Multivariate Beta Mixture Model: Probabilistic Clustering With Flexible Cluster Shapes

Yung-Peng Hsu and Hung-Hsuan Chen

National Central University, Taiwan
yungpeng1998@gmail.com, hhchen1105@acm.org

Abstract. This paper introduces the multivariate beta mixture model (MBMM), a new probabilistic model for soft clustering. MBMM adapts to diverse cluster shapes because of the flexible probability density function of the multivariate beta distribution. We introduce the properties of MBMM, describe the parameter learning procedure, and present the experimental results, showing that MBMM fits diverse cluster shapes on synthetic and real datasets. The code is released anonymously at https://github.com/hhchen1105/mbmm/.

Keywords: Mixture model · EM algorithm · Clustering.

1 Introduction

Data clustering groups data points into components so that similar points are within the same component. Data clustering is commonly used for data exploration and is sometimes used as a preprocessing step for later analysis [11].

In this paper, the multivariate beta mixture model (MBMM), a new probabilistic model for soft clustering, is proposed. As the MBMM is a mixture model, it shares many properties with the Gaussian mixture model (GMM), including its soft cluster assignment and parametric modeling. In addition, the MBMM allows the generation of new (synthetic) instances based on a generative process. Because the beta distribution is highly flexible (e.g., unimodal, bimodal, straight line, or exponentially increasing or decreasing), MBMM can fit data with versatile shapes. Figure 1 shows that various cluster shapes can be obtained with a bivariate beta distribution. On the contrary, the shape of a Gaussian distribution is symmetric and unimodal, which limits its fitting capacity.

The multivariate beta distribution is defined in different ways. In some studies, the Dirichlet distribution is considered a multivariate beta distribution (e.g., [9]) because the beta distribution is a special case of the Dirichlet distribution with two parameters. However, we apply the definition provided in [6], which is even more general than the Dirichlet distribution. The relationship between the Dirichlet distribution and our multivariate beta distribution will be discussed in Section 2.1 when we introduce the details of the multivariate beta distribution.

This paper presents several contributions. First, we propose a new probabilistic model for soft clustering. Our model is similar to the Gaussian Mixture Model

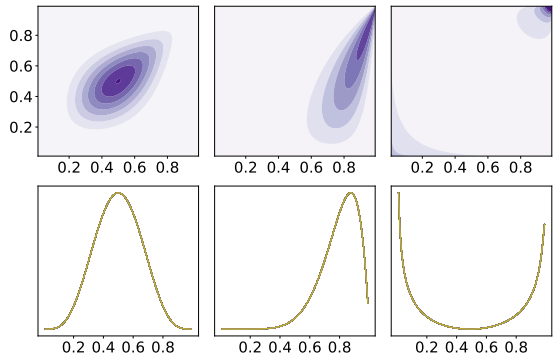

Fig. 1. Examples of the versatile shape of the bivariate beta distribution. The upper row shows three bivariate beta distributions with different parameters. The bottom row shows the marginal distribution of x_1 (i.e., the variable on the horizontal axis in the top row). This distribution can be symmetric unimodal (e.g., the left subfigure), skewed unimodal (e.g., the middle subfigure), or bimodal (e.g., the right subfigure).

(GMM), but the shape of each cluster is more versatile than those generated by GMM. Second, we compare MBMM with well-known clustering algorithms on synthetic and real datasets to demonstrate its effectiveness. Finally, we release the code for reproducibility. Our implemented class offers `fit()`, `predict()`, and `predict_proba()`, the common methods provided by `scikit-learn`'s clustering algorithms, making it convenient to apply MBMM to new domains.

The rest of the paper is organized as follows. Section 2 introduces the multivariate beta distribution and the proposed MBMM. Section 3 describes experiments on synthetic and real datasets. Section 4 reviews previous work on data clustering. We conclude by discussing the limitations of the MBMM and the ongoing and future work on the MBMM in Section 5.

