Impact of Similarity Measures on Web-page Clustering

Alexander Strehl, Joydeep Ghosh, and Raymond Mooney
The University of Texas at Austin, Austin, TX, 78712-1084, USA

Email: strehl@ece.utexas.edu, ghosh@ece.utexas.edu, and mooney@cs.utexas.edu

Abstract

Clustering of web documents enables (semi-)automated categorization, and facilitates certain types of search. Any clustering method has to embed the documents in a suitable *similarity space*. While several clustering methods and the associated similarity measures have been proposed in the past, there is no systematic comparative study of the impact of similarity metrics on cluster quality, possibly because the popular cost criteria do not readily translate across qualitatively different metrics. We observe that in domains such as YAHOO that provide a categorization by human experts, a useful criteria for comparisons across similarity metrics is indeed available. We then compare four popular similarity measures (Euclidean, cosine, Pearson correlation and extended Jaccard) in conjunction with several clustering techniques (random, self-organizing feature map, hyper-graph partitioning, generalized k-means, weighted graph partitioning), on high dimensional sparse data representing web documents. Performance is measured against a human-imposed classification into news categories and industry categories. We conduct a number of experiments and use t-tests to assure statistical significance of results. Cosine and extended Jaccard similarities emerge as the best measures to capture human categorization behavior, while Euclidean performs poorest. Also, weighted graph partitioning approaches are clearly superior to all others.

Introduction

The increasing size and dynamic content of the world wide web has created a need for automated organization of web-pages. Document clusters can provide a structure for organizing large bodies of text for efficient browsing and searching. For this purpose, a web-page is typically represented as a vector consisting of the suitably normalized frequency counts of words or terms. Each document contains only a small percentage of all the words ever used in the web. If we consider each document as a multi-dimensional vector and then try to cluster documents based on their word contents, the problem differs from classic clustering scenarios in several ways. Document clustering data is high dimensional, characterized by a highly sparse word-document matrix with positive ordinal attribute values and a significant amount of outliers.

Clustering has been widely studied in several disciplines, specially since the early 60's (Hartigan 1975). Some classic approaches include partitional methods such as k-means, hierarchical agglomerative clustering, unsupervised Bayes, and soft, statistical mechanics based techniques. Most classical techniques, and even fairly recent ones proposed in the data mining community (CLARANS, DBSCAN, BIRCH, CLIQUE, CURE, WAVECLUSTER etc. (Rastogi & Shim 1999)), are based on distances between the samples in the original vector space. Thus they are faced with the "curse of dimensionality" and the associated sparsity issues, when dealing with very high dimensional data. Indeed, often, the performance of such clustering algorithms is demonstrated only on illustrative 2–dimensional examples.

When documents are represented by a bag of words, the resulting document-word matrix typically represents data in 1000+ dimensions. Several noteworthy attempts have emerged to efficiently cluster documents that are represented in such high dimensional space[1]. In (Dhillon & Modha 1999), the authors present a spherical k-means algorithm for document clustering. Graph-based clustering approaches, that attempt to avoid the curse of dimensionality by transforming the problem formulation include (Karypis, Han, & Kumar 1999; Boley *et al.* 1999; Strehl & Ghosh 2000). Note that such methods use a variety of similarity (or distance) measures, literature, and we are unaware of any solid comparative study across different similarity measures.

In this paper, we first compare similarity measures analytically and illustrate their semantics geometrically. Secondly, we propose an experimental methodology to compare high dimensional clusterings based on mutual information, entropy, and purity. We conduct a series of experiments on YAHOO news pages to evaluate the performance and cluster quality of four simi-

[1] There is also substantial work on *categorizing* such documents. Here, since at least some of the documents have labels, a variety of supervised or semi-supervised techniques can be used (Mooney & Roy 1999; Yang 1999)

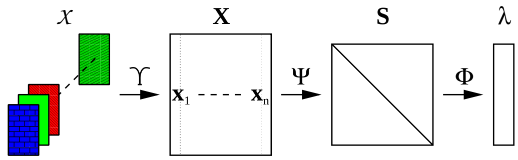

Figure 1: Overview of a similarity based clustering framework.

larity measures (Euclidean, cosine, Pearson correlation, extended Jaccard) in combination with five algorithms (random, self-organizing feature map, hyper-graph partitioning, generalized k-means, weighted graph partitioning).

