CS999: Web Data and Text Mining
Annotated Bibliography

Justin Kamerman 3335272

April 1, 2011

Bibliography

[1] M. Ackerman and S. Ben-David, "Measures of clustering quality: A working set of axioms for clustering," in *Proceedings of the 22nd Annual Conference on Neural Information Systems Processing*, 2008.

The paper analyses cluster quality measures in general and proposes a set of requirements that well behaved quality measures should exhibit. The paper is an extension of [15] in which Kleinberg advocates the development of a cluster theory independent of algorithm, objective function, or generative model. The paper shows how Kleinberg's *cluster function* axioms, which he found to be mutually inconsistent, can be transformed into axioms about a *clustering-quality measure*. These transformed axioms are relaxed to such a point that they are no longer inconsistent.

The axioms proposed are as follows:

- **Scale Invariance**: the quality measure is not effected by uniform scaling of the distance function output.
- **Isomorphism Invariance**: the quality measure is not effected by the by the individual indentity of the clustered elements i.e. a permutation on point labels should not effect the output of a clustering function.
- **Weak Local Consistency**: the quality measure does not change value if the distances between pairs of points within each cluster shrink , and distances between pairs of points in different clusters expands. This shrinkage and expansion need not occur uniformly.

- **Co-Final Richness**: requires that the quality of any clustering can be improved arbitrarily via consistent changes of the distance function.

In order for any clustering-quality measure to be considered good, it must satisfy all proposed axioms. Clustering-quality measures for various common cluster paradigms are propsed and analysed in the context of these axioms: *loss-based*, *center-based*, and *linkage-based*.

[2] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, "A framework for clustering evolving data streams," in *Proceedings of the 29th international conference on Very large data bases - Volume 29*, ser. VLDB '2003. VLDB Endowment, 2003, pp. 81–92.

[3] A. V. Aho and M. J. Corasick, "Efficient string matching: an aid to bibliographic search," *Commun.ACM*, vol. 18, no. 6, pp. 333–340, June 1975.

[4] K. Baker, "Singular value decomposition tutorial," " 2008.

This is tutorial covering vector and matrix methods in support of explaining the mechanics of *singular value decomposition* and its use in *information retrieval*. It describes vectors in terms of points in a three dimensional space and then extends the definition to an arbitrary number of dimensions. This n dimensional space is called textithyperspace or *n-space*. The hyperspace document representation is described as a method of representing a document as a vector whose components correspond in some way to the words occurring in the document. The tutorial details the primary vector operations: *addition*, *scalar multiplication*, and *inner product* (also called the *dot product* or *scalar product*). Vectors are *orthogonal* to one another if their inner product is zero. In two-dimensional space, this is equivalent to saying that the angle between them is 90 degrees. The implications of this to document retrieval is that the *cosine distance* between two such vectors

is zero. The *Gram-Schmidt* process for converting a set of vectors into *orthonormal vectors* (orthogonal unit vectors) is described.

Matrices are introduced as collections of vectors and the general notation thereof is covered in detail. The basic operations of multiplication and transposing matrices are described. A matrix is orthogonal if multiplying it by its transpose produces the identity matrix. Intuitively, if the vectors are all orthogonal unit vectors, the diagonal elements, produced by the inner product of each vector with itself, will be one, and all other entries will be zero. Calculation of the matrix *determinant* is described but its significance is not explained. Also, *eigenvalues* and *eigenvectors* are covered briefly and a sample calculation thereof shown. Intuitively, eigenvectors of a rotational matrix are those vectors which will be scaled but not rotated by the transform. The amount of scaling that occurs is the eigenvalue.

The paper concludes with a description of singular value decomposition (SVD) and gives some useful illustrations to convey how the process may interpreted. In the context of document retrieval, SVD breaks down the term-document matrix into linearly independent (orthogonal) components along which most variation occurs and reduces noisy correlations between documents (In [33], this process is interpreted as introducing noise, not removing it. I am inclined to go with the published paper [33]). If the components exhibiting the smallest variation in this new representation are eliminated, we end up with a reduced dimension approximation which best captures the original data. This feature reduction makes documents which share common substructure (topic) more similar and those which do not, more dissimilar. In practical terms, this means that documents about a particular topic become more similar in the new representation, even if the exact same words don't appear in all of them.

[5] M. W. Berry, S. T. Dumais, and G. W. O'Brien, "Using linear algebra for intelligent information retrieval," *SIAM Review*, vol. 37, no. 4, pp. 573–595, 1995.

[6] R. Cooley, B. Mobasher, and J. Srivastava, "Data preparation for mining world wide web browsing patterns," *KNOWLEDGE AND INFORMATION SYSTEMS*, vol. 1, pp. 5–32, 1999.

[7] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 391–407, 1990.

[8] M. M. Gaber, A. Zaslavsky, and S. Krishnaswamy, "Mining Data Streams: A Review," *SIGMOD Rec.*, vol. 34, no. 2, pp. 18–26, June 2005.

This paper reviews the theoretical foundations of data stream analysis. Mining of data streams uses techniques based on well established statistical and computational approaches. These techniques can be categorized into *data-based* and *task-based*.

Data-based techniques involve summarizing the whole data set or choosing a subset thereof to analyze:

- **Sampling**: an old statistical technique involving the probabilistic choice of whether to process a data item or not. Boundaries of the error rate of the computation are given as a function of time. In stream analysis the unknown data set size requires special analysis to derive this error bound function. Sampling does not address the problem of fluctuating data rates.
- **Load Shedding**: involves dropping a sequence of data streams. This makes the technique difficult to use with mining techniques as the dropped portions may contain patterns of interest in time series analysis. Load shedding has been used successfully in querying data streams but suffers from teh same problems as sampling.

- **Sketching**: random projection of a subset of features through vertical sampling of the input stream. Sketching has been applied in comparing different data streams and in aggregate queries but it is hard to use in the context of data stream mining; the major drawback is that of poor accuracy.
- **Synopsis Data Structures**: applying summarization techniques to creatre a synopsis of incoming data suitable for further analysis. Wavelet analysis, histograms, quantiles, and frequency moments have been proposed as synopsis data structures. Since not all aspects of the data set are retained during summarization, approximate answers are produced.
- **Aggregation**: computing statistical measures that characterize the data stream and using these measures with mining algorithms. Aggregation does not perform well with highly fluctuating data distributions.

Task-based techniques are methods which modify existing or invent new techniques to specifically address the computational challenges of data stream processing:

- **Approximation Algorithms**: designing algorithms for computationally hard problems which create approximate solutions with error bounds. Approximation algorithms have attracted reserachers as a direct solution to data stream mining problems however, the technique does not solve the problem of data rates with regard to resource availability.
- **Sliding Window**: detailed analysis is done over the most recent data window and summarized versions of old data.
- **Algorithm Output Granularity (AOG)**: the first resource-aware data analysis approach that can cope with fluctuating very high data rates. Data mining followed by adaptation to resources and data stream rates represents the first two stages of AOG. Lastly, generated knowledge

structures are merged when running out of memory. (This looks like a promising technique)

A number of algorithms have been propsed for mining of streaming data:

Clustering
Guha et al [27,28] have applied K-median in a divide and conquor fashion. Fixed-size data samples are individually clustered and the resulting cluster centers are clustered. This process is repeated to a fixed number of levels.