2 Multivariate Beta Mixture Model

2.1 Multivariate beta distribution

The probability density function (PDF) of a multivariate beta distribution (MB) has been defined in different ways [2, 6]. Here, we apply the definition in [6]: given an instance $\boldsymbol{x} = [x_1, \ldots, x_M]^T$ with M variates (i.e., features) and the shape parameters $a_m > 0, b > 0$ $(m = 1, \ldots, M)$, its PDF is given by

$$MB(\boldsymbol{x}|a_{1:M}, b) = \frac{1}{Z} \times \frac{\prod_{m=1}^{M} \frac{x_m^{a_m-1}}{(1-x_m)^{a_m+1}}}{(1+\sum_{k=1}^{M} \frac{x_k}{1-x_k})^{a_1+\ldots+a_M+b}}, \tag{1}$$

where $x_m \in (0, 1), a_m > 0, b > 0$, and the normalizer Z is defined by

$$Z = \frac{\Gamma(b) \prod_{m=1}^{M} \Gamma(a_m)}{\Gamma(b + \sum_{j=1}^{M} a_j)}, \tag{2}$$

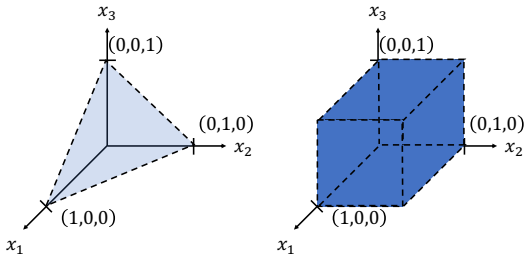

Fig. 2. A comparison of the support of the Dirichlet distribution (left) and our multivariate beta distribution (right) with 3 variates. The Dirichlet distribution is only defined on $x_i \in (0, 1)$ such that $x_1 + x_2 + x_3 = 1$ (the standard 2-simplex in R^3). On the contrary, our multivariate beta distribution is defined on $(0, 1)^3$ (the unit cube in R^3), which is a superset of the Dirichlet distribution.

where Γ is the gamma function.

In some previous studies, the Dirichlet distribution was treated as a multivariate generalization of the beta distribution (e.g., [9]) since the Dirichlet distribution falls back to the beta distribution when the number of parameters is 2. However, we describe a more general definition of the multivariate beta distribution that regards the Dirichlet distribution as a special case. The relationship between the Dirichlet distribution and the proposed multivariate beta distribution is illustrated in Figure 2. Specifically, the support of an n-variate Dirichlet distribution is restricted to a standard $(n-1)$-simplex. However, the support of our multivariate beta distribution is a hypercube in an n-dimensional space with a length of 1 on each side. In other words, the Dirichlet distribution is the multivariate beta distribution subject to $\|\boldsymbol{x}\|_1 = 1$.

2.2 MBMM density function and generative process

In Table 1, we list the notations that will be used in this paper hereafter.

In MBMM, it is assumed that the data points are generated from a mixture of multivariate beta distributions (whose PDF is defined in Equation 1). Consequently, the probability of the MBMM given C components is

$$p(\boldsymbol{x}_n|\boldsymbol{\theta}) = \sum_{c=1}^{C} \pi_c MB(\boldsymbol{x}_n|\boldsymbol{\theta}_c). \tag{3}$$

The parameter π_c determines the probability that a random instance $\boldsymbol{x}_n$ belongs to cluster c (before knowing the values of the variates in $\boldsymbol{x}_n$), and $MB(\boldsymbol{x}_n|\boldsymbol{\theta}_c)$ gives the PDF if $\boldsymbol{x}_n$ indeed belongs to cluster c.

Figure 3 shows a graphical representation of the multivariate beta mixture model. To generate a sample $\boldsymbol{x}_n$, we first sample a latent variable z_n

