Consensus Based Ensembles of Soft Clusterings

Kunal Punera
Dept. of Electrical and Computer Engineering
University of Texas at Austin
Austin, Texas 78712
Email: kunal@ece.utexas.edu

Joydeep Ghosh
Dept. of Electrical and Computer Engineering
University of Texas at Austin
Austin, Texas 78712
Email: ghosh@ece.utexas.edu

***Abstract*— Cluster Ensembles is a framework for combining multiple partitionings obtained from separate clustering runs into a final consensus clustering. This framework has attracted much interest recently because of its numerous practical applications, and a variety of approaches including Graph Partitioning, Maximum Likelihood, Genetic algorithms, and Voting-Merging have been proposed. The vast majority of these approaches accept hard clusterings as input. There are, however, many clustering algorithms such as EM and fuzzy c-means that naturally output soft partitionings of data, and forcibly hardening these partitions before obtaining a consensus potentially involves loss of valuable information. In this paper we propose several consensus algorithms that work on soft clusterings and experiment with many real-life datasets to empirically show that using soft clusterings as input does offer significant advantages, especially when dealing with vertically partitioned data.**

Keywords: Soft clustering, ensemble methods, graph based algorithms, information theoretic kmeans.

I. Introduction

Cluster Ensemble is a framework for combining multiple clusterings of a set of objects without accessing the original features of the objects. This problem was first proposed in a knowledge reuse framework by Strehl and Ghosh [1] who applied it for improving clustering quality and for distributed clustering. Since then cluster ensembles have been shown to be useful in many application scenarios, such as knowledge-reuse [2], multi-view clustering [3], distributed computing [4], and in improving the quality and robustness of clustering results [5], [6], [7], [8].

Several approaches have been proposed to solve cluster ensembles in the recent past. Strehl and Ghosh [1] proposed three graph-theoretic approaches for finding the consensus clustering. A bipartite graph partitioning based approach has been proposed in [9]. Topchy et al. [10] proposed the use of mixture of multinomial distributions to model the ensemble of labels along the lines of classical latent class analysis in marketing literature [3]. While these techniques are very varied in the algorithms they employ, the common thread is that they only work with hard constituent clusterings. It is the goal of this paper to investigate *Soft* Cluster Ensembles.

A. Ensembles of Soft Clusterings

There are several clustering algorithms, such as EM [11] and fuzzy c-means [12], that naturally output soft partitions of data. A soft partition assigns a value for the degree of association of each instance to each output cluster. So instead of a label vector for all the instances we have a matrix of values in which each instance is associated with every cluster with some membership value; often these values are the posterior probabilities and sum up to one. In order to solve an ensemble of soft clusterings using one of the existing algorithms mentioned above, we would have to "harden" the cluster assignments. This process involves completely assigning each instance to the cluster to which it is most associated. This results in the loss of the information contained in the uncertainties of the cluster assignments. This is especially true for application settings where underlying clustering algorithms access partial views of the data, such as in distributed data mining.

In the present study, we propose flexible frameworks for combining multiple soft clusterings directly without "hardening" the individual solutions first. We introduce a new consensus function ITK, based on the Information-Theoretic KMeans algorithm [13], that is more efficient and effective than existing approaches. For evaluation purposes, we create a large number of ensembles of varying degrees of difficulty, and report clustering results achieved by the various existing and new algorithms on them. In order to objectively evaluate ITK we extend existing algorithms to operate on soft cluster ensembles as well.

In Section II we describe some past work on solving cluster ensembles. In Section III we introduce *soft* cluster ensembles formally and also propose several new consensus functions which operate on them. The experimental setup and the empirical results comparing various soft-ensemble methods then follow in Section IV and Section V respectively. Finally, in Section VI we conclude the paper and briefly mention possible directions for future research.

II. Related Work on Cluster Ensembles

In this section we briefly describe past work on cluster ensembles. Readers are referred to the original papers for more details.

A. Ensembles of Hard Clusterings

Strehl and Ghosh [1] proposed three graph-theoretic approaches (CSPA, HGPA, and MCLA) for finding consensus clusterings. In CSPA the similarity between two data-points is defined to be directly proportional to number of constituent clusterings of the ensemble in which they are clustered together. The intuition is that the more similar two data-points are the higher is the chance that constituent clusterings will place them in the same cluster. This similarity graph between data-points is partitioned using METIS [14] to obtain the desired number of clusters. The HGPA algorithm seeks to directly partition the hyper-graph defined by the clusters in the ensemble. Hyper-graph partitioning seeks produce the final clustering by eliminating the minimal number of hyper-edges. This partitioning is performed using the package HMETIS [15]. The MCLA algorithm takes a slightly different approach to finding the consensus clustering than the previous two methods. First, it tries to solve the cluster correspondence problem by clustering the clusters to form meta-clusters. It then uses voting to place data-points into the meta-clusters obtaining the final consensus clustering. On similar lines, Fern and Brodley [9] proposed the HBGF algorithm that forms a bipartite graph between clusters and data-points, and then partitions the graph to obtain the final consensus clustering.