Let n be the number of objects (web-pages) in the data and d the number of features (words, terms) for each sample $\mathbf{x}_j$ with $j \in \{1, \ldots, n\}$. The input data can be represented by a $d \times n$ word-document matrix $\mathbf{X}$ with the j-th column representing the sample $\mathbf{x}_j$. Hard clustering [2] assigns a label λ_j to each d–dimensional sample $\mathbf{x}_j$, such that similar samples tend to get the same label. The number of distinct labels is k, the desired number of clusters. In general the labels are treated as nominals with no inherent order, though in some cases, such as self-organizing feature maps (SOFMs) or top-down recursive graph-bisection, the labeling may contain extra ordering information. Let $\mathcal{C}_\ell$ denote the set of all objects in the ℓ-th cluster ($\ell \in \{1, \ldots, k\}$), with $\mathbf{x}_j \in \mathcal{C}_\ell \Leftrightarrow \lambda_j = \ell$ and $n_\ell = |\mathcal{C}_\ell|$. Figure 1 gives an overview of a batch clustering process from a set of raw object descriptions $\mathcal{X}$ via the vector space description $\mathbf{X}$ and similarity space description $\mathbf{S}$ to the cluster labels λ: $(\mathcal{X} \in \mathcal{I}^n) \stackrel{\Upsilon}{\rightarrow} (\mathbf{X} \in \mathcal{F}^n \subset \mathbb{R}^{d \times n}) \stackrel{\Psi}{\rightarrow} (\mathbf{S} \in \mathcal{S}^{n \times n} = [0,1]^{n \times n} \subset \mathbb{R}^{n \times n}) \stackrel{\Phi}{\rightarrow} (\lambda \in \mathcal{O}^n = \{1, \ldots, k\}^n)$. The next section briefly describes the compared algorithms.

Algorithms

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 $O(n)$.

Self-organizing Feature Map

We use a 1–dimensional SOFM as proposed by Kohonen (Kohonen 1995). To generate k clusters we use k cells in a line topology and train the network for $m = 5000$ epochs or 10 minutes (whichever comes first). All experiments are run on a dual processor 450 MHz Pentium and for this clustering technique we use the SOFM implementation in the MATLAB neural network tool-box. 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.

Generalized k-means

We also employed the well-known k-means algorithm and three variations of it using non-Euclidean distance measures. The k-means algorithm is an iterative algorithm to minimize the least squares error criterion (Duda & Hart 1973). A cluster $\mathcal{C}_\ell$ is represented by its center μ_ℓ, the mean of all samples in $\mathcal{C}_\ell$. The centers are initialized with a random selection of k data objects. Each sample is then labeled with the index i of the *nearest* or *most similar* center. In the following subsections we will describe four different semantics for closeness or similarity $s(\mathbf{x}_a, \mathbf{x}_b)$ of two objects $\mathbf{x}_a$ and $\mathbf{x}_b$. Subsequent re-computing of the mean for each cluster and re-assigning the cluster labels is iterated until convergence to a fixed labeling after m iterations. The complexity of this algorithm is $O(n \cdot d \cdot k \cdot m)$.

Weighted Graph Partitioning

The objects to be clustered can be viewed as a set of vertices $\mathcal{V}$. Two web-pages $\mathbf{x}_a$ 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 *non-zero* 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 sub-graphs $\mathcal{G}_\ell = (\mathcal{V}_\ell, \mathcal{E}_\ell)$, is called an edge separator. Our objective 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 decided to use OPOSSUM (Strehl & Ghosh 2000), which produces balanced (equal sized) clusters from the similarity matrix using multi-level multi-constraint graph partitioning (Karypis & Kumar 1998). Balanced clusters are desirable because each cluster represents an equally important share of the data. However, some natural classes may not be equal size. By using a higher number of clusters we can account for multi-modal classes (e.g., XOR-problem) and clusters can be merged at a latter stage. 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. In web-page clustering sparsity is induced by all non-Euclidean similarities proposed in this paper, and may be increased by a thresholding criterion.

Hyper-graph Partitioning

A hyper-graph is a graph whose edges can connect more than two vertices (hyper-edges). The clustering problem is then formulated as a finding the minimum-cut of

[2] In *soft* clustering, a record can belong to multiple clusters with different degrees of "association" (Kumar & Ghosh 1999).

a hyper-graph. A minimum-cut is the removal of the set of hyper-edges (with minimum edge weight) that separates the hyper-graph into k unconnected components. Again, an object $\mathbf{x}_j$ maps to a vertex v_j. Each word (feature) maps to a hyper-edge connecting all vertices with non-zero frequency count of this word. The weight of this hyper-edge is chosen to be the total number of occurrences in the data-set. Hence, the importance of a hyper-edge during partitioning is proportional to the occurrence of the corresponding word. The minimum-cut of this hyper-graph into k unconnected components gives the desired clustering. We employ the HMETIS package 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 hyper-graph partitioner. However, sample-wise frequency information gets lost in this formulation since there is only one weight associated with a hyper-edge.

Similarity Measures

Metric Distances

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 the standard metrics for geometrical problems. For $p = 1$ ($p = 2$) we obtain the Manhattan (Euclidean) distance. For Euclidean space, we chose to relate distances d and similarities s using $s = e^{-d^2}$. Consequently, we define Euclidean $[0,1]$ normalized similarity as $s^{(\mathrm{E})}(\mathbf{x}_a, \mathbf{x}_b) = e^{-\|\mathbf{x}_a - \mathbf{x}_b\|_2^2}$ which has important properties (as we will see in the discussion) that the commonly adopted $s(\mathbf{x}_a, \mathbf{x}_b) = 1/(1 + \|\mathbf{x}_a - \mathbf{x}_b\|_2)$ lacks.