Table 1. Notation list

Indices:	
M	Dimensions of an observed instance ($m \in \{1, \ldots, M\}$)
C	Number of clusters ($c \in \{1, \ldots, C\}$)
N	Number of instances ($n \in \{1, \ldots, N\}$)
Parameters:	
$a_{c,m}$	the m-th shape parameter for cluster c; $a_{c,m} > 0$
b_c	the $(M+1)$-th shape parameter for cluster c; $b_c > 0$
π_c	Mixture weight of cluster c, $0 < \pi_c < 1, \sum_{c=1}^{C} \pi_c = 1$
z_n	Cluster that $\boldsymbol{x}_n$ belongs to; $z_n \in \{1, \ldots, C\}$
$\gamma_{n,c}$	Probability that $\boldsymbol{x}_n$ belongs to cluster c; $\sum_{c=1}^{C} \gamma_{n,c} = 1$
$\boldsymbol{\theta}_c$	Set of parameters for cluster c; $\boldsymbol{\theta}_c = \{a_{c,1}, \ldots, a_{c,M}, b_c\}$
$\boldsymbol{\theta}$	Set of parameters for the MBMM; $\boldsymbol{\theta} = \{a_{1:C,1:M}, b_{1:C}, \pi_{1:C}\}$
Observed random variables:	
$\boldsymbol{x}_n$	Observed instance; $\boldsymbol{x}_n = [x_{n,1}, \ldots, x_{n,M}]^T \in R^M$

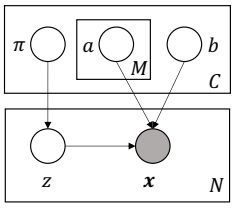

Fig. 3. Graphical representation of the multivariate beta mixture model

(the cluster ID of the sample $\boldsymbol{x}_n$) from a multinomial distribution with parameters $\pi_1, \ldots, \pi_C$. Suppose that $z_n = c$ after sampling, we further sample an instance $\boldsymbol{x}_n$ from the multivariate beta distribution with parameters $\boldsymbol{\theta}_c$: $MB(\boldsymbol{x}_n|\boldsymbol{\theta}_c) = MB(\boldsymbol{x}_n|a_{c,1}, \ldots, a_{c,M}, b_c)$.

2.3 Parameter Learning for the MBMM

In reality, we do not know the values of the parameters $\boldsymbol{\theta} = \{a_{1:C,1:M}, b_{1:C}, \pi_{1:N}\}$ (referring to Figure 3). We hope to recover these parameters based on the observed $\boldsymbol{x}_n$-s to maximize the likelihood function:

$$L(\boldsymbol{\theta}) = p(\boldsymbol{x}_{1:N}, z_{1:N}|\boldsymbol{\theta}) = \prod_{n=1}^{N} \prod_{c=1}^{C} \left[\pi_c MB(\boldsymbol{x}_n|\boldsymbol{\theta}_c)\right]^{I(z_n=c)}, \tag{4}$$

where I is the indicator function.

As the likelihood function (Equation 4) involves the multiplication of $N \times C$ terms, the result is numerically unstable. Instead, we compute the log-likelihood

Data: Input data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$, cluster number C
Result: Parameters $\boldsymbol{\theta} = \{\pi_{1:C}, a_{1:C,1:M}, b_{1:C}\}$
Initialize $\boldsymbol{\theta}$ randomly;
while not converge **do**
  // E-step
  for $n \leftarrow 1$ **to** N **do**
    for $c \leftarrow 1$ **to** C **do**
      Update $\gamma_{n,c}$ by Equation 7;
    end
  end
  // M-step
  Update $a_{1:C,1:M}$ and $b_{1:C}$ with the SQP solver [10];
  Update $\pi_{1:C}$ by Equation 8;
  if iteration count reaches a pre-defined value **then**
    Exit while loop;
  end
end

Algorithm 1: Parameter learning algorithm for the MBMM

function to convert multiplications to additions, as shown in Equation 5. As a result, the computation is more numerically stable.

$$\log L(\boldsymbol{\theta}) = \sum_{n=1}^{N} \sum_{c=1}^{C} I(z_n = c) \left(\log \pi_c + \log MB(\boldsymbol{x}_n | \boldsymbol{\theta}_c)\right). \tag{5}$$