Topchy et al. [10] model the cluster ensemble using a generative model and use EM to estimate the parameters of the model. In

	$\lambda^{(1)}$	$\lambda^{(2)}$	$\lambda^{(3)}$
x_1	1	2	1
x_2	1	2	1
x_3	1	3	2
x_4	2	3	2
x_5	2	3	3
x_6	3	1	3
x_7	3	1	3

TABLE I
A SET OF THREE CLUSTERINGS

	$S^{(1)}$			$S^{(2)}$			$S^{(3)}$		
	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9
x_1	0.7	0.2	0.1	0.1	0.7	0.2	0.6	0.3	0.1
x_2	0.9	0.1	0.0	0.0	0.8	0.2	0.8	0.2	0.0
x_3	0.9	0.0	0.1	0.1	0.4	0.5	0.5	0.5	0.0
x_4	0.2	0.6	0.2	0.1	0.2	0.7	0.2	0.7	0.1
x_5	0.1	0.9	0.0	0.0	0.1	0.9	0.0	0.5	0.5
x_6	0.0	0.2	0.8	0.8	0.1	0.1	0.1	0.2	0.7
x_7	0.1	0.2	0.7	0.7	0.1	0.2	0.1	0.3	0.6

TABLE II
ENSEMBLE OF SOFT CLUSTERINGS

this approach, it is assumed that the ensemble has been generated from a mixture of multi-dimensional multinomial distributions. Each data point is generated by first picking a multinomial distribution according to the priors, and then picking the cluster label in each clustering from the chosen multinomial distribution over the cluster labels. The cluster labels of different constituent clusterings are assumed to be *i.i.d.*. A drawback of this approach is that the number of parameters to be estimated increases with both the number of constituent clusterings as well as with the number of clusters in them. Experiments in [10] do not involve datasets with more than 3 clusters. In this paper we will evaluate the performance of this consensus function on more complex real-life datasets. One advantage of this approach is that it enables us to conveniently model final clusters of different sizes via priors in the mixture model. Graph partitioning methods tend to yield roughly balanced clusters. This is a disadvantage in situations where the data distribution is not uniform.

B. Ensembles of Soft Clusterings

A landmark work on "collaborative" fuzzy clustering was done by Pedrycz [16]. The author considered a vertical partitioning scenario, and captured the collaboration between multiple partitionings via pair wise interaction coefficients. This resulted in an extended cost function to accommodate the collaboration effect in the optimization process. This approach is restricted in scope in many ways: each partition needs to have the same number of clusters; the difficult cluster correspondence problem is assumed to be already solved; and the distances between each point and its representative in each of the solutions need to be known. Despite these constraints, it was illustrated that, at least for simple 2 and 3 cluster problems, collaboration had a positive effect on cluster quality. The success of this work further motivates our research on more flexible consensus functions for ensembles of soft clusterings.

III. SOFT CLUSTER ENSEMBLES

In this section we will first formally define the soft cluster ensemble problem. We will then introduce a new algorithm based on Information Theoretic KMeans [13], and describe enhancements to existing techniques mentioned in Section II, to solve ensembles of soft clusterings.

A. The Soft Cluster Ensemble Problem

Let $X = \{x_1, x_2, ..., x_n\}$ denote a set of instances/objects. Also, let $\lambda^{(q)} \in \{1, 2, ...k^{(q)}\}^n$ denote the label vector of the q^{th} clustering of X; *i.e.* $\lambda_i^{(q)}$ is the label of x_i in the q^{th} clustering. This is a hard labeling produced by clusterings algorithms such as KMeans (see Table I). In cases where the underlying clustering algorithm outputs soft cluster labels, $\lambda_i^{(q)}$ is defined as $argmax_j P(C_j|x_i)$, where $P(C_j|x_i)$ is the posterior probability of instance x_i belonging to cluster C_j. An ensemble of soft clusterings is shown in Table II.

While constructing a soft cluster ensemble, instead of *hardening* the posterior probabilities into cluster labels we construct a matrix $S^{(q)}$ representing the solution of the q^{th} soft clustering algorithm. $S^{(q)}$ has a column for each cluster generated in the clustering and the rows denote the instances of data with $S_{ij}^{(q)}$ being the probability of x_i belonging to cluster j of the q^{th} clustering. Hence, the values in each row of $S^{(q)}$ sum up to 1. There are r such clusterings ($S^{(1,\dots,r)}$) each with $k^{(q)}$ clusters. Our goal is to find a consensus function Γ which combines these clusterings into a combined labeling, λ, of the data. It should be noted that the cluster ensemble framework doesn't specify whether the final clusterings should be hard or soft. In this paper we only work with algorithms that output hard final clusterings.

B. Solving Soft Ensembles with Information-Theoretic KMeans (ITK)

Information-Theoretic KMeans was introduced by Dhillon et al. [13] as way to cluster words in order to reduce dimensionality in the document clustering problem. It is very similar to the KMeans algorithm, differing only in the fact that as a measure of distance it uses the KL-divergence [17] instead of the Euclidean distance. The reader is referred to the original paper for more details. Here we just describe the mapping of the soft cluster ensemble problem to the information-theoretic KMeans problem.

Each instance in a soft ensemble is represented by a concatenation of r posterior membership probability distributions obtained from the constituent clustering algorithms (see matrix S in Table II). Hence, we can define a distance measure between two instances using the Kullback-Leibler (KL) divergence [17], which calculates the "distance" between two probability distributions. The distance between two instances x_a and x_b can be calculated as

$$KL_{x_a,x_b} = \sum_{q=1}^{r} w^{(q)} \sum_{i=1}^{k^{(q)}} S_{x_a i}^{(q)} log\left(\frac{S_{x_a i}^{(q)}}{S_{x_b i}^{(q)}}\right) \tag{1}$$

where, $w^{(q)}$ are clustering specific weights, so that $\sum_{q=1}^{r} w^{(q)} = 1$.

