Refined Shared Nearest Neighbors Graph for Combining Multiple Data Clusterings

Hanan Ayad and Mohamed Kamel

Pattern Analysis and Machine Intelligence Lab,
Systems Design Engineering, University of Waterloo,
Waterloo, Ontario N2L 3G1, Canada
{hanan,mkamel}@pami.uwaterloo.ca

Abstract. We recently introduced the idea of solving cluster ensembles using a Weighted Shared nearest neighbors Graph (*WSnnG*). Preliminary experiments have shown promising results in terms of integrating different clusterings into a combined one, such that the natural cluster structure of the data can be revealed. In this paper, we further study and extend the basic *WSnnG*. First, we introduce the use of fixed number of nearest neighbors in order to reduce the size of the graph. Second, we use refined weights on the edges and vertices of the graph. Experiments show that it is possible to capture the similarity relationships between the data patterns on a compact refined graph. Furthermore, the quality of the combined clustering based on the proposed *WSnnG* surpasses the average quality of the ensemble and that of an alternative clustering combining method based on partitioning of the patterns' co-association matrix.

1 Introduction

Cluster analysis is an unsupervised learning method that constitutes a cornerstone of an intelligent data analysis process. It is used for the exploration of inter-relationships among a collection of patterns, by organizing them into homogeneous clusters. It is called unsupervised learning because unlike classification (known as supervised learning), no *a priori* labelling of some patterns is available to use in categorizing others and inferring the cluster structure of the whole data.

Cluster analysis is a difficult problem because many factors (such as effective similarity measures, criterion functions, algorithms and initial conditions) come into play in devising a well tuned clustering technique for a given clustering problem. Moreover, it is well known that no clustering method can adequately handle all sorts of cluster structures (shape, size and density). In fact, the cluster structure produced by a clustering method is sometimes an artifact of the method itself that is actually imposed on the data rather than discovered about its true structure.

Combining of multiple classifiers has been successful in improving the quality of data classifications. Significant theoretical advances and successful practical

M.R. Berthold et al. (Eds.): IDA 2003, LNCS 2810, pp. 307–318, 2003.

applications have been achieved in this area [1]. An experimental comparison of various combining schemes can be found in [2]. In a recent review of multi-classifier systems [1], the integration of multiple clustering is considered as an example to further broaden and stimulate new progress in the area. However, the theoretical foundation of combining multiple clusterers, is still in its early stages. Some of the important related contributions can be found in [3] and [4].

In fact, combining multiple clusterings is a more challenging problem than combining multiple classifiers. In [5], the reasons that impede the study of clustering combination have been identified as: (1) It is believed that the quality of clustering combination algorithms can not be evaluated as precisely as combining classifiers. A priori knowledge or user's judgement plays a critical role in estimation of clustering performance. This problem represents an obstacle to proposing a mathematical theory to design clustering combination algorithms. (2) As various clustering algorithms produce largely different results due to different clustering criteria, combining the clustering results directly with integration rules, such as sum, product, median and majority vote can not generate a good meaningful result. In [1], combining multiple clusterings has been stated to be a more difficult problem, since cluster labels are symbolic and a correspondence problem must also be solved. Other difficulties include variability of the number and shape of clusters provided by individual solutions, and the desired or the unknown "right" number and shape.

However, as noted in [1,3], cluster ensembles can lead to improved quality and robustness of clustering solutions across a wider variety of data sets, enable *"knowledge reuse"*, and *"distributed clustering"* both in terms of objects or features. Moreover, we believe that one of the potential benefits of cluster ensembles lies in leveraging the ability of cluster analysis to reveal the natural cluster structure through careful analysis and consolidation of multiple different clustering of the input data. Therefore, cluster ensembles have a potential to play a critical role in the field of knowledge extraction from data, distributed learning and decision fusion of multiple unsupervised learners.