However, since we cannot observe the latent z_n in practice, direct optimization of Equation 5 is difficult. As an alternative, we compute the expected value of the log-likelihood function with respect to the latent variables $z_{1:N}$, which involves the expected (but not the true) values of z_n:

$$E_{z_{1:N}} \left[\log L(\boldsymbol{\theta})\right] = \sum_{n=1}^{N} \sum_{c=1}^{C} \gamma_{n,c} \left(\log \pi_c + \log MB(\boldsymbol{x}_n | \boldsymbol{\theta}_c)\right). \tag{6}$$

After the above reformulation, the parameters $(\boldsymbol{\theta}_{1:C}, \pi_{1:C})$ that are used to maximize the expected value of the log-likelihood function (Equation 6) can be learned via the EM algorithm, as given by the pseudocode in Algorithm 1. In the E-step, we compute $\gamma_{n,c}$ (the probability that instance $\boldsymbol{x}_n$ belongs to cluster c) that maximizes Equation 6 by assuming that the randomly initialized or currently estimated $\pi_{1:C}$ and $\boldsymbol{\theta}_{1:C}$ are correct. The assignment of $\gamma_{n,c}$ has a simple closed-form solution, as shown below

$$\gamma_{n,c} = \frac{\pi_c MB(\boldsymbol{x}_n | \boldsymbol{\theta}_c)}{\sum_{k=1}^{C} \pi_k MB(\boldsymbol{x}_n | \boldsymbol{\theta}_k)}. \tag{7}$$

In the M-step, we search for the parameters $\pi_{1:C}$, $a_{1:C,1:M}$, and $b_{1:C}$ by assuming that the estimated $\gamma_{n,c}$ values in the E-step are correct. However, since

the $a_{1:C,1:M}, b_{1:C}$ parameters seem to lack a closed-form solution, we resort to numerical optimization strategies, specifically the sequential quadratic programming (SQP) iterative method, as the minimization strategy [10] because SQP allows linear constraints on the parameters (i.e., $a_{c,m} > 0$ and $b_c > 0 \; \forall c, m$). For parameters $\pi_1, \ldots, \pi_C$, we rely the efficient closed-form solution:

$$\pi_c = \frac{1}{N} \sum_{n=1}^{N} \gamma_{n,c}. \tag{8}$$

We compute the difference between the log-likelihood estimation in successive rounds for the convergence check. Additionally, if the number of iterations reaches a predefined value, we terminate the loop.

2.4 The similarity score between data points

Most clustering algorithms define the distance between two samples by converting them into a non-negative real value, i.e., given $\boldsymbol{x}_i, \boldsymbol{x}_j \in R^M$, the distance function is represented by $d_{i,j} : \boldsymbol{x}_i \times \boldsymbol{x}_i \to \{0, R^+\}$. However, if we define the distance of two samples based on their coordinates and assign samples to the closest cluster centroid, the output shapes of the clusters are inevitably convex.

In MBMM, we define the distance between two data points from a different perspective. Since the PDF of a data point is an affine combination of C multivariate beta distributions (Equation 3), we consider $MB(\cdot|\boldsymbol{\theta}_1), \ldots, MB(\cdot|\boldsymbol{\theta}_C)$ as the basis to form a function space. Consequently, the coordinate of the data point $\boldsymbol{x}_n$ becomes $\boldsymbol{\gamma}_n = [\gamma_{n,1}, \gamma_{n,2}, \ldots, \gamma_{n,C}]^T$ with respect to the basis functions. The vector $\boldsymbol{\gamma}_n$ is a discrete probability distribution since $\sum_{c=1}^{C} \gamma_{n,c} = 1$. Thus, we can define the distance between the data points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ as the distance between the discrete probability distributions $\boldsymbol{\gamma}_i$ and $\boldsymbol{\gamma}_j$. We use the Kullback-Leibler divergence (KL divergence) to determine this distance:

$$d_{i,j}^{KL} := \sum_{c=1}^{C} \gamma_{i,c} \log \left(\frac{\gamma_{i,c}}{\gamma_{j,c}} \right). \tag{9}$$

3 Experiments

3.1 Comparisons on the synthetic datasets

Baseline models Clustering algorithms can be classified into four types: centroid-based, density-based, distribution-based, and hierarchical clustering algorithms. We select representative models from each of these categories. For centroid-based models, we choose the k-means algorithm, which is likely the most widely used clustering algorithm. We select the DBSCAN algorithm for density-based models, which received the Test of Time award at KDD 2014. We choose the GMM as the distribution-based model and the agglomerative clustering (AC) algorithm as the hierarchical clustering model.