Babcock et al [7] have used exponential histogram (EH) data structures to improve (27) by addressing the problem of merging clusters when the two sets of cluster centers are far apart. Other k-median based techniques have been proposed which overcome the problem of increasing approximation factors with the increase in the number of cluster aggregation levels.

Domingos et al [15,16,35] proposed Very Fast Machine Learning (VFML), a general method for scaling up machine learning algorithms. The method depends on determining an upper bound on the the learning loss as a function of the number of data items to be examined in each step of the algorithm. VFML has been applied to k-means clustering (VFKM) and decision tree classification (VFDT).

Ordonez [46] proposed improvements to k-means to cluster binary data streams. Also, an incremental one-pass k-means variant has been developed and demonstrated to outperform scalable k-means in the majority of cases.

O'Callaghan et al [45] proposed STREAM and LOCALSEARCH algorithms for high quality data stream clustering.

Aggarwal et al [1] have proposed a data stream cluster-

ing framework called CluStream which divides the clustering process into an online component, which summarizes datastream statistics, and an offline component which clusters the summarized data.

Keogh et al [39] have proved empirically that most time series data clustering algorithms propsed so far produce meaningless results in subsequence clustering (what is this ?). They propose a solution using k-motif to choose subsequences that would provide meaningful results.

Gaber et al [21] have developed LIghtweight Clustering (LWC), an AOG based algorithm which is sensitive to resource availability.

Classification
Wang et al [53] propsed an agorithm to account for mining concept drifting data streams (what are this ?) usiing weighted classifier ensembles.

Ganti et al [18] developed analytically an algorithm for model maintenance under insertion and deletion of data records. The algorithm can be applied to any incremental data mining model (promising). Also, they have described a general framework for change detection between two data sets in terms of the data mining results they induce (does it apply to data streams ?). These two techniques are called GEM and FOCUS.

Papdimitiou et al [48] have proposed a single-pass incremental algorithm for pattern discovery from sensor data. The algorithm uses wavelet coefficients as compact information representation and correlation data structure detection, and then apply a linear regression model in the wavelet domain. (Ghorbani used wavelet analysis for anomaly detection in IDS. Related ?)

Aggarwal et al [3] have adapted the idea of micro-clusters

introduced in CluStream to use clustering results to classify data using the statisitical class distibution in each cluster.

Gaber et al [21] have developed LIghtweight Classification (LWClass), a variation of LWC.

Frequency Counting
Gianella et al [20] have developed a frequent item sets mining algorithm over data streams. They proposed the use of a tilted window to calculate the frequent patterns for the most recent transactions. Manku and Motwani [43] have propsed and implemented a frequency counting algorithm which uses group testing to find the most frequent items. The algorithm is used with the turnstile data stream model which allows additions and deletion of data items. Gaber et al [21] have developed Light Weight Frequency Counting (LWF), an AOG based algorithm.

Time Series Analysis
Indyk et al [36] have propsed approximate solutions with probabilistic error bounding to the probems of relaxed periods and queuing trends. The algorithms use dimasionality reduction sketching techniques and have been shown experimentally to be efficient in running time and accuracy.

Perlman and Java [49] have propsed an approach to mine astonomical time series streams. Sliding window patterns are clustered and an association rule technique used to create affinity analysis results among the created clusters.

Zhu and Sasha [54] have proposed techniques to compute statisitical measures using discrete Fourier Transforms.

Lin et al [42] proposed using symbolic representation to reduce dimensionality and numerosity.

Chen et al [12] proposed the application of multidimensional regression analysis to create compact cubes that can

be used for answering aggregate queries over the incoming streams.

Research Issues
Handling the continuous flow of data is not a capability native to most database management systems. Novel indexing, storage and querying techniques are required to handle this non-stop fluctuated flow of data. Processing of data streams has an unbounded memory requirement. This places limits on the machine learning techniques that can be applied since most methods require data to be resident in memory while the algorithm runs. It is importatnt to design space efficient techniques that can have only one look or less over an incoming stream.

How do we model the change of mining results over time ? Dynamics of data structures using changes in the knowledge structures generated would benefit many temporal-based analysis applications. Also, traditional mining algorithms do not produce results that show the change of the results over time. We need to develop algorithms for mining such changes.

Data stream preprocessing is a way of reducing the amount of memory required and the amount of effort to process a data stream. Light-weight preprocessing algorithms that can guarantee the quality of the mining results woud be of great benefit.

[9] A. Ghorbani and I. Onut, *Y-Means: An Autonomous Clustering Algorithm*, ser. Hybrid Artificial Intelligence Systems. Springer Berlin / Heidelberg, 2010, vol. 6076, pp. 1–13.

The paper describes a new clustering technique, *Y-Means*, based on the seminal *K-Means* algorithm. *Y-Means* addresses three main limitations of the *K-Means* method:

- **Dependence on choice of initial centroids**: *K-Means* is based on the mean squared error and converges to a local minima. In *Y-Means* the final result is independent of the choice of initial centroids. **WHY ?**
- **Dependence number of centroids**: finding the optimal number of centroid is NP-hard. *Y-Means* aims to find a semi-optimal approximation by exploiting statistical properties of the data. *Y-Means* starts with an arbitrary number of clusters and iteratively splits clusters based on an outlier detection function. Once cluster structure has stabilized, clusters may be merged based on a merging threshold.
- **Degeneracy**: there is no mechanism in **K-Means** to eliminate empty clusters at the end of the clustering process. *H-Means+* and *X-Means* are *K-Means* variants which attempt to deal with the degeneracy issue.

Y-Means begins by normalizing the data set to remove the dominating effect of large-scale features. Then, an arbitrary number of cluster centroids are randomly chosen and the algorithm enters an iterative loop until the cluster stabilizes. At the start of each iteration *K-Means* is executed and empty clusters are eliminated from the resulting structure. Then an outlier detection function is applied to each cluster and outliers are removed to form new cluster centres. This process repeats until the cluster structure stabilizes. Finally, similar clusters are merged based on a merging threshold and the resulting clusters are labelled. Labelling is domain dependent and only used when it applies to the data set. During experimentation, the authors used *size-based* and *distance-based* labelling which are specific to the intrusion detection domain.

Various outlier identification functions were used, based on Mahalanobis, Tukey, and Radusu Based metrics. The authors experimented with two popular statistical rules for

outlier threshold definition: the *Empirical Rule* (assumes a normal distribution) and the *Chebyshev's Inequality* (applies to any kind of distribution). Six different point-to-point distance metrics were used in *Y-Means* experiments: Euclidean, Manhattan, Minkowski of order 3, Chebyshev, Canberra, and Pearson's Coefficient of Correlation.

Merging of two clusters occurs if the distance between their centroids is not greater than a threshold. As with cluster splitting, this threshold is based on the statistical distribution of each cluster. The threshold is calculated as a weighted sum of the σ of two clusters being considered for merging. *Y-Means* uses the *linking* technique to merge clusters, creating multi-centroid clusters which better model the data as opposed to *fusing* which combines centroids to form a new one.