Cluster ensembles can be formed in a number of different ways, such as (1) the use of a number of different clustering techniques (either deliberately or arbitrarily selected). (2) The use of a single technique many times with different initial conditions. (3) The use of different partial subsets of features or patterns. In this paper, we use the first approach. As to the selection of the techniques, we use a set of techniques from those proposed in [3] for creating a diverse collection of clustering methods.

2 Shared Nearest-Neighbors Based Combiner

In a recent contribution [6], we introduced the idea of combining multiple different clusterings of a set of data patterns based on a *Weighted Shared nearest neighbors Graph* $WSnnG$. We compared the performance of the Shared nearest neighbors-based (Snn-based) combiner with the supra-consensus function introduced in [3]. Results were promising and showed the ability of the Snn-based combiner to reveal cluster structures that may be unbalanced.

2.1 Similarity Based on Sharing of Neighbors

Based on the principle of evidence accumulation introduced in [4] and which was inspired by the work in sensor fusion and classifier combination techniques in [2], a voting mechanism can be adopted to determine the similarity between patterns based on their co-occurrence in the given set of clusterings.

An alternative, or rather a complement to direct similarity between patterns is their shared nearest neighbors, an approach to similarity proposed in 1973 by Jarvis and Patrick [7] and recently extended in [8,9]. The philosophy of the approach is that one can confirm the similarity between two patterns by their common (i.e shared) nearest neighbors. For instance, if patterns i and j are similar, and if they are both similar to a set of shared patterns S, then one can have greater confidence in the similarity between i and j, since their similarity is confirmed by the set of their shared neighbors S.

In the proposed combiner, we emphasize the shared neighborhood relationships between patterns. The extracted relationships are used to construct the $WSnnG$ whose vertices correspond to the patterns and the edges represent the links between the patterns based on their neighborhood sharing. Weights are assigned on both edges and vertices, as will be further explained later.

The advantages of $WSnnG$ are: first, we find that the shared nearest neighbors approach to similarity (co-association) can be particularly useful in combining different data clusterings. In combining different clusterings based on co-associations, one faces the issue of transitivity of co-associations in merging patterns. We believe that re-defining co-associations in terms of neighborhood sharing provides a means to carefully analyze the co-associations among patterns, as revealed by the consensus and conflicts among the different clusterings. Therefore, ideas based on this concept, such as those used in this paper, can help in integrating different cluster structures of the data, beyond applying strict consensus or transitivity on co-associations.

Second, we derive from this approach, the definition of the *shared nearest neighbors population* associated with each pattern i as the sum or weighted sum of shared neighbors with pattern i. We use this number in determining the relative weights of patterns. This is analogous to the notion of probability density of a point defined in [8]. The idea is that we assume that if a point has a lot of highly associated shared nearest neighbors, then it is more likely to be a member of a large and/or dense cluster. On the other hand, a point that has fewer weakly associated shared nearest neighbors, then it is more likely to be a member of a small and/or dispersed cluster.

2.2 Problem Formulation

Let $X = \{x_1, x_2, \cdots, x_n\}$ denote a set of n data patterns, and $C = \{C_1, C_2, \cdots, C_r\}$ is a set of r clusterings (cluster ensemble) of the n data patterns. From the given set of clusterings C of the n patterns, we construct a $WSnnG(V, E)$, where $V = \{v_1, v_2, \cdots v_n\}$ is the set of weighted vertices corresponding to the set of patterns. $E = \{e_1, e_2, \cdots, e_m\}$ represents the set of weighted edges connecting the patterns, where m is the number of edges. An edge between any pair of

vertices is counted only once, i.e. edge connecting vertex v_i to vertex v_j is not counted separately from edge connecting v_j to v_i.

Since we combine the different clusterings of the ensemble without accessing the data patterns X in their feature space, we will refer, from now on, to a pattern by its index, that is, we use i rather than x_i. Before building the $WSnnG$, the following preprocessing steps are performed.