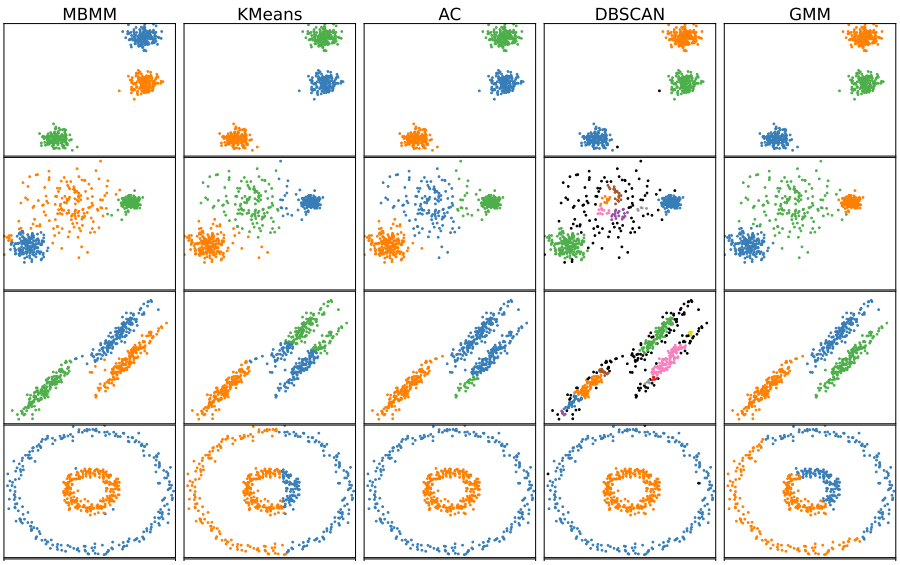

Fig. 4. Clustering algorithms on synthetic datasets

Synthetic dataset generation We generate synthetic datasets for the experiments. First, we create data points from three isotropic 2D Gaussian distributions whose means are distant and whose variances are small in each dimension. Consequently, a data point is close to other data points of the same Gaussian distribution but far from those in other Gaussian distributions (Figure 4, first row). This dataset represents an ideal case for data clustering.

Second, we generate two distant 2D Gaussian distributions with small variances in each dimension. However, the third distribution has a large variance. Consequently, several data points sampled from the third distribution are mixed with those from other distributions (Figure 4, second row).

Third, we generate three 2D Gaussian distributions with isolated means as before. However, we introduce a high correlation between two covariates for each Gaussian distribution. Consequently, data points are sometimes closer to data points generated from other distributions (Figure 4, third row).

Fourth, we generate concentric circles (i.e., one circle within another). If a point from the outer circle is selected, the most distant data point is located on the other side of the same circle. Consequently, such a synthetic dataset is highly challenging for centroid-based and distribution-based methods to group the outer circle as one cluster (Figure 4, fourth row).

We scale the range of every feature to within 0 and 1 because the support of each variate in the MB distribution is between 0 and 1, as explained in Section 2.1. This constraint has little influence in practice because clustering algorithms generally require the availability of all data sets prior to training; thus, scaling each variate as a preprocessing step is straightforward.

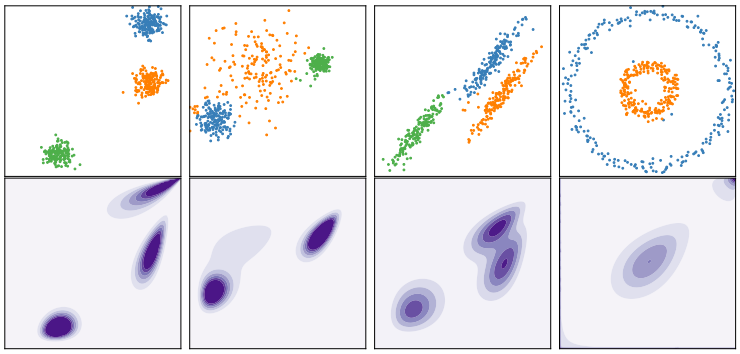

Fig. 5. Upper row: data points clustered by the MBMM; bottom row: PDFs based on the fitted parameters.