Cosine Measure

Similarity can also be defined by the angle or cosine of the angle between two vectors. The cosine measure is given by $s^{(\mathrm{C})}(\mathbf{x}_a, \mathbf{x}_b) = \frac{\mathbf{x}_a^\dagger \mathbf{x}_b}{\|\mathbf{x}_a\|_2 \cdot \|\mathbf{x}_b\|_2}$ and captures a scale invariant understanding of similarity. An even stronger property is that the cosine similarity does not depend on the length: $s^{(\mathrm{C})}(\alpha \mathbf{x}_a, \mathbf{x}_b) = s^{(\mathrm{C})}(\mathbf{x}_a, \mathbf{x}_b)$ for $\alpha > 0$. This allows documents with the same composition, but different totals to be treated identically which makes this the most popular measure for text documents. Also, due to this property, samples can be normalized to the unit sphere for more efficient processing (Dhillon & Modha 1999).

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 $s^{(\mathrm{P})}(\mathbf{x}_a, \mathbf{x}_b) = \frac{1}{2}\left(\frac{(\mathbf{x}_a - \bar{x}_a)^\dagger (\mathbf{x}_b - \bar{x}_b)}{\|\mathbf{x}_a - \bar{x}_a\|_2 \cdot \|\mathbf{x}_b - \bar{x}_b\|_2} + 1\right)$, where $\bar{x}$ denotes the average feature value of $\mathbf{x}$ over all dimensions.

Extended Jaccard Similarity

The binary Jaccard coefficient measures 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 $\mathbf{x}_b$. It is often used in retail market-basket applications. Jaccard similarity can be extended to continuous or discrete *non-negative* features using $s^{(\mathrm{J})}(\mathbf{x}_a, \mathbf{x}_b) = \frac{\mathbf{x}_a^\dagger \mathbf{x}_b}{\|\mathbf{x}_a\|_2^2 + \|\mathbf{x}_b\|_2^2 - \mathbf{x}_a^\dagger \mathbf{x}_b}$ (Strehl & Ghosh 2000).

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 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 variant while cosine is translation variant but scale invariant. The extended Jaccard has aspects of both properties as illustrated in figure 2. Iso-similarity lines at $s = 0.25$, 0.5 and 0.75 for points $\mathbf{x}_1 = (3\ 1)^\dagger$ and $\mathbf{x}_2 = (1\ 2)^\dagger$ 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 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 extended Jaccard similarity.

In Euclidean space, iso-similarities are concentric hyper-spheres around the considered sample point (vector). Due to the finite range of similarity, the radius decreases hyperbolically as s increases linearly. The radius is constant for a given similarity regardless of the center-point. The only location with similarity of 1 is the considered point itself and no location at finite distance has a similarity of 0 (no sparsity). Using the cosine measure renders the iso-similarities to be hyper-cones all having their apex at the origin and axis aligned with the given sample (vector). Locations with similarity 1 are on the 1–dimensional sub-space defined by this axis and the locus of points with similarity 0 is the hyper-plane perpendicular to this axis. For the extended Jaccard similarity, the iso-similarities are non-concentric hyper-spheres. The only location with $s = 1$ is the point itself. The hyper-sphere radius increases with the 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 approaches 0 the radius becomes infinite. The resulting iso-similarity surface is the hyper-plane perpendicular to the considered point through the origin. Thus, for $s \to 0$, extended Jaccard behaves like the cosine measure, and for $s \to 1$, behaves like the Euclidean distance.

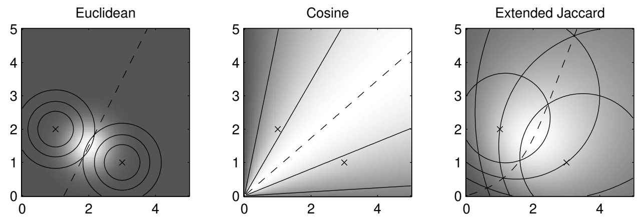

Figure 2: Properties of various similarity measures. The extended Jaccard adopts the middle ground between Euclidean and cosine based similarity.

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

$$\mathbf{c}_\ell = \arg\min_{\mathbf{z}\in\mathcal{F}} \sum_{\mathbf{x}_j\in\mathcal{C}_\ell} \|\mathbf{x}_j - \mathbf{z}\|_2^2. \tag{1}$$

In the following, we proof 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 $\mathbf{z}$: $f(\mathcal{C}_\ell, \mathbf{z}) = \sum_{\mathbf{x}_j\in\mathcal{C}_\ell} d(\mathbf{x}_j, \mathbf{z})^2 = \sum_{\mathbf{x}_j\in\mathcal{C}_\ell} \|\mathbf{x}_j - \mathbf{z}\|_2^2$. Mapping from distances to similarities yields $f(\mathcal{C}_\ell, \mathbf{z}) = \sum_{\mathbf{x}_j\in\mathcal{C}_\ell} -\log(s(\mathbf{x}_j, \mathbf{z}))$, and therefore $f(\mathcal{C}_\ell, \mathbf{z}) = -\log\prod_{\mathbf{x}_j\in\mathcal{C}_\ell} s(\mathbf{x}_j, \mathbf{z})$. Finally, we transform the objective using a strictly monotonic decreasing function: Instead of minimizing $f(\mathcal{C}_\ell, \mathbf{z})$, we maximize $f'(\mathcal{C}_\ell, \mathbf{z}) = e^{-f(\mathcal{C}_\ell, \mathbf{z})}$. Thus, in similarity space $\mathcal{S}$, the least squared error representative $\mathbf{c}_\ell \in \mathcal{F}$ for a cluster $\mathcal{C}_\ell$ satisfies

$$\mathbf{c}_\ell = \arg\max_{\mathbf{z}\in\mathcal{F}} \prod_{\mathbf{x}_j\in\mathcal{C}_\ell} s(\mathbf{x}_j, \mathbf{z}). \tag{2}$$