A. Computation of co-associations (direct-similarities):
Similar to the voting mechanism described in [4], a similarity measure between patterns co-occurring in the same cluster is computed as

$$co - assoc(i,j) = \frac{votes_{ij}}{r}$$

where $votes_{ij}$ is the number of times patterns i and j are assigned to the same cluster among the r clusterings. That is $0 \leq co - assoc(i,j) \leq 1.0$.

B. Sparsification of the nearest neighbors:
Based on the co-associations between patterns, a list of nearest neighbors P^i to each pattern i is defined as the list of patterns whose co-associations with i satisfy a selected $vote_{Thresh}$, where $0 < vote_{Thresh} \leq 1.0$. That is, $\forall(i,j), i \in P^j$ and $j \in P^i \iff co - assoc(i,j) \geq vote_{Thresh}$.

A number of factors can play a role in choosing the value of the $vote_{Thresh}$, such as the size of the ensemble, knowledge about the relative performance of the individual clustering techniques. Other issue such as the number of clusters relative to the size of the data. Increasing the value of $vote_{Thresh}$ corresponds to imposing stricter consensus among the clusterings, while decreasing it, corresponds to relaxes this condition and leads to revealing of more nearest neighbors. A majority vote corresponds to $vote_{Thresh} = 0.5$. In the experiments described, we maintain a majority vote threshold, as we find it suitable for the diverse ensemble that we use. The ensemble consist of most of the techniques originally proposed in [3].

The construction of the $WSnnG$, is described in the next two subsections. After its construction, the graph is then partitioned using the graph partitioning package METIS [10], in which the underlying algorithms described in [10,11, 12] are based on the state-of-the-art multilevel paradigm that has been shown to produce high quality results and scale to very large problems. METIS can partition a graph in the presence a balancing constraint. The idea is that each vertex has a weight associated with it, and the objective of the partitioning algorithm is to minimize the edge-cut subject to the constraint that the weight of the vertices is equally distributed among the clusters.

2.3 Basic WSnnG

The computation of the weights in the basic WSnnG [6] are described as follows.

i. Evaluation of neighborhood sharing: For each pair of patterns i and j, determine the size (count) of their shared nearest neighbors, i.e., the size of

the intersection $|P^i \cap P^j|$ of their respective nearest neighbors lists P^i and P^j. Using the Jaccard measure of similarity, we compute the strength σ_{ij} of the neighborhood sharing between the two objects i and j as

$$\sigma_{ij} = 2 \times \frac{|P^i \cap P^j|}{|P^i| + |P^j|} \tag{1}$$

If σ_{ij} satisfies a strength threshold $\sigma_{ijThresh}$, where this threshold is evaluated dynamically for each pair of objects i and j, and is given as, $\sigma_{ijThresh} = \mu \times max(|P^i|, |P^j|)$. The parameter μ determines the required ratio of overlap. In the reported experiments $\mu = 0.5$, indicating that objects must share at least 50% of their nearest neighbors in order to satisfy the strength threshold.

Then

- Create an edge between i and j with weight σ_{ij}
- Increment shared nearest neighbors populations θ_i, θ_j for objects i and j

ii. Assignment of weights to the vertices of the graph: The weight ω_i of each object i is computed as the ratio of a balancing factor l to the shared nearest neighbors population θ_i associated with object i. That is, $\omega_i = \frac{l}{\theta_i}$ The idea of the relative weights ω_i is to in reflect in the graph the varying cluster sizes.

This graph is characterized by the variability of the size of each pattern's nearest neighbors. This type of graph, however, can grow into a significantly large one reaching an $O(n^2)$. For instance, in such cases where n is large and the number of clusters is small, this leads to large lists of nearest neighbors P^i to each pattern i. Furthermore, the weighting scheme used is binary, it is sensitive to the number of shared nearest neighbors as shown in Equation 1, rather than their corresponding co-associations (values of votes ratios).

