

# Lambda Consensus Clustering

Douglas R. Heisterkamp  
Computer Science Department  
Oklahoma State University  
Stillwater, Oklahoma 74078  
Email: doug@cs.okstate.edu

**Abstract**—This paper introduces an extension to consensus clustering that allows a feedback of the results of the consensus to the original clustering processes. The original clustering processes may use this information to update their partitioning of the data. An exponential weighting approach, called lambda consensus, is presented as a method to merged the consensus information into graph based and vector space based clustering algorithms. Successful consensus clustering is highly dependent on the quality and diversity of the partitions in the ensemble. The feedback signal allows the clustering processes to adapt their algorithms to attempt to improve quality and diversity of the set of partitions in the ensemble. Communication requirements are on the same order as consensus clustering as only the consensus labels are returned to the clustering processes. The method is evaluated on real world data sets.

**Keywords**-consensus clustering; ensemble clustering; k-means clustering; pinch ratio clustering;

## I. INTRODUCTION

Clustering [1] is an unsupervised learning problem and a fundamental tool of data analysis that creates a partitioning of a data set. Consensus clustering is an ensemble approach that uses the labellings from a set of partitions to create a partitioning without accessing the original data set. Consensus clustering [2] is useful in distributed computing as only the labels need to be communicated from an individual process. It is also useful for privacy preservation as access to the original data is not needed. Even in non-distributed situations, consensus clustering is often used to improve quality and robustness of a clustering. The success of consensus clustering in finding a good partitioning is highly dependent on the quality and diversity of the clusterings in the ensemble [3], [4].

With the goals of improving quality and managing diversity, this paper proposes extending the consensus clustering framework with a feedback loop of the results of the consensus to the individual clustering processes. The individual clusterings processes can then utilize the consensus knowledge to create a new partitioning of the data. If only the consensus labelling is returned then the extension maintains the low communication and privacy preservation benefits consensus clustering. Since many clustering algorithms stop in a local minimum of their objective function, the extra information from the consensus may allow them to transition to a better local minimum and hence improve the quality

of the partitions in the ensemble. An exponential weighting approach, called  $\lambda$ -consensus clustering, is proposed as a method of merging the consensus results into the similarity or distance function of an individual clustering algorithm. An advantage of modifying the distance or similarity measure is that the individual clustering algorithms can be used with little or no modifications. The parameter  $\lambda$  controls the weighting between the original measure and consensus measure with the contribution of the original data decaying exponentially by  $\lambda^t$  for feedback iteration  $t$ . An alternative usage of consensus feedback would be to use the difference of the local partitioning and the consensus partitioning as a search direction for incremental updates to the local partitioning. This approach requires a customization of individual clustering algorithms and is not explored in this paper.

Consensus clustering is briefly reviewed in section II. The  $\lambda$ -consensus clustering approach is presented in section III, followed by preliminary experimental results in section IV and the summary in section V.

## II. CONSENSUS CLUSTERING

Denote a set of  $n$  data objects  $\{\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_{n-1}\}$  as  $\mathcal{X}$ . A partition  $\pi$  of  $\mathcal{X}$  is a set of  $k$  subsets  $C_i$  of  $\mathcal{X}$  such that  $\bigcup_{i=0}^{k-1} C_i = \mathcal{X}$  and  $C_i \cap C_j = \emptyset$  for  $0 \leq i, j < k$  and  $i \neq j$ . Let  $\Pi = \{\pi_0, \pi_1, \dots, \pi_{r-1}\}$  denote a set of  $r$  different partitions of  $\mathcal{X}$ . The consensus clustering problem is to create a partition  $\pi_{cc}$  using only the structures of the partitions in set  $\Pi$  and not the original data  $\mathcal{X}$ .

Similarity based clustering algorithms can be used for clustering  $\Pi$  by creating a *consensus matrix*  $M$  [5], (also known as an *ensemble co-association matrix* [6]), where  $M_{i,j}$  is the number of times  $\mathbf{x}_i$  was placed in the same cluster as  $\mathbf{x}_j$  with possible weighting of the partitions and normalization of  $M$ . The entries of  $M$  are used as the similarity measurements for a consensus clustering algorithm.

