R Package CEC

P. Spurek, K. Kamieniecki, J. Tabor, K. Misztal, M. Śmieja

Faculty of Mathematics and Computer Science, Jagiellonian University, Łojasiewicza 6, 30-348 Kraków, Poland.

Abstract

Cross-Entropy Clustering (CEC) is a model-based clustering method which divides data into Gaussian-like clusters. The main advantage of CEC is that it combines the speed and simplicity of k-means with the ability of using various Gaussian models similarly to EM. Moreover, the method is capable of the automatic reduction of unnecessary clusters. In this paper we present the **R** Package **CEC** implementing CEC method.

Keywords: clustering, Gaussian models, density estimation, R package

1. Introduction

Gaussian Mixture Model (GMM) is one of the most popular parametric clustering models implemented in various R packages, such as **mclust** [7], **Rmixmod** [8], **pdfCluster** [2], **mixtools** [3], etc. The model focuses on finding the mixture of Gaussians $f = p_1 f_1 + \ldots + p_k f_k$, where $p_1, \ldots, p_k > 0$ and $\sum_i p_i = 1$, which provides an optimal estimation of data set $X \subset \mathbb{R}^N$, measured by the negative log-likelihood cost function:

$$\mathrm{EM}(f, X) = -\frac{1}{|X|} \sum_{x \in X} \log\left(p_1 f_1(x) + \ldots + p_k f_k(x)\right), \tag{1}$$

where $|X|$ denotes the cardinality of X. Its minimization is iteratively performed with use of EM (Expectation Maximization) algorithm. While the expectation step is relatively simple, the maximization step usually needs

[1]The work was supported by the National Centre of Science (Poland) [grants no. 2013/09/N/ST6/01178, 2014/13/B/ST6/01792, 2012/07/N/ST6/02192, 2014/13/N/ST6/01832].

complicated numerical optimization which is a source of the high computational cost in the case of large, high dimensional data sets.

This paper presents **R** Package **CEC**, the first open source implementation of a novel Cross-Entropy Clustering method (CEC) [4, 5, 6] which is a fast hybrid between k-means and GMM. Similarly to GMM, CEC searches for Gaussian densities $f_1, \ldots, f_k$ and numbers $p_1, \ldots, p_k \geq 0$, such that $\sum_i p_i = 1$, which minimizes the generalized cross-entropy function:

$$\mathrm{CEC}(f, X) = -\frac{1}{|X|} \sum_{x \in X} \log\left(\max(p_1 f_1(x), \ldots, p_k f_k(x))\right). \tag{2}$$

Although the difference between (2) and (1) is slight and relies on substituting the sum operation by the maximum, it occurs that the optimization can be realized in a comparable time to k-means algorithm by a modified Hartigan approach. From an information-theoretic point of view we construct k-encoders (identified by densities f_i) which allow to optimally compress, with respect to differential entropy, data set X. Since every encoder (cluster) has defined its own cost then CEC allows to reduce unnecessary clusters on-line (some of p_i can be zeros).

2. Implementation and functionalities

The R package is divided into the R part and a compiled library. The R part contains the main function `cec`, various auxiliary functions and a test framework with a set of end-to-end tests. The core of the package is written in C and consists of two layers: the implementation of CEC algorithm with corresponding data structures and functions that handle interactions with R environment.

The package provides a main clustering method:

```
cec(x = ..., type = ..., centers = ..., card.min = ..., nstart = ...).
```

The parameter `type` specifies the type of clusters models. Six types of Gaussian distributions are available to represent the clusters models: general (unconstrained) Gaussians (`type = "all"`), spherical Gaussians(`type = "spherical"`), spherical Gaussians with the fixed radius(`type = "fixedr", param = ...`), diagonal Gaussians (`type = "diagonal"`), Gaussians with the fixed covariance (`type = "covariance", param = ...`) or Gaussians with fixed eigenvalues (`type = "eigenvalues", param = ...`). The unconstrained Gaussian can be used for exploring the data structure in the case when no information about the relations in the dataset is available, see Fig. 1(a). After the analysis of the outcome, the decision can be made to use more specific types of Gaussian families, e.g., if we look for spherically shaped clusters, as in the case of mouse-like set presented in the Fig 1(b), the value `type = "spherical"` should be used. The illustration of various types of Gaussian models is presented in Fig. 2.

