KERNEL K-MACE CLUSTERING

Faizan Rahman, Soosan Beheshti

Ryerson University, Department of Electrical and Computer Engineering, Toronto, Canada
faizan.rahman@ryerson.ca, soosan@ee.ryerson.ca

ABSTRACT

We propose a kernel k-means based unsupervised clustering algorithm. The kernel k-means approaches require finding not only the Correct Number of Clusters (CNC) but also the optimum kernel function parameters in the clustering procedure. Existing index validation approaches use a criterion different from the k-means criterion to find the optimum CNC and also choose kernel parameter by trial and error. The proposed algorithm denoted by kernel k-Minimum Average Central Error (Kernel k-MACE), estimates the CNC while simultaneously providing the optimum value of the Gaussian kernel parameter. The advantage of the method in theory is in its consistency in using only one criterion for all the three steps of clustering, CNC estimation, and kernel function parameter estimation. A novel cluster initialization technique enables Kernel k-MACE to converge in less iterations compared to the existing approaches. Simulation results illustrate superiority of Kernel K-MACE over multiple state-of-the-art unsupervised clustering methods for both real data sets and self-generated data sets having 10% - 50% overlap. The method outperforms the existing methods by not only providing more accurate CNC estimates, but also by providing better clustering results evaluated by Adjusted Random Index (ARI) and Normalized Variation Index (NVI).

***Index Terms*—** Clustering, Kernel Functions, Kernel k-means, Overlapping Clusters

1. INTRODUCTION

Data clustering is one of the important topics of machine learning [1, 2, 3]. Clustering algorithms can be divided into partition and hierarchical clustering methods [4]. Partitional clustering methods are generally used for clusters following uniform or Gaussian distributions whereas hierarchy based clustering algorithms are mostly used for data containing arbitrary shaped clusters. K-means is one of the most important and widely used partition based clustering algorithms [5] which is the focus of this paper. The Correct Number of Clusters (CNC) is usually not known in real world applications and finding CNC is a very important challenge in the clustering procedure. There are standalone methods for estimating the number of clusters in a data set that are used prior to the use of partition or hierarchical clustering method [6]. Many existing algorithms address this issue by attempting to cluster the data set simultaneously along with estimating the number of clusters. These algorithms are known as fully unsupervised clustering algorithms [7]. On the other hand data sets with overlapping clusters in input space can usually be separated in high dimensional space. This is the motivation behind the use of kernel functions in clustering approaches. kernel functions are used to transform a data set to high dimensional space which is also called feature space [8]. Note that kernel based clustering methods require the knowledge of the kernel function parameter in addition to the number of clusters. The kernel function parameter is important in governing the separability of clusters in feature space and its optimum value corresponds to the separation of data in feature space that results in the estimation of CNC. There are several methods that cluster data using kernel functions while also estimating the number of clusters in a data set. However, these methods use trial and error to obtain the optimum value of the kernel function parameter. Other kernel based clustering methods use internal validation indices including Gap, Calinski-Harabasz, Davies-Bouldin and Silhouette index not only to find the optimum kernel parameter but also to estimate the CNC [9].

The proposed algorithm in this paper denoted by Kernel k-MACE is able to cluster the data while finding the optimum value of the Gaussian kernel parameter and the CNC simultaneously. Kernel k-MACE has three major steps. In the first step we propose a method for initializing the kernel k-means algorithm for the purpose of fast and robust convergence, the second step uses the ACE criterion in recently developed validation index method denoted by k-MACE [12] [13], for optimum CNC calculation for a range of kernel parameters. This is done by lifting k-MACE into feature space. The third step uses the result of the previous step to find the optimum Gaussian kernel parameter and its associated optimum CNC based on the behavior of the ACE itself.

The paper is arranged as follows. In Section 2 the clustering problem is defined. Section 3 presents the proposed clustering algorithm. Section 4 contains the simulations and results and Section 5 has the concluding remarks.