Using the concave evaluation function f', we can obtain optimal representatives for non-Euclidean similarity spaces. The values of the evaluation function $f'(\{\mathbf{x}_1, \mathbf{x}_2\}, \mathbf{z})$ are used to shade the background in figure 2. 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.

Experimental Evaluation

Methodology

We conducted experiments with all five algorithms, using four variants each for k-means and graph partitioning, yielding eleven techniques in total. Since clustering is unsupervised, success is generally measured by a cost criterion. Standard cost functions, such as the sum of squared distances from cluster representative depend on the the similarity (or distance) measure employed. They cannot be used to compare techniques that use different similarity measures. However, in situations where the pages are categorized (labelled) by an external source, there is a plausible way out! Given g categories (classes) $\mathcal{K}_h$ ($h \in \{1,\ldots,g\}$, $\mathbf{x}_j \in \mathcal{K}_h \Leftrightarrow \kappa_j = h$), we use the "true" classification labels κ to evaluate the performance. For evaluating a single cluster, we use *purity* and *entropy*, while the entire clustering is evaluated using *mutual information*.

Let $n_\ell^{(h)}$ denote the number of objects in cluster $\mathcal{C}_\ell$ that are classified to be h as given by κ. Cluster $\mathcal{C}_\ell$'s *purity* can be defined as

$$\Lambda^{(\mathrm{P})}(\mathcal{C}_\ell) = \frac{1}{n_\ell}\max_h(n_\ell^{(h)}). \tag{3}$$

Purity can be interpreted as the classification rate under the assumption that all samples of a cluster are predicted to be members of the actual dominant class for that cluster. Alternatively, we also use $[0, 1]$ *entropy*, which is defined for a g class problem as

$$\Lambda^{(\mathrm{E})}(\mathcal{C}_\ell) = \sum_{h=1}^{g} \frac{n_\ell^{(h)}}{n_\ell}\log\left(\frac{n_\ell^{(h)}}{n_\ell}\right)/\log(g). \tag{4}$$

Entropy is a more comprehensive measure than purity since rather than just considering the number of objects "in" and "not in" the most frequent class, it considers the entire distribution.

While the above two criteria are suitable for measuring a single cluster's quality, they are biased to favor smaller clusters. In fact, for both these criteria, the globally optimal value is trivially reached when each cluster is a single sample! Consequently, for the overall (not cluster-wise) performance evaluation, we use a measure based on *mutual information*:

$$\Lambda^{(\mathrm{M})}(\kappa, \lambda) = \frac{1}{n}\sum_{\ell=1}^{k}\sum_{h=1}^{g} n_\ell^{(h)} \frac{\log\left(\frac{n_\ell^{(h)} n}{\sum_{i=1}^{k} n_i^{(h)} \sum_{i=1}^{g} n_\ell^{(i)}}\right)}{\log(k\cdot g)} \tag{5}$$

Mutual information is a symmetric measure for the degree of dependency between the clustering and the categorization. Unlike correlation, mutual information also takes higher order dependencies into account. We use the symmetric mutual information criterion because it successfully captures how related the labeling and categorizations are without a bias towards smaller clusters.

Since these performance measures are affected by the distribution of the data (e.g., *a priori* sizes), we normalize the performance by that of the corresponding random clustering, and interpret the resulting ratio as "performance lift".

Findings on Industry Web-page Data

From the YAHOO industry web-page data (CMU Web KB Project (Craven *et al.* 1998)), the following ten industry sectors 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. The frequencies of 2896 different words that are not in a standard English stop-list (e.g., `a`, `and`, `are`, ...) and do occur on average between 0.01 and 0.1 times per page, were extracted from Html. Word location was not considered. This data is far less clean than e.g., the Reuters data. Documents vary significantly in length, some are in the wrong category, same are out-dated 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. Sample sizes of 50, 100, 200, 400, and 800 were used for clustering 966 documents from the above 10 categories. The number of clusters k was set to 20 and each setting was run 10 times (each technique gets the same data) to capture the random variation in results.

Figure 3 shows the results of the 550 experiments. In table 1, the t-test results indicate that graph-partitioning with cosine similarity performs best closely followed by the other two non-Euclidean measures. The second tier are non-Euclidean k-means variations. Hyper-graph partitioning performs reasonably well. The Sofm and Euclidean similarity with k-means as well as with graph partitioning, fail to capture the relationships of the high dimensional data.

