success of SOM is explained by the fact that the Euclidean distance becomes locally meaningful once the cell centroids are locked onto a good cluster.

All approaches behaved consistently over the four data sets with only slightly different scale caused by the different data sets' complexities. The performance was best on `YAH` and `WKB` followed by `N20` and `REU`. Interestingly, the gap between GP and KM techniques is wider on `YAH` than on `WKB`. The low performance on `REU` is probably due to the high number of classes (82) and their widely varying sizes.

In order to assess those approaches that are more suitable for a particular amount of objects n, we also looked for intersects in the performance curves

[8] For verification purposes we also computed entropy values for our experiments and compared with, e.g., [463] to ensure validity.

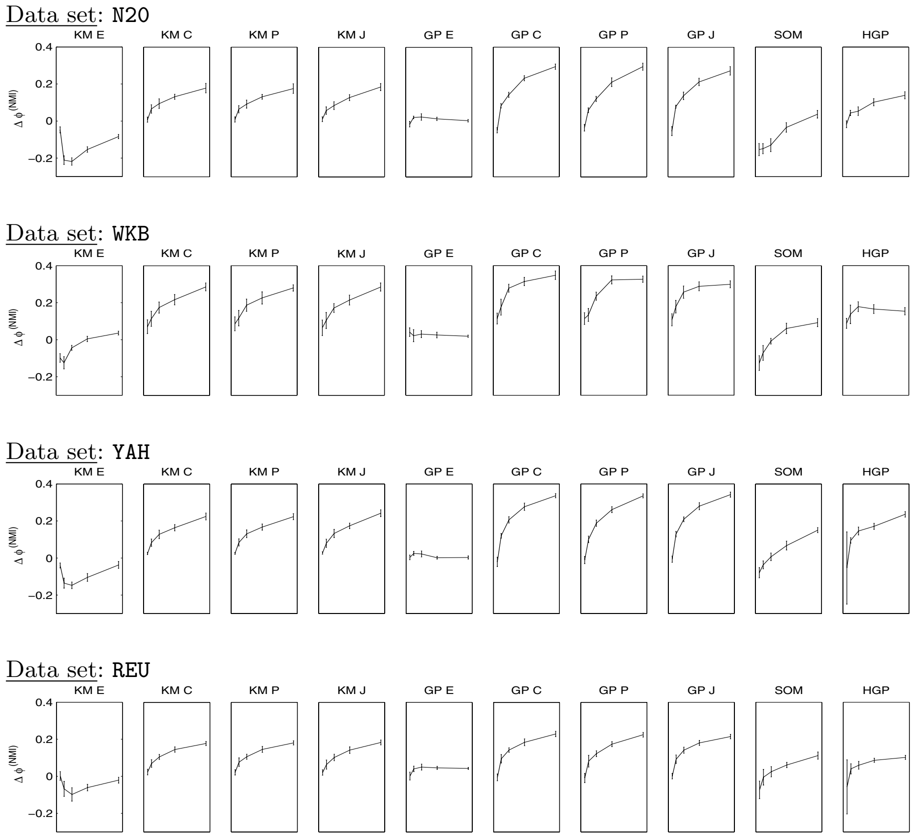


Fig. 4. Mutual information performance curves comparing ten algorithms on four data sets. Each curve shows the difference in mutual information-based quality $\phi^{(\mathrm{NMI})}$ compared to random for five sample sizes (at 50, 100, 200, 400, and 800). Error bars indicate ±1 standard deviations over ten experiments

of the top algorithms (nonmetric GP and KM, HGP).[9] In our experiments, the curves do *not* intersect indicating that ranking of the top performers does not change in the range of dataset sizes considered.

In terms of balance (Figs. 5 and 6b) the advantages of GP are clear. GP explicitly tries to achieve balanced clusters ($n = 800 : \phi^{(\mathrm{BAL})} \approx 0.9$). The second tier is HGP, which is also a balanced technique ($n = 800 : \phi^{(\mathrm{BAL})} \approx 0.7$) followed by nonmetric KM approaches ($n = 800 : \phi^{(\mathrm{BAL})} \approx 0.5$). Poor balancing

[9] Intersections of performance curves in classification (learning curves) have been studied recently, e.g., in [359].