2. CLUSTERING PROBLEM FORMULATION

Given a data set of length N, $x^N = [x_1, x_2, \ldots, x_N]^T$ where $x_i \in \mathbb{R}^{1\times d}$. Each data sample x_i is an array of length d and we have:

$$X = \bar{c}_X + \overline{W}_X \tag{1}$$

Each x_i is a sample of random variable X with mean $\bar{c}_X$ added to a zero mean Gaussian noise denoted by $\overline{W}_X$. The data set consists of $\bar{m}$ clusters where each data sample only belongs to one cluster:

$$x^N = \overline{C}_1 \cup \overline{C}_2 \cup ... \cup \overline{C}_{\overline{m}} \tag{2}$$

Data samples belonging to each cluster C_j can be represented as

$$x_j = \bar{c}_{x_j} + \mathcal{N}(0, \overline{\Sigma}_{\bar{c}_{x_j}}) \tag{3}$$

The jth cluster has N_j elements. Using the available data of length N, the goal of the clustering method is to find the CNC ($\bar{m}$) and assign the data samples to the proper clusters.

3. KERNEL K-MACE CLUSTERING METHOD

Kernel k-MACE clustering is performed in three steps that will be explained in this section. Firstly, the observed data is transformed to feature space. The Data set in feature space is given by $\phi^N = [\phi_1, \phi_2, \ldots, \phi_N]^T$ where $\phi_i \in \mathbb{R}^{1\times N}$. Each data sample ϕ_i is an array of length N, and each element of ϕ_i represents a feature. Data samples in feature space (ϕ_is) are obtained by applying Singular Value Decomposition (SVD) on the kernel distance matrix of the data set [10]. In this case (1), (2), and (3) are transformed in the feature space as follows:

$$\Phi = \bar{c}_\Phi + \overline{W}_\Phi \tag{4}$$

$$\phi^N = \overline{C}_1 \cup \overline{C}_2 \cup ... \cup \overline{C}_{\overline{m}} \tag{5}$$

$$\phi_i = \bar{c}_{\phi_j} + V \tag{6}$$

where V is the additive noise with covariance matrix $\overline{\Sigma}_j$.

3.1. Step 1: Initial Cluster Assignment in Kernel k-MACE

In the conventional k-means algorithm, initially, data is randomly segmented and the algorithm converges to the final clustering result through an iterative procedure. kernel k-means algorithm also uses the same randomized initialization procedure. However, kernel k-means shows greater sensitivity to these initial values in converging to the correct clustering result. To avoid this problem we are expanding the approach in [11] which was initially proposed for the k-means algorithm. In this approach we choose the initial clusters by iteratively assigning data samples to clusters based on the distance between them in feature space. The procedure reduces sensitivity of kernel k-means to the initial values and in addition, reduces the number of iterations required for clustering convergence.

3.2. Step 2: CNC and Minimum Average Central Error (ACE)

ACE, denoted by Z_{s_m}, is defined as the average distance between the estimated cluster center and the true cluster center. In m clustering the data is clustered with k-means by setting the number of clusters to m:

$$\bar{c}(\phi_i) \to \phi_i \to \hat{c}_{\phi_{mj}}(\phi_i) \tag{7}$$

Therefore while $\bar{c}(\phi_i)$ has generated the data, the data after clustering is assigned to a center denoted by $\hat{c}_{\phi_{mj}}(\phi_i)$. Therefore ACE is the average distance between the unavailable true cluster center, $\bar{c}(\phi_i)$, and the estimated cluster center $\hat{c}_{\phi_{mj}}$ over the complete data set:

$$Z_{s_m} = \frac{1}{N}\sum_{j=1}^{m} Z_{s_{mj}}, \quad Z_{s_{mj}} = \|\bar{c}_{\phi_{mj}} - \hat{c}_{\phi_{mj}}\|_F^2 \tag{8}$$

where $||\cdot||_F$ is the Frobenius norm. While this desired criterion is not available, it is possible to use the available data error, denoted by Y_{s_m}, to find probabilistic bounds on this criterion. Data error is defined as the average distance between each data sample and its assigned estimated cluster center, and is also known as cluster compactness:

$$Y_{s_m} = \frac{1}{N}\sum_{j=1}^{m} Y_{s_{mj}}, \quad Y_{s_{mj}} = \|\hat{c}_{\phi_{mj}} - \phi_{mj}\|_F^2 \tag{9}$$