Vector space based clustering algorithms may be used for clustering  $\Pi$  by following the approach in [4] if an appropriate utility function  $U$  can be created. The consensus clustering problem is then expressed as a maximization problem

$$\pi_{cc} = \operatorname{argmax}_{\pi} \sum_0^{r-1} w_i U(\pi, \pi_i)$$

where  $w_i$  is a weight for partition  $\pi_i$ . A binary data set  $\mathcal{X}^b$  was used in [4] to prove the equivalence of k-means clustering over  $\mathcal{X}^b$  and consensus clustering when using appropriate distance functions and their corresponding utility functions. The binary data set  $\mathcal{X}^b$  is a unary encoding of the hypergraph from [2] where the unary encoding is the one typically used for encoding categorical data for input to an artificial neural networks and not the standard unary representation. That is, the unary encoding of a value 2 out of the range 0-3 is represented by  $[0, 0, 1, 0]$  and not 110. A row  $i$  of  $\mathcal{X}^b$  corresponds to the labelling of  $x_i$  in  $\Pi$  with column ‘jk’ equals to 1 if partition  $\pi_j$  has  $x_i$  in subset  $C_k$  else it is zero. Note that different partitions may have different number of labels, so ‘jk’ means the  $k$ th label of the  $j$ th partition. For example, if  $x_i$  has labels 0, 2, and 1 in three partitions with 4, 3, and 2 being the number of possible labels in each partition respectively, then the row  $i$  of  $\mathcal{X}^b$  would be  $\underbrace{[1, 0, 0, 0]}_{\pi_0}, \underbrace{[0, 0, 1]}_{\pi_1}, \underbrace{[0, 1]}_{\pi_2}$ . A consensus clustering  $\pi_{cc}$  can be found by applying k-means clustering to the rows of  $\mathcal{X}^b$ .

### III. $\lambda$ CONSENSUS CLUSTERING

In consensus clustering, no access to the original data  $\mathcal{X}$  is required and no feedback to the  $i$ th clusterer (the process that creates  $\pi_i$ ) is provided. This paper proposes to extended consensus clustering by adding a feedback signal to the clustering algorithms. The entire consensus matrix  $M$  or binary data set  $\mathcal{X}^b$  could be returned, but that would be a significance increase in communication requirements over consensus clustering. What is proposed is the feedback of only the consensus labelling  $\pi_{cc}$ . The communication requirement is on the same order as consensus clustering with each iteration requiring twice the communication of consensus clustering (the labels to and from each clusterer). If communication is not a bottleneck, then  $M$  or  $\mathcal{X}^b$  could be sent back, but experiments show that  $\pi_{cc}$  is an effective approximation or summary of the information in  $M$  and  $\mathcal{X}^b$ .

Since many clustering algorithms stop in a local minimum of their objective function, the extra information from the consensus may allow them to transition to a better local minimum and hence improve the quality of the partitions in the ensemble. An exponential weighting approach, called  $\lambda$ -consensus clustering, is proposed as a method of merging  $\pi_{cc}$  into the similarity or distance function of an individual clustering algorithm. An advantage of modifying the distance or similarity measure is that the individual clustering algorithms can be used with little or no modifications. The parameter  $\lambda$  controls the weighting between the original measure and consensus measure with the contribution of the original data decaying exponentially by  $\lambda^t$  for feedback iteration  $t$ .

For similarity based clusterings with the  $r$ th clusterer’s similarity matrix at iteration  $t$  denoted as  $S_{r,t}$  and receiving the feedback of the consensus labelling  $\pi_{cc,t}$ , the new similarity matrix for the next iteration  $t + 1$  is

$$S_{r,t+1} = \lambda S_{r,t} + (1 - \lambda) M_{\pi_{cc,t}}$$

where  $M_{\pi_{cc,t}}$  is a co-association matrix created from  $\pi_{cc,t}$  (element  $M_{\pi_{cc,t}}(i, j) = 1$  if  $x_i$  and  $x_j$  are in the same cluster in  $\pi_{cc,t}$  else 0 with normalization depending on the clusterer’s algorithm). The initial similarity matrix  $S_{r,0}$  is the  $r$ th clusterer’s original similarity matrix of its view of the data. Note that  $S_{r,t+1}$  may not be positive semi-definite. If the clusterer, such as kernel k-means [7], [8], requires a positive semi-definite matrix then  $S_{r,t+1}$  can be shifted by adding positive values along the diagonal. Since kernel k-means often converges when applied to indefinite matrices and often to a better partitioning than the clustering on shifted positive definite matrix, shifting is only applied with kernel k-means if it fails to converge using  $S_{r,t+1}$ .

For distance based vector space algorithms, such as k-means, the consensus clustering  $\pi_{cc,t}$  needs to update the distances in a manner that the  $r$ th clusterer can use. One proper way of doing this would be to define a distance  $dist_{\pi_{cc,t}}(x_i, x_j)$  as zero or one depending on if  $x_i$  and  $x_j$  are in the same cluster or not. Then create a distance matrix  $D_{r,t+1}(i, j) = \lambda D_{r,t} + (1 - \lambda) dist_{\pi_{cc,t}}(x_i, x_j)$  where  $D_{r,0}$  is a matrix of the pairwise Euclidean distances of the  $r$ th clusterer’s view of  $\mathcal{X}$ . A new Euclidean space can be created from  $D_{r,t+1}$  using Torgerson approach from multidimensional scaling [9], [10] which requires an expensive SVD computation. A computationally cheaper alternative is to extend the data vector with virtual dimensions corresponding to the unary encoding of  $x_i$ ’s label in  $\pi_{cc}$ . For example, if there are four class labels in  $\pi_{cc}$  and  $x_i$ ’s label is 1 then  $[0, 1, 0, 0]$  is appended to the vector. Let  $z_{r,i,t}$  be the  $r$ th clusterer’s view of data  $x_i$  at iteration  $t$  and let  $b_{i,\pi_{cc,t}}$  be the unary encoding of  $x_i$ ’s label from  $\pi_{cc,t}$  then

$$z_{r,i,t+1} = [\lambda z_{r,i,t}, (1 - \lambda) b_{i,\pi_{cc,t}}].$$

This shrinks the magnitude of the current vector and appends the extra dimensions for the label. The vector  $z_{r,i,t}$  grows in dimensions with each iteration  $t$ . Typically, only a few iterations are conducted so the growth may not cause any problems. In case where it may cause a problem,  $z_{r,i,t+1}$  may be approximated by

$$z_{r,i,t+1} \approx [\lambda^t z_{r,0,i}, (1 - \lambda^t) b_{i,\pi_{cc,t}}]$$

which ignores the contributions of the previous consensus clusterings by just using the terms involving the original view of the data and that latest consensus clustering. In the experiment section, this is called a *truncated* update.

#### IV. EXPERIMENTAL RESULTS

In this section we present preliminary experiments to show the efficacy of  $\lambda$ -consensus clustering. We used three real world data sets that are typically used for classification so the class labels can be used to evaluate the quality of the results. The real world datasets include iris and vote from [11] and 14cancer from [12]. For a distance based vector space algorithm, we used the k-means algorithm. For graph or similarity based algorithms, we used pinch ratio clustering (PRC) [13]. Each clusterer used a different random projection of the data to a three dimensional subspace. When comparing clusterings, the adjusted rand index (ARI) [14] is used. For evaluating quality, the ARI is calculated between a clustering and the known class labels. The number of clusters  $k$  was given to each clusterer instead of trying to estimate  $k$ , such as in [15].

The first experiment investigates the truncation approximation of  $z_{r,i,t+1}$  for k-means. The average results of ten runs with 100 k-means clusterers is presented in Figure 1 with  $\lambda=0.9$  for the iris, vote, and 14cancer datasets. As can be seen from the boxplots, there is an improvement from iteration 0 and iteration 1. Standard consensus clustering would provide the results from iteration 0. The final  $\lambda$ -consensus labelling, the red dot at iteration 9, is very similar for the truncated and not truncated cases. In two of the datasets, not truncating  $z_{r,i,t+1}$  causes the distribution of the 100 clusterers to converge faster.

The second experiment investigates the  $M_{\pi_{cc},t}$  approximation of the consensus matrix  $M_t$ . The average results of ten runs with 100 PRC clusterers [13] is presented in Figure 2 for  $\lambda=0.9$  for the iris, vote, and 14cancer datasets. The clusterers used a Gaussian similarity function  $s(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$  over all dimensions of the data. The values of  $\sigma$  were chosen using the heuristics presented in [16] of  $\sigma$  being the average distance to the  $k$ th nearest neighbor using  $k = \log(N) + 1$  where  $N$  is the number of data points. The final  $\lambda$ -consensus labelling using the approximation similar to that using the entire matrix. Note that the medians, represented by green lines, of the last three iterations on iris using  $M_{\pi_{cc},t}$  is higher than the  $\lambda$ -consensus result. The diversity of the clusterers over the vote dataset was so small that consensus clustering provided no benefit.

#### V. SUMMARY

With the goals of improving quality and managing diversity of consensus clustering, this paper proposed  $\lambda$ -consensus clustering which extends the consensus clustering framework with a feedback loop of consensus labels to the individual clustering processes. The individual clusterings merge the consensus knowledge with its view of the data using an exponential weighting scheme for another round of clustering. The extension maintains the low communication and privacy preservation benefits consensus

clustering. Initial experimental results demonstrate that the approximations to the consensus information preform as well as using the entire consensus information.

#### REFERENCES

- [1] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010. [Online]. Available: <http://linkinghub.elsevier.com/retrieve/pii/S0167865509002323>
- [2] A. Strehl and J. Ghosh, "Cluster ensembles—a knowledge reuse framework for combining multiple partitions," *The Journal of Machine Learning Research*, vol. 3, pp. 583–617, 2002.
- [3] Y. Senbabaoglu, G. Michailidis, and J. Z. Li, "Critical limitations of consensus clustering in class discovery," *Scientific reports*, vol. 4, 2014.
- [4] J. Wu, H. Liu, H. Xiong, J. Cao, and J. Chen, "K-means-based consensus clustering: A unified view," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 27, no. 1, pp. 155–169, Jan 2015.
- [5] C. Meyer, S. Race, and K. Valakuzhy, "Determining the number of clusters via iterative consensus clustering," in *2013 SIAM Proceedings of the International Conference on Data Mining (SDM13)*, Austin Texas, May 2013, pp. 94–102.
- [6] J. Ghosh and A. Acharya, "Cluster ensembles," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 1, no. 4, pp. 305–315, 2011.
- [7] I. S. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means: spectral clustering and normalized cuts," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2004, pp. 551–556.
- [8] —, "Weighted graph cuts without eigenvectors a multilevel approach," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 11, pp. 1944–1957, 2007.
- [9] W. S. Torgerson, "Multidimensional scaling: I. theory and method," *Psychometrika*, vol. 17, no. 4, pp. 401–419, 1952.
- [10] J. B. Kruskal, "Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis," *Psychometrika*, vol. 29, no. 1, pp. 1–27, 1964.
- [11] A. Frank and A. Asuncion, "UCI machine learning repository," 2010. [Online]. Available: <http://archive.ics.uci.edu/ml>
- [12] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, 2nd ed. Springer Verlag, 2008. [Online]. Available: <http://http://www-stat.stanford.edu/~tibs/ElemStatLearn>
- [13] D. R. Heisterkamp and J. Johnson, "Pinch ratio clustering from a topologically intrinsic lexicographic ordering," in *2013 SIAM Proceedings of the International Conference on Data Mining (SDM13)*, Austin Texas, May 2013, pp. 560–568.
- [14] L. Hubert and P. Arabie, "Comparing partitions," *Journal of Classification*, vol. 2, no. 1, pp. 193–218, 1985.

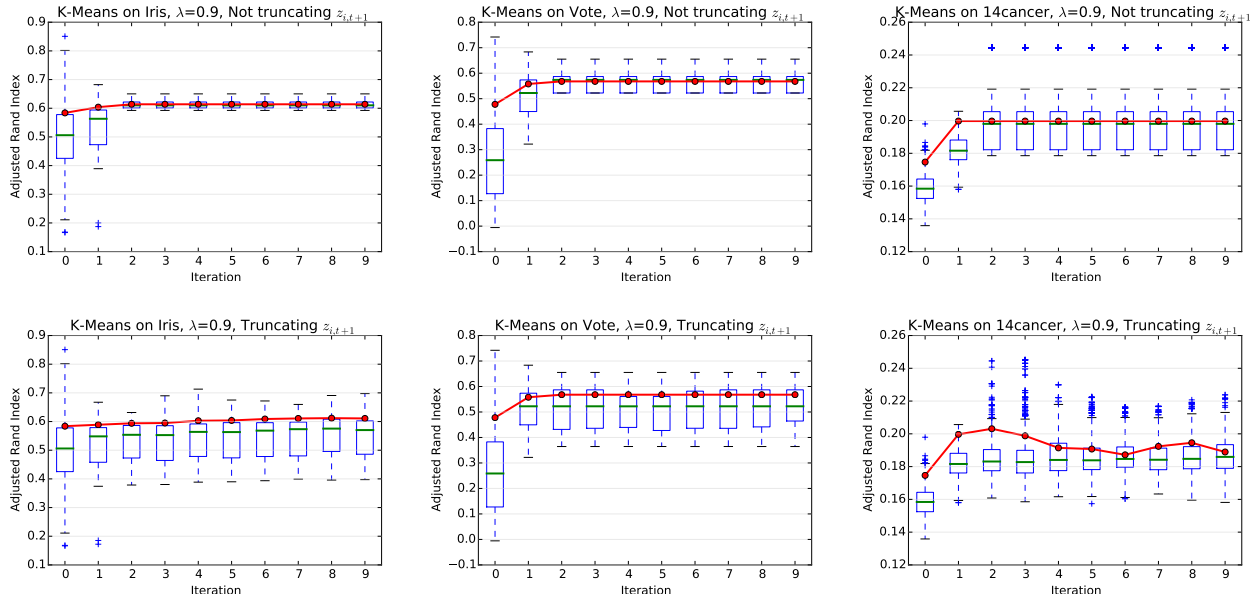


Figure 1.  $\lambda$ -consensus clustering with 100 clusterers using k-means on subspace projections, consensus clustering with k-means on  $\mathcal{X}^b$ , and using  $\lambda=0.9$ . Results are averaged over 10 runs. ARI of partitions are measured against class labels. The red line is the partitioning from that iteration's consensus clustering.

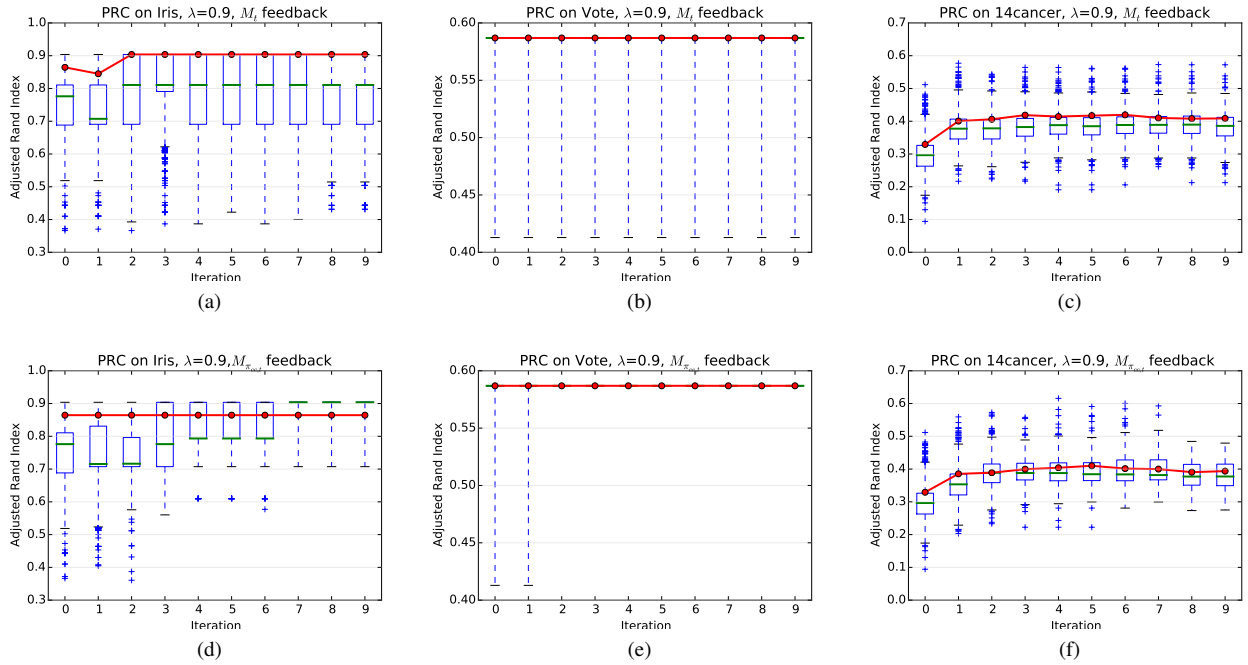


Figure 2.  $\lambda$ -consensus clustering with 100 clusterers using PRC, consensus clustering with PRC, and using  $\lambda=0.9$ . The clusterers all used the same Gaussian similarity. Results are averaged over 10 runs. ARI of partitions are measured against class labels. The red line is the partitioning from that iteration's consensus clustering. The green bars are the median ARI value of the 100 clusterers.

[15] M. R. Daliri and V. Torre, "Shape and texture clustering: Best estimate for the clusters number," *Image Vision Comput.*, vol. 27, no. 10, pp. 1603–1614, Sep. 2009.

[16] U. Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, Dec. 2007.