2.4 Refined WSnnG

In the refined graph, we want to use only the k-nearest neighbors of each pattern, thus ensuring a compact graph of size $O(kn)$ and consequently reducing the computational cost. In addition, we use a refined weighting of the edges and vertices that is sensitive to the corresponding vote ratios instead of the binary scheme used in the basic graph.

We refer to the list of k-nearest neighbors of pattern i as L^i. A characteristic of the lists P^i (defined in step B. of Section 2.2) that is not found in L^i is their of *mutuality*. We have, $\forall(i,j), i \in P^j \iff j \in P^i$, whereas this does not hold for the lists L^i.

Therefore, two algorithmic options present themselves in dealing with the issue of mutuality, while building the $WSnnG$ based on k-nearest neighbors. It is possible that one chooses to connect two patterns i and j only when they are mutual neighbors. Alternatively, one can choose to connect them whenever either $i \in L^j$ or $j \in L^i$. In this paper, we choose the second option because it reveals more relationships at smaller values of k and leads to faster convergence.

The weights assigned to the vertices are meant to reflect an estimate of the probability density of the patterns, thus estimating the probability distribution of the clusters to which the patterns belong. In fact, this is an important aspect of the solution that we propose for accommodating natural cluster structures that may be unbalanced. In the refined $WSnnG$, we use the weighted sum of the shared nearest neighbors as the density estimates of the patterns. Again, at this stage, two options present themselves. One option is to compute the weight of pattern i based only neighborhood sharing with the elements of its pruned k-nearest neighbors list L^i. The second option is to use all the elements of the P^i list. In this paper, we choose the second option in order to maintain a stability to the partitioning solutions at different values of k. This is achieved due to the global computation of the weights assigned to the vertices. Therefore, choosing k becomes less of an issue. The computations of the weights are described below followed by a description of the algorithm *WSnnG-Builder* in Algorithm 1.

i. Evaluation of neighborhood sharing: Let $S^{ij} = \{s_1^{ij}, s_2^{ij} \cdots s_{n^{ij}}^{ij}\} = P^i \cap P^j$, be the set of shared nearest neighbors between pattern i and j, and $W^{ij} = \{w_1^{ij}, w_2^{ij}, ..., w_{n^ij}^{ij}\}$ is a corresponding list of weights. n^{ij} is the number of shared patterns between patterns i and j. The weight $w_p^{ij} \in W^{ij}$ is computed as the product of ϑ_p^i and ϑ_p^j where ϑ_p^i is the $co-assoc(i, s_p^{ij})$, and ϑ_p^j is the $co-assoc(j, s_p^{ij})$. That is, the corresponding actual vote ratios. The formulas for computing w_p^{ij} and the total weight w^{ij} are as follows.

$$w_p^{ij} = \vartheta_p^i \times \vartheta_p^j.$$

$$w^{ij} = \sum_{p=1}^{n^{ij}} w_p^{ij}. \tag{2}$$

Let W^i and W^j be the lists of corresponding weights to P^i and P^j, as determined from the vote ratios (in the preprocessing steps of Section 2.2). Using on the extended Jaccard measure [13], we compute the strength of neighborhood sharing σ_{ij} as

$$\sigma_{ij} = \frac{w^{ij}}{||W^i||^2 + ||W^j||^2 - w^{ij}} \tag{3}$$

ii. Assignment of weights to the vertices of the graph: The weight ω_i of each pattern i is computed as the ratio of a balancing factor l to the weighted sum of the shared nearest neighbors population θ_i, where $\theta_i = \sum_{j=1}^{n^i} w^{ij}$, and n_i is size of the list P^i. Finally, ω_i is given by $\omega_i = \frac{l}{\theta_i}$.

Notice that we use only the k-nearest neighbors L^i of each pattern i, to connect edges between the vertices of the graph, but we use the nearest neighbors P^i to evaluate the strength of neighborhood sharing, and to evaluate the patterns' weights (density estimates). Consequently, the size of the graph is reduced from a worst case complexity of $O(n^2)$ (Basic Graph) to $O(kn)$.