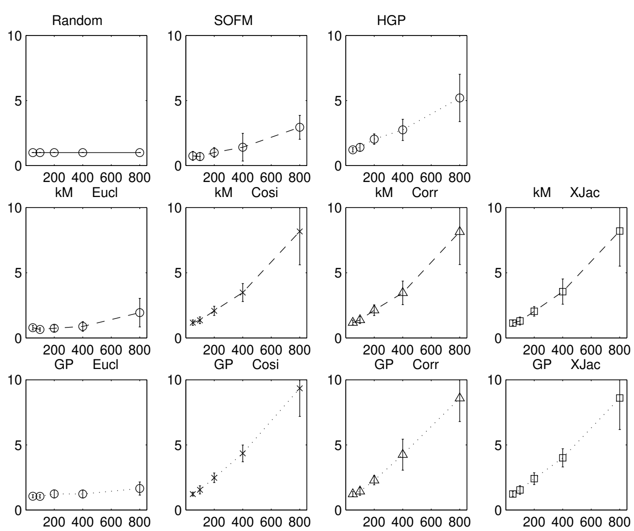

Figure 3: Performance lift (normalized to the random baseline) on Yahoo industry web-page data of mutual information $\Lambda^{(M)}$ for various sample sizes n. The bars indicate ± 2 standard deviations.

Findings on News Web-pages

The 20 original Yahoo news categories in the data are `Business`, `Entertainment (no sub-category`, `art`, `cable`, `culture`, `film`, `industry`, `media`, `multimedia`, `music`, `online`, `people`, `review`, `stage`, `television`, `variety`), `Health`, `Politics`, `Sports`, `Technology` and correspond to $\kappa = 1, \ldots, 20$, respectively. The data is publicly available from `ftp://ftp.cs.umn.edu/dept/users/boley/` (K1 series) and was used in (Boley *et al.* 1999). The raw 21839×2340 word-document matrix consists of the non-normalized occurrence frequencies of stemmed words, using Porter's suffix stripping algorithm (Frakes 1992). Pruning all words that occur less than 0.01 or more than 0.10 times on average because they are insignificant (e.g., `abdrazakof`) or too generic (e.g., `new`), respectively, results in $d = 2903$.

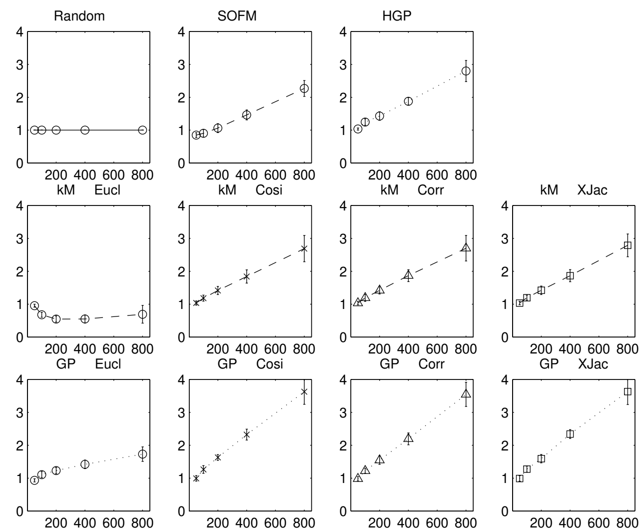

Figure 4: Performance lift (normalized to the random baseline) on Yahoo news web-page data of mutual information $\Lambda^{(M)}$ for various sample sizes n. The bars indicate ± 2 standard deviations.

Sample sizes of 50, 100, 200, 400, and 800 were used for clustering 2340 documents from the above 20 categories. The number of clusters k was set to 40 and each setting was run 10 times (each technique gets the same sub-sample) to capture the random variation in results. We chose 40 clusters, two times the number of categories, since this seemed to be the more natural number of clusters as indicated by preliminary runs and visualization. Using a greater number of clusters than classes can be viewed as allowing multi-modal distributions for each classes. For example, in an Xor like problem, there are two classes, but four clusters. 550 clustering runs were conducted (figure 4) and the results were evaluated in 55 one-sided t-tests (table 2) for the $n = 800$ sample level.

Non-Euclidean graph partitioning approaches work best on the data. The top performing similarity measures are extended Jaccard and cosine. We initially expected cosine to perform better than the extended Jac-

	$\Lambda^{(M)}$	GP Cosi	GP Corr	GP XJac	kM XJac	kM Cosi	kM Corr	HGP	SOFM	kM Eucl	GP Eucl	Random
GP Cosi	0.192	-	0.998	0.997	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
GP Corr	0.177	-	-	-	0.957	0.967	0.968	1.000	1.000	1.000	1.000	1.000
GP XJac	0.177	-	-	-	-	0.956	0.958	1.000	1.000	1.000	1.000	1.000
kM XJac	0.168	-	-	-	-	-	-	1.000	1.000	1.000	1.000	1.000
kM Cosi	0.167	-	-	-	-	-	-	1.000	1.000	1.000	1.000	1.000
kM Corr	0.167	-	-	-	-	-	-	1.000	1.000	1.000	1.000	1.000
HGP	0.107	-	-	-	-	-	-	-	1.000	1.000	1.000	1.000
SOFM	0.061	-	-	-	-	-	-	-	-	1.000	1.000	1.000
kM Eucl	0.039	-	-	-	-	-	-	-	-	-	0.971	1.000
GP Eucl	0.034	-	-	-	-	-	-	-	-	-	-	1.000
Random	0.021	-	-	-	-	-	-	-	-	-	-	-

Table 1: Industry web-page data with $n = 966$, $d = 2896$, $g = 6$, and $k = 20$. Comparison of 10 trials of techniques at 800 samples in terms of $\Lambda^{(M)}$ performance and t-test results (confidences below 0.950 are marked with '-').