In experiments using the KDD Cup 1999 data set, *Y-Means* exhibited good performance compared with four well known unsupervised algorithms: EM, K-Means, SOM, and ICLN.

[10] I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection," *J.Mach.Learn.Res.*, vol. 3, pp. 1157–1182, March 2003.

Variable and ferature selection are meant to improve the accuracy and speed of predictors and to provide a better understanding of the underlying process that produced the data. The paper describes various techniques and methods used in the process of variable and feature selection:

Variable Ranking
Variable ranking is a *filter* method applied during preprocessing and independent of the choice of predictor. It is simple, scalable, and exhibits good empirical success. Statistically it is robust against overfitting because it introduces bias.

A scoring or correlation function is computed based on the features and used to sort variables. To use variable ranking to build predictors, nested subsets incorpororating progressively more variables of decreasing relevance are defined.

Using a correlation criteria like *Pearson's Coefficient*, one can only detect linear dependencies between variable and target (supervised). This restriction may be lifted by making a non-linear fit of the target with single variables and rank according to the goodness of fit. Overfitting can be avoided by using non-linear preprocessing and then using a simple correlation coefficient.

As opposed to using a correlation function for variable ranking, variables can be selected according to their individual predictive power, using as criterion the performance of a classifier built with that variable alone.

Several approaches to variable selection using information theoretic criteria have been propsed. Many rely on mutual information between each variable the the target (supervised).

The paper presents some informative examples that highlight the shortcoming of variable ranking techniques which evaluate variables predictive power individually:

- **Can presumably redundant variables help each other out?**: noise reduction and consequently better class separation may be obtained by adding variables that are presumably redundant.
- **How does variable correlation impact variable redundancy?**: perfectly correlated variables are truley redundant in the sense that no additional information is gained by adding them. Very high variable correlation (or anti-correlation) does not mean that said variables cannot

complement one another.

- **Can a variable that is useless by itself be useful with others?**: yes it can; also, two variables that are useless by themselves can be useful together.

Variable Subset Selection
Variable subset selection considers the predicitve power of groups of variables as opposed to individually. Such techniqies are divied into *wrappers*, *filters*, and *embedded methods*.

- **Filters**: fileters select subsets of variables as a preprocessor step, independently of the chosen predictor. Filters are faster than wrappers but not tuned to a specific learning machine. Filters can be used to reduce space dimensionality and overcome overfitting.
- **Wrappers**: use the learning machine of interest to score variable subsets according to their predictive power. An exhaustive search through the variable subset space is conceivable but NP-hard. A wide range of search stategies can be used to prevent the search becoming comutationally intractable. These methods are universal and simple.
- **Embedded Methods**: perform variable selection in the process of training and usually specific to the learning machine. They make better use of available data and reach a solution faster by avoiding retraining the predictor from scratch for every variable subset investigated.

Feature Construction and Space Dimensionality Reduction
Feature construction is concerned with improving predictor performance and building more compact feature subsets. Two distinct goals may be pursued for feature construction: achieving best reconstruction of the data (unsupervised) or being most efficient for making predictions (supervised). Feature construction is an opportunity to incorporate domain knowledge into the model and and can be very application specific, however there are a number of generic techniques,

some of which are described:

- **Clustering**: a group of similar variables by a cluster centroid, which becomes a feature. Some supervised may be introduced to obtainmore discriminant features. Clustering is commonly used for feature selection in text processing. Here the supervision comes from a priori document categories.
- **Matrix Factorization**: *singular value decomposition* (SVD) forms sets of features that are linear combinations of the original variables and which provide the best possible reconstruction thereof in the least square sense. The method is unsupervised.
- **Supervised Feature Selection**: the paper reviews three approaches for selecting features in cases where features should be distinguished from variables because both appear simultaneously in the system (**WHAT DOES THIS MEAN ?**)

Validation Methods It is important to distinguish between the problem of model selection and final evaluation of the predictor. For predictor evaluation an independent test should be kept aside. For model selection, the remaining data should be further split between fixed training and validation , or cross validation can be used. Statisitcal tests can be used to estimate the significance of differences in validation errors.

Unsupervised Variable Selection In unsupervised variable selection, there are a number of variable ranking criterion, a number of which are useful across applications, including: *saliency*, *entropy*, *smoothness*, *density*, and *reliability*.

Forward vs Backward Selection Forward selection (add variables) is computationally more efficient than backward selection (eliminate variables). However, it is argued that forward finds weaker subsets because the importance of variables is not assessed within the context of other variables

not yet included.

[11] E. R. C. Ide, "Relevance feedback in an automatic document retrieval system," " 1969.

[12] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Comput.Surv.*, vol. 31, no. 3, pp. 264–323, September 1999.

Paper is an overview of data clustering concepts and techniques. Clustering is an exploratory undertaking (unsupervised), as opposed to classification (supervised). During clustering, a collection of patterns are organised based on a notion of their similarity to one another. Patterns are typically represented as feature vectors and their organisation occurs within this feature space. Pattern selection involves feature selection as well as feature extraction, transforming input features to produce new salient features. All clustering algorithms will produce clusters regardless of the underlying data so how do we evaluate a cluster algorithm ? Cluster validation studies can be *external*, comparing the recovered structure to an *a priori* structure; *internal*, which examines whether the structure is intrinsically appropriate for the data; or *relative*, which compares two structures and measures their relative merit.

A measure of the similarity between two patterns is essential to most clustering procedures. The most common measure is the Euclidean distance. It works well when a data set has compact, isolated clusters but large scale features tend to dominate unless weighted or normalized. Salient cluster algorithm properties include: agglomerative vs divisive; monothetic vs polythetic; hard vs fuzzy; deterministic vs stochastic; incremental vs non-incremental; and hierarchical vs partitioning.

Hierarchical algorithms produce a *dendrogram* representing nested groupings of patterns and the similarity thresholds

at which they change. Most hierarchical cluster algorithms are variants of the single-link, *single-link* (distance between clusters is the minimum between any two patterns drawn from different clusters); *complete-link* (distance between clusters is the maximum between any two patterns from different clusters); and *minimum-variance* algorithms.

Partitioning algorithms are less demanding computationally compared to hierarchical algorithms. A problem of these algorithms is the choice of the number of desired output clusters. The most common and intuitive criterion function used in partitional clustering is the *squared-error* criterion of which *K-Means* is the simplest and most commonly used. *K-Means* is one of the most efficient in terms of execution time and one of the few methods appropriate for use on large data sets. *K-Means* requires one to specify the number of clusters to create which is difficult to do optimally. Variants of *K-Means* have been proposed which dynamically merge and/or spilt clusters based on a variance threshold.

Clusters are typically represented by their centroid, a simple scheme if clusters are compact and iso-tropic. If clusters are elongated or non-isotropic, then this representation weak, better replaced by a collection of points.

Search based cluster techniques can be either deterministic or stochastic. Deterministic techniques guarantee an optimal partition by performing exhaustive enumeration. Stochastic search techniques generate near optimal partitions reasonably quickly and guarantee asymptotic convergence to optimal partition.