Note that Equation (1) is equivalent to computing the KL divergence between instances represented by a matrix S in which each row sums up to one. This normalization can be performed by multiplying each value in $S^{(q)}$ by $\frac{w^{(q)}}{\sum_{q=1}^{r} w^{(q)}}$. Now that we have a distance measure between instances based on KL-divergence, we can use the information-theoretic KMeans algorithm to obtain the final consensus clusters.

Computing Equation (1) with $w^{(q)} = \frac{1}{r}$ gives all clusterings equal importance. We can, however, imagine scenarios where we have different importance values for the constituent clusterings; possibly our confidence in the accuracy of these clusterings based on the number of features they access. These confidence values can be easily integrated into the cost function using the weights $w^{(q)}$.

C. Soft version of CSPA (sCSPA)

The CSPA algorithm [1] works by first creating a graph of all objects where edges are weighted by pair-wise similarities. This graph is then partitioned using METIS [14] to produce the desired number of clusters.

sCSPA extends CSPA by using values in S to compute pair-wise similarities. If we visualize each object as a point in $\sum_{q=1}^{r} k^{(q)}$ dimensional space, with each dimension corresponding to probability of its belonging to a cluster, then SS^T is the same as finding the dot product in this new space. Thus the technique first transforms the objects into a *label-space*, and then interprets the dot product between the vectors representing the objects as their similarity. In our experiments we use Euclidean distance in the label space to obtain our similarity measure. The dot product is highly co-related with the Euclidean measure, but Euclidean distance provides for cleaner semantics. The distance between two instances x_a and x_b is calculated as

$$d_{x_a,x_b} = \sqrt{\sum_{q=1}^{r}\sum_{i=1}^{k^{(q)}}\left(S_{x_a i}^{(q)} - S_{x_b i}^{(q)}\right)^2}$$

This can be interpreted as a measure of the difference in the membership of the objects for each cluster. This dissimilarity measure is converted into a similarity measure $s_{x_a,x_b} = e^{-d^2_{x_a,x_b}}$.

We can also define distances between instances using KL-divergence [17] as in Section III-B. In our experiments we observed that while all versions of the sCSPA (with Euclidean distance, KL divergence, and cosine similarity) performed similarly, the results obtained using Euclidean distance were slightly better. So in this paper we will only report results on sCSPA using Euclidean distance. sCSPA (like CSPA) is impractical for large datasets, and hence we will only report results for datasets with less than 2000 data-points.

D. Soft version of MCLA (sMCLA)

In MCLA each cluster is represented by a n-length binary association vector. The idea is to group and collapse related clusters into meta-clusters, and then assign each object to the meta-cluster in which it belongs most strongly. The clusters are grouped by graph partitioning based clustering.

sMCLA extends MCLA by accepting soft clusterings as input. sMCLA's working can be divided into the following steps (similar steps are followed in MCLA too).

Construct Soft Meta-Graph of Clusters: All the $\sum_{q=1}^{r} k^{(q)}$ clusters or indicator vectors s_j (with weights), can be considered as vertices of another regular undirected graph. The weight of the edge between two clusters s_a and s_b is set to $W_{a,b} = Euclidean\_dist(s_a, s_b)$. This distance is a measure of the difference in membership of all objects to these two clusters. As in the sCSPA algorithm, the Euclidean distance is converted into a similarity value.

Group the Clusters into Meta-Clusters: The meta-graph constructed in the previous step is partitioned using METIS to produce k balanced meta-clusters. Since each vertex in the meta-graph represents a distinct cluster label, a meta-cluster represents a group of corresponding cluster labels.

Collapse Meta-Clusters using Weighting: We now collapse all the clusters contained in each meta-cluster to form its association vector. Each meta-cluster's association vector contains a value for every object's association to it. This association vector is computed as the mean of the association vectors for each cluster that is grouped into the meta-cluster. This is a weighted form of the step performed in MCLA.

Compete for Objects: Each object is assigned to the meta-cluster to which it is most associated. This can potentially lead to a soft final clustering, since the ratio of the winning meta-cluster's association value to the sum of association values of all final meta-clusters can be the confidence of assignment of an object to the meta-cluster.

Name	Type of features	#features	#classes	#instances
8D5K	real	8	5	1000
Vowel	real	10	11	990
Pendigits	real	16	10	10992
Glass	real	9	6	214
HyperSpectral	real	30	13	5211
Yeast	real	8	10	1484
Vehicle	real	18	4	846

TABLE III
DATASETS USED IN EXPERIMENTS

There is, however, one issue with this approach. Because we are using soft clusterings as inputs, the meta-graph of the clusters is almost complete. More specifically, even clusters from the same clusterings have non-zero similarity to each other. This is not the case with MCLA since it uses a binary Jaccard measure, and for hard clusterings Jaccard similarity between clusters in the same clusterings is necessarily zero. Empirically, we get better consensus clustering results after making the meta-graph r-partite. Hence, sMCLA forces the similarity of clusters coming from the same clustering to be zero. This is, however, only done when the number of clusters in all the constituent clusterings is equal to the desired final number of clusters. In ensembles where the number of clusters in each underlying clustering vary the algorithm does not force the meta-graph to be r-partite.