The user chooses the maximal number of groups in the parameter `centers`. CEC reduces unnecessary clusters on-line and consequently the final partition might result in less number of groups than a given value. To enable faster reduction of unnecessary clusters an additional parameter `card.min = "5%"` is introduced: a group is removed if it contains less number of elements then 5% of data set cardinality. The nature of the algorithm is non-deterministic and analogously to k-means it depends on the initial clusters memberships. We can initialize clustering by kmeans++ algorithm [1] (by specifying `centers.init` parameters) instead of random initialization. The parameter `nstart` determines the number of membership initialization. In the basic use of this package the input dataset (`data`) in the form of simple array or matrix and the initial number (`centers`) of clusters have to be specified, other parameters take their default values: `card.min = "5%", nstart=1, type="all", iter.max = 25, centers.init = "kmeans++"`.

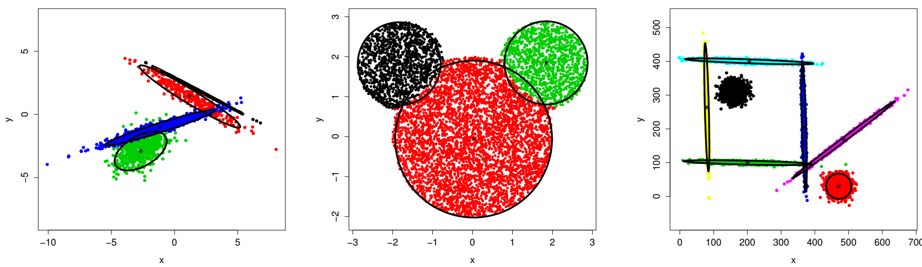
One of the most important properties of CEC is the possibility of mixing different types of models. This allows to distinguish various patterns on the image, e.g. matches from coins [5]. If we know that image contain two clusters described by spherical Gaussians with a fixed radius $r = 350$ and five clusters with fix eigenvalues `c(9000, 8)` we can find them (see Fig. 1(c)) by specifying parameters `type` and `param`:

```
type=c("fixedr","fixedr","eigen","eigen","eigen","eigen","eigen")
param=list(350,350,c(9000,8),c(9000,8),c(9000,8),c(9000,8),c(9000,8)).
```

3. Empirical results

We present a basic session with **R**:

```
R> library("CEC")
R> data("fourGaussians")
R> cec <- cec(fourGaussians, centers = 10, type = "all",
     nstart = 20)
R> plot(cec, xlim = c(0, 1), ylim = c(0, 1), asp = 1)
```



(a) (b) (c)

Figure 1: The effect of CEC algorithm in the case of (a) all Gaussian distribution, (b) spherical Gaussian distribution, (c) mixed model.

Figure 2: Confidence ellipse of spherical Gaussians, spherical Gaussians with fixed radius, diagonal Gaussians, Gaussians with fixed covariance and Gaussians with fixed eigenvalues.

The results of the general Gaussian CEC algorithm presented in the Fig. 1(a) give similar results to those obtained by the Gaussian Mixture Models. In Tab. 1 we present comparison between CEC, EM and k-mens on typical real datasets with using Rand index measure. Usually CEC and EM discovered close to correct number of cluster and obtain higher value of Rand index then k-means method. However, the author's method does not use the EM approach for minimization but a faster iterative Hartigan's algorithm. Consequently, larger datasets can be processed in shorter time. In the experiments we compared the computational times between **CEC** and alternative packages **mclust** and **Rmixmod** implementing EM algorithm when increasing the number of data set instances and the dimension of data. For this purpose a modified version of mouse-like set given in Fig 1(b) was considered. One can observe that EM implementations, contrary to k-means and CEC, do not scale well in the case of large amount of high dimensional data, see Fig 3.