Clustering is subjective by nature. Subjectivity is usually incorporated into some phase of clustering, whether it be in selection of a pattern representation, choosing a similarity measure, or cluster representation. The incorporation of domain knowledge consists of ad-hoc approaches with little in common.

Clustering of large data sets is computationally demanding and many clustering algorithms do not scale adequately. The emerging discipline of data mining has spurred developments and optimizations in this area. Clustering is used in the data mining process for segmentation of databases into homogeneous groups, predictive modelling, and visualization. If the data set is too large to fit in main memory, techniques like *divide-and-conquer*, incremental clustering, and parallel algorithm implementations have been used.

The paper review several application domains in which clustering has been successfully employed: image segmentation, object and character recognition, information retrieval, and data mining.

[13] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 6/1 2010.

Although *K-Means* was devised in 1955, it is still widely used because of its simplicity, efficiency, and empirical success. The paper looks at the difficulties of developing better algorithms. Clustering algorithms can be broadly divided into *hierarchical* and *partitional*. Hierarchical algorithms recursively find nested clusters in either *agglomerative* (bottom up) or *divisive* (top down) mode, taking an $n * n$ similarity matrix as input. Partitional algorithms take as input an $n * d$ pattern matrix or a similarity matrix.

The *K-Means* algorithm finds a partition such that the squared error between the empirical mean of a cluster and the points in the cluster is minimized. The goal is to minimize the squared error over all clusters however this problem is NP-hard so, out of necessity, *K-Means* is a greedy algorithm which converges to a local minima. Research does show however that if the clusters are well separated, the algorithm

will converge with high probability to the global optimum. The main steps of *K-Means* are

1. Select an initial partition and repeat steps 2 and 3 until cluster membership stabilizes.
2. Generate a new partition by assigning each pattern to its nearest cluster centre.
3. Compute new cluster centres.

K-Means requires three user-specified parameters: number of clusters, cluster initialization, and distance metric. Selection of number of clusters is difficult and usually based on heuristics and/or repeated execution with different number of clusters, adjudicated by a domain expert. *K-Means* typically uses the Euclidean distance metric and as a result, finds hyperspherical shaped clusters.

Clustering algorithms have been developed that model pattern density by a probabilistic mixture model viz. EM algorithm and several Bayesian approaches. These methods are attractive because of their ability to deal with arbitrary shaped clusters but have difficulty dealing with high dimensional data the feature space is characteristically sparse, making it difficult to distinguish high density regions from low. Graph theoretic clustering is another class of clustering algorithms. These algorithms represents data points as nodes in a graph with connecting edges weighted by their pair-wise similarity. The central idea is to partition the nodes into two groups such that the weights of the edges between the two groups is minimized.

All the variables involved in a clustering project make it inherently difficult. One of the most important decisions is that of data representation. A good data representation will result in compact, well separated clusters however there is no universally good representation and the process must be guided by domain knowledge. Another variable is the number

of clusters. Automatic determination of this variable has been one of the most difficult problems in clustering. Alternatively, the optimal number of clusters must be determined through trial and error.

Since clustering algorithms tend to find clusters irrespective of whether they exist, it is important to objectively evaluate whether the data has a natural tendency to cluster. *Cluster validation* is the formal evaluation of clustering results in a quantitative and objective manner. Cluster validity measures can be *internal*, *external*, or *relative*. Internal measures asses the fit between the structure imposed by the algorithm and the data itself. Relative measures compare the structure imposed by different algorithms on the same data. External measures compare cluster structure to some a priori information, namely "true" class labels.

Stability of a clustering solution is a measure of how much variation occurs in the structure imposed over different sub-samples drawn from the input data. Different measures of variation can be used to obtain different stability measures. Since many algorithms are asymptotically stable, it may be important to consider the rate at which stability is reached.

Some recent clustering trends include:

- **Clustering Ensembles**: combine the resulting partitions resulting from application of differing clustering methods on the same data.
- **Semi-Supervised Clustering**: a subset of the data is labelled and these are used to impose pairwise constraints (*must-link* and *cannot-link*) on the cluster algorithm.
- **Large-Scale Clustering**: algorithms developed to handle large data sets can be classified as: efficient nearest neighbour (NN), data summarization, distributed computing, incremental clustering, or sampling-based methods.

[14] N. Jardine and C. J. van Rijnsbergen, "The use of hierarchical clustering in information retrieval," *Inform. Stor. & Retr*, no. 7, pp. 217–240, 1971.

[15] J. Kleinberg, "An impossibility theorem for clustering," " 2002.

[16] T. G. Kolda and D. P. O'Leary, "A semidiscrete matrix decomposition for latent semantic indexing information retrieval," *ACM Trans.Inf.Syst.*, vol. 16, no. 4, pp. 322–346, October 1998.

The paper proposes a *semi-discrete decomposition* (SDD) to replace *singular value decomposition* (SVD) as used in *latent semantic indexing* (LSI) for approximating a document matrix. LSI has been shown to overcome many difficulties associated with literal term matching and the method proposed in the paper claims to exceed the performance thereof in certain aspects.

As background, the paper describes the *vector space method* for document retrieval, of which LSI (both SVD- and SDD-based methods) is an extension. The creation of the *term-document matrix* is described and extensive coverage is given to the different methods of weighting terms in the document matrix. A term weight has three components:

- **local weight**: based on the occurrence of the term within a specific document.
- **global weight**: based on the the occurrence of the term within the document collection as a whole.
- **normalization**: specifies whether or not the columns (documents) are normalized.

When executing queries, weightings are also applied to the *pseudo document* vector, though these need not be the same as the term weightings. Normalizing the query vector has no effect on document ranking so it is never done.

The paper gives a good illustration as to how queries based on both SVD and SDD approximations of the term-document matrix can actually improve document retrieval.

The decomposition introduces 'noise' as the rank of the original matrix is reduced. This noise makes similar documents appear more similar while remaining close to the original data. The amount of noise introduced depends on the extent to which the rank of the original matrix is compressed.

The SDD method proposed was originally introduced by [**?**] and is used here in a bid to save storage space and query time. The method does not reproduce the original matrix exactly, even with no rank compression, but for equal reduction of rank, the SVD method requires nearly 32 times more storage than SDD. SDD is calculated iteratively, terminating when the improvement in a residual stagnates. Queries using the SDD decomposition use fewer floating point operations than queries on an SVD decomposition.

An empirical comparison of the vector space method, SVD-based LSI, and SDD-based LSI, was conducted over three different corpora. The results indicate that SDD-based LSI retrieves documents as well as SVD-based LSI but requires only about half the query time. The disadvantage of the SDD method is that it takes five times as long as computing the SVD approximation.

The paper proposes some techniques for updating the SDD approximation when the document collection changes as well as methods to improve the decomposition if it is found to be inadequate (in terms of retrieval accuracy). The SDD update methods are easier than updating SVD. SVD update methods can take as long as the original decomposition, but require less memory.