E. Soft version of HBGF (sHBGF)

HBGF models the ensemble as a bipartite graph with each cluster and instance represented as nodes, and edges between the instances and the clusters they belong to. This approach can be adapted to consider soft ensembles by weighing the edges of the graph by the degree of association of instance to the cluster. This graph is then partitioned using METIS which accepts weighted edges. In sHBGF, the graph has $n+t$ vertices, where t is the total number of underlying clusters. The weights on the edges are set as follows:

- $W(i,j) = 0$ if i, j are both clusters or both instances
- $W_{(i,j)} = S_{i,j}$ otherwise, where i is the instance and j is the cluster

IV. EXPERIMENTAL SETUP

We empirically evaluate the various algorithms presented in Sections II and III on soft cluster ensembles generated from various datasets. In this section we describe the experimental setup in detail.

A. Datasets Used

We perform the experimental analysis using the six real-life datasets and one artificial dataset. Some basic properties of these datasets are summarized in Table III. These datasets were selected so as to present our algorithms with problems of varying degrees of difficulty – in terms of number of desired clusters, number of attributes, and number of instances. All these datasets, with the exception of 8D5K[1] and HyperSpectral[2], are publicly accessible from the UCI data repository[3].

[1]www.strehl.com
[2]csrserv.csr.utexas.edu/rs/research/hyper.html
[3]www.ics.uci.edu/~mlearn/MLRepository.html

Name	# attributes	Numatts options	#clusterings/Numatts-option
8D5K	8	3,4,5,6	10
Vowel	10	3,4,5,6,7	10
Pendigits	16	3,4,6,9,12	15
Glass	9	3,4,5,6,7	10
HyperSpectral	30	5,10,15,20,25	15
Yeast	8	2,3,4,5	10
Vehicle	18	4,5,8,11	15

TABLE IV
DATASET SPECIFIC OPTIONS FOR CREATING ENSEMBLES

B. Ensemble Test-set Creation

In order to compare the hard and soft ensemble methods, as well as to evaluate the our Information-Theoretic KMeans (ITK) based approach, we created soft cluster ensembles of varying degrees of difficulty. Note here that for each soft cluster ensemble we also stored its corresponding hardened version to evaluate methods that only accept hard clusterings.

The individual clusterings in our ensembles were created using the EM algorithm [11] with a mixture of Gaussian distributions model, but any algorithm that outputs soft probabilities could have been used. Further, each constituent clustering was created using vertically partitioned subsets of the datasets. This partial view of the data as well as the dependence of EM on initialization resulted in the diversity in the individual clustering solutions in an ensemble.

As mentioned above, we wanted to evaluate our algorithms on ensembles of varying degrees of difficulty. For this purpose we created ensembles by varying two parameters that controlled the degree of difficulty. The first parameter is the number of attributes that the EM algorithm accesses while creating the constituent clusterings. We expect the difficulty of an ensemble containing clusterings created from less attributes to be higher. The second parameter is the number of constituent clusterings in the ensemble. In general, we expect that as the number of constituent clusterings increase the consensus clusterings obtained should be more accurate. For most datasets the number of clusterings in the ensembles is varied from 2 to 10, and in some cases to 15. The entire set of options for all the datasets is listed in Table IV. The second column in the table describes the different settings for number of features used to create clusterings. For instance, for the 8D5K dataset we can obtain ensembles with constituent clusterings created using 3,4,5, or 6 attributes. Also, for each such setting we can select from 10 clusterings to form an ensemble. Of course, each of these 10 clusterings is created with a randomly selected set of attributes.

Hence, while creating an ensemble we specify three parameters: the dataset name, the number of attributes, and the number of clusterings. For each set of parameter values, we create multiple ensembles by randomly selecting the clusterings to combine. Also, non-deterministic consensus functions are run multiple times in order to average out variations in results due to initialization.

Here we must note that each individual clustering as well as the consensus function is given the true number of clusters to find. The use of ensembles for finding the true number of clusters, or the effect of different k in constituent clusterings on ensemble accuracy are not investigated in this study.

C. Evaluation Criteria

In order to evaluate the final consensus clusterings obtained we used Normalized Mutual Information (NMI) which was introduced by [1]. NMI can be used to compare the obtained clustering to the true labels of the instances.

The NMI of two labellings of instances can be measured as

$$NMI(X,Y) = \frac{I(X,Y)}{\sqrt{H(X)H(Y)}} \tag{2}$$

where, $I(X,Y)$ denotes the mutual information between two random variables X and Y and $H(X)$ denotes the entropy of X. In our evaluation, X will be consensus clustering while Y will be the true labels. More details about NMI can be obtained from [1].

Since we are varying the ensemble parameters over a very wide range for each dataset, we end up with a lot of different points of comparison. In order to report some sort of overall score for each algorithm on all the ensembles used, we use the Geometric Mean Ratio [18]. The GMR is calculated as follows. Suppose we have n ensembles that we tested our algorithms on, and NMI_A and NMI_B are vectors of the average NMI values w.r.t. to true labels obtained by algorithms A and B on these runs. GMR is calculated as

$$GMR(A,B) = \left(\prod_{i=1}^{n} \frac{NMI_{Bi}}{NMI_{Ai}} \right)^{\frac{1}{n}} \tag{3}$$

In later sections we display the GMR values in tables with rows and columns representing the algorithms being compared. In these tables element (i,j) represents the value $GMR(algo(i), algo(j))$, where $algo(i)$ and $algo(j)$ are the algorithms represented in row i and column j respectively. Hence, values > 1 along a column mean that the algorithm corresponding to the column performs better than the other algorithms. Similarly, the values < 1 along the rows indicates that the algorithm corresponding to the row scores better than the other algorithms.