	$\Lambda^{(M)}$	GP XJac	GP Cosi	GP Corr	HGP	kM XJac	kM Corr	kM Cosi	SOFM	GP Eucl	Random	kM Eucl
GP XJac	0.240	-	-	0.991	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
GP Cosi	0.240	-	-	0.990	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
GP Corr	0.234	-	-	-	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
HGP	0.185	-	-	-	-	-	0.982	0.988	1.000	1.000	1.000	1.000
kM XJac	0.184	-	-	-	-	-	0.992	0.994	1.000	1.000	1.000	1.000
kM Corr	0.178	-	-	-	-	-	-	-	1.000	1.000	1.000	1.000
kM Cosi	0.178	-	-	-	-	-	-	-	1.000	1.000	1.000	1.000
SOFM	0.150	-	-	-	-	-	-	-	-	1.000	1.000	1.000
GP Eucl	0.114	-	-	-	-	-	-	-	-	-	1.000	1.000
Random	0.066	-	-	-	-	-	-	-	-	-	-	1.000
kM Eucl	0.046	-	-	-	-	-	-	-	-	-	-	-

Table 2: News web-page data with $n = 2340$, $d = 2903$, $g = 20$, and $k = 40$. Comparison of 10 trials of techniques at 800 samples in terms of $\Lambda^{(M)}$ performance and t-test results (confidences below 0.950 are marked with '-').

card and correlation due to its length invariance. The middle ground viewpoint of extended Jaccard seems to be successful in web-page as well as market-basket applications. Correlation is only marginally worse in terms of average performance. Hyper-graph partitioning is in the third tier, outperforming all generalized k-means algorithms except for the extended Jaccard. All Euclidean techniques including SOFM performed very poorly. Surprisingly, SOFM and graph partitioning were still able to do significantly better than random despite the limited expressiveness of Euclidean similarity. Euclidean k-means performed even worse than random in terms of entropy and equivalent to random in terms of purity (not shown).

Table 3 shows the results of the best performing OPOSSUM clustering (Strehl & Ghosh 2000). For each cluster the dominant category, its purity and entropy are given along with the top three descriptive and discriminative words. Descriptiveness is defined as occurrence frequency (a notion similar to singular item-set support). Discriminative terms for a cluster have the highest occurrence multipliers compared to the average document (similar to the notion of singular item-set lift). Unlike the YAHOO categories, which vary in size from 9 to 494 pages (!), all our clusters are well-balanced: each contains between 57 and 60 pages. Health ($\mathcal{K}_{\mathrm{H}}$) turned out to be the category most clearly identified. This could have been expected since its language separates quite distinctively from the others. However, our clustering is better than just matching the YAHOO given labels, because distinguishes more precisely. For example there are detected sub-classes such as HIV ($\mathcal{C}_9$) and genetics related ($\mathcal{C}_{10}$) pages. Cluster 8, for example is described by our system through the terms `vaccin`, `strain`, `antibiot` indicating an infection related cluster. Similarly, in the `Entertainment people` category, our algorithm identifies a cluster dealing with Princess Diana's car accident ($\mathcal{C}_{24}$) and funeral ($\mathcal{C}_{25}$). Some smaller categories, such as `Entertainment no sub-category` pages ($\mathcal{K}_{\mathrm{E}}$) have been absorbed into more meaningful clusters. Most `Technology` pages are found in cluster $\mathcal{C}_{12}$. Interestingly, documents from the technology category have also been grouped with a Entertainment-online dominated and a Business dominated cluster indicating an overlap of topics. Clusterings of this quality may be used to build a fully automated web-page categorization engine yielding cleaner cut groups than currently seen.

Concluding Remarks

The key contribution of this work lies in providing a framework for comparing several clustering approaches *across* a variety of similarity spaces. The results indicate that graph partitioning is better suited for word frequency based clustering of web documents than generalized k-means, hyper-graph partitioning, and SOFM. The search procedure implicit in graph partitioning is far less local than the hill-climbing approach of k-means. Moreover, it also provides a way to obtain balanced clusters and exhibit a lower variance in results.

Metric distances such as Euclidean are not appropriate for high dimensional, sparse domains. Cosine, correlation and extended Jaccard measures are successful in capturing the similarities implicitly indicated by manual categorizations as seen for example in YAHOO.