Algorithm 1 WSnnG-Builder

for all pattern i in the dataset **do**
  for all pattern $j \in P^i$, such that the symmetric pair (j, i) was not processed **do**
    compute w^{ij} as given by Equation 2
    $\theta_i \leftarrow \theta_i + w^{ij}$
    $\theta_j \leftarrow \theta_j + w^{ij}$
    if $j \in L^i$ OR $i \in L^j$ **then**
      connect patterns i and j with an edge of strength σ_{ij} as given by Equation 3
    end if
  end for
  $\omega_i \leftarrow \frac{l}{\theta_i}$
  assign the weight ω_i to pattern i.
end for

Also, notice that we eliminated the μ parameter that was used in the basic graph. This is done to avoid filtering of relationships, since we only consider a limited number of neighbors. We rely instead on the added sensitivity to the weighting scheme in determining the strength of the links, and weights of patterns. However, one may still choose to use a threshold and eliminate weak relationships.

Basically, the idea is to extract from the relationships revealed by the ensemble, representative interconnections, and corresponding weights, in a concise graph. An important characteristic of the graph is to be flexible enough to capture different clusters distributions, which may naturally occur in the data, making it more difficult to clustering techniques to reveal them, and also causing variability in the clustering generated using different techniques.

2.5 Partitioning of the WSnnG

The $WSnnG$ is partitioned into q clusters, so as to minimize the edge-cut subject to the constraint that the weights on the vertices are equally distributed among the q clusters. The issue of how to determine q is not addressed in this work. An estimate of q is rather assumed to be available for the combiner and the individual clustering techniques forming the ensemble. By assigning relative weights ω_i to the vertices based on their corresponding weighted sum of neighborhood sharing, we embed in the graph information about the sizes of neighborhood associated with each pattern, thus accommodating for varying cluster sizes and structure imbalances. The weights σ_{ij} on the edges, are based on the strength of the link connecting patterns i and j. Stronger links are less likely to be cut, again subject to the balance constraint.

3 Experimental Analysis

In [3], an ensemble of diverse techniques was proposed to be used as a portfolio of methods across a wide variety of data from different domains and dimensionality. We use most of the techniques of this portfolio, totalling to 9 different techniques.

They are the graph partitioning and k-means, each using 4 different similarity measures (Euclidean, cosine, extended Jaccard and correlation), in addition to hypergraph partitioning. We used ClusterPack, a cluster analysis package by Alexander Strehl available at http://strehl.com/ to generate the ensemble.

3.1 Quality Evaluation

We use the ensemble of clusterings to build a refined $WSnnG$, at variable values of K in the range [10-100], in order to study the behavior of the method. For each K, we generate the combined clustering based on the graph partitioning. In order to study the merit of our approach, we compare it to solution generated by the partitioning of the original co-association matrix, which is the cluster ensemble method called CSPA (Cluster-based Similarity Partitioning Algorithm) in [3]. The partitioning is generated using METIS on the original co-association matrix (direct similarities based on votes of co-associations). We also compare the quality of the combined clustering to the average quality of the individual clustering techniques forming the ensemble.

We use the F-measure [14], which combines both precision and recall to evaluate the quality of the different clustering solutions. The F-measure is an external criteria based on additional information not given to the clusterers, which is the labelled categorization assigned externally by humans.

3.2 Artificial Data

We artificially generated 3 datasets of 2 dimensional points. The datasets are shown in Figure 1. Dataset 2D-2C consists of two clusters of points of sizes (50, 150) sampled from two Gaussian distributions with different means and co-variance matrices. Dataset 2D-3C consists of three clusters of points of sizes (50, 75, 150) also sampled from three Gaussian distributions using different means, and co-variance matrices. Dataset 2C-NonConvex consists of points forming two non-convex clusters of sizes (120, 300). The clusters were generated by concatenating several smaller Gaussian clusters with means moving along half circles.