Visualizing clustering results Figure 4 shows a visualized comparison of the clustering algorithms for the four synthetic datasets.

All algorithms compared perform well on the first synthetic dataset. However, some data points belonging to the middle cluster are incorrectly grouped as the right cluster for the second dataset when using k-mean and AC. This is because the middle cluster has a wider spread, making the centroid far away from some points in the same cluster. With DBSCAN, many data points from the middle cluster are regarded as outliers because density-based algorithms usually have difficulty when the intracluster distance (the distance between members of a cluster) differs greatly. For similar reasons, unsatisfied performance is obtained using the k means and DBSCAN on the third dataset. Mediocre results are also obtained using the AC algorithm, likely because Ward's linkage function merges the wrong groups. On the concentric circles dataset, poor performance is observed for the k-means and GMM algorithms because they can only group geometrically neighboring nodes in one cluster. Since the AC and DBSCAN algorithms can recursively group adjacent points into the same cluster, it is possible to group two distant points into the same cluster, so reasonable performance is achieved. Excellent results are obtained using our proposed MBMM on all synthetic datasets because the shape of a multivariate beta distribution is versatile. In particular, for the fourth dataset, because a multivariate beta distribution can be bimodal, the MBMM can group the data points in the outer circle as one cluster even though they are geometrically distant.

We visualize the PDFs by fitting the MBMM to the four synthetic datasets (Figure 5). The upper row shows the data points, and the bottom row shows the PDFs estimated by MBMM, which indeed fits these datasets adequately.

3.2 Comparison on the real datasets

Real datasets We used two open real datasets. The first dataset is MNIST, which includes grayscale images of handwritten digits. The size of each image

Table 2. Comparison of the clustering algorithms on the MNIST dataset and the breast cancer dataset (mean ± standard deviation)

	MNIST		breast cancer	
	ARI	AMI	ARI	AMI
MBMM	**.937** ± .000	**.884** ± .000	<u>.664</u> ± .000	<u>.558</u> ± .000
k-means	.913 ± .000	.850 ± .000	.491 ± .000	.464 ± .000
AC	<u>.933</u> ± .000	<u>.878</u> ± .000	**.689** ± .000	**.568** ± .000
DBSCAN	.854 ± .009	.745 ± .011	.554 ± .018	.447 ± .010
GMM	.909 ± .000	.846 ± .000	<u>.664</u> ± .000	<u>.558</u> ± .000

is 28×28. Since image pixels should have spatial correlations, directly using the pixel values as input features for clustering algorithms could be problematic. Eventually, we reduce the dimension of each image to 2 dimensions using the following procedure. First, we train a vanilla convolutional neural network (ConvNet) using the Fashion MNIST dataset (not the MNIST dataset). Then, we feed each MNIST image into the Fashion MNIST-trained ConvNet and take the hidden layer before the output (a vector with 512 neurons) as the image representation. Finally, we use a standard autoencoder to reduce the vector into 2 dimensions, which are the inputs of the clustering algorithms. Ultimately, we include only images of number 1 and number 9 in MNIST for experiments.

The second dataset, the breast cancer Wisconsin (diagnostic) dataset, consists of 569 instances. Each instance includes 32 attributes and a binary class label indicating the status of the tumor (benign or malignant). We download the dataset from the UCI Machine Learning Repository.

Results We compare clustered IDs with ground truth labels to calculate the adjusted Rand index (ARI) [12, 16] and adjusted mutual information (AMI) [15], two standard metrics for clustering evaluation. If a clustering result perfectly matches the referenced clusters (labels), both metrics return a score of 1. However, ARI and AMI are biased toward different types of clustering results: ARI prefers balanced partitions (clusters with similar sizes), and AMI prefers unbalanced partitions [13]. For a fair comparison, we report both metrics.