[1] D. Arthur, S. Vassilvitskii, k-means++: The advantages of careful seeding, Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

Dataset	Nr. of clusters				Rand index		
	original	EM	CEC	k-means	EM	CEC	k-means
wine	3	3	3	3	0.94668	0.96033	0.71866
diabetes	10	10	8	5	0.74683	0.71013	0.67711
glass	7	5	5	5	0.69361	0.69791	0.66311
diabetes	2	6	6	8	0.49321	0.50768	0.49742

Table 1: Comparison of the CEC with clustering by EM (the number of cluster is obtained by Bayesian information criterion) and k-means (the number of cluster is obtained by gap statistic).

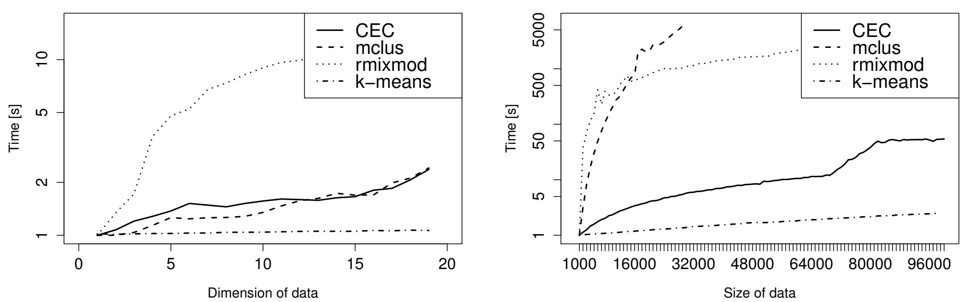


Figure 3: Comparison of computational efficiency between **R** packages **CEC**, **Rmixmod**, **Mclust** (times is shown in the logarithmic scale).

[2] A. Azzalini, G. Menardi, Clustering via nonparametric density estimation: The **R** package **pdfCluster**, Journal of Statistical Software 57 (11).

[3] T. Benaglia, D. Chauveau, D. R. Hunter, D. S. Young, **mixtools**: An **R** package for analyzing mixture models, Journal of Statistical Software 32 (6).

[4] J. Tabor, P. Spurek, Cross-entropy clustering, Pattern Recognition 47 (9) (2014) 3046–3059.

[5] J. Tabor, K. Misztal, Detection of elliptical shapes via cross-entropy clustering, Pattern Recognition and Image Analysis (2013) 656–663.

[6] M. Smieja, J. Tabor, Spherical Wards clustering and generalized Voronoi diagrams, IEEE International Conference DSAA'2015 (2015) 1–10.

[7] C. Fraley, A. E. Raftery, MCLUST: Software for model-based cluster analysis, Journal of Classification 16 (2) (1999) 297–306.

[8] B. Auder, R. Lebret, S. Iovleff, F. Langrognet, Rmixmod: An interface for MIXMOD, r package version 2.0.2 (2014).

Required Metadata

Current executable software version

Ancillary data table required for sub version of the executable software: (x.1, x.2 etc.) kindly replace examples in right column with the correct information about your executables, and leave the left column as it is.

Nr.	**(executable) Software metadata description**	**Please fill in this column**
S1	Current software version	0.9.4
S2	Permanent link to executables of this version	*https : //cran.r − project.org/web/packages/CEC/index.html*
S3	Legal Software License	GPL-3
S4	Computing platform/Operating System	Linux, OS X, Microsoft Windows, Unix-like
S5	Installation requirements & dependencies	
S6	If available, link to user manual - if formally published include a reference to the publication in the reference list	*https : //github.com/azureblue/cec*
S7	Support email for questions	przemyslaw.spurek.at.gmail.com

Table 2: Software metadata (optional)

Current code version

Ancillary data table required for subversion of the codebase. Kindly replace examples in right column with the correct information about your current code, and leave the left column as it is.

Nr.	Code metadata description	Please fill in this column
C1	Current code version	0.9.4
C2	Permanent link to code/repository used of this code version	*https : //github.com/azureblue/cec*
C3	Legal Code License	GPL-3
C4	Code versioning system used	git
C5	Software code languages, tools, and services used	C, R
C6	Compilation requirements, operating environments & dependencies	
C7	If available Link to developer documentation/manual	*https : //github.com/azureblue/cec*, *https : //cran.r − project.org/web/packages/CEC/CEC.pdf*
C8	Support email for questions	przemyslaw.spurek.at.gmail.com

Table 3: Code metadata (mandatory)