From the results shown in Figure 2, we notice the superiority of the combined clustering based on the refined $WSnnG$ over the ensemble's mean and the CSPA clustering. In addition, we find an improvement in terms of the discovered cluster distribution relative to the natural clusters inherent in the data. In contrast to the combined clustering based on the CSPA method which always generate equal size clusters, thus imposing well balanced clusters, our approach thrives to discover the true probability distributions of the clusters.

The clustering of the datasets 2D-2C, 2D-3C and 2C-NonConvex based on the CSPA, discovered clusters of sizes as follows (100, 100), (91, 91, 93) and (210, 210), respectively. The WSnnG solutions for different K discovered clusters of sizes around (44, 156), (27, 78, 170) and (108, 312) respectively. By comparing these values to the original cluster sizes, we find that the WSnnG solutions represent better estimates of the cluster sizes.

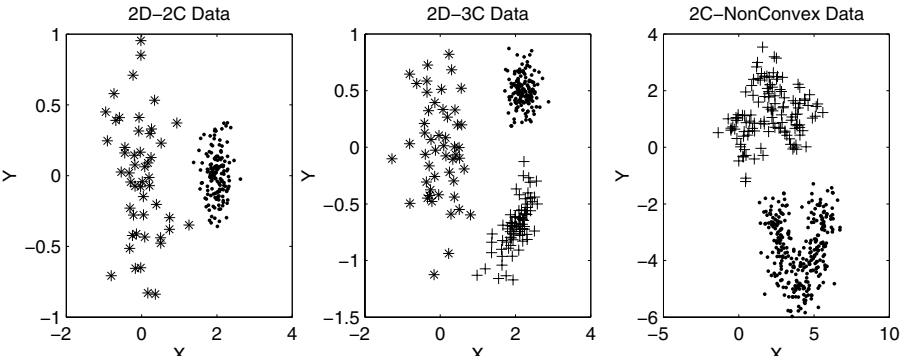

Fig. 1. Artificial datasets

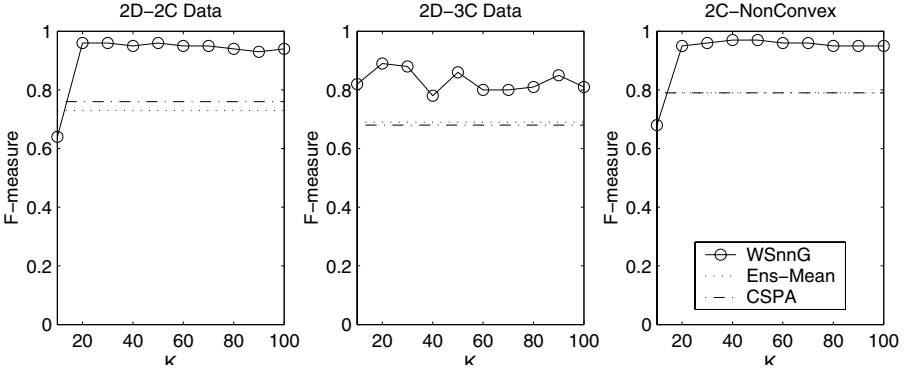

Fig. 2. Artificial data results. The ensemble's mean (Ens-Mean) and the CSPA are not function of K, they have constant values. We plot them on the same plot of WSnnG vs K to facilitate the comparison. With the 2D-3C and the 2C-NonConvex datasets, the Ens-Mean and the CSPA overlap.

3.3 Real Data

We experimented with datasets from the UCI machine learning repository available at http://www.ics.uci.edu/ mlearn/MLRepository.html. The datasets selected are generally used in classification and pattern recognition problems. The results are shown in Figure 3, and further discussed below.