Table 2 shows the results. We repeat each experiment five times and report the mean ± standard deviation. We highlight each metric's first and second highest values in bold and underlined. For MNIST, the top 3 methods are our MBMM, followed by AC, and then k-means. For the cancer dataset, the best performance is achieved using the AC algorithm, followed by our MBMM and GMM. In general, our MBMM and AC perform best among all.

3.3 Distance between data points

As explained in Section 2.4, we define the distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ based on the KL divergence between $[\gamma_{i,1}, \ldots, \gamma_{i,C}]^T$ and $[\gamma_{j,1}, \ldots, \gamma_{j,C}]^T$. Consequently,

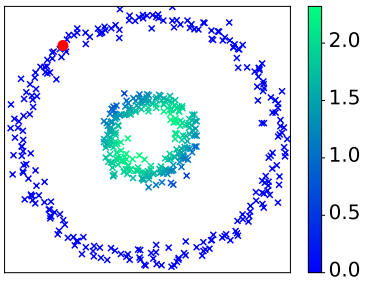

Fig. 6. Distance from red point to other points in concentric circles dataset

Table 3. Popular clustering algorithms and their properties

	MBMM	***k*-means**	**DBSCAN**	**AC**	**GMM**
Type	Distribution-based	Centroid-based	Density-based	Hierarchical clustering	Distribution-based
Assignment	Soft	Hard	Hard	Hard	Soft
Cluster shape	Versatile	Convex	Versatile	Versatile	Convex
Generative/Discriminative	Gen.	Discr.	Discr.	Discr.	Gen.

even if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are distant based on the Euclidean distance, they could still have a small distance score if $\gamma_{i,c} \approx \gamma_{j,c}$ for most c-s.

We illustrate the red data point's distances to others using concentric circles in Figure 6. Outer circle points are closer, showing that MBMM's distance function can assign small values even with large Euclidean distances.

4 Related Work

Clustering algorithms can be classified into four types based on how they partition data points: hierarchical, centroid-based, density-based, and distribution-based clustering.

Hierarchical clustering algorithms are top-down or bottom-up, corresponding to the iterative division of each cluster into smaller clusters and the aggregation of smaller clusters into larger clusters. Hierarchical clustering algorithms allow dynamically adjusting the cluster numbers. However, users must define the distance between not only data points but also between clusters, which could sometimes be counterintuitive. Well-known hierarchical clustering algorithms include agglomerative clustering (AC) and BIRCH [17].

Centroid-based algorithms represent each cluster using a centroid, assigning each data point to the closest cluster. However, these algorithms generate clusters

with convex shapes, eliminating the possibility of fitting a bimodal cluster. Well-known algorithms include k-means, k-medoids, k-medians, and k-means++.

Density-based algorithms determine clusters by assuming that densely distributed areas are clusters. Typical algorithms include DBSCAN [4] and OPTICS [1]. Although they discover clusters of various shapes, hyperparameter tuning can be time-consuming and heavily influence clustering results [7]. Additionally, density-based algorithms sometimes have difficulty clustering data points when the distances between different data points vary widely.

Distribution-based models assume that each cluster follows a probability distribution. One well-known is GMM, which assumes that each cluster follows a Gaussian distribution. Distribution-based models naturally generate synthetic data points by sampling a cluster ID and then one data point from the data distribution of cluster i. One problem with GMM is that the shape of each cluster must be convex since this is a fundamental property of a Gaussian distribution. As a result, GMM cannot differentiate between inner and outer circles.

Table 3 gives an overview of these clustering algorithms and their properties. Both MBMM and GMM are distribution-based models, which thus allow for soft clustering and synthetic generation of the data points. DBSCAN and AC allow non-convex cluster shapes in which each data point within a cluster is close to only a few data points from the same cluster. MBMM also supports non-convex cluster shapes due to the versatility of the multivariate beta distribution.

The beta mixture model has been studied in the bioinformatics and biochemical domains [5, 14]. However, they assumed that each cluster follows a standard beta distribution, which limits the practical usage of these models because each data point must be univariate. This constraint is probably due to the fact that the definition of a multivariate beta distribution is still ambiguous. Our study is more practical because we allow each data point to be multivariate.