The corpora used in the evaluation are relatively small (¡2000 documents) and it remains to be seen how SDD-based LSI performs on large collections. Given the large initial approximation computation time, SDD may not scale adequately to be of practical use in such situations.

[17] J. Lei and A. Ghorbani, "Improved competitive learning neural networks for network intrusion and fraud detection."

The authors have developed two new clustering algorithms, the *Improved Competitive Learning Network* (ICLN) and the *Supervised Improved Competitive Learning Network* (SICLN), specifically for use in the intrusion and fraud detection domains. Data mining-based intrusion and fraud detection is categorized into *misuse detection* and *anomaly detection*. Misuse detection is a classification exercise based on the supervised learning from labelled data. Anomaly detection establishes patterns of normal behaviour and thereby identifies deviations. Both algorithms are derived from the *Standard Competitive Learning Network* (SCLN).

Standard Competitive Learning Network
SCLN is a two layer neural network: distance measure layer and competitive layer. During training, the distance measure layer calculates the distance between the weight vectors and the training example. Of these distances, the competitive layer finds the shortest and the winning weight vector is adjusted (rewarded) towards the training example. Eventually each of the weight vectors converges towards the centroid of one cluster. Clustering centers are the output of the network. SCLN performance depends heavily on the number of initial neurons and the initialization of their weight vectors.

Improved Competitive Learning Network
ICLN changes the SCLN's reward only rule to include a punishment for losing weight vectors. These vectors are moved away from the training example based on a kernel function and learning rate. This change accelerates the learning process without additional iterations.

ICLN initializes weight vectors to random training examples or all to the mean of the training data. ICLN can

exclude redundant neurons via the punishment rule, but not add to, the number of clusters so the number of initial clusters is usually set higher than expected.

During training, the ICLN iterates until the maximum update to a weight vector falls below a minimum update threshold or until a preset number of iterations.

Supervised Improved Competitive Learning Network

SICLN modifies the ICLN learning rule to train on both labelled and unlabelled data. It uses an objective function to measure the quality of the clustering result w.r.t the produced cluster centers and the data set. The purpose of the objective function is to optimize the purity and number of the clusters.

SICLN is initialized in the same way as ICLN but before learning begins, the initial weight vectors of the network neurons are labelled with their member data points. A weight vector is labelled to a class if this class is the biggest population of the members of this neuron of this weight vector. In the case where only a portion of the data are labelled, neurons may be labelled as "unknown" if no other members of the same neuron are labelled.

During learning, SICLN uses labelled data, if available, to update cluster centers. For labelled traiing examples, only output neurons that are the same class as the training example or "unknown" class can compete. For unlabelled training examples, SICLN functions updates as per ICLN.

After the learning step, SICLN constructs a new network. A neuron will be split in two if it contains many members of other classes. Neurons are merged if they belong to the same class. The training step is repeated in this new network. Training stops when then the objective function or the number of iterations reaches satisfies a certain threshold.

Results
ICLN and SICLN were compared with k-means and SOM over three different data sets. ICLN exhibited similar accuracy to the traditional algorithms. SICLN outperformed all other algorithms over all three data sets (**DOES SICLN QUALIFY AS SEMI-SUPERVISED ?**

[18] T. A. Letsche and M. W. Berry, "Large-scale information retrieval with latent semantic indexing," *Information Sciences*, vol. 100, no. 1-4, pp. 105–137, 8 1997.

The paper describes the implementation of a *latent semantic indexing* (LSI) search function library called LSI++. The library is implemented in C++ and designed with a modular API to facilitate easy integration into other applications. A test application incorporating LSI++ and exposing a WWW interface was built by the authors and is used to benchmark the implementation against a previous implementation at Bellcore [7]. In serial mode, the new system was found to be six times faster than the Bellcore system. These improvements can be accounted for by more efficient programming. The authors also implemented a parallel version of the LSI search function and managed to achieve nearly 180 times improvement over the previous implementation over some document collections.

During the search phase of LSI, each vector of the term-document space must be loaded into memory and compared to the query *pseudo-document* vector. Because of finite memory capacity, it is not always possible to load the entire collection of document vectors into primary storage and vectors must be loaded and unloaded to and from main memory as the search progresses. It is this transfer to and from primary storage that the authors direct their optimizations. The collection of document vectors are partitioned between a number of workstations, each loading the entire partition

into main memory. A root node receives the search request, creates the query pseudo document and broadcasts it to the other members. Each workstation then computes, in parallel, the similarity measure between the pseudo document and each local vector and returns the results to the root node which performs a global sort and returns a ranked list to the user. An interesting result of benchmark testing is that adding more processors to the smaller document collections generally did not have a great effect but larger collections benefited significantly from increased parallelism. This is probably because smaller collections are able to fit a larger portion of the document representation collection in primary storage and do not suffer the paging effect to the same extent as the larger collections. Also, the overhead imposed by distributing and reconstituting the search across the workstations is relatively insignificant for large collections but not so for small collections.

Although LSI is capable of achieving significant retrieval performance gains over standard lexical techniques, it's execution efficiency lags behind these simpler methods and can become prohibitively slow on very large data sets. However, most of the processing effort can be attributed to the pre-processing phase which the the paper does not address. The system described in the paper assumes that this pre-processing is a one-time cost and that software to perform this step already exists. This point of view holds for static evaluation corpora, however, by the authors own admission, a functional real-world system would require some method of updating and downdating the document set without having to pre-process the entire document collection from scratch. Also, speeding up the search process through concurrent execution obviously yields improvements however, this method can be applied to any retrieval algorithm which performs an exhaustive search on a document collection, decomposed or not.

[19] T. Li, Q. Li, S. Zhu, and M. Ogihara, "A survey on wavelet applications in data mining," *SIGKDD Explor.Newsl.*, vol. 4, no. 2, pp. 49–68, December 2002.

[20] B. Liu, "Sentiment Analysis: A Multifaceted Approach," *IEEE Intell.Syst.IEEE Intelligent Systems*, vol. 25, no. 3, pp. 74–76, 2010.

> The World Wide Web has made large numbers of opinionated texts available, driving the study of *sentiment analysis*, a field which combines problems from many different sub-fields and in which significant progress has been made over the last few years.
>
> Originally, research treated the problem of sentiment analysis as one of text classification: *Sentiment Classification* classifies a document as expressing a positive or negative opinion; *Subjectivity Classification* aims to determines whether a sentence is subjective or objective. This treatment has since been expanded to encompass more detailed analysis required for many real-world applications. In particular, users are interested in determining the subject of an opinion.
>
> In defining the sentiment analysis or opinion mining problem, we make the following definitions:
>
> - **opinion target**: the target entity or object about which an opinion is being expressed. An opinion target can have a set of components, and/or atributes about which an opinion can be expressed, in addition to the object itself.
> - **opinion holder**: the entity expressing the opinion.
> - **opinion**: a positive or negative appraisal of an opinion target. Positives and negatives are called the orientation of the opinion. Opinions may be direct or comparative.
>
> These aspects combine to form the *feature based sentiment analysis model*. In this model, opinionated documents are analysed to extract the following information.

- **opinion quintuples**: capture orientation of an opinion expressed regarding a particular object feature by a particular opinion holder at a specific time.
- **synonyms** of each object feature.