Ecoli Dataset. The Ecoli Dataset is used in classification problems for predicting the cellular localization sites of proteins [15]. The dataset consists of 336 instances represented by 7 numeric predictive attributes. There are 8 classes of unbalanced distribution, with class sizes ranging from 2 to 143 patterns per cluster (ppc). From the results shown in Figure 3, we find that the quality of the combined clustering based on the $WSnnG$ surpasses both the ensemble's average and combined clustering based on CSPA. In addition, the CSPA method

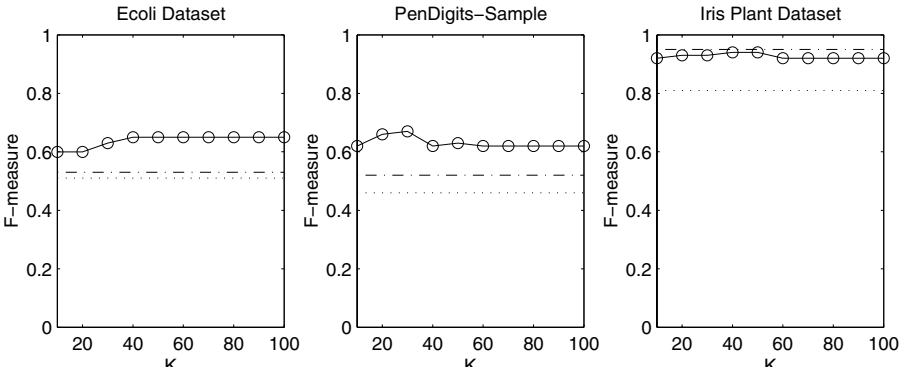

Fig. 3. Results on real datasets

generated equal size clusters ranging from 41 to 43 ppc, while the solutions based on $WSnnG$ generated cluster sizes approximately ranging from 1 to 105 ppc.

Pen-Based Recognition of Handwritten Digits. The Pen Digits dataset is used as a benchmark problem for the classification of pen-based recognition of handwritten digits 0 to 9 from 16 spatial features [16]. Samples from this datasets have also been used in experiments with cluster ensembles [3]. The whole dataset consists of ten classes of of roughly equal sizes. We generated a sample from the dataset with unequal class distribution. PenDigits-Sample consists of 490 instances with varying class sizes ranging from 10 to 150 ppc. From the results shown in Figure 3, the F-measure of the clusterings based on the $WSnnG$ surpasses both the ensemble's mean and the CSPA-based clustering. We also find that while the combined clustering based on CSPA consists of equal size clusters ranging from 48 to 50 ppc, the $WSnnG$ clusters were approximately ranging from 1 to 114 ppc.

Iris Plant Dataset. The Iris plant dataset is one of the best known datasets in the pattern recognition literature. The dataset contains 3 classes of 50 instances each, where each class refers to a type of Iris plant, and the data is represented by 4 attributes. One class is linearly separable from the other 2; the latter are not linearly separable from each other. As opposed to the other datasets used in this paper, this is an example of a well balanced dataset with patterns equally distributed among the clusters. The challenge for the individual clustering methods of the ensemble is rather due to the interleaving of two classes. In this particular example, the CSPA performed well, since the natural clusters are well balanced. The $WSnnG$ solutions also generated balanced clusters of comparable performance to CSPA while achieving an improvement over the ensemble's mean.

4 Conclusions

We introduced in this paper a compact and refined version of the $WSnnG$. The shared nearest neighbors approach to similarity provides a complement to direct similarities and is a more reliable approach to analyze and extract complex similarity relationships among patterns. We believe that this approach can be particularly useful in combining clusterings. As shown in the experiments, the quality of combined clustering based on the refined $WSnnG$ surpasses the ensemble's mean, and enhances the combined clustering based the CSPA method, in terms of the F-measures and the cluster distributions revealed.

