Incremental Cluster Updating Using Gaussian Mixture Model

Elnaz Bigdeli[1(✉)], Mahdi Mohammadi[2], Bijan Raahemi[2], and Stan Matwin[3,4]

[1] School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, ON, Canada
ebigd008@uottawa.ca
[2] Knowledge Discovery and Data Mining Lab, Telfer School of Management, University of Ottawa, Ottawa, ON, Canada
{mmohamm6,braahemi}@uottawa.ca
[3] Department of Computing, Dalhousie University, Halifax, NS, Canada
[4] Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland
stan@cs.dal.ca

Abstract. In this paper, we present a new approach for updating clusters incrementally. The proposed incremental approach preserves comprehensive statistical information of the clusters in form of Gaussian Mixture Models (GMM). As each GMM needs the number of Gaussian (component) as an input parameter, we proposed a method to determine the number of components automatically with introducing the concept of core points. In the updating phase, instead of processing each new sample individually, we collect the new incoming samples and cluster them. By employing the concepts of core points and GMMs, we build a number of GMMs for the new samples and we label the new GMMs based on their similarity to the already existing GMMs. To find the similarity among GMMs, we introduce a new modified version of Kullback-Leibler as a distance function. For merging the current GMMs and the new GMMs, we proposed a new merging mechanism in which the closest components in both GMMs are merged to create a new GMM. Since GMM structure is a compact representation of clusters, there is no increase in the time neither in clustering side nor in updating phase. We measured the accuracy of clusters based on different clustering validity metrics (DB, Dunn, SD and purity) and the results show that our algorithm outperforms other incremental clustering algorithms in terms of quality of the final clusters.

Keywords: Incremental clustering · Gaussian Mixture Model · Stream data clustering

1 Introduction

Nowadays, numerous applications ranging from business and market analysis to computer networks deal with overwhelming quantities of data. The demand to work with large and fast incoming data creates a new trend to tailor the traditional algorithms to the online environment. Additionally, the main part of the data is unlabelled this is why the clustering algorithms receive attention to deal with the unlabeled data [1] [2].

© Springer International Publishing Switzerland 2015
D. Barbosa and E. Milios (Eds.): Canadian AI 2015, LNAI 9091, pp. 264–272, 2015.
DOI: 10.1007/978-3-319-18356-5_23

Since in online environment the behavior of data changes by passing time, there is a great need to propose algorithms which are able to update themselves with new trends in data behavior. The primitive way to update clusters based on the new incoming data is to add the new data to the existence data and then apply the clustering algorithm on the entire data. Obviously, this approach is not practical in many online applications. To solve this problem incremental clustering algorithms are suggested in which the clustering algorithms keep a history of data and then add the new data to the current data in an incremental way. Incremental clustering algorithms need to have specific characteristics. First, the incoming sample has to be processed quickly in order to find the closest cluster. Second, the clusters have to be updated very fast to be adapted to the new changes in data. Third, the clusters have to be compact and well separated over the time, and they should not be grown fast by every incoming sample. Another important feature, which needs to be considered, is to make the incrementally generated clusters as close as possible to the clusters generated using the whole data.

Incremental clustering algorithms such as STREAM [3] and Clustream [4] update clusters efficiently in terms of memory and time complexity. In these algorithms, finding the closest cluster to the new incoming sample is based on finding the distance of the new sample to the clusters centers. There is another trend in incremental clustering that estimates the entire data with one GMM in which each component is a representative of a cluster. Considering a single GMM for the whole data has its own problems. The most important problem is that the model is complex and updating step involves the whole samples in the dataset. To conquer this problem, in this paper, we employ the advantages of both trends in incremental clustering algorithms. In our proposed approach, instead of finding a single GMM for the whole data, first, the data is clustered and then each cluster is represented using a GMM. It has two advantages: first, the GMM is able to generate the original data with a good accuracy; second, the GMM formula finds the closest cluster to the new sample with a simple calculation. From updating perspective, in our approach instead of updating clusters based on each new incoming sample, the new coming samples are collected, and a group of samples are used to update GMMs. In this way, updating phase is less sensitive to noise. Moreover, since the updating is happening offline, it is able to deal with a huge amount of data in the online applications.

The rest of the paper is organized as follows: In Section 2 a review of incremental clustering algorithms is presented. The general architecture of our GMM-based incremental clustering algorithm is presented in Section 3. The clustering approach used in the paper is discussed in Section 4. The new algorithm for creating GMM is presented in Section 5. GMM-based updating approach is introduced in Section 6. The experimental results and the all the comparisons with all other algorithms are presented in Section 7. The conclusion and future work of the paper are presented in Section 8.

2 Related Work

There are different approaches for clustering data in online applications. The first well-known algorithm in online clustering is BIRCH [5] that was introduced by Zhang et al. The BIRCH algorithm creates a hierarchical structure of clusters in the form of a tree that makes this algorithm fast for searching purposes. In spite of being fast, this algorithm has its own disadvantages. First, creating a hierarchy of data is usually a difficult

task which involves many parameter setting, plus the fact that updating a tree structure is not a straightforward task. Moreover, BIRCH employs the idea of the centers and a radius to create a cluster which makes it inappropriate for non-convex clusters. In addition, the center-based approaches are sensitive to noise and also they have low accuracy in clustering new samples. STREAM [3] is an incremental algorithm that uses the idea of preserving weighted medians over the time. In this algorithm in each step the LSEARCH algorithm finds the clusters medians. Every new incoming data is added to the current medians and LSEARCH is applied to find the new medians. To have an accurate clustering, a large number of medians need to be preserved which is not appealing in the online environments. Clustream [4] is an incremental algorithm that combines the idea of BIRCH and STREAM in a framework. In pyramidal time steps, the clusters are updated, and the current clusters are replaced by new ones. Clustream still carries the problems related to BIRCH and STREAM.

In the area of density-based clustering algorithms, DenStream [6] is used to make DBSCAN an incremental clustering algorithm. The main algorithm for initialization and updating stages is DBSCAN. Each new coming sample is assigned to a cluster with distance less than a radius to a core micro cluster. Dealing with the whole dataset makes the algorithm slow and unpleasant for online applications. Grid-based algorithms are also changed to be used in incremental and online environments [7]. The idea is pretty much similar to DenStream while finding dense regions and connecting them is applied to the grid that has its own problems. Incremental Gaussian Mixture Model algorithms are another group of incremental clustering algorithms that consider all data is generated based on a Gaussian Mixture Model [8]. In this group, a single GMM is found for the entire dataset. After generation of the first GMM, each new sample is fed to the current GMM and the GMM is updated based on the new sample [9] [10] [11]. Our approach is a combination of both GMM-based incremental clustering and traditional incremental clustering algorithms, and it also introduces a new structure which is fast and accurate for online data clustering and is explained in next section.

3 The Proposed Incremental Cluster Updating

In this section, we present the proposed method CUGMM (Cluster Updating based on GMM) in more details. Our approach has three main phases which are shown in Figure 1. As shown in this figure, the input dataset is available and a clustering algorithm is applied to partition the dataset into some clusters (cluster creation phase is explained in Section 4).

In the second phase, for each cluster, a GMM is employed to estimate the distribution and shape of the cluster. In this paper, we modify the standard GMM estimation phase in order to improve the time and memory complexity of the previous methods. The proposed approach is described in Section 5. In updating phase, instead of updating the current clusters using each new sample, we collect the incoming samples and after collecting a specific number of samples, we cluster the collected samples into some clusters. Then we apply the same procedure on the new clusters and represent each cluster by a GMM as shown in Figure 1. In this step, we compare the newly generated clusters with the exciting ones. If the new GMM is close enough to one of the existing cluster GMMs, they are merged and they create an updated cluster; otherwise, it is either added as a new cluster or deleted. In the following, we discussed each step in more detail.

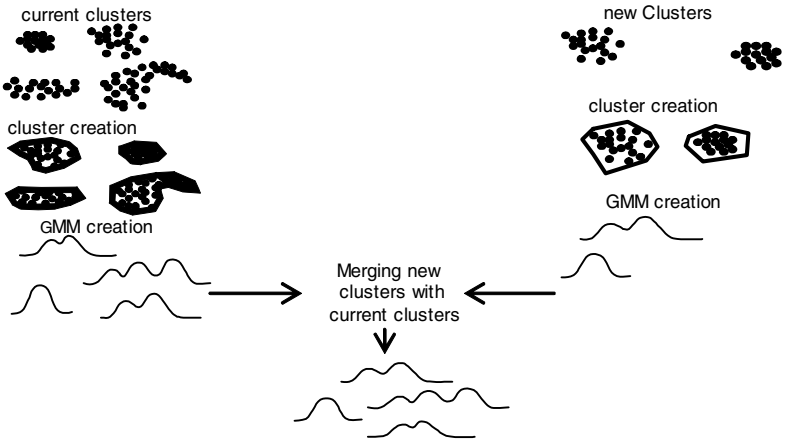

Fig. 1. The General Structure of Incremental cluster updating

4 Creating Clusters

In each dataset, based on the requirements of an application and nature of data, different kinds of clustering algorithms are applied. In the proposed model, any kind of clustering algorithms can be employed. This means that the proposed algorithm is able to work with the clusters produced by any clustering algorithm. To preserve the shape of a cluster either the majority of samples in each cluster should be stored or the boundary of each cluster must be detected. Either way, the memory complexity for these methods is not negligible. Moreover, we need to use a technique that not only preserves the shape of the cluster, but also retains all valuable information for that cluster to use for further investigation of clusters. Therefore, we proposed our GMM-based approach called SGMM [2] to summarize the clusters as a GMM. In the next section, we explain the proposed SGMM algorithm in details.

5 Creating Gaussian Mixture Model

Summarization based on Gaussian Mixture Model (SGMM), has three main steps to represent each cluster as a GMM. First, a set of objects called core objects are detected. These objects are representative of a cluster, and they are able to generate the original cluster as needed. After detecting the backbone objects, there comes the absorption step, where the attached objects to the core point are absorbed and represented by the core object. Then by introducing a new feature set for each core point, the cluster is summarized while its original distribution is preserved using a GMM.

-*Finding core objects:* In this phase, a radius is considered as an input parameter for the algorithm. The radius is employed to find dense regions and the core points which are the center of these dense regions. Every point with a greater number of neighbors in comparison with other objects is a good candidate to be a final core point.

-Absorption and cluster feature extraction: Each core object is represented by a triple $CF_i = \langle c_i, \Sigma_i, \omega_i \rangle$. In this triple c_i is the core point and Σ_i is the covariance calculated using the core point as the center and all samples in its neighbourhood. $\omega_i = n/CS$ is the weight of core point, n is the number of objects in the neighbourhood of core point c_i , and CS is the cluster size.
-GMM representation: After finding the core objects, we generate a Gaussian Mixture Model for each cluster. Given feature space, $f \subset R^d$ a Gaussian Mixture Model $g: f \rightarrow R$ with n components is defined as:

$$g(x) = \sum_{i=1}^{n} w_i \, N_{\mu_i \Sigma_i}(x); \; N_{\mu_i \Sigma_i}(x) = \frac{1}{\sqrt{(2\pi)^d |\Sigma_i|}} e^{-\frac{1}{2}(x-\mu_i)\Sigma_i^{-1}(x-\mu_i)^T} \tag{1}$$

6 Updating Cluster

In comparison to the previous online clustering algorithms that update the clusters based on each new sample, our algorithm updates clusters in a batch way. To update the clusters, first, the new incoming samples are collected, and some new clusters are generated and by GMMs. The updating procedure consists of two main steps; finding two closest clusters and then merging the close clusters.

6.1 Finding Two Closest GMMs

To update clusters, first we need to find the closest clusters. Based on the definition of GMM in Equation (1), the Kullback-Leibler [12] distance for two normal distributions is:

$$KL(N_0|N_1) = \frac{1}{2}(tr(\Sigma_1^{-1}\Sigma_0) + (\mu_1 - \mu_0)^T \Sigma_1^{-1}(\mu_1 - \mu_0) - k - ln\left(\frac{\det(\Sigma_0)}{\det(\Sigma_0)}\right) \tag{2}$$

This distance is used to measure the distance of two normal distributions not two mixtures of the normal distributions. To propose a symmetric distance, we need to find both $kl(N^b|N^a)$ and $kl(N^a|N^b)$, and then take the average. In our proposed distance measure, we first find the distance of first GMM components to the second GMM components using weights of the first GMM. Then, we find the distance of second GMM's components to the first GMM's components using weights of the second GMM.

$$D(G_b|G_a) = \sum_{i=1}^{n}\sum_{j=1}^{m} w_i^b \, kl(N_i^b|N_j^a) \qquad D(G_a|G_b) = \sum_{i=1}^{m}\sum_{j=1}^{n} w_i^a \, kl(N_i^a|N_j^b) \qquad D_{cluster} = \frac{(D(G_a|G_b)+D(G_b|G_a))}{2} \tag{3}$$

6.2 Merging Two GMMs

After finding the two closest GMMs, the next step is to merge them. According to our proposed approach, the new data samples $\{x_1,\ldots,x_m\}$ are clustered to clusters $\{c_1,\ldots,c_k\}$. SGMM algorithm finds all equivalent GMMs; $\{G_1,..,G_k\}$, for clusters $\{c_1,\ldots,c_k\}$. Each GMM consists of a set of components $G_i = \{g_i^1, \cdots, g_i^s\}$ where s is the number of GMM components. Each GMM component is represented by $g_i^j = \{\mu_i^j, \Sigma_i^j, \omega_i^j\}$. Considering that G_a and G_b are two close GMMs. In G_a the i_{th} component that is g_a^i is close to j_{th} component of G_b that is g_b^j. Therefore, the new GMM component is created based on merging $g_a^i = \{\mu_a^i, \Sigma_a^i, N_a^i\}$ and $g_b^j = \{\mu_b^j, \Sigma_b^j, N_b^j\}$.

$$\mu_{new} = \frac{N_a^i \mu_a^i + N_b^j \mu_b^j}{N_a^i + N_b^j} \quad (4)$$

$$\Pi = \frac{N_a^i + N_b^j}{\sum_{i=1}^{s} N_a^i + \sum_{j=1}^{l} N_b^j} \quad (5)$$

$$\Sigma_{new} = \frac{N_a^i \Sigma_a^i + N_b^j \Sigma_b^j}{N_a^i + N_b^j} + \frac{N_a^i \mu_a^i {\mu_a^i}^T + N_b^j \mu_b^j {\mu_b^j}^T}{N_a^i + N_b^j} - \mu_{new} {\mu_{new}}^T \quad (6)$$

6.3 Discussion on the Efficiency of the Proposed Method

Summarization is the main feature of the proposed method. Instead of keeping the entire samples in each cluster for preserving the arbitrary shape of the cluster, the proposed method summarizes each cluster into a set of Gaussian models. Each Gaussian model contains a mean, a variance, and a weight. In terms of memory complexity, summarizing the clusters with Gaussian mixtures are more beneficial than holding the entire samples in each cluster. It can estimate the shape of the clusters as well as distribution of samples in each cluster by estimating each cluster with a set of Gaussian models. In arbitrary shape clustering algorithms which keep the entire samples, to label the new incoming sample, the distances between the new test samples and entire samples in the cluster (or at least a part of the samples) have to be calculated. In the proposed method, a few number of core points are kept based on which a GMM is estimated on each cluster. Summarizing the clusters with a set of core points and estimating the characteristics of the clusters by the GMMs benefits the proposed method in a noisy environment. If the updating process is triggered whenever a new sample is introduced to the model, the noisy samples may bias or change the attributes of the clusters based on their noisy behavior. It may end to inaccurate final clusters which do not follow the real behavior of the clean data. But the proposed method collects the incoming samples, and then group them into some clusters and consider the extracted clusters to represent the new samples. After clustering the new samples, the outliers and the clusters which have the number of the samples less than a given threshold are removed from the data. It prevents the final clusters to be persuaded by the input noisy data.

In some cases, the number of noisy samples, which are placed in the same neighborhood, is enough to shape a component of a GMM. In this case, the noisy component should be close to any other clean components in other clusters. If the algorithm decides to combine two GMMs (one of the GMMs consists of the noisy component), there will be two approaches for the algorithm. First, while the noisy component is not close enough to any components in the other GMM and the number of samples in the component is less that a specific value, the proposed method eliminates the com-

ponent from its GMM. In the second approach, if the noisy component is not close to any component in the other GMM but the number of samples is not small enough to be eliminated, the noisy component is departed from the GMM and forms a new GMM (cluster) which has one component representing the whole samples in the new cluster. If the newly formed cluster is not really a noisy cluster, by adding new samples, it may have the chance to find the clusters close enough to be merged and shape a new cluster, otherwise it may not have a chance to be merged with other clusters.

7 Experimental Results

In this part, we evaluate the proposed method based on different criteria consisting of five different metrics as Dunn [13], DB [14], SD, SSQ [4], and Purity [4]. In this paper, we first divide the input dataset into 6 parts and pick the first part to generate the initial clusters and the rest is added to the existing clusters incrementally in different rounds (iterations). That is to say, in each round we introduce a new part of data to the previously seen data to examine the learning of the proposed method. Some of the UCI repository datasets are used in our experiments, as well as a 2D synthetic dataset to visualize the final clustering results. The datasets, which are chosen from UCI repository, are KDDCup99, MagicGamma. Figure (2) shows the result of SSQ for KDDCup99 dataset for CUGMM and Clustream [3] and Clustering the Whole data in each Round (CWR).

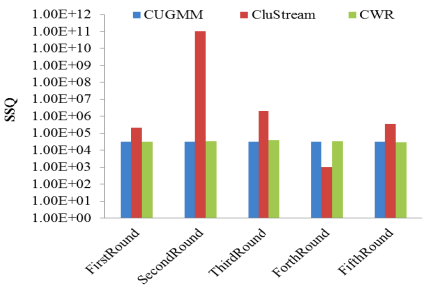

Fig. 2. SSQ for KDD dataset

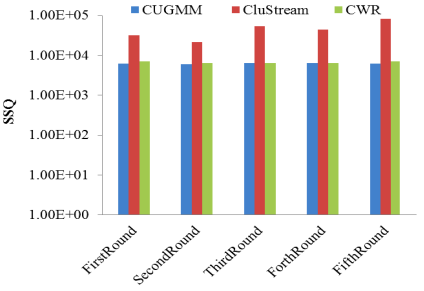

Fig. 3. SSQ for MagicGamma dataset

In CWR, in the first step, we cluster the first part of data using K-means algorithm. In second round, we add the second part of the data into the existing one and apply the K-means algorithm on the whole data. That is to say, in each round we forget about the previous clusters and build the new clusters based on the whole data received by the current round. This means that there is no incremental learning process in CWR that is why we expect the CWR outperforms other algorithms in most cases. But the main point is that we can measure how far we are from the CWR by introducing the CUGMM, which works in an incremental way. These results indicate the accuracy of clusters by considering each cluster as a Gaussian Mixture Model. In the Clustream, the clustering is based on considering core clusters and the distance of the

new sample to the center of the cluster. As shown in the Figure 3, CUGMM shows better performance in clustering the new incoming samples incrementally since the results is very close that of CWR while CWR considers the whole samples in each round in clustering and CUGMM updates the clusters incrementally. Figure (3) shows the same result on MagicGamma dataset which confirms that the CUGMM works better than Clustream algorithm, and it almost ties with the CWR method. For example, in the first round the CUGMM SSQ is 6221 in comparison to SSQ 32210 for Clustream and SSQ 7100 for the basic model. The lower error rate shows that the approach does not lose any accuracy in terms of clustering over the time. In next experiment, we evaluate the proposed method using Dunn, DB, SD, and Purity. For DB and SD the lowest value shows the best performance while for Dunn and Purity the highest is the best (. 1). For KDD dataset in each round the CUGMM algorithm has a better result than Clustream, and it is close to the results of CWR. For DB index, the CUGMM shows a stable performance during the whole rounds while Clustream and CWR fluctuate between [0.4-0.8] and [0.2-0.7] respectively. In terms of SD, most of the time the CUGMM shows better results than that of Clustream and also CUGMM is very close to CWR. It shows the proposed method (CUGMM) outperforms the Clustream, and it is able to generate the same cluster as CWR. The result for DB index shows that the generated clusters are good in all rounds. The result for SD confirms the previous results. Result shows that with CUGMM algorithm, the purity inside clusters based on label of the data is higher than that of Clustream, and it is close to the CWR.

Table 1. Performance of clustering based on Dunn, DB, SD, Purity indexes

Clustering index		Dunn			DB			SD			Purity		
DataSet/ Algorithm		CUGMM	Clu-stream	CWR	CUGMM	Clu-stream	CWR	CUGMM	Clu-stream	CWR	CUGMM	Clu-stream	CWR
KDD	R1	1e-6	2e-8	0.0003	0.613	0.81	0.35	2.01	3.2	2.22	0.76	0.63	0.79
	R4	0.00025	1e-6	0.0006	0.634	0.71	0.69	2.3	2.27	2.27	0.74	0.66	0.73
Shuttle	R1	7e-6	2e-8	3e-05	0.332	0.49	0.2	6.35	7.15	6.42	0.96	0.81	0.95
	R4	9 e-7	2e-8	2e-05	0.363	0.59	0.29	3.88	4.98	3.98	0.96	0.85	0.95
Magicc Gamma	R1	0.00489	0.0001	0.0222	0.777	0.87	0.66	1.89	2.89	1.9	0.76	0.64	0.73
	R4	0.01960	0.0039	0.0339	0.727	0.94	0.7	1.89	2.71	1.9	0.75	0.64	0.75
Synthetic	R1	2e-09	1e-011	1e-05	0.40	0.45	0.35	2.25	3.25	2.08	0.89	0.79	0.91
	R4	2e-07	4e-09	6. e-06	0.38	0.68	0.28	2.33	4.47	1.77	0.9	0.81	0.9

8 Conclusions

This paper presents a new approach to update clusters based on the new incoming samples. In our approach, all clusters are represented using a GMM that keeps a summary of each cluster and is able to be updated very fast and more accurate. In-

stead of processing each new incoming sample individually, we first collect the samples and then group them into some clusters. For each cluster a GMM is defined to estimate the samples scattering and distribution. In updating phase, we proposed a modified Kullback-Leibler distance measure to find the closest cluster to the new clusters. After finding a pair of close clusters from current and new clusters, their corresponding GMMs are merged. We introduce a merging strategy which enables the algorithm to detect noisy samples and remove them from the rest of the process. We evaluate the proposed method in comparison with two other clustering algorithms, CWR and Clustream based on different cluster validity indices. The result shows that our updating approach tends to follow the original shape of clusters during time. The result on different dataset based on different factors shows that updating clusters using GMMs is more accurate than Clustream.

References

1. Mohammadi, M., Akbari, A., Raahemi, B., Nasersharif, B., Asgharian, H.: A fast anomaly detection system using probabilistic artificial immune algorithm capable of learning new attacks. Evolutionary Intelligence **6**(3), 135–156 (2014)
2. Bigdeli, E., Mohammadi, M., Raahemi, B., Matwin, S.: Arbitrary shape cluster summarization with Gaussian Mixture Model. In: KDIR, Roma (2014)
3. O'Callaghan, L., Mishra, N., Meyerson, A., Guha, S., Motwani, R.: Streaming-Data Algorithms for High-Quality Clustering. IEEE Conference on Data Engineering (2001)
4. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for clustering evolving data streams. In: VLDB 2003 (2003)
5. Zhang, T., Ramakrishnan, R., Livny, M.: BIRCH: A New Data Clustering Algorithm and Its Applications. Data Mining and Knowledge Discovery **1**, 141–182 (1997)
6. Cao, F., Ester, M., Qian, W., Zhou, A.: Density-based clustering over an evolving data stream with noise. In: SIAM Conference on Data Mining (2006)
7. Chen, Y., Tu, L.: Density-based clustering for real time stream data. In: ACM SIGKDD (2007)
8. Hajji, H.: Statistical Analysis of Network Traffic forAdaptive Faults Detection. Transactions on Neural Networks, 16(5) (2005)
9. Song, M., Wang, H.: Highly efficient incremental estimation of Gaussian Mixture Models for online data stream clustering. In: SPIE Conference on Intelligent Computing: Theory And Applications III, Orlando (2005)
10. Declercq, A., Piater, J.H.: Online learning of Gaussian Mixture Models: a two-level approach. In: Third International Conference on Computer Vision Theory and Applications (2008)
11. Hennig, C.: Methods for merging Gaussian mixture components. Advances in Data Analysis and Classification **4**(1), 3–34 (2010)
12. Kullback, S., Leibler, R.A.: On Information and Sufficiency. Annals of Mathematical Statistics, 79–86 (1951)
13. Dunn, K., Dunn, J.: Well separated clusters and optimal fuzzy partitions. Cybernetics **4**, 95–104 (1997)
14. Davies, L.D., Bouldin, W.D.: A cluster separation measure. IEEE Trans. Pattern Anal. Machine Intell **1**(4), 224–227 (1979)