V. SOFT VS HARD CLUSTER ENSEMBLES

In this section we present results from our evaluation of the algorithms we described in earlier sections using the experimental setup described Section IV.

A. Soft Versions of Existing Algorithms

In this section we evaluate the performance of CSPA, MCLA, and HBGF, their soft counterparts, and the Mixture of Multinomials method. The evaluation measure we employ is the Geometric Mean Ratio (GMR), which is calculated over all the ensembles that were created as described in Subsection IV-B. There were, however, some exceptions to the direct application of the GMR formula over all datasets. HBGF, CSPA and their soft versions were not run on the HyperSpectral and Pendigits datasets because these datasets are too large to expect solutions in reasonable time. Hence, when we compare one of these algorithms to the others we do not consider ensembles of these large datasets. Also, in certain cases (especially for tough ensembles) the consensus functions output clusterings that score 0 on the NMI measure. This would happen, for example, if all the instances were placed in a single consensus cluster. In such cases the GMR either becomes 0 or ∞ depending on where the zero score appears. Hence, we assign a very small nominal value (0.00001) to the NMI score whenever it is zero. The effect of this nominal score vanishes because we normalize by taking the n^{th} root of the product.

Table V shows the GMR values of the NMI measure comparing the three original algorithms as well as their soft versions. We can see that for each algorithm the soft version performs better than the corresponding hard version. Keep in mind that algorithms with values < 1 on their row are performing better than the others. These results show that the soft versions of the algorithms are able to use the

	CSPA	sCSPA	MCLA	sMCLA	HBGF	sHBGF	MixMns
CSPA	1	1.05	0.718	0.999	0.978	1.02	0.802
sCSPA	0.94	1	0.68	0.948	0.928	0.967	0.76
MCLA	1.163	1.22	1	1.17	1.136	1.18	0.913
sMCLA	1.00	1.05	0.56	1	0.978	1.019	0.77
HBGF	1.02	1.076	0.73	1.02	1	1.04	0.82
sHBGF	0.98	1.03	0.705	0.98	0.959	1	0.787
MixMns	1.25	1.31	0.73	1.297	1.219	1.269	1

TABLE V

GEOMETRIC MEAN RATIO OF NMI SCORE OVER ALL ENSEMBLES. THE VALUE $table_{i,j}$ INDICATES RATIO OF ALGORITHMS j/i

	CSPA	sCSPA	MCLA	sMCLA	HBGF	sHBGF	MixMns
CSPA	1	1.085	0.652	0.997	0.97	1.06	0.655
sCSPA	0.92	1	0.60	0.919	0.897	0.98	0.604
MCLA	1.53	1.665	1	1.47	1.49	1.63	0.922
sMCLA	1.003	1.088	0.46	1	0.976	1.06	0.627
HBGF	1.028	1.113	0.67	1.025	1	1.09	0.673
sHBGF	0.94	1.024	0.62	0.94	0.92	1	0.618
MixMns	1.53	1.656	0.73	1.59	1.485	1.617	1

TABLE VI

GEOMETRIC MEAN RATIO OF NMI SCORE OVER TOUGH ENSEMBLES. THE VALUE $table_{i,j}$ INDICATES RATIO OF ALGORITHMS j/i

	ITK 10K	sHBGF	sMCLA
ITK 10K	1	0.856	0.875
sHBGF	1.167	1	0.98
sMCLA	1.142	1.012	1

TABLE VII

GEOMETRIC MEAN RATIO OF NMI SCORE OVER ALL ENSEMBLES. THE VALUE $table_{i,j}$ INDICATES RATIO OF ALGORITHMS j/i

	ITK 10K	sHBGF	sMCLA
ITK 10K	1	0.816	0.798
sHBGF	1.226	1	0.94
sMCLA	1.253	1.06	1

TABLE VIII

GEOMETRIC MEAN RATIO OF NMI SCORE OVER TOUGH ENSEMBLES. THE VALUE $table_{i,j}$ INDICATES RATIO OF ALGORITHMS j/i

extra information in the soft ensembles to obtain better consensus clusterings.

We notice that the mixture of Multinomials algorithm (MixMns) performs worse than all other algorithms other than MCLA. This may be because many of the datasets we used had a large number of clusters, resulting in parameter estimation problems for the mixture model. Topchy et al. [10] only evaluated their algorithm on real datasets with very low number of clusters.

Another key observation is the dramatic difference in the performance of the sMCLA and MCLA algorithms. The performance improvement of sMCLA over MCLA is by far larger than the improvements by other soft versions like sCSPA and sHBGF. This is because MCLA's performance is very bad when the input clusterings are not accurate. This can be seen by its performance values over tough ensembles (Table VI) as well as ensembles with very low number of attributes in constituent clusterings (Figure 1). sMCLA doesn't get misled during the meta-clustering phase because the distances in the meta-graph are now determined from soft probabilities. Hence, an error in a input clustering which assigns an instance into the wrong cluster could be alleviated in sMCLA's case if the posterior probabilities of the wrong assignment are small. This phenomenon, however, needs to be investigated further since sMCLA performs on par with the best algorithms shown in Table V.

In order to evaluate the intuition that the information obtained from soft ensembles is especially useful when dealing with *tough* ensembles, we have populated the Table VI with GMR values calculated over only the *tough ensembles*. Tough ensembles are defined as those comprising a small number of clusterings, each of which are obtained using very few features. In our experiments, tough ensembles contained only 2-4 clusterings which were obtained using the minimum "Numatts" option for each dataset shown in Table IV. As we can see from Table VI, soft versions of algorithms perform better than their hard counterparts and the improvements in their performance are slightly higher than those in Table V. The fact that the improvements in performances are higher shows that the extra information in soft clusterings is useful in tough situations.

B. Information-Theoretic KMeans (ITK)

We compare the Information-Theoretic KMeans algorithm with two of the best algorithms from the analysis in the previous section. Table VII displays the GMR values for the ITK, sHBGF, and sMCLA algorithms over all the ensembles. As we can see, ITK performs appreciably better than both sHBGF and sMCLA. Moreover, sHBGF and sMCLA are fairly similar to each other in overall performance.

In order to find whether ITK performs better for tougher or simpler ensembles we calculate GMR over only the tough ensembles. Here again the tough ensembles are defined as in Subsection V-A. The results of this experiment are listed in Table VIII. As we can see, the improvement in ITK's performance over sHBGF/sMCLA is higher when considering the subset of tougher ensembles.

Some of the datasets used for this study present tougher challenges to the clustering algorithms than others. In terms of the NMI score of clusterings 8D5K is the simplest dataset while Yeast is the toughest. We display in Table IX and Table X the GMR value matrix for ensembles of datasets 8D5K and Yeast respectively. As we can see from these tables, in the case of the Yeast dataset ITK is by far the best performing algorithm. But for the 8D5K dataset all algorithms are fairly comparable with sHBGF slightly better than the rest. One reason is that for soft ensembles where most probability values are close to 1 or 0, more complex algorithms like ITK do not perform better than simple graph-theoretic approaches.

Another explanation for ITK's performance on the Yeast dataset can be provided based on the characteristics of the algorithms. The graph partitioning based consensus algorithms are constrained to provide roughly balanced clusters. This can be a problem in cases where the underlying data does not have balanced classes. The 8D5K dataset has perfectly balanced clusters (200 instances each) while the Yeast dataset has classes that range from 5 instances to 463 instances in size. The ITK algorithm is not constrained to find balanced clusters and hence can adapt the clustering solution to better match the natural distribution of instances in the data. This is why we see ITK outperform sHBGF and sMCLA on the Yeast dataset by a large margin.

C. Performance Variation with Increasing Attributes

In this section we examine how the performances of different consensus functions change as the number of attributes used for

	ITK 10K	sHBGF	sMCLA
ITK 10K	1	1.03	0.97
sHBGF	0.968	1	0.944
sMCLA	1.025	1.05	1

TABLE IX
GEOMETRIC MEAN RATIO OF NMI SCORE FOR ONLY THE 8D5K DATASET. THE VALUE $table_{i,j}$ INDICATES RATIO OF ALGORITHMS j/i

	ITK 10K	sHBGF	sMCLA
ITK 10K	1	0.84	0.68
sHBGF	1.18	1	0.817
sMCLA	1.454	1.222	1

TABLE X
GEOMETRIC MEAN RATIO OF NMI SCORE FOR ONLY THE YEAST DATASET. THE VALUE $table_{i,j}$ INDICATES RATIO OF ALGORITHMS j/i

the constituent clusterings is changed. The number of attributes is an ad-hoc measure of the quality of clustering obtained and hence the difficulty of the ensemble. In general, the lesser the number of attributes in the constituent clusterings the more the confusion in the clustering solutions obtained, and hence, the more the difficulty of obtaining a consensus labeling using these clustering solutions.

Figure 1 shows the variation in the performance of the hard ensemble methods and their soft variations on two datasets. The mixture of multinomial model method is not shown since its performance was much lower than the others. The datasets selected for these plots are of intermediate difficulty. As we can see, as we increase the number of attributes in the constituent clusterings the accuracies of all algorithms increase. For Pendigits Figure 1(a) only has curves for MCLA and sMCLA since we did not run HBGF and CSPA on it.

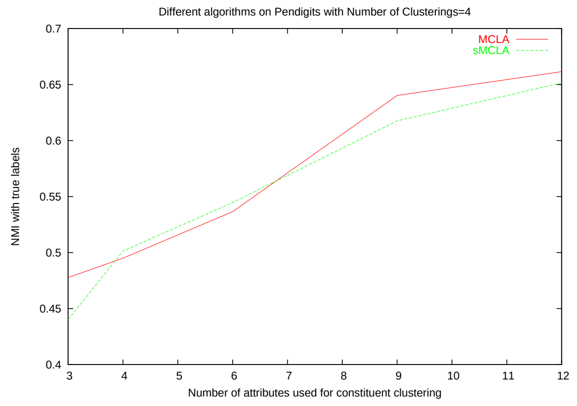
Figure 2 shows the performances of ITK, sHBGF, and sMCLA. As we can see ITK outperforms the other algorithms over the whole range of attributes. But as the number of attributes is increased the accuracies of all algorithms tend to plateau.

Fern and Brodley [5] show experimentally that for high dimensional domains combining clusterings on subspace projections of the data outperforms clustering on the whole data. They also found that the impact of subspace clustering is more prominent if the number of dimensions is higher (> 60). We have not experimented with datasets that have very high dimensionality, and hence we did not observe the reduction in accuracy when using the full set of attributes.

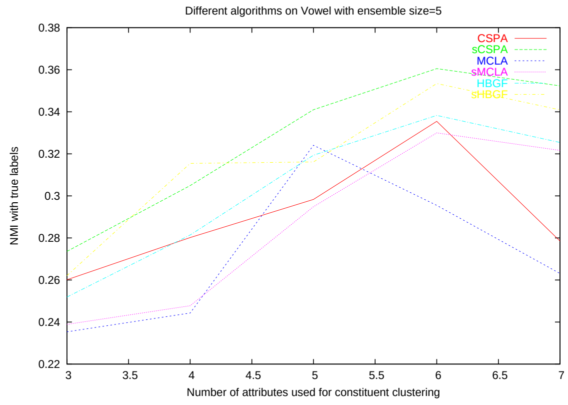
D. Performance Variation with Increasing Ensemble Size

In this section examine the effect of increasing the number clusterings used in the ensemble on the accuracy of final clustering. Say, we set the number of attributes used to create constituent clusterings to some constant value. We would then expect that as more clusterings are added to the ensemble the combining function would have more information available to create the final clustering. This has been previously seen in the classifier ensemble literature where increasing the size of the ensemble increases the accuracy until a saturation point is reached [19], [20], [21]. Hence, the number of clusterings in an ensemble can also be said to be a measure of the difficulty of the task of combining them.

Figure 3 shows the variation in accuracy as number of clusterings is increased in the ensembles. We can see that as the ensembles become easier to solve the accuracy of all algorithms increases. We can also see that the increasing accuracy of most algorithms reaches a plateau once the number of clusterings grows very large. Figure 4 shows the variation in accuracy of ITK, sMCLA, and sHBGF over the Pendigits and Vowel dataset as we increase the size of the ensembles. The accuracies of all the algorithms rise but the ITK algorithm performs significantly better than the others.

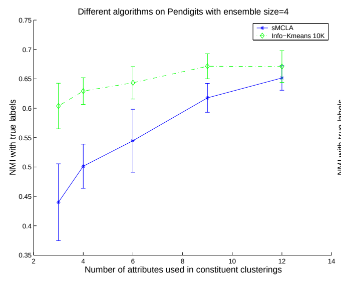


(a) Pendigits

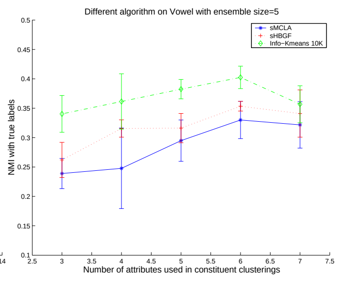


(b) Vowel

Fig. 1. Performance of CSPA, MCLA, HBGF, sCSPA, sMCLA, and sHBGF while varying the number of attributes used in constituent clusterings



(a) Pendigits

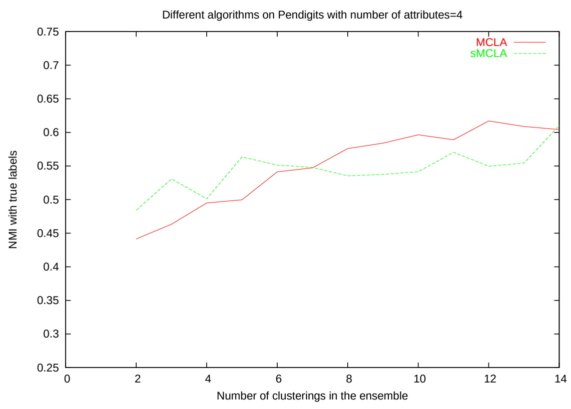


(b) Vowel

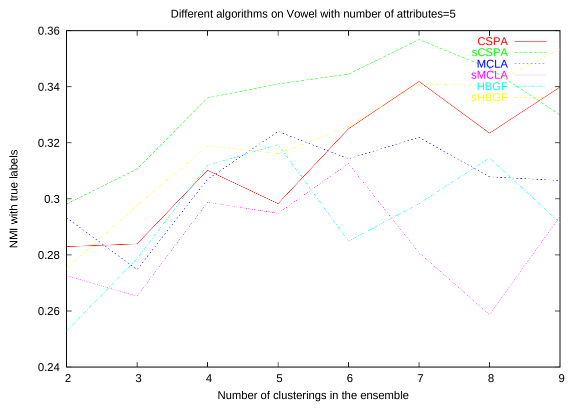
Fig. 2. Performance of ITK, sMCLA, and sHBGF while varying the number of attributes used in constituent clusterings

VI. CONCLUSIONS AND FUTURE WORK

In this paper we presented several approaches to solving ensembles of soft clusterings. We introduced a new approach based on Information-Theoretic KMeans (ITK), and also presented extensions of existing approaches for hard ensembles (like sCSPA, sMCLA, and sHBGF), These approaches were extensively evaluated using datasets

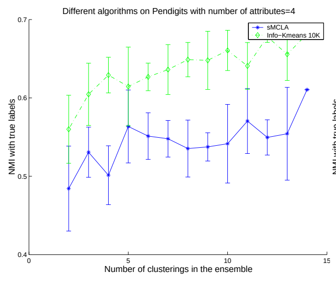


(a) Pendigits

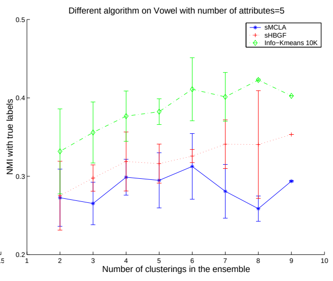


(b) Vowel

Fig. 3. Performance of CSPA, MCLA, HBGF, sCSPA, sMCLA, and sHBGF while varying the number of constituent clusterings



(a) Pendigits



(b) Vowel

Fig. 4. Performance of ITK, sMCLA, and sHBGF while varying the number of constituent clusterings

and ensembles of varying degrees of difficulty. Empirical evidence seems to suggest that soft ensembles contain useful information that can be exploited by our algorithms to obtain better consensus clusterings, especially in situations where the constituent clusterings are not very accurate. Also, ITK significantly outperforms existing approaches over most datasets, with the improvement in performance is especially large when dealing with *tough* ensembles.

Though the experimental results given in this chapter all assume the same number of clusters in each solution, the approaches do allow for varying resolution in the individual solutions. Moreover, the match of the consensus solution at different resolutions with respect to the individual solutions along the lines of [2] provides a good way of model selection. A further challenge is to identify scenarios where the use of soft ensembles provides significantly improved performance over hard ensembles, and if needed devise specialized algorithms to deal with these domains.

References

[1] A. Strehl and J. Ghosh, "Cluster ensembles – a knowledge reuse framework for combining multiple partitions," *Journal on Machine Learning Research (JMLR)*, vol. 3, pp. 583–617, December 2002. [Online]. Available: http://strehl.com/download/strehl-jmlr02.pdf

[2] J. Ghosh, A. Strehl, and S. Merugu, "A consensus framework for integrating distributed clusterings under limited knowledge sharing," in *Proceedings of NSF Workshop on Next Generation Data Mining*, November 2002, pp. 99–108. [Online]. Available: citeseer.nj.nec.com/ghosh02consensus.html

[3] A. M. Kreiger and P. Green, "A generalized rand-index method for consensus clustering of separate partitions of the same data base," *Journal of Classification*, vol. 16, pp. 63–89, 1999.

[4] S. Merugu and J. Ghosh, "Privacy-preserving distributed clustering using generative models," in *Proceedings of The Third IEEE International Conference on Data Mining (ICDM)*, 2003, pp. 211–218. [Online]. Available: citeseer.ist.psu.edu/705042.html

[5] X. Z. Fern and C. E. Brodley, "Random projection for high dimensional clustering: A cluster ensemble approach," in *Proceedings of the Twentieth International Conference on Machine Learning*. ACM Press, 2003.

[6] A. Fred and A. K. Jain, "Data clustering using evidence accumulation," in *Proceedings of the Sixteenth International Conference on Pattern Recognition (ICPR)*, 2002, pp. 276–280.

[7] L. Kuncheva and S. Hadjitodorov, "Using diversity in cluster ensembles," in *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, 2004, pp. 1214–1219.

[8] S. Hadjitodorov, L. Kuncheva, and L. Todorova, "Moderate diversity for better cluster ensembles," *Information Fusion*, vol. 7(3), pp. 264–275, 2006.

[9] X. Z. Fern and C. E. Brodley, "Solving cluster ensemble problems by bipartite graph partitioning," in *Proceedings of the Twenty-first International Conference on Machine Learning*. ACM Press, 2004.

[10] A. Topchy, A. Jain, and W. Punch, "Mixture Model for Clustering Ensembles," in *Proceedings of The Fourth SIAM Conference on Data Mining (SDM)*, 2004, pp. 379–390.

[11] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the em algorithm," in *Journal of the Royal Statistical Society*, vol. 39 Series B, 1977, pp. 1–38.

[12] J. C. Dunn, "A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters," *Journal of Cybernetics*, vol. 3, pp. 32–57, 1973.

[13] I. S. Dhillon, S. Mallela, and R. Kumar, "A divisive Information-Theoretic feature clustering algorithm for text classification," *Journal of Machine Learning Research*, vol. 3, pp. 1265–1287, 2003.

[14] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs," *SIAM Journal on Scientific Computing*, vol. 20(1), pp. 359–392, 1998.

[15] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar, "Multilevel hypergraph partitioning: application in VLSI domain," in *Proceedings of the Thirty-fourth Annual Conference on Design Automation*, 1997, pp. 526–529.

[16] W. Pedrycz, "Collaborative fuzzy clustering," *Pattern Recognition Letters*, vol. 23, no. 14, pp. 1675–86, 2002.

[17] S. Kullback and R. A. Leibler, "On information and sufficiency," *Annals of Mathematical Statistics*, vol. 22, pp. 79–86, 1951.

[18] G. I. Webb, "Multiboosting: A technique for combining boosting and wagging," *Machine Learning*, vol. 40, no. 2, pp. 159–196, 2000.

[19] L. Hansen and P. Salamon, "Neural network ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 993–1001, 1990.

[20] P. Melville and R. J. Mooney, "Constructing diverse classifier ensembles using artificial training examples," in *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI)*, 2003, pp. 505–510.

[21] D. Opitz and R. Maclin, "Popular ensemble methods: An empirical study," *Journal of Artificial Intelligence Research*, vol. 11, pp. 169–198, 1999. [Online]. Available: citeseer.ist.psu.edu/opitz99popular.html