The WSnnG-based approach thrives to discover the natural cluster structure, using the relationships revealed by the ensemble of individual clustering techniques. We argue that this approach can successfully capture and flexibly adapts to natural cluster distributions that may be unbalanced. We believe that it can provide a new tool to solve cluster ensembles for the purpose of enhancing the clustering quality while preserving fidelity to the true clusters inherent in the data. In this paper, experiments have shown promising results in terms of capturing the significant inter-relationships among patterns on a compact and refined graph.

In Future work, we want to further analyze and extend the $WSnnG$. Moreover, we want to explore its applicability in the domain of very high dimensional data, overlapping clusters, and multi-class patterns. Finally, it is worth noting that in this work, we emphasized the underlying model for combining a given number of clusterings, while the issue of which clustering techniques to use, as an ensemble, is not emphasized. Nonetheless, the study of different ensembles and which combination of clustering techniques would produce the best results with the proposed combiner, will be addressed in future work.

Acknowledgements. This work was partially funded by an NSERC strategic grant.

References

1. J. Ghosh. Multiclassifier systems: Back to the future. In J. Kittler F. Roli, editor, *Multiple Classifier Systems: Third International Workshop, MCS 2002, Proceedings. LNCS 2364*, pages 1–15, Cagliari, Italy, June 2002. Springer.
2. K. Kittler, M. Hatef, R.P.W Duin, and J. Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, March 1998.
3. A. Strehl and J. Ghosh. Cluster ensembles – a knowledge reuse framework for combining partitionings. In *Conference on Artificial Intelligence (AAAI 2002)*, pages 93–98, Edmonton, July 2002. AAAI/MIT Press.
4. A. Fred and A.K. Jain. Data clustering using evidence accumulation. In *Proceedings of the 16th International Conference on Pattern Recognition. ICPR 2002*, volume 4, pages 276–280, Quebec City, Quebec, Canada, August 2002.
5. Y. Qian and C. Suen. Clustering combination method. In *International Conference on Pattern Recognition. ICPR 2000*, volume 2, pages 732–735, Barcelona, Spain, September 2000.

6. H. Ayad and M. Kamel. Finding natural clusters using multi-clusterer combiner based on shared nearest neighbors. In *Multiple Classifier Systems: Fourth International Workshop, MCS 2003, Guildford, Surrey, United Kingdom, June 11–13. Proceedings. To Appear*, 2003.
7. R.A. Jarvis and E.A. Patrick. Clustering using a similarity measure based on shared nearest neighbors. *IEEE Transactions on Computers*, C-22(11):1025–1034, November 1973.
8. L. Ertoz, M. Steinbach, and V. Kumar. Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data. Technical report, Department of Computer Science, University of Minnesota, 2002.
9. L. Ertoz, M. Steinbach, and V. Kumar. A new shared nearest neighbor clustering algorithm and its applications. In *Workshop on Clustering High Dimensional Data and its Applications at 2nd SIAM International Conference on Data Mining*, Arlington, VA, 2002.
10. George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. Technical Report TR 95–035, Department of Computer Science and Engineering, University of Minnesota, 1995.
11. George Karypis and Vipin Kumar. Multilevel k-way partitioning scheme for irregular graphs. Technical Report TR 95–064, Department of Computer Science and Engineering, University of Minnesota, 1995.
12. George Karypis and Vipin Kumar. Multilevel algorithms for multi-constraint graph partitioning. In *Conference on High Performance Networking and Computing. Proceedings of the 1998 ACM/IEEE conference on Supercomputing*, San Jose, CA, 1998.
13. A. Strehl and J. Ghosh. Relationship-based clustering and visualization for high dimensional data mining. *INFORMS Journal on Computing*, pages 1–23, 2002.
14. M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, 2000.
15. Paul Horton and Kenta Nakai. A probablistic classification system for predicting the cellular localization sites of proteins. *Intelligent Systems in Molecular Biology*, pages 109–115, 1996.
16. F. Alimoglu. Combining multiple classifiers for pen-based handwritten digit recognition. Master's thesis, Institute of Graduate Studies in Science and Engineering, Bogazici University, 1996.