Acknowledgments: This research was supported in

$\mathcal{C}_\ell$	$\mathcal{K}_{\hat{h}}$	$\Lambda^{(\mathrm{P})}$	$\Lambda^{(\mathrm{E})}$	top 3 descriptive terms	top 3 discriminative terms
1	H	45.76%	0.60533	abus, label, addict	mckinnei, grammer, addict
2	H	35.09%	0.68312	suicid, israel, jet	broccoli, tractor, weizman
3	H	74.14%	0.19081	surgeri, arteri, kidnei	vander, stent, pippen
4	H	88.14%	0.15943	fda, safeti, seate	cartilag, latex, fda
5	H	100.00%	0	smok, smoker, lung	nonsmok, clozapin, prostat
6	H	100.00%	0	weight, pregnanc, obes	insulin, calor, heparin
7	H	100.00%	0	breast, vitamin, diet	miner, fatti, estrogen
8	H	100.00%	0	vaccin, strain, antibiot	aureu, vancomycin, influenza
9	H	100.00%	0	hiv, depress, immun	chemotherapi, hiv, radiosurgeri
10	H	100.00%	0	mutat, genet, protein	chromosom, mutat, prion
11	o	60.00%	0.34318	apple, intel, electron	gorman, ibm, compaq
12	T	63.79%	0.44922	java, advertis, sun	nader, lucent, java
13	P	18.97%	0.81593	miami, fcc, contract	panama, pirat, trump
14	P	56.90%	0.47765	appeal, suprem, justic	iraqi, nato, suprem
15	P	84.75%	0.20892	republican, committe, reform	teamster, government, reno
16	S	88.14%	0.17215	smith, coach, marlin	oriol, homer, marlin
17	S	70.18%	0.35609	goal, yard, pass	touchdown, defenseman, yard
18	i	43.10%	0.59554	usa, murdoch, channel	biondi, viacom, pearson
19	B	73.33%	0.28802	cent, quarter, revenu	ahmanson, loral, gm
20	B	82.76%	0.24307	dow, greenspan, rose	dow, greenspan, treasuri
21	p	54.24%	0.52572	notabl, canadian, magazin	stamp, notabl, polanski
22	f	26.32%	0.73406	opera, bing, draw	bing, pageant, lange
23	cu	46.55%	0.59758	bestsell, weekli, hardcov	hardcov, paperback, bestsell
24	p	64.41%	0.43644	crash, paparazzi, pari	merced, stephan, manslaught
25	p	45.76%	0.59875	funer, royal, prince	buckingham, grief, spencer
26	mu	22.41%	0.69697	meredith, classic, spice	burgess, meredith, espn
27	t	23.73%	0.69376	radio, prodigi, station	cybercast, prodigi, fo
28	mu	53.45%	0.39381	concert, band, stage	bowie, ballad, solo
29	p	68.33%	0.29712	showbiz, academi, south	cape, showbiz, calendar
30	p	32.76%	0.63372	albert, stone, tour	jagger, marv, forcibl
31	f	77.59%	0.30269	script, miramax, sequel	sequel, bon, cameron
32	f	76.27%	0.30759	cast, shoot, opposit	showtim, cast, duvall
33	r	43.10%	0.49547	tom, theater, writer	cusack, selleck, rep
34	r	64.41%	0.37305	script, scot, tom	nichola, horse, ira
35	r	93.22%	0.10628	camera, pic, sound	juliett, narr, costum
36	S	48.28%	0.48187	japanes, se, hingi	porsche, hingi, quarterfin
37	t	39.66%	0.51945	nomin, hbo, winner	miniseri, hbo, kim
38	t	55.17%	0.42107	king, sitcom, dreamwork	winfrei, dreamwork, oprah
39	f	76.67%	0.29586	weekend, gross, movi	gross, weekend, monti
40	t	70.69%	0.34018	household, timeslot, slot	denot, timeslot, datelin