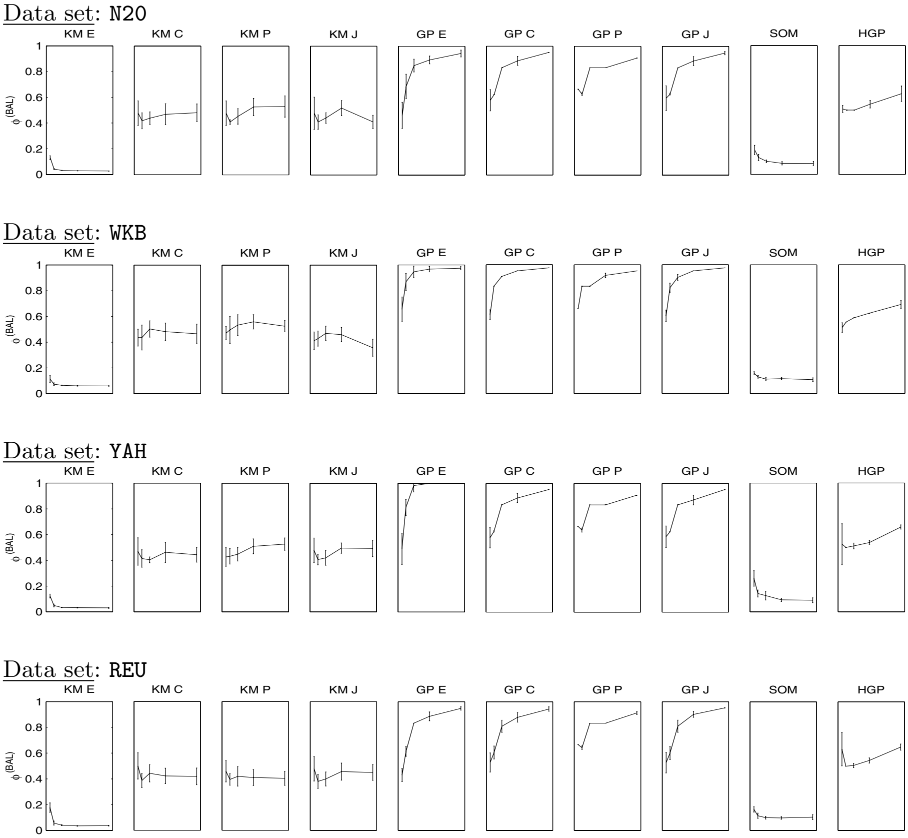
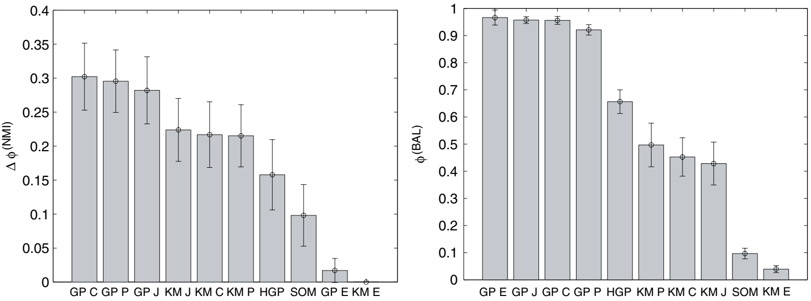


Fig. 5. Amount of balancing achieved for four data sets in combination with ten algorithms. Each curve shows the cluster balance $\phi^{(\mathrm{BAL})}$ for five sample sizes (at 50, 100, 200, 400, and 800). Error bars indicate ±1 standard deviations over ten experiments.

is shown by SOM and Euclidean KM ($n = 800 : \phi^{(\mathrm{BAL})} \approx 0.1$). Interestingly, balancedness does not change significantly for the KM-based approaches as the number of samples n increases. GP-based approaches quickly approach perfect balancing as would be expected since they are explicitly designed to do so.

Nonmetric GP is significantly better in terms of mutual information as well as balance. There is no significant difference in performance amongst the nonmetric similarity measures using cosine, correlation, and extended Jaccard. Euclidean distance-based approaches do not perform better than random clustering.



(a) (b)

Fig. 6. Comparison of cluster quality in terms of **(a)** mutual information and **(b)** balance on average over four data sets with ten trials each at 800 samples. Error bars indicate ±1 standard deviation. Graph partitioning is significantly better in terms of mutual information as well as in balance. Euclidean distance-based approaches do not perform better than random clustering

7 Conclusions

This work provides a mutual information-based comparison of several similarity-based clustering approaches to clustering of unannotated text *across* several similarity measures. It also provides a conceptual assessment of a variety of similarity measures and evaluation criteria.

The comparative results indicate that for the similarity measures considered, graph partitioning is better suited for word frequency-based clustering of web documents than generalized KM, HGP, and SOM. The search procedure implicit in GP is far less local than the hill-climbing approach of KM. Moreover, it also provides a way to obtain clusters of comparable sizes and exhibit a lower variance in results. Note that while this extra constraint is helpful for datasets that are reasonably balanced, it can degrade results when the classes are highly skewed. With regard to the appropriateness of various distance/similarity measures, it was very clear that metric distances such as the L_2 norm (Euclidean distance) are not appropriate for the high-dimensional, sparse domains that characterize text documents. Cosine, correlation, and extended Jaccard measures are much more successful and perform comparably in capturing the similarities implicitly indicated by manual categorizations of document collections. Note that all three measures tune to different degrees to the directional properties of the data, which is the likely reason for their effectiveness. This intuition is supported by the recent development of a generative model using mixture of von Mises–Fisher distributions from directional statistics and tailored for high-dimensional data, which has been applied to text clustering with clearly superior results [43]. Such generative models are also attractive since their computational complexity can be linear in the number of objects, as compared a mimimum of quadratic complexity for any similarity-based method that involves a comparison between each pair of objects.

Since document clustering is currently a popular topic, a comparative study such as that undertaken in this chapter is by nature an unfinished one as new techniques and aspects emerge regularly. For example, a recent paper introduces a similarity measure based on the number of neighbors two points share, and shows promising results on earth sciences data and word clustering [149]. It will be interesting to see how suitable this measure is for clustering a variety of text collections.

Acknowledgments

We thank Raymond Mooney and Inderjit Dhillon for helpful comments. This work was supported in part by NSF under grants IIS-0307792 and IIS-0325116.