For opinion extraction, existing approaches are based on different supervised and unsupervised methods using opinion words and phases and grammar information. This task is difficult because of the scope of how opinions may be expressed across different domains and between different opinion holders.

Correlating the attributes of the opinion quintuples requires a high level of integration. *Natural language processing* (NLP) techniques have been applied to this task but even within this well defined field there are many aspects to which accurate solutions have not been discovered viz. coreference resolution and wordsense disambiguation.

In evaluating semantic analysis systems, *precision* and *recall* are common measures. In most applications high precision is critical but high recall may not be necessary as long as the system can extract enough opinions to ensure a statistical balance of errors and not destroy the natural distribution of sentiment.

In practice, completely automated solutions are not imminent however it is possible to devise effective semi-automated systems.

[21] M. Makki and A. Ghorbani, "Ensemble of word clusters as the feature space for document clustering," " 2009.

[22] T. M. Mitchell, *Machine Learning.* New York: McGraw-Hill, 1997.

[23] G. W. O'Brien, Master's thesis, Department of Computer Science, University of Tennessee, Knoxville, TN, 1994.

[24] C. H. Papadimitriou, H. Tamaki, P. Raghavan, and S. Vempala, "Latent semantic indexing: a probabilistic analysis," in *Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, ser. PODS '98. New York, NY, USA: ACM, 1998, pp. 159–168.

The paper makes a formal analysis of *latent-semantic indexing* (LSI) in an attempt to understand the apparent strengths of this information retrieval technique. It has been previously held that LSI captures the underlying semantics of a corpus and is thereby able to outperform more conventional vector-based methods with reduced storage requirements and reduced query times. These claims have thus far only been supported by empirical evidence are proved formally, under certian consitions, in the paper. Also, a *random projection* technique is proposed to speed up LSI, which involves considerable pre-processing.

Capturing the underlying (latent) semantics of documents and queries during information retrieval addresses the problems of *synonymy* and *polysemy*. LSI attempts to capture this hidden structure using linear algebra techniques. The *term-document* matrix is reduced by *single value decomposition* (SVD). A portion of the largest singular values are retained, forming a reduced matrix which allows faster retrienval while still adequately capturing the structure of the corpus. Therefore, LSI preserves (to the extent possible) the relative distances (and hence the retrieval capabilities) of the term-document matix while projecting it onto a lower dimensional space.

During its formal analysis, the paper concludes that the LSI transformation of the term-document matrix aligns the vectors of documents on the same topic and orthogonalizes vectors of documents on different topics. This notion is confirmed during empirical evaluations which measure the angle between intra-topic and inter-topic documents. NOTE: from a clustring perspective, would this not create tighter,

better separated structures ? Also, the paper descibes how synonymous terms would have very similar representations in the term-document matrix and would be "projected out" by the LSI transform, as would be expected from a method that claims to capture the semnatics of the corpus. The paper proposes a method for combininig *random projection* with LSI. LSI is computationally demanding and, in that respect, would benefit from the reduction in dimension of the document space provided by random projection.

[25] M. Rosell, "Introduction to Information Retrieval and Text Clustering," " 2006.

This is a collection of chapters adopted from the authors licentiate theses *Clustering in Swedish*. The first chapter introduces the field of Information Retrieval (IR), as a large and growing field within Natural Language Processing (NLP). IR is the theoretical foundation of text search engines. Texts are represented as vectors, each dimension corresponding to a distinct word in the set of words appearing in all texts. The vector fields are weights which model how important the corresponding word is deemed to be in the context of the text. There are many weighting schemes but in the most common the weights are the product the *term-frequency* (tf) and *inverse document frequency* (idf). The term frequency is a function of the number of occurrences of a particular word in a document divided by the number of words in the entire document. The inverse document frequency models the distinguishing power of the word in the text set; the fewer documents that contain the word, the more information about the text int he text set it gives.

In a text query, a search is conducted for texts similar to the search vector which is represented in the same way as the texts. The most common measure of similarity is the *cosine measure*, the cosine of the angle between the query

and texts. The texts are returned are ranked by similarity. It is difficult to evaluate search results. In a controlled text set, query results can be compared against results of human opinion. By comparing these perspectives, we may define performance measures for the search engine. *Precision* and *recall* are common measures. To further characterize search engine performance over a range of operating conditions, the precision at different levels of recall can be plotted in a graph.

Modifications can be made to the vector space model described to improve search performance:

- **Stoplist and Word Classes**: stoplist words are excluded from the model, usually very common words whose occurrence do not separate one text significantly from another.
- **Phases**: treat phrases as separate dimensions for phrase based searches.
- **Lemmatizing and Stemming**: extracting word fragments that appear frequently in documents.
- **Related Words**: the vector model does not account for the fact that words may be related (synonyms, homonyms etc). Many attempts have been made to attempt to address this phenomenon viz. word sense disambiguation, query expansion.
- **Statistically Related Words**: statistical examination of the word-by-document matrix gives information regarding words that appear together often. This information can be used by search engines to improve performance. *Latent Semantic Analysis* (LSA) is such a technique but is computationally heavy. *Random Indexing* (RI) is a much faster, less memory intensive alternative but does not use the entire word-by-document matrix.
- **Meta-data**: meta-data found in web pages provides additional information that can be used when indexing.

Text clustering can be used to discover structures within a text set that were not previously known. This is as opposed to text categorization where texts are assigned to predefined categories. IR and text clustering are related in that they both employ the same pattern representation and search function. Researchers believe that credible text clustering could make search times shorter by retrieving clusters of texts instead of individual documents. Similar (clustered) documents are probably relevant to the same queries but that does not mean that pre-clustering of the entire text set can take all future queries into account (I think this means that clustering is coarse grained relative to search queries). The authors argue for text clustering after ordinary search engine retrieval and have shown through experimentation that this can improve search result quality.

It is hard to objectively evaluate clustering results since the value thereof is subjective. It is common to distinguish between intrinsic and external measures. Intrinsic measure use no external knowledge other than what was available to the cluster algorithm. External measures use external knowledge.

[26] S. Sun and Y. Wang, "K-nearest neighbor clustering algorithm based on kernel methods," in *Intelligent Systems (GCIS), 2010 Second WRI Global Congress on*, vol. 3, 2010, pp. 335–338.

[27] W. tau Yih, J. Goodman, L. Vanderwende, and H. Suzuki, "Multi-document summarization by maximizing informative content-words," in *Proceedings of the 20th international joint conference on Artifical intelligence*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc, 2007, pp. 1776–1782.

The paper describes a new multi-document summarization technique which enhances the simple yet very effective *SumBasic* system. *SumBasic* computes the probability of content words within the document set and scores individual

sentences based on the average probabilities of the constituent words. The summary is generated by a greedy algorithm that iteratively selects the sentences with the highest scoring content word, breaking ties using the average sentence scores. This continues until the maximum summary length is reached.