	B	E	a	c	cu	f	i	m	mm	mu	o	p	r	s	t	v	H	P	S	T
19	**44**	-	-	-	-	-	6	-	-	-	-	1	-	-	1	-	-	-	-	8
20	**48**	-	-	-	-	-	3	2	-	-	-	-	-	-	2	-	-	2	-	1
23	-	1	1	3	**27**	5	1	2	1	1	-	10	4	-	2	-	-	-	-	-
22	-	-	3	-	8	**15**	1	1	-	4	3	7	6	3	5	-	-	-	1	-
31	-	-	-	2	-	**45**	1	-	-	-	1	2	-	-	3	4	-	-	-	-
32	-	-	-	3	-	**45**	-	-	-	-	-	2	-	2	4	3	-	-	-	-
39	-	2	1	-	-	**46**	-	-	-	-	-	5	-	-	4	2	-	-	-	-
18	10	1	-	7	2	-	**25**	3	-	2	-	1	-	-	4	3	-	-	-	-
26	-	-	3	4	-	9	1	-	1	**13**	-	11	5	1	8	2	-	-	-	-
28	-	1	2	-	1	1	-	-	-	**31**	-	3	19	-	-	-	-	-	-	-
11	8	-	-	-	-	-	2	-	-	-	**36**	-	-	-	-	-	-	-	-	14
21	-	-	4	-	6	6	2	-	-	4	2	**32**	1	-	2	-	-	-	-	-
24	-	-	2	-	3	4	1	4	-	2	-	**38**	-	-	5	-	-	-	-	-
25	-	1	-	2	3	5	-	-	-	8	1	**27**	2	1	7	2	-	-	-	-
29	-	-	-	-	1	7	-	-	-	11	-	**41**	-	-	-	-	-	-	-	-
30	-	-	1	1	5	9	-	1	1	14	-	**19**	3	-	1	1	-	-	2	-
33	-	-	-	1	2	19	2	-	-	-	-	4	**25**	2	-	3	-	-	-	-
34	-	-	-	-	-	8	1	-	1	-	-	9	**38**	-	1	1	-	-	-	-
35	-	-	-	-	-	2	1	-	-	-	1	-	**55**	-	-	-	-	-	-	-
27	-	-	-	-	6	5	3	1	2	12	10	3	-	2	**14**	1	-	-	-	-
37	-	-	-	6	-	16	1	-	-	2	-	1	-	1	**23**	8	-	-	-	-
38	-	-	1	1	-	5	2	-	-	-	-	2	-	-	**32**	15	-	-	-	-
40	-	-	-	1	-	5	-	-	-	-	-	1	-	5	**41**	5	-	-	-	-
1	3	-	1	1	1	1	-	-	1	2	3	10	-	-	5	-	**27**	4	-	-
2	-	1	1	2	1	2	1	1	-	3	-	6	-	-	6	-	**20**	10	3	-
3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**43**	-	15	-
4	2	-	-	-	-	-	-	-	-	-	-	1	-	-	-	-	**52**	4	-	-
5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**59**	-	-	-
6	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**58**	-	-	-
7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**59**	-	-	-
8	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**58**	-	-	-
9	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**60**	-	-	-
10	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**58**	-	-	-
13	3	2	2	5	3	1	8	2	-	7	1	3	-	1	7	2	-	**11**	-	-
14	11	-	2	2	2	-	-	1	-	-	3	1	-	-	3	-	-	**33**	-	-
15	5	-	-	-	1	-	1	1	-	-	-	1	-	-	-	-	-	**50**	-	-
16	-	-	-	-	1	1	3	-	-	-	-	-	-	-	2	-	-	-	**52**	-
17	1	-	-	-	-	-	3	-	6	5	-	1	-	-	1	-	-	-	**40**	-
36	-	-	-	2	1	16	-	-	-	3	-	5	-	-	1	2	-	-	**28**	-
12	7	-	-	1	-	-	1	2	1	1	4	1	-	-	3	-	-	-	-	**37**

Table 3: Best clustering for $k = 40$ using OPOSSUM and extended Jaccard similarity on 2340 YAHOO news pages. Cluster evaluations, their descriptive and discriminative terms (left) as well as the confusion matrix (right).

part by NSF grant ECS-9900353. We thank Inderjit Dhillon for helpful comments.

References

Boley, D.; Gini, M.; Gross, R.; Han, E.; Hastings, K.; Karypis, G.; Kumar, V.; Mobasher, B.; and Moore, J. 1999. Partitioning-based clustering for web document categorization. *Decision Support Systems* 27:329–341.

Craven, M.; DiPasquo, D.; Freitag, D.; McCallum, A.; Mitchell, T.; Nigam, K.; and Slattery, S. 1998. Learning to extract symbolic knowledge from the world wide web. In *AAAI98*, 509–516.

Dhillon, I. S., and Modha, D. S. 1999. Concept decompositions for large sparse text data using clustering. Technical Report RJ 10147, IBM Almaden Research Center. To appear in Machine Learning.

Duda, R. O., and Hart, P. E. 1973. *Pattern Classification and Scene Analysis*. New York: Wiley.

Frakes, W. 1992. Stemming algorithms. In Frakes, W., and Baeza-Yates, R., eds., *Information Retrieval: Data Structures and Algorithms*. New Jersey: Prentice Hall. 131–160.

Hartigan, J. A. 1975. *Clustering Algorithms*. New York: Wiley.

Karypis, G., and Kumar, V. 1998. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal of Scientific Computing* 20(1):359–392.

Karypis, G.; Han, E.-H.; and Kumar, V. 1999. Chameleon: Hierarchical clustering using dynamic modeling. *IEEE Computer* 32(8):68–75.

Kohonen, T. 1995. *Self-Organizing Maps*. Berlin, Heidelberg: Springer. (Second Extended Edition 1997).

Kumar, S., and Ghosh, J. 1999. GAMLS: A generalized framework for associative modular learning systems. In *Proceedings of the Applications and Science of Computational Intelligence II*, 24–34.

Mooney, R. J., and Roy, L. 1999. Content-based book recommending using learning for text categorization. In *Proceedings pf the SIGIR-99 Workshop on Recommender Systems: Algorithms and Evaluation*.

Rastogi, R., and Shim, K. 1999. Scalable algorithms for mining large databases. In Han, J., ed., *KDD-99 Tutorial Notes*. ACM.

Strehl, A., and Ghosh, J. 2000. Value-based customer grouping from large retail data-sets. In *Proceedings of the SPIE Conference on Data Mining and Knowledge Discovery, 24-25 April 2000, Orlando, Florida, USA*, volume 4057, 33–42. SPIE. To appear.

Yang, Y. 1999. An evaluation of statistical approaches to text categorization. *Journal of Information Retrieval*.