Therefore each element of this error is fully available. The expected value and variance of data error and ACE are connected and using the available sample of data error and it is possible to provide probabilistic bounds on expected value and variance of ACE [12][13]. The probabilistic upper-bound is:

$$\overline{Z_{s_m}} = E[Z_{s_m}] + \beta_N\sqrt{var[Z_{s_m}]} \tag{10}$$

where β_N is a confidence value. The estimate of this value is only function of the available data error (details in [12][13]). This criterion is calculated for values of m in the range of possible number of clusters, $[m_{min}, ..., m_{max}]$ and the value that minimizes the criterion is chosen as the estimate of CNC:

$$\hat{m} = arg\min_m(\overline{Z_{s_m}}) \tag{11}$$

3.3. Step 3: Optimum Gaussian Kernel Parameter

The Gaussian kernel feature map is defined below:

$$\phi_{d,j}(x) = e^{-\frac{\|x\|^2}{2\sigma^2}}\frac{1}{\sigma^d\sqrt{d!}}\prod_{i=0}^{d} x_{j_i} \qquad where \quad x \in \mathbb{R}^{1\times d} \tag{12}$$

where σ_k is the Gaussian kernel parameter and x is a data sample from a data set with d dimensions and N samples and j enumerates over all selections of d elements of x [14]. To find the optimum σ_k we first find the clustering results for a range of values of $\sigma_k : [\sigma_{k_{min}}, ..., \sigma_{k_{max}}]$. We propose a method to obtain the optimum value of σ_k that corresponds to the correct clustering result. To obtain the best σ_k, we obtain the gradient of the straight line connecting the minimum $\overline{Z_{s_m}}$ at each σ_k and for each σ_k after the peak, we add the absolute value of the gradient between the previous and the next value of σ_k. The value of σ_k which corresponds to the maximum value of this sum is chosen and the corresponding $\hat{m}$ and clustering result are chosen as the correct result:

$$\max_{\sigma_k}\left(\left|\frac{\partial(\min(\overline{Z_{s_m}}(m-1 \to m, \sigma_k)))}{\partial\sigma_k} + \frac{\partial(\min(\overline{Z_{s_m}}(m \to m+1, \sigma_k)))}{\partial\sigma_k}\right|\right)$$
$$for \quad \sigma_k > \max_{\sigma_k}(\min(\overline{Z_{s_m}}(m, \sigma_k))) \tag{13}$$

As the value of σ_k increases, the minimum $\overline{Z_{s_m}}$ rises as the clusters do not have any structure for very small values of σ_k i.e. for $\sigma_k < 1$. For the value of σ_k at which $\hat{m}$ increases to the value of CNC, a rapid decrease in minimum $\overline{Z_{s_m}}$ occurs corresponding to the estimated value of σ_k and the correct clustering result. The pseudo-code for Kernel k-MACE is given in Algorithm 1. Note that the computational complexity of Kernel k-MACE remains the same as kernel k-means which is $O(N^2)$.

Algorithm 1 Kernel k-MACE Algorithm

```
Require: Estimate the number of clusters $\hat{m}$ and provide a clustering solution.
Input: Data set x = $[x_1, x_2, ..., x_N]$, range of m $[m_{min}, m_{max}]$ and range of values for $\sigma_k = [\sigma_{k_{min}}, ..., \sigma_{k_{max}}]$ with a fixed interval
Output: Estimated number of clusters $\hat{m}$, the clustering solution $[\hat{C}_1, \hat{C}_2, .., \hat{C}_{\hat{m}}]$ and the optimum value of $\sigma_k$
 1: for ($\sigma_k = \sigma_{k_1}; \sigma_k \leq \sigma_{k_{max}}; \sigma_{k_{++}}$) do
 2:     for ($m = m_{min}; m \leq m_{max}; m_{++}$) do
 3:         $[C_{m1}, C_{m2}, .., C_{mj}] = kernelkmeans(x, m)$ with non-random initial cluster assignments
 4:         for each cluster $C_{mj}$ $j = 1, .., m$ do
 5:             Solve cluster compactness $y_{smj}$ of cluster $C_{mj}$ using (9)
 6:         end for
 7:         Solve for total cluster compactness $Y_{sm}$ using (9)
 8:     end for
 9:     for ($m = m_{min}; m \leq m_{max}; m_{++}$) do
10:         Calculate the covariance of each cluster for each $m_i$
11:     end for
12:     Estimate the optimum $\overline{Z_{s_m}}$ and $\hat{m}$ using the cluster covariances and $Y_{s_m}$ using (10)
13:     $[\hat{C}_1, \hat{C}_2, .., \hat{C}_{\hat{m}}] = kernelkmeans(x, \hat{m})$ for each $\sigma_k$ with non-random initial cluster assignments
14: end for
15: Calculate the gradient of the minimum $\overline{Z_{s_m}}$ curve following the condition in (13) to obtain the optimum $\sigma_k$, the final clustering solution and $\hat{m}$
```

4. SIMULATIONS AND RESULTS

Here we compare the performance of Kernel k-MACE with kernel k-means based methods for both synthetic and real data sets. The method is compared to popular internal index validation methods such as Gap, Calinski-Harabasz, Davies-Bouldin and Silhouette [16, 17, 18, 19]. These validation indices have been used alongside kernel k-means. Clustering results have also been compared to other well-known fully unsupervised clustering methods including G-means, DBSCAN and k-MACE, the clustering scheme that inspired Kernel k-MACE [20, 21, 12]. Adjusted Random Index (ARI) and Normalized Variation Index (NVI) are the external clustering validation indices used to evaluate the results where both range between 0 and 1 [22][23]. The results in Tables 1 and 2 are provided in the format: $\hat{m} + std[\hat{m}], (ARI, NVI)$; where $\hat{m}$ and $std[\hat{m}]$ correspond to the estimate for CNC and its standard deviation and ARI and NVI are the respective ARI and NVI values. The values in bold font in the tables correspond to the most accurate estimate for CNC and the best ARI and NVI values. The kernel parameter is chosen such that logσ_k^2 in the range of -10 and 30 has been used with Kernel k-MACE as suggested in [15]. While Kernel k-MACE refers to the algorithm that uses the Gaussian kernel function, Kernel k-MACE using the Polynomial kernel function is also used in comparisons. Clustering results are generated from an average of 50 runs.

4.1. Synthetic Data Sets

Synthetic data sets S1 - S6 are shown in Figure 1. They are generated using Gaussian clusters with varying degrees of overlap. Each synthetic data set consists of 300 data samples and has a CNC of 6. Data sets S1 - S4 have almost no overlap. However, S5 and S6 have 30% and 50% overlap respectively. Data sets S2 - S6 have varying variance while data set S1 has a uniform variance between clusters. Typical behavior of $\overline{Z_{s_m}}$ against σ_k is shown in Figure 2. The figures are from the data set S1. The behavior of (13) in this figure shows how the optimum choice of kernel parameter is at the 10th red square from the right that occurs at log$\sigma_k^2 = 3$ and the optimum choice of CNC is shown in Figure 2 (top Figure) which corresponds to $\hat{m}$ = 6.

Clustering results for data sets S1 - S6 are shown in Table 1. Kernel k-MACE is the only method that is able to identify the CNC for data sets S1 - S5 and outperforms the other methods. A $std[\hat{m}]$ of 0 shows that the method is consistent over multiple runs due to the proposed cluster initialization method which results in better ARI and NVI values as compared to other methods for most data sets. Kernel k-MACE with Polynomial kernel can identify the CNC for data sets S1, S2 and S4 - S6. Kernel k-MACE is also able to produce the best clustering result for data sets S1 - S5. k-MACE is able to correctly identify the correct number of clusters in data sets S1 - S3 (which can be easily clustered in input space) but has problem identifying clusters with significantly different variances that overlap or are in very close proximity of each other. G-means and DBSCAN are not able to identify CNC for most data sets while validation index based methods produce mostly correct results but have a high $std[\hat{m}]$ due to random initialization of clusters for each run.

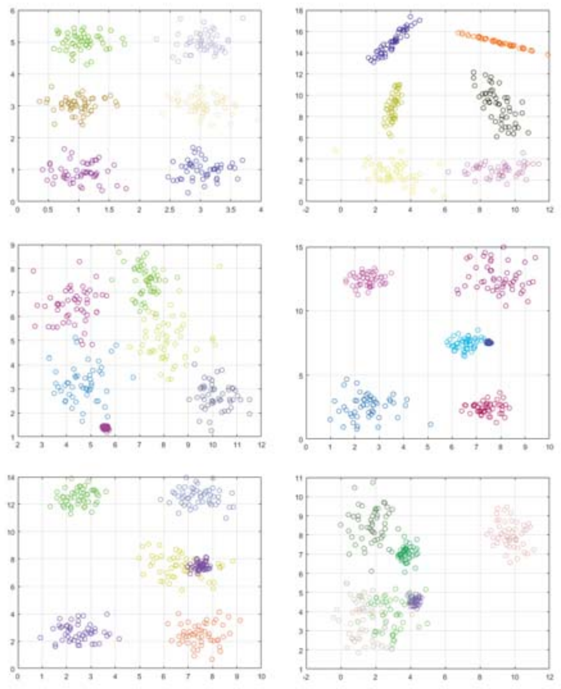


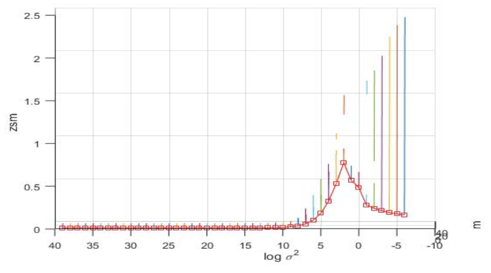
Fig. 1: Synthetic Data sets S1 - S6 with S1, S3, S5 on the left side and S2, S4, S6 on the right side

4.2. Real Data Sets

Clustering results for real data sets are shown in Table 2. The data sets have been obtained from the UCI machine learning repository website [1].

Kernel k-MACE provides the best estimate of $\hat{m}$ for Seeds, Iris, Wine, Glass and Soybean. Kernel k-MACE works well for clustering high dimensional data sets such as Wine, Glass and Soybean. G-means is unsuccessful in estimating CNC for all data sets while DBSCAN is able to identify CNC for Iris data set but fails for Thyroid and Vertebral data sets. Kernel k-means + GAP index stands out from index validation methods by generating accurate results for most data sets but has a high $std[\hat{m}]$ (> 1) for most data sets.

[1]http://archive.ics.uci.edu/ml/



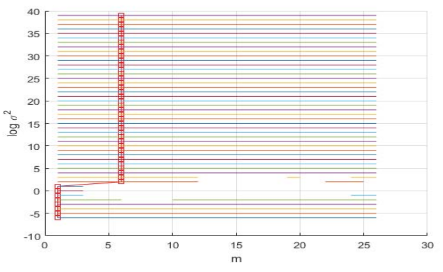


Fig. 2: Side and top view of plot showing $\overline{Z_{s_m}}$ for data set S1 for different values of σ_k

5. CONCLUSION

We proposed a kernel based clustering scheme denote by Kernel k-MACE that estimates the number of clusters and obtains the optimum value of the Gaussian kernel parameter simultaneously. A new cluster initialization technique improves the clustering results and overall consistency of the algorithm over multiple runs. Clustering results from synthetic data sets show that Kernel k-MACE can successfully cluster data sets containing overlapping Gaussian clusters with significantly different variances and outperforms the competitive methods.

6. REFERENCES

[1] H. Zhang, B. Pang, K. Xie, H. Wu. "An Efficient Algorithm for Clustering Search Engine Results", *International Conference on Computational Intelligence and Security*, 2006.

[2] S. Dolnicar. "Data-driven Market Segmentation in Tourism - Approaches, Changes Over Two Decades and Development Potential", *15th International Research Conference of the Council for Australian University Tourism and Hospitality Education*, pp. 346-360, 2006.

[3] K. Samuelowicz, J. D. Bain. "Revisiting Academics Beliefs about Teaching and Learning", *Higher Education*, pp. 299-325, 2001.

[4] H. G. Wilson, B. Boots, A. A. Millward. "A Comparison of Hierarchical and Partitional Clustering Techniques for Multispectral Image Classification", *IEEE International Geoscience and Remote Sensing Symposium*, 2002.

[5] A. K. Jain. "Data clustering: 50 years beyond K-means", *Pattern Recognition Letters*, vol. 31, pp. 651-666, 2010.

[6] M. Shahbaba, S. Beheshti. "Signature test as statistical testing in clustering", *Signal, Image and Video Processing*, vol. 10, pp. 1343-1351, 2016.

[7] A. Lorette, X. Descombes, J. Zerubia. "Fully Unsupervised Fuzzy Clustering with Entropy Criterion", *Proceedings 15th International Conference on Pattern Recognition*, vol. 3, pp. 986-989, 2000.

Data sets	S1	S2	S3	S4	S5	S6
$\overline{m}$ (CNC)	6	6	6	6	6	6
Kernel k-MACE	$\mathbf{6 \pm 0, (1, 0)}$	$\mathbf{6 \pm 0, (1, 0)}$	$\mathbf{6 \pm 0, (0.9, 0.2)}$	$\mathbf{6 \pm 0, (0.9, 0.1)}$	$\mathbf{6 \pm 0, (0.9, 0.1)}$	$5 \pm 0, (0.7, 0.3)$
Kernel k-MACE (Polynomial kernel)	$6 \pm 0, (0.9, 0.1)$	$6 \pm 0, (0.9, 0.2)$	$5 \pm 0, (0.7, 0.3)$	$\mathbf{6 \pm 0, (0.9, 0.1)}$	$6 \pm 0, (0.8, 0.2)$	$\mathbf{6 \pm 0, (0.7, 0.4)}$
k-MACE	$\mathbf{6 \pm 0, (1, 0)}$	$6 \pm 0, (0.9, 0.1)$	$6 \pm 0, (0.9, 0.2)$	$5 \pm 0, (0.8, 0.1)$	$5 \pm 0, (0.8, 0.1)$	$5 \pm 0, (0.7, 0.3)$
G-means	$\mathbf{6 \pm 0, (1, 0)}$	$12 \pm 0, (0.7, 0.2)$	$15 \pm 0, (0.7, 0.3)$	$14 \pm 0, (0.7, 0.3)$	$12 \pm 0, (0.8, 0.2)$	$7 \pm 0, (0.8, 0.3)$
DBSCAN	$7 \pm 0, (0.7, 0.4)$	$7 \pm 0, (0.7, 0.4)$	$7 \pm 0, (0.8, 0.2)$	$4 \pm 0, (0.3, 0.7)$	$4 \pm 0, (0.3, 0.7)$	$9 \pm 0, (0.4, 0.5)$
Kernel k-means + GAP	$5.8 \pm 1.5, (0.8, 0.2)$	$6.4 \pm 0.9, (0.9, 0.1)$	$6.2 \pm 0.8, (0.8, 0.2)$	$4.3 \pm 1.3, (0.6, 0.3)$	$3.9 \pm 1.7, (0.5, 0.4)$	$5 \pm 0.8, (0.6, 0.4)$
Kernel k-means + Calinski-Harabasz	$6.8 \pm 1, (0.9, 0.1)$	$9.5 \pm 3.5, (0.8, 0.2)$	$6.8 \pm 1.3, (0.8, 0.2)$	$5.2 \pm 0.8, (0.7, 0.2)$	$5.4 \pm 1.2, (0.7, 0.3)$	$5.9 \pm 1.1, (0.7, 0.3)$
Kernel k-means + Davies-Bouldin	$6.1 \pm 0.7, (0.8, 0.1)$	$7.1 \pm 1.2, (0.9, 0.1)$	$6 \pm 0.6, (0.8, 0.2)$	$5.4 \pm 1, (0.7, 0.2)$	$5 \pm 1, (0.7, 0.2)$	$5.1 \pm 1.6, (0.6, 0.4)$
Kernel k-means + Silhouette	$6.5 \pm 0.7, (0.9, 0.1)$	$6.8 \pm 0.8, (0.9, 0.1)$	$6.2 \pm 0.6, (0.8, 0.3)$	$5.3 \pm 1.2, (0.7, 0.3)$	$5.7 \pm 0.7, (0.7, 0.2)$	$5.5 \pm 1.6, (0.6, 0.4)$

Table 1: CNC Clustering results for synthetic data sets S1 - S6

Data sets	Seeds	Iris	Wine	Glass	Soybean	Vertebral	Thyroid
$\overline{m}(CNC)$	3	3	3	7	19	3	3
Dimensions	7	4	13	10	35	6	5
Kernel k-MACE	$\mathbf{3 \pm 0, (0.7, 0.5)}$	$\mathbf{3 \pm 0, (0.8, 0.3)}$	$\mathbf{3 \pm 0, (0.4, 0.7)}$	$\mathbf{7 \pm 0, (0.2, 0.7)}$	$\mathbf{18 \pm 0, (0.4, 0.4)}$	$2 \pm 0, (0.3, 0.7)$	$2 \pm 0, (0.1, 0.9)$
Kernel k-MACE (Polynomial kernel)	$\mathbf{3 \pm 0, (0.7, 0.5)}$	$3 \pm 0, (0.7, 0.5)$	$2 \pm 0, (0.3, 0.7)$	$4 \pm 0, (0.2, 0.7)$	$16 \pm 0, (0.4, 0.5)$	$\mathbf{3 \pm 0, (0.1, 0.9)}$	$2 \pm 0, (0.1, 0.9)$
k-MACE	$\mathbf{3 \pm 0, (0.7, 0.5)}$	$3 \pm 0, (0.7, 0.4)$	$\mathbf{3 \pm 0, (0.4, 0.7)}$	$5 \pm 0, (0.2, 0.8)$	$12 \pm 0, (0.5, 0.4)$	$2 \pm 0, (0.3, 0.7)$	$2 \pm 0, (0.1, 0.9)$
G-means	$4 \pm 0, (0.3, 0.7)$	$4 \pm 0, (0.5, 0.6)$	$2 \pm 0, (0.2, 0.7)$	$11 \pm 0, (0.1, 0.9)$	$16 \pm 0, (0.3, 0.8)$	$5 \pm 0, (0.3, 0.8)$	$7 \pm 0, (0.1, 0.9)$
DBSCAN	$2 \pm 0, (0.1, 0.9)$	$3 \pm 0, (0.5, 0.6)$	$1 \pm 0, (0.1, 0.9)$	$2 \pm 0, (0.2, 0.8)$	$4 \pm 0, (0.1, 0.5)$	$1 \pm 0, (0.1, 0.9)$	$1 \pm 0, (0.1, 0.9)$
Kernel k-means + GAP	$2.9 \pm 0.6, (0.6, 0.5)$	$3.2 \pm 0.6, (0.7, 0.4)$	$2.5 \pm 4.5, (0.1, 0.9)$	$6.4 \pm 1.5, (0.2, 0.8)$	$17.8 \pm 3.7, (0.4, 0.5)$	$1.7 \pm 2.2, (0.1, 0.9)$	$8.5 \pm 3.6, (0.2, 0.8)$
Kernel k-means + Calinski-Harabasz	$\mathbf{3 \pm 0, (0.7, 0.5)}$	$3.4 \pm 0.5, (0.7, 0.4)$	$9.6 \pm 7, (0.1, 0.9)$	$2.4 \pm 0.5, (0.2, 0.8)$	$4.1 \pm 2.3, (0.2, 0.7)$	$2.2 \pm 0.4, (0.2, 0.9)$	$4.7 \pm 2.9, (0.1, 0.9)$
Kernel k-means + Davies-Bouldin	$2 \pm 0, (0.5, 0.6)$	$2 \pm 0, (0.6, 0.4)$	$3.3 \pm 2.7, (0.1, 0.9)$	$6.9 \pm 3, (0.2, 0.8)$	$15.9 \pm 4.1, (0.4, 0.5)$	$2.6 \pm 2, (0.2, 0.9)$	$5.4 \pm 1.8, (0.3, 0.8)$
Kernel k-means + Silhouette	$2.1 \pm 0.3, (0.5, 0.6)$	$2 \pm 0, (0.6, 0.4)$	$2 \pm 0, (0.1, 0.9)$	$2.5 \pm 0.8, (0.2, 0.8)$	$14 \pm 3, (0.4, 0.5)$	$2.3 \pm 0.7, (0.2, 0.9)$	$\mathbf{3.8 \pm 0.9, (0.4, 0.8)}$

Table 2: CNC Clustering results for real data sets

[8] M. Filipponea, F. Camastrab, F. Masullia, S. Rovettaa. "A survey of kernel and spectral methods for clustering", *Pattern Recognition*, vol. 41, pp. 176-190, 2008.

[9] C. Gustavo, R. Jos, M. Manel. "Kernel Clustering for Knowledge Discovery in Clinical Microarray Data Analysis in Chapter 3: Kernel Methods in Bioengineering, Signal and Image Processing", *Idea Group Pub*, 2007

[10] A. Golts, M. Elad. "Linearized Kernel Dictionary Learning", *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 4, 2016

[11] K. A. Abdul Nazeer, M. P. Sebastian. "Improving the Accuracy and Efficiency of the k-means Clustering Algorithm", *Proceedings of the World Congress on Engineering*, vol 1, 2009

[12] E. W. Nidoy. "k-MACE Clustering for Gaussian Clusters", *M.A.Sc Thesis*, Ryerson University, 2016

[13] M. Shahbaba and S. Beheshti, "MACE-means Clustering", *Signal Processing*, vol. 105, pp. 216-225, December 2014

[14] A. Cotter, J. Keshet and N. Srebro, "Explicit Approximations of the Gaussian Kernel", *CoRR*, vol. 1109.4603, 2011

[15] K. Wu, S. Wang. "Choosing the Kernel Parameters for Support Vector Machines by the Inter-cluster Distance in the Feature Space", *Pattern Recognition, Elsevier*, vol 42, pp. 710-717, 2009

[16] Tibshirani, Robert, Walther, Guenther, Hastie, Trevor. "Estimating the Number of Clusters in a Data Set via the Gap Statistic", *Journal of the Royal Statistical Society*, vol. 63, pp. 411-423, 2001.

[17] T. Calinski, J. Harabasz. "A Dendrite Method for Cluster Analysis", *Communications in Statistics*, vol. 3, no.1, pp. 1-27, 1974.

[18] D. L. Davies, D. W. Bouldin. "A Cluster Separation Measure", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-1, no. 2, pp. 224-227, 1979.

[19] L. Kaufman, P.J. Rousseeuw. "Finding Groups in Data: An Introduction to Cluster Analysis", New York, NY: Wiley, 1990.

[20] Z. Zhao , S. Guo , Q. Xu , T. