Split-and-merge Tweak in Cross Entropy Clustering

Krzysztof Hajto, Konrad Kamieniecki, Krzysztof Misztal, and Przemysław Spurek

Faculty of Mathematics and Computer Science,
Jagiellonian University,
Łojasiewicza 6, 30-348 Kraków, Poland
krzysztof.hajto@ii.uj.edu.pl, konrad.kamieniecki@alumni.uj.edu.pl,
krzysztof.misztal@ii.uj.edu.pl, przemyslaw.spurek@ii.uj.edu.pl

Abstract. In order to solve the local convergence problem of the Cross Entropy Clustering algorithm, a split-and-merge operation is introduced to escape from local minima and reach a better solution. We describe the theoretical aspects of the method in a limited space, present a few strategies of tweaking the clustering algorithm and compare them with existing solutions. The experiments show that the presented approach increases flexibility and effectiveness of the whole algorithm.

Keywords: cross entropy clustering, clusters splitting, clusters merging

1 Introduction

Clustering plays a basic role in many parts of data engineering, pattern recognition, data mining, data quantization and image analysis [6, 5, 15]. Some of the most important clustering algorithms are based on density estimation.

In the probabilistic model construction for univariate and multivariate data, finite mixture models have been widely used. The capability of representing arbitrary complex probability density functions (pdfs) enables it to have many applications not only in unsupervised learning [8], but also in (Bayesian) supervised learning or in parameter estimation of class-conditional pdfs [4].

One of the most important clustering method is based on GMM (Gaussian Mixture Models), which uses the Expectation maximization (EM) algorithm. Unfortunately, GMM has strong limitations related to its optimization procedure, which has to be applied in each iteration of the EM algorithm. While the expectation step is relatively simple, the maximization step usually needs complicated numerical optimization [2, 9]. Because of its greedy nature, the EM algorithm is sensitive to the initial configuration and usually gets stuck at local maxima. Moreover, there is a problem with choosing the correct number of clusters.

A feasible way for solving this problem is to choose several sets of initial values, then proceed respectively with the EM algorithms, and finally choose the best outcome set as the estimation. In most cases, the Bayesian information

criterion (BIC) is used to establish the best result and final number of clusters. However, this will certainly increase computational complexity, since we have to apply a method many times with different initial parameters.

In order to solve this problem, many various methods were introduced. In [14] the authors proposed a split-and-merge EM (SMEM) algorithm in which they applied a split-and-merge operation to the EM algorithm. The basic idea of the SMEM algorithm is: after the convergence of the usual EM algorithm, we first use the split-and-merge operation to update the values of some parameters among all the parameters, then we perform the next round of the usual EM algorithm, and alternatively iterate the split-and-merge operation and the EM algorithm until some criterion is met. However, the split or merge method is a linear heuristic procedure without theoretical support. Moreover, the split or merge of the mean vector is independent of the covariance matrix, and vice versa.

In [17] authors propose two split methods based on SVD and the Cholesky decomposition of the covariance matrices. Shoham presented a robust clustering algorithm by creating a deterministic agglomeration EM (DAGEM) with multivariate t-distributions [11]. It was derived from the DAEM algorithm and achieved encouraging performance. Because the initial component number is much larger than the true number, the computation load is one to two orders of magnitude heavier than EM [11]. In [16] authors present Competitive EM (CEM) which uses an information theory based criterion for split and merge operations. The initial component number and model parameters can be set arbitrarily and the split and merge operation can be selected efficiently. In [7] the authors present a method that uses two different split and merge criteria. Homogeneity criterion decides whether two clusters should be merged or not (clusters which touch each other or slightly overlap can be merged, if they fulfill the homogeneity criterion). The split criterion is based on a penalized Bayesian information criterion (BIC), evaluated for the actual clusters and hypothetically split clusters, updated in the previous incremental learning step.

In all of the methods the basic idea is to construct a split merge strategy by analyzing cluster shapes. The idea is to avoid a local minima by applying some unconventional operation. The main problem is that such an operation does not depend on the cost function, that is minimized by the EM algorithms. Moreover, after the split or merge operation it is non trivial how to update the parameters of the components, since each point belongs to all cluster with different probability.

In this paper we present a split and merge strategy which solves these two basic problems. First of all, a simpler optimization procedure Cross Entropy Clustering (CEC) [13] is used instead of EM. The goal of CEC is to optimally approximate the scatter of a data set $X \subset \mathbb{R}^d$ by a function which is a small modification of EM (for more information see Section 2). It occurs that at the small cost of having a minimally worse density approximation [13], we gain an efficient method which can be easily adapted for more complicated density models. Moreover, we can treat clusters separately which allows us to update the parameters of clusters more easily(each point belongs to only one group).

Furthermore, we can treat each cluster as a new dataset, that is separated from other data, and apply the CEC algorithm in that single cluster. The new division of the cluster is accepted if the global value of the cost function is lower. Similarly, we can verify if the merge of two clusters decreases the cost function.

Let us discuss the contents of the paper. In the first part of our work we briefly describe the CEC algorithm together with the basic structures which we can use in model construction. Then we describe the split-and-merge tweak in detail. At the end of this paper we present results of numerical experiments and conclusions.

2 Split-and-merge Cross Entropy Clustering

In this section the Split-and-merge Cross Entropy Clustering method will be presented. Our method is based on the CEC approach. Therefore, we start with a short introduction to the method. Since CEC is similar to EM in many aspects, let us first recall that, in general, EM aims to find $p_1,\ldots,p_k \geq 0$ ($\sum_{i=1}^{k} p_i = 1$) and $f_1,\ldots,f_k$ Gaussian densities (where k is given beforehand and denotes the number of densities for which the convex combination builds the desired density model) such that the convex combination $f = p_1 f_1 + \ldots + p_k f_k$ optimally approximates the scatter of our data X with respect to the MLE cost function

$$\mathrm{MLE}(f,X) = -\sum_{x\in X} \ln(p_1 f_1(x) + \ldots + p_k f_k(x)). \tag{1}$$

A goal of CEC is to minimize the cost function, which is a minor modification of the one given in (1) by substituting the sum with the maximum:

$$\mathrm{CEC}(f,X) = -\sum_{x\in X} \ln(\max(p_1 f_1(x),\ldots,p_k f_k(x))). \tag{2}$$

Instead of focusing on the density estimation as its main task, CEC aims at solving the clustering problem directly. As it turns out, at the small cost of having a minimally worse density approximation [13], we gain speed in implementation[1] and the ease of using less complicated density models. This is an advantage, roughly speaking, because the models do not mix with each other since we take the maximum instead of the sum.

To explain cross entropy clustering (CEC), we need to first introduce an energy function for the purpose of minimizing, which uses cross entropy. But let's start with the definition of cross entropy itself.

By the cross-entropy of the dataset $X \subset \mathbb{R}^d$ with respect to density f we understand

$$H^{\times}\big(X\|f\big) = -\frac{1}{|X|}\sum_{x\in X} \ln f(x).$$

[1] We can often use the Hartigan approach to clustering, which is faster and typically finds better minimas.

Cross entropy corresponds to the theoretical code-length of compression. Let us consider the case of partitioning $X \subset \mathbb{R}^N$ into pairwise disjoint sets $X_1, \ldots, X_k$, such that elements of X_i are encoded by the optimal density from family $\mathcal{F}$. In this case, the cross-entropy with respect to a family of coding densities $\mathcal{F}$ is given by $H^{\times}\big(X\|\mathcal{F}\big) = \inf_{f\in\mathcal{F}} H^{\times}\big(X\|f\big)$. Thus, the mean code-length of a randomly chosen element x equals

$$E(X_1, \ldots; X_k, \mathcal{F}) := \sum_{i=1}^{k} p_i \cdot (-\ln(p_i) + H^{\times}\big(X_i\|\mathcal{F}_i\big)), \tag{3}$$

where $p_i = \frac{|X_i|}{|X|}$.

The aim of CEC is to find a partitioning of $X \subset \mathbb{R}^N$ into pairwise disjoint sets X_i, $i = 1, \ldots, k$ which minimizes the function given by (3). The minimization of (3) is equivalent to optimization of (2). In our case we consider as a $\mathcal{F}$ a family of all Gaussian distributions[2] $\mathcal{G}$. According to [13], for single piece $X_i \subset \mathbb{R}^N$ considered with respect to $\mathcal{N}(\mu, \Sigma) \in \mathcal{G}$ we can get that

$$H^{\times}\big(X_i\|\mathcal{N}(\mu, \Sigma)\big) = \frac{N}{2}\ln(2\pi) + \frac{1}{2}\mathrm{tr}(\Sigma^{-1}\Sigma_{X_i}) + \frac{1}{2}\ln\det(\Sigma), \tag{4}$$

where Σ_{X_i} is a covariance matrix of X_i, which allows as to easily calculate the cost function (3).

Let us now briefly introduce the algorithm step by step. The CEC clustering method starts from an initial clustering, which can be obtained randomly or by the use of the k-means++ [1] approach. Then the following two simple steps are applied simultaneously. First, we estimate the parameters of the optimal Gaussian function in each cluster. In the second step, we construct a new division of X by adding points to the closest cluster, or rather, to the closest Gaussian density. Specifically, we assign a point $x \in X$ to the cluster $i \in \{1, \ldots, k\}$ such that

$$-\ln(p_i) - \ln\big(\mathcal{N}(x; \mu_i, \Sigma_i)\big)$$

is minimal.

We apply the above steps simultaneously until the change of the cost function is smaller than a predefined threshold or if the clusters did not change at all.

This approach causes a problem with local minima. Therefore, we aply a two point strategy for increasing the performance of the algorithm. More precisely, we apply a split and merge strategy.

2.1 Merge strategies

Let us consider Gaussian densities $\mathcal{G}$. For two disjoint sets X and Y ($X, Y \subset \mathbb{R}^N$), we want to develop a condition under which we should combine them into one

[2] We can also consider same Gaussians subfamilies [13, 10, 12].

cluster, rather then consider them separately – namely, energy/cost of $X \cup Y$ is less then sum of energy of X and Y. This condition is given by

$$E(X \cup Y, \mathcal{G}) \leq E(X, \mathcal{G}) + E(Y, \mathcal{G}). \tag{5}$$

For the general case, it is very difficult to solve the above inequality and give an analytical solution. Thus, for simplicity of the problem, it is necessary to put some restrictions on the sets X and Y. In this section we solve this in a one dimensional real space under same constraint about sets X and Y. But before that, let us now recall the following important remark which simplifies our situation.

Remark 1. Let X, Y be given as finite subsets of $\mathbb{R}^N$. Assume additionally that $X \cap Y = \emptyset$. Then

$$\begin{aligned} \mathrm{m}_{X\cup Y} &= p_X \mathrm{m}_X + p_Y \mathrm{m}_Y \\ \Sigma_{X\cup Y} &= p_X \Sigma_X + p_Y \Sigma_Y + p_X p_Y (\mathrm{m}_X - \mathrm{m}_Y)(\mathrm{m}_X - \mathrm{m}_Y)^T \end{aligned}$$

where $p_X = \frac{|X|}{|X|+|Y|}, p_Y = \frac{|Y|}{|X|+|Y|}$.

Theorem 1. *Let X, Y be given as finite subsets of $\mathbb{R}$. Assume additionally that*

- $X \cap Y = \emptyset$,
- $|X| = |Y|$,
- $\Sigma_X = \Sigma_Y = \Sigma = (\sigma^2)$ *for an arbitrary* $\sigma > 0$.

Then the distributions $\mathcal{N}(\mathrm{m}_X, \Sigma)$ and $\mathcal{N}(\mathrm{m}_Y, \Sigma)$ should be combined into one Gaussian distribution $\mathcal{N}(\frac{1}{2}(\mathrm{m}_X + \mathrm{m}_Y), (\sigma^2 + \frac{1}{4}(\mathrm{m}_X - \mathrm{m}_Y)^2))$ with respect to condition (5) iff

$$|\mathrm{m}_X - \mathrm{m}_Y| < 2\sqrt{3}\sigma,$$

where $\mathrm{m}_X, \mathrm{m}_Y$ denote the mean values of sets X, Y.

Proof. Let us consider equation (5) under the terms of our theorem. This leds us to

$$-\ln(1) + H^{\times}\big(X \cup Y \| \mathcal{G}\big) \leq \frac{1}{2} \cdot (-\ln(\frac{1}{2}) + H^{\times}\big(X \| \mathcal{G}\big)) + \frac{1}{2} \cdot (-\ln(\frac{1}{2}) + H^{\times}\big(Y \| \mathcal{G}\big)).$$

By equation (4) we get

$$\begin{aligned} \frac{N}{2}\ln(2\pi e) + \frac{1}{2}\ln\det(\Sigma_{X\cup Y}) \leq & \frac{1}{2}\bigg(\ln 2 + \frac{N}{2}\ln(2\pi e) + \frac{1}{2}\ln\det(\Sigma)\bigg) \\ & + \frac{1}{2}\bigg(\ln 2 + \frac{N}{2}\ln(2\pi e) + \frac{1}{2}\ln\det(\Sigma)\bigg), \end{aligned}$$

and consequently $\ln\det(\Sigma_{X\cup Y}) \leq \ln(4\det(\Sigma))$. Finally, by Remark 1 we obtain

$$\det(\Sigma_{X\cup Y}) \leq 4\det(\Sigma), \tag{6}$$

where $\Sigma_{X\cup Y} = \Sigma + \frac{1}{4}(\mathrm{m}_X - \mathrm{m}_Y)(\mathrm{m}_X - \mathrm{m}_Y)^T$.

In the case of one dimensional space the equation (6) simplifies to

$$\sigma^2 + \frac{1}{4}(\mathrm{m}_X - \mathrm{m}_Y)^2 \leq 4\sigma^2$$

which ends the proof.

Figure 1 presents a simple illustration of the above theorem in the case of $\sigma = 1$.

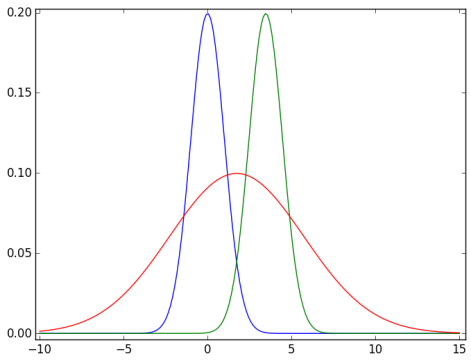


Fig. 1: Comparision of density descriptions of two sets. First, the sum of two densities $\mathcal{N}(0,1)$ (the blue one) and $\mathcal{N}(2\sqrt{3},1)$ (the green one) describe separated clusters, while the density $\mathcal{N}(\sqrt{3},4)$ (the red one) presents the best description of their set combination.

It needs to be highlighted that even in one dimensional space our considerations were limited to strong constraints, which shows how it is a very hard task in the general case.

In the approach proposed by the authors for the merge problem we will always check the condition (5) directly. However, we will use Remark 1, according to which we do not need to recalculate the covariance matrix for a cluster combination, which simplifies and speeds up the calculations.

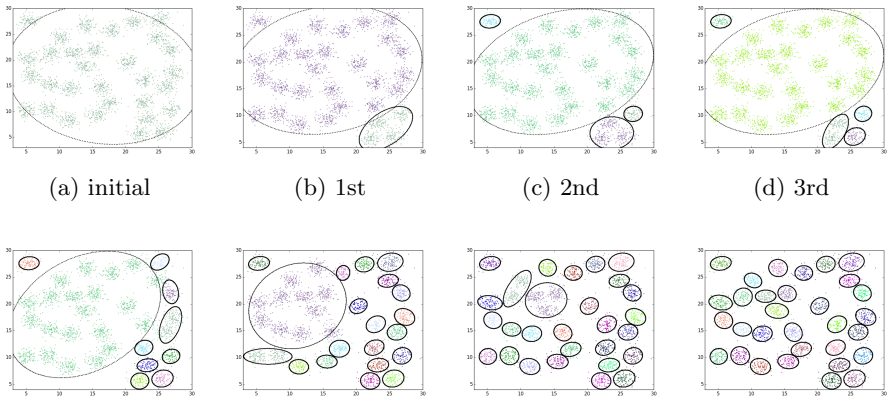
2.2 Split strategies

The decision about splitting a given cluster $X \subset \mathbb{R}^N$ into two parts X_1, X_2 ($X_1 \cup X_2 = X$, $X_1 \cap X_2 = \emptyset$) for CEC clustering in our case is given by the following

– run CEC clustering with two clusters for set X,

– for obtained clusters X_1 and X_2, if $E(X, \mathcal{G}) \geq E(X_1, \mathcal{G}) + E(X_2, \mathcal{G})$ then replace cluster X by X_1 and X_2.

The approach, in this case, is similar to the merge strategy, since we also want to decrease energy. Figure 2 presents division steps for a sample set from CEC clustering.



(a) initial (b) 1st (c) 2nd (d) 3rd

(e) 6th (f) 11th (g) 12th (h) 20th, last

Fig. 2: Split strategy. An initial clustering (with one cluster) is divided into as many clusters as needed using the split strategy. The numbers below the images present the numbers of waves of divisions. Those do not correspond to the number of clusters since multiple divisions can happen simultainiously.

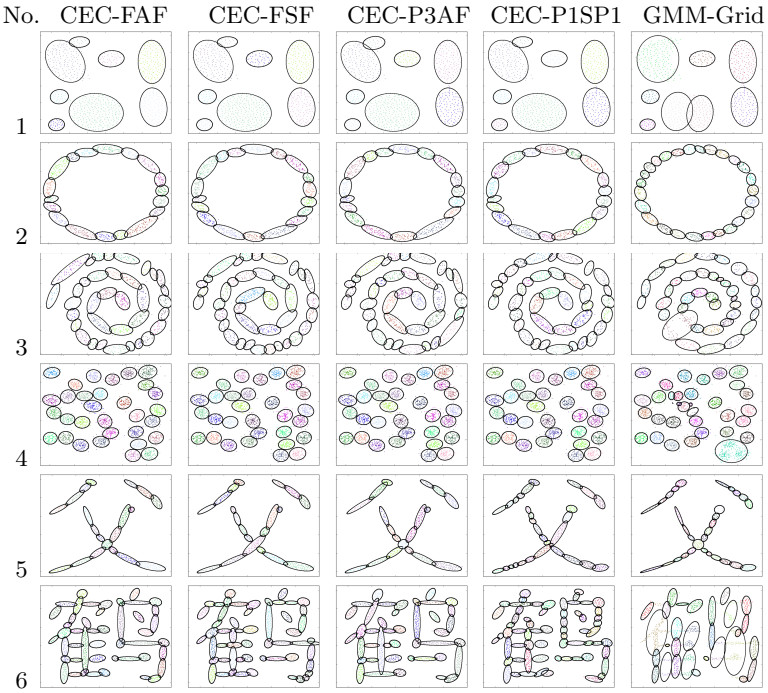
3 Proposed strategies and experiments

We develop a few strategies to check when split and merge steps should be applied during CEC clustering. We denote them as follows:

– CEC-FAF – the merge and split are performed when a CEC iteration did not reduce energy, the merge step is performed as many times as possible;
– CEC-FSF – the merge and split are performed when a CEC iteration did not reduce energy, the merge is performed only once at a time;
– CEC-P3AF – the merge is performed every third iteration as many times as possible, the split is perfmormed if a CEC iteration(including the merge) did not reduce energy,
– CEC-P1SP1 – the merge and split are performed in every iteration, the merge is performed once at most in each iteration.

We compare them with

Table 1: Comparision of different CEC split and merge strategies with GMM clustering.



- CEC-Grid – we apply CEC clustering with a starting number of clusters from 2 to 70 and then we choose the best result according to the lowest Bayesian information criterion (BIC);
- GMM-Grid – we used the Project R package mclust [3] which performs a grid search for finding the optimal numbers of mixture components (clusters) according to the BIC criterion. The considered range of the number of clusters was the same as in CEC-Grid.

3.1 Split-and-merge strategies effectiveness

Illustrations: Table 1 presents the results of clustering of the sample sets with the strategies listed above. The results of all strategies are pretty similar except for GMM-Grid.
Accuracy: Table 2 presents the comparision of the results obtained by investigating different strategies. We show the calculated MLE (maximum likelihood estimation – the higher the better), BIC (Bayesian information criterion – the lower the better), AIC (Akaike information criterion – the lower the better) and also give the values of the enegry function obrained by CEC clusterings (the lower the better). In this case results seem to favor the strategies CEC-Grid and

Table 2: Effectiveness of clustering for datasets from Figure 1. Table presents a comparison of results for various criteria with the best strategies selected (the marked ones).

Set	Method	MLE	AIC	BIC	cost	No. clust
1	CEC-Grid	-5028.89	10135.77	10317.88	6.387748	7
	CEC-FAF	-4971.75	10090.16	10274.28	6.372357	8
	CEC-FSF	-4972.59	10092.01	10274.28	6.368866	8
	CEC-P3AF	-4985.80	10088.99	10274.28	6.375692	8
	CEC-P1SP1	-4973.65	10092.16	10274.27	6.369666	8
	GMM-Grid	-5052.78	10171.55	10325.65		8
2	CEC-Grid	-4772.61	9828.48	10164.06	4.820493	12
	CEC-FAF	-4910.83	9840.92	10190.31	5.086688	18
	CEC-FSF	-4826.43	9840.41	10181.07	5.086773	18
	CEC-P3AF	-4831.82	9841.64	10183.92	5.086012	17
	CEC-P1SP1	-4827.30	9840.52	10181.90	5.088752	18
	GMM-Grid	-4844.54	9923.09	10497.30		29
3	CEC-Grid	-5302.07	10933.03	11523.40	5.337437	24
	CEC-FAF	-5296.87	10949.07	11511.24	5.545499	35
	CEC-FSF	-5297.57	10954.84	11516.37	5.544520	38
	CEC-P3AF	-5295.65	10953.27	11503.95	5.546854	36
	CEC-P1SP1	-5293.99	10943.12	11494.34	5.546223	38
	GMM-Grid	-5476.76	11275.40	12065.55		40
4	CEC-Grid	-17345.28	35275.38	36420.90	5.626151	28
	CEC-FAF	-17469.87	35247.75	36177.78	5.791727	31
	CEC-FSF	-17469.87	35247.75	36177.78	5.791727	31
	CEC-P3AF	-17469.87	35247.75	36177.78	5.791727	31
	CEC-P1SP1	-17469.87	35247.75	36177.78	5.791727	31
	GMM-Grid	-17857.81	35933.63	36591.90		27
5	CEC-Grid	1169.98	-2142.05	-1908.32	-1.55705	7
	CEC-FAF	1260.44	-2259.60	-1780.56	-2.23154	20
	CEC-FSF	1261.10	-2252.04	-1765.01	-2.39991	20
	CEC-P3AF	1259.68	-2260.20	-1781.17	-2.47543	19
	CEC-P1SP1	1279.09	-2227.21	-1613.51	-1.76784	29
	GMM-Grid	1048.00	-1854.02	-1312.30		30
6	CEC-Grid	1585.14	-2822.28	-2088.01	-1.00290	25
	CEC-FAF	1762.23	-2876.52	-1554.79	-1.09039	46
	CEC-FSF	1738.68	-2876.18	-1566.58	-0.96549	54
	CEC-P3AF	1758.68	-2893.73	-1658.30	-1.19903	42
	CEC-P1SP1	1790.03	-2855.14	-1260.18	-0.85854	63
	GMM-Grid	815.81	-1413.62	-834.12		27

CEC-P1SP1 as the best ones, although the results are usually very close (except for GMM-Grid).

Table 3 presents an adjusted rank index for different strategies. In this case all strategies gave the same result on their best runs.

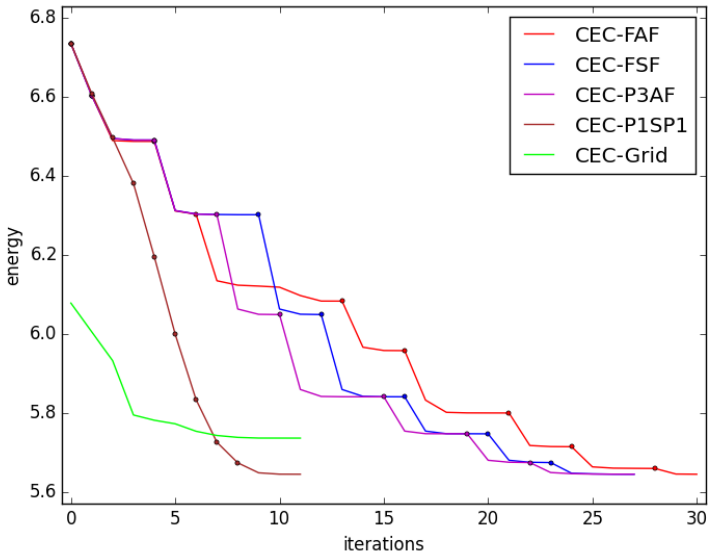


Fig. 3: Energy during clustering iterations under different strategies. The split and merge strategies were all started with a single cluster.

Table 3: Adjusted rank index (the higher the better) for datasets 1 and 4 from Figure 1 under different strategies.

Set	CEC-P1SP1	GMM-Grid	CEC-Grid
1	0.9687795	0.801032	0.997804
4	0.9471447	0.847442	0.877827

Energy: Figure 3 presents the comparison of the energy function during clustering under different strategies. The results show that CEC-Grid and CEC-P1SP1 reach minimum in the smallest number of steps.
Speed: While the above could suggest the CEC-P1SP1 strategy was the fastest, take note that because it performs both split and merge operations at each iteration, the iterations themself are more expensive computational-wise. A direct time measurement of the strategies revealed CEC-FSF to be the fastest, followed closely by CEC-P3AF.

3.2 Split-and-merge CEC in higher dimensions

Table 4 presents the results obtained with the same strategies and algorithms on higher dimensional data. In this experiment there was much more variation both between the strategies and between different runs of the same strategy. It appears that the algorithm suffers from the curse of dimensionality and has

Table 4: The results of algorithms in the case of data from UCI repository.

Set	Method	MLE	AIC	BIC	cost	No. clust
cancer	CEC-Grid	-5397.75	12541.50	14248.87	12.6645	26
	CEC-FAF	-5724.97	12311.94	14124.35	11.3153	8
	CEC-FSF	-5276.11	11738.21	13842.75	11.1355	11
	CEC-P3AF	-5383.41	12222.18	13691.71	10.5129	15
	CEC-P1SP1	-4399.87	12037.74	15161.96	11.4671	30
	GMM-Grid	-5556.90	11551.97	12548.34		16
iris	CEC-Grid	-147.29	537.78	619.07	0.7153	11
	CEC-FAF	-166.91	529.32	619.07	0.7874	66
	CEC-FSF	-206.48	522.96	619.07	1.2747	53
	CEC-P3AF	-203.29	516.58	653.35	1.3382	49
	CEC-P1SP1	-205.56	541.42	658.23	1.2750	56
	GMM-Grid	-221.20	506.40	602.74		4
seeds	CEC-Grid	2751.91	-5291.82	-4937.03	-2.4818	6
	CEC-FAF	2088.59	-3897.19	-3428.59	-2.3327	5
	CEC-FSF	2082.18	-3880.36	-3405.07	-2.5921	6
	CEC-P3AF	2302.52	-4463.04	-4225.39	-2.1047	4
	CEC-P1SP1	2868.88	-5056.88	-4581.59	-2.4040	13
	GMM-Grid	1157.24	-2064.48	-1646.09		11
wine	CEC-Grid	-2455.36	6452.88	7984.43	11.17177	14
	CEC-FAF	-2306.49	6674.17	7190.19	11.64549	12
	CEC-FSF	-2604.09	6667.63	7562.31	16.40669	10
	CEC-P3AF	-2805.89	6738.39	7403.39	13.86455	8
	CEC-P1SP1	-2378.48	6873.21	8524.56	11.52989	11
	GMM-Grid	-2966.22	6320.44	6937.71		7