5 Discussion

This paper proposes a new probabilistic model, the multivariate beta mixture model, for data clustering. We demonstrate MBMM's effectiveness by thorough experiments on synthetic and real datasets. Furthermore, MBMM is a generative model that allows for the generation of new data points. Compared to another famous generative clustering algorithm, the Gaussian mixture model, MBMM allows for a more flexible cluster shape. To ensure reproducibility, we have released our experimental code and encapsulated the MBMM module as a class with typical class methods supported by the clustering algorithms in `scikit-learn`, facilitating the utilization of the MBMM in various applications.

Although MBMM has these nice properties, we believe that different clustering algorithms should be used in combination to jointly partition data points for the following reasons. First, data clustering is ill-defined due to the lack of ground truth labels during both training and testing, making the choice of training objective and evaluation ad hoc [3]. Additionally, it has been shown that,

under reasonably general conditions, no single clustering algorithm can satisfy the three fundamental properties introduced in [8].

The capacity of MBMM is limited by the need for a positive correlation among all variates due to parameter b [6]. We plan to explore multivariate beta distributions allowing both positive and negative correlations based on [2].

References

1. Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J.: OPTICS: Ordering points to identify the clustering structure. ACM Sigmod record **28**(2), 49–60 (1999)
2. Arnold, B.C., Ng, H.K.T.: Flexible bivariate beta distributions. Journal of Multivariate Analysis **102**(8), 1194–1202 (2011)
3. Caruana, R., Elhawary, M., Nguyen, N., Smith, C.: Meta clustering. In: International Conference on Data Mining. pp. 107–118. IEEE (2006)
4. Ester, M., Kriegel, H., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Simoudis, E., Han, J., Fayyad, U.M. (eds.) International Conference on Knowledge Discovery and Data Mining. pp. 226–231. AAAI Press (1996)
5. Ji, Y., Wu, C., Liu, P., Wang, J., Coombes, K.R.: Applications of beta-mixture models in bioinformatics. Bioinformatics **21**(9), 2118–2122 (2005)
6. Jones, M.: Multivariate t and beta distributions associated with the multivariate f distribution. Metrika **54**(3), 215–231 (2002)
7. Karami, A., Johansson, R.: Choosing dbscan parameters automatically using differential evolution. International Journal of Computer Applications **91**(7), 1–11 (2014)
8. Kleinberg, J.: An impossibility theorem for clustering. Advances in neural information processing systems pp. 463–470 (2003)
9. Kotz, S., Balakrishnan, N., Johnson, N.L.: Continuous multivariate distributions, Volume 1: Models and applications. John Wiley & Sons (2004)
10. Kraft, D.: A software package for sequential quadratic programming. Wiss. Berichtswesen d. DFVLR (1988)
11. Lien, C.Y., Bai, G.J., Chen, H.H.: Visited websites may reveal users' demographic information and personality. In: International Conference on Web Intelligence. pp. 248–252. IEEE (2019)
12. Rand, W.M.: Objective criteria for the evaluation of clustering methods. Journal of the American Statistical association **66**(336), 846–850 (1971)
13. Romano, S., Vinh, N.X., Bailey, J., Verspoor, K.: Adjusting for chance clustering comparison measures. The Journal of Machine Learning Research **17**(1), 4635–4666 (2016)
14. Schröder, C., Rahmann, S.: A hybrid parameter estimation algorithm for beta mixtures and applications to methylation state classification. Algorithms for Molecular Biology **12**(1), 1–12 (2017)
15. Vinh, N.X., Epps, J., Bailey, J.: Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. The Journal of Machine Learning Research **11**, 2837–2854 (2010)
16. Wagner, S., Wagner, D.: Comparing clusterings: an overview. Universität Karlsruhe, Fakultät für Informatik Karlsruhe (2007)
17. Zhang, T., Ramakrishnan, R., Livny, M.: Birch: an efficient data clustering method for very large databases. ACM sigmod record **25**(2), 103–114 (1996)