The proposed system differs from the *SumBasic* implementation in the following ways:

- **Scoring:** enhanced sentence scoring method using a discriminative machine learning algorithm to combine term position information with term frequency. Seven features are defined using frequency and position information and the model is trained using human generated summaries to predict the probability of a term appearing in a summary.
- **Sentence Selection:** a more complex algorithm than that of *SumBasic* is used to select sentences fof the summary. This algorithm, based on a stack decoder, searches explicitly for the combination of sentences which will maximize the sum of term scores for a specific sentence.
- **Sentence Simplification:** sentences are simplified to remove phrases with little expected value. The simplified sentences compete for selection by the summarizer with the original sentences.

The technique proposed in the paper gives the best results reported on the DUC-2004 and MSE-05 summarization tasks for the ROUGE-1 score, although only the for the DUC-2004 task is the difference statistically significant. The algorithm is considered significantly simpler than the previous best system. Sentence simplification seems to have a small positive effect for certain but not all evaluation scores.

[28] E. M. Voorhees, "The cluster hypothesis revisited," in *Proceedings of the 8th annual international ACM SIGIR conference on Research and*

development in information retrieval, ser. SIGIR '85. New York, NY, USA: ACM, 1985, pp. 188–196.

The paper revisits the notion proposed by [14] that not only can clustering be used to reduce the number of documents which have to be compared to the query, but that the clusters themselves embody useful information which can be exploited to improve the effectiveness as well as the efficiency of a retrieval search. Specifically, [14] stated the *cluster hypothesis*: documents that are similar to one another (by some clustering distance function) are relevant to the same queries, and proposed *cluster-based retrieval* to exploit this relationship. Cluster-based retrieval retrieves one or more clusters in their entirety in response to a query, as opposed to most other cluster methods which identify clusters which are are likely to contain good documents and then compute the similarity between the query and each of the documents in the identified clusters.

Research involving relevance feedback by [11], brings the cluster hypothesis into question and thereby the effectiveness of cluster-based retrieval. This paper proposes a new test as to whether the cluster hypothesis holds for a given document collection and conducts cluster-based and non-cluster-based searches on these document collections to empirically examine the validity of the hypothesis.

The original cluster hypothesis test plots and compares frequency distributions of pairwise similarity between documents relevant to a particular query vs between documents which are not relevant to the same query. The paper regards this test as not sufficiently granular and proposes a new test based on relevance of a fixed number of nearest neighbours.

By comparing the assesment of the new cluster hypothesis test to the relative performance of hierarchical cluster-based searches with sequential searches, the paper concludes the the relative performance seems to be independent of how

well the cluster hypothesis characterizes the collection. Also, the paper concludes that cluster searches that retrieve individual documents almost always performed better than a cluster-based search (returns entire cluster).

I think, given the conclusions above, the paper's claims that its cluster-hypothesis test is an improvement over [15] are weak. The results of the research brings into question whether the cluster hypothesis holds universally at all or whether it is largely dependent on the choice of our notion of similarity (document compared to document) vs relevance (document compared to query).

[29] X. Wan and J. Yang, "Single document summarization with document expansion," in *Proceedings of the 22nd national conference on Artificial intelligence - Volume 1*. AAAI Press, 2007, pp. 931–936.

The paper proposes a document summarization technique that makes use of document expansion to improve single document summarization. The technique focuses on generic, as opposed to query-specific, summarization and is based on the assumption that topic-related documents can provided an enlarged context within which a better summary may be extracted from a specific document.

The proposed approach begins by expanding the target document into a small document set of the k nearest neighbors of the target. In this case, the similarity measure used in the the standard *cosine Measure*.

Given the sentence collection of the document set, the *affinity* weight between pairs of sentences is calculated using the cosine measure. In this calculation, sentences are represented as weighted vectors where the weights are the product of term frequency within the sentence and inverse sentence frequency. Then an *affinity graph* is constructed in

which sentences are represented as nodes, and edges reflect a non-zero affinity weight between sentences. Edges are weighted based on affinity weight and the *cosine distance* between the sentences. For comparison sake, two additional graphs were constructed at this point, containing only inter-document links and intra-document links respectively.

Based on the *affinity graphs*, the *informativeness* score of each sentence is calculated iteratively, based on the scores of those sentences linked to it. A greedy ranking algorithm is then applied which penalizes sentences based on redundancy. The highest ranked sentences are supposedly highly informative and novel and are included in the final document summary.

Empirical evaluation of this method on the DUC2002 corpus, indicates that document expansion does benefit single document summarization. Also, using inter-document links alone produces better results than using inter-link documents to construct summaries. The size of the expanded document set was found to be optimal at less than ten documents, an encouraging result given the added computational complexity over the single document method. The paper suggests using clustering to derive document sets, as a possible optimization.

These results lend themselves to the notion that inter-document relations reflect latent document semantics.

[30] Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorization," in *Proceedings of the Fourteenth International Conference on Machine Learning*, ser. ICML '97. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc, 1997, pp. 412–420.

The paper is a comparitive study of five different feature selection techniques as applied to the problem of text categorization: *document frequency* (DF), *information gain* (IG),

mutual information (MI), χ^2-test (CHI), and *term strength* (TS). All of these techniques use a term-goodness criterion threshold to achieve the desired degree of term elimination from the full vocabulary of a document corpus. These methods all fall into the wrapper class of feature selection techniques, as opposed to filters or embedded methods. To evaluate the effectiveness of the feature selection in each case, two well known, highly scalable, classification algorithms are used: *k-nearest neighbour* (KNN) and a regression method named *Linear Least Squares Fit* (LLSF). The choice of classifiers is based on the fact that they differ statistically and therefore should reduce classifier bias in the results. Feature sets were evaluated by measuring precision and recall over the Reuter-22173 collection and OHSUMED.

- **Document Frequency Thresholding:** document frequency of a term is the number of documents in which a term appears. The DF for each unique term in the training corpus was calculated and those whose DF was below a certain threshold were eliminated from the feature space. The assumption here is that rare terms are either non-informative for classification, or not influential in global performance. Improvement in accuracy is also possible if rare terms happen to be noise terms. DF is considered an ad-hoc approach but scales well by virtue of its simplicity and performs relatively well in this case. I think that simple techniques, like this one especially, that are purely lexical do not have much scope for improvement. Although semantic analysis is complicated and not well developed, it offers much more scope for improving text processing in general.
- **Information Gain:** measures the amount of information gained for classification with knowledge of the presence or absence of a term in a document. Feature terms whose information gain falls below a certain threshold, are removed from the feature space.

- **Mutual Information:** criterion commonly used in statistical language modelling of word associations. It is based on joint and marginal probablities of terms and categories in the training copus. It is strongly influenced by marginal term probabilities, favouring rare terms.
- χ^2 **Statistic:** measures the lack of independence between term and document co-occurence. For each category, CHI is calculated between each unique term in the training corpus and that category, and then the category specific scores are combined into two scores for each term. Like MI, CHI has a strong statistical basis however CHI is a normalized value and can be compared across terms for the same category. CHI is known not to be reliable for low-frequency terms.
- **Term Strength:** estimates term relevance based on how likely term is to appear in closely related (by *cosine rule* thresholding) documents. This method is radically different from the others. It is based on document clustering, assuming that documents with many shared words are similar.

Results DF, IG and CHI perform well and are strongly correlated. DF's basis on common terms would indicate that, contrary to popuar belief, common terms are often informative (perhaps this is particular to text classification). Given that removing stop words seems generally to improve classification performance, I think there must be some kind of occurence frequency threshold above which terms no longer give usefull information for classification puposes. MI exhibited inferior performance compared to other methods due to its bias for rare terms and/or a strong sensitivity to probability estimation errors.
Interestingly, MI is *task- sensitive* (uses category information) but underperforms TS and DF which are *task-free*. It would seem that using category information for feature selection is not crucial for good performance.

[31] R. Zafarani and A. Ghorbani, "Dynamic clustering of large scale data using random sampling."

[32] ——, "Oracle clustering: Dynamic partitioning based on random observations."

[33] H. Zha and H. D. Simon, "On updating problems in latent semantic indexing," *SIAM Journal on Scientific Computing*, vol. 21, no. 2, pp. 782–791, 1999.

The paper describes a new *singular value decomposition* (SVD) updating algorithm to support the use of *latent semantic indexing* (LSI) on evolving document collections. LSI is a type of vector space information retrieval method that uses a *reduced dimension representation* (RDR) computed by the SVD of the *term-document* matrix. For each of the dimensions reduced, a *pseudoconcept* is introduced which may not have any explicit semantic content but which helps to discriminate documents.

Three types of updating problems exist in LSI:

- **Updating documents**: documents are added to the collection.
- **Updating terms**: new terms are added to the model.
- **Term weight corrections**: term weights in the original term-document matrix need to be adjusted.

A method for updating the LSI-generated RDR, called *folding*, is described in [7]. However, this method is based on the original RDR and does not adjust the representation of existing terms and documents, causing retrieval accuracy to suffer. This paper focuses on the three SVD updating methods derived in [5, 23] and proposes enhancements thereto. The enhanced methods are proven formally to be more accuracy approximations of the original term-document matrix, albeit at additional computation cost. The paper also makes a formal justification for using the RDR as the

basis for updating, as opposed to the original term-document matrix. The proof shows that the information lost in the RDR would also be lost in decomposing an updated term-document matrix. This result has important practical consequences as, in many cases, the original term-document matrix is not maintained on disk or in memory.

The new update methods are tested on three document collections and compared against the methods from [5, 23] as well as an update method based on the entire term-document matrix. In the tests, the document collection is broken into partitions and the collection updated incrementally and the accuracy of queries against the updated RDRs are compared. For one document collection, the new method shows significant improvement over [5, 23] and marginally exceeds the performance of the update method based on the entire term-document matrix. For the other document collections, the new method shows marginal improvement over [5, 23] and is comparable to the update method based on the entire term-document matrix. In all cases, the computation cost for the new method is higher than that of [5, 23].

As is the case with many LSI-based studies, the test collections are small and do not qualify the results for large-scale practical use. However the formal proofs in this paper provide some confidence in the basis of the decompositions used and how accurately we can expect them to approximate the original corpus.

[34] G. P. Zhang, "Neural networks for classification: a survey," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 30, no. 4, pp. 451–462, 2000.

This paper is a review of the use of Artificial Neural Networks (ANN) for classification tasks. It compares ANNs to statistically based classification procedures and explains advantages

and disadvantages of ANN relative to these more traditional approaches. The paper shows how ANNs are able to estimate posterior classification probabilities by virtue of the fact that ANNs are typically trained by attempting to minimize mean squared errors. This provides a direct link between ANN classifications and statistical methods, particularly Bayesian.

Direct comparison of ANN and statistical classifiers may not be possible because ANNs are non-linear and model-free, while statistical methods are linear and model-based. However, by appropriately encoding ANN outputs, we can use ANNs to directly model some high order discriminant functions. Analysis along these lines has shown that the hidden layers of an MLP project the data onto different clusters in a way that these clusters can be further aggregated into different classes. However, the added flexibility of ANNs due to hidden layers does not automatically guarantee their superiority over logistical regression due to possible overfitting and other inherent problems.

Due to the variables associated with constructing ANN classifiers and the local minima problem associated with training ANNs, there is an inherent error between true posterior probabilities and the least square estimates provided by ANN. This prediction error is composed of two components, the *approximation error* and the *estimation error*. The *approximation error* reflects a inherent irreducible consequence of the randomness of the training data. The *estimation error* is a reflection of the effectiveness of the ANN to approximate the target function.

The paper describes how a bias-plus-variance decomposition of the ANN prediction error provides useful information on how the estimate differs from the target function. The model bias quantifies how the average estimates over all possible data sets of the same size differ from the target function. Bias is an indication of the limitations of the model itself. Model variance is an indication of the sensitivity of the estimation

function to the training data set. Bias and variance are generally conflicting goals. ANNs are flexible and tend to have low bias but high variance.

Ensemble methods are described where classifiers are combined by averaging or voting prediction results from multiple ANNs. Improvements in prediction results are attributed to reduction of variance. The technique seems to work best when the voting models disagree with one another strongly i.e. are biased. Averaging seems to offset this bias and reduce sensitivity to the data. Methods of constructing biased models include statistical resampling techniques and using different feature variables.

Feature selection methods for ANNs are mostly heuristic in nature and and lack statistical justification.

Taking misclassification costs into account seems to improve the performance of ANNs in terms of classification and feature selection. Various techniques are described for incorporating misclassification cost information and prior knowledge of relative class importance, however, little research has been done in the this area.

[35] A. Zhou, F. Cao, W. Qian, and C. Jin, "Tracking clusters in evolving data streams over sliding windows," *Knowledge and Information Systems*, vol. 15, no. 2, pp. 181–214, 2008.

[36] L. Zhou, L. Wang, X. Ge, and Q. Shi, "A clustering-based knn improved algorithm clknn for text classification," in *Informatics in Control, Automation and Robotics (CAR), 2010 2nd International Asia Conference on*, vol. 3, 2010, pp. 212–215.

The KNN classification algorithm is widely used in large scale text classification applications. Although many other classification algorithms have been devised over the years, KNN scales better than most. KNN does not, however,

perform well where training samples are unevenly distributed within the feature space. This paper proposes a new algorithm, *CLKNN*, that performs pre-processing of training data sets using unsupervised clustering to even out training data distribution before applying KNN classification.

Previously, in dealing with the problem of unevenly distributed training samples, a denisity reduction technique has been shown to improve the speed and accuracy of KNN. However, this technique does not address the issue of low density regions. In *CLKNN*, a clustering algorithm is applied (which clustering algorithm exactly is not mentioned in the paper) to the training data set to partition it into a number of small mutually exclusive neighbourhoods. If the number of samples in a cluster exceeds a certain threshold and they all belong to the same class, the cluster centriod is substituted in the training data to represent all samples in the cluster. This process evens out the training data which is then processed by modified KNN algorithm.

Experimental results show that *CLKNN* exhibits an improvement over regular KNN classification accuracy and execution time. Experiments however only used two types of data sets and more work needs to be donw to validate the effect of this technique.

The language in this paper is poor and the logic difficult to follow.