trouble finding the right number of clusters in higher dimensional data. This is most evident in the iris dataset, where it overshots that number tenfold.

4 Conclusions and future work

We have proposed an approach for the split and merge problem in CEC clustering. It was shown that even in a one dimensional space the condition is not so easy to assess. Thus, in the proposed approach, we implicitly use the formula to decide if two clusters should be combined. Our experiments present that the strategy when the merge and split are performed in every iteration and the merge is performed once at most in each iteration can be competitive to classical CEC or even CEC-Grid.

Our future work will focus on developing new measures which will allow us to solve the split problem using information theory.

5 Acknowledgement

The research of Krzysztof Misztal is supported by the National Science Centre (Poland) [grant no. 2012/07/N/ST6/02192]. The research of Przemysław Spurek by the National Science Centre (Poland) Grant No. 2015/19/D/ST6/01472].

References

1. David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.
2. Gilles Celeux and Gérard Govaert. Gaussian parsimonious clustering models. *Pattern recognition*, 28(5):781–793, 1995.
3. Chris Fraley and Adrian E Raftery. MCLUST: Software for model-based cluster analysis. *Journal of Classification*, 16(2):297–306, 1999.
4. Geoffrey E Hinton, Peter Dayan, and Michael Revow. Modeling the manifolds of images of handwritten digits. *IEEE transactions on Neural Networks*, 8(1):65–74, 1997.
5. Anil K Jain. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010.
6. Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.
7. Edwin Lughofer. A dynamic split-and-merge approach for evolving cluster models. *Evolving Systems*, 3(3):135–151, 2012.
8. Ujjwal Maulik and Sanghamitra Bandyopadhyay. Genetic algorithm-based clustering technique. *Pattern recognition*, 33(9):1455–1465, 2000.
9. Geoffrey McLachlan and Thriyambakam Krishnan. *The EM algorithm and extensions*, volume 382. John Wiley & Sons, 2007.
10. K. Misztal, P. Spurek, E. Saeed, K. Saeed, and J. Tabor. Cross entropy clustering approach to iris segmentation for biometrics purpose. *Schedae Informaticae*, 24:31–40, 2015.
11. Shy Shoham. Robust clustering by deterministic agglomeration em of mixtures of multivariate t-distributions. *Pattern Recognition*, 35(5):1127–1142, 2002.
12. J. Tabor and K. Misztal. Detection of elliptical shapes via cross-entropy clustering. In *Pattern Recognition and Image Analysis*, volume 7887, pages 656–663. Springer Berlin Heidelberg, 2013.
13. Jacek Tabor and Przemysław Spurek. Cross-entropy clustering. *Pattern Recognition*, 47(9):3046–3059, 2014.
14. Naonori Ueda, Ryohei Nakano, Zoubin Ghahramani, and Geoffrey E Hinton. Smem algorithm for mixture models. *Neural computation*, 12(9):2109–2128, 2000.
15. Rui Xu and Don Wunsch. *Clustering*. Wiley-IEEE Press, 2009.
16. Baibo Zhang, Changshui Zhang, and Xing Yi. Competitive EM algorithm for finite mixture models. *Pattern recognition*, 37(1):131–144, 2004.
17. Zhihua Zhang, Chibiao Chen, Jian Sun, and Kap Luk Chan. Em algorithms for gaussian mixtures with split-and-merge operation. *Pattern recognition*, 36(9):1973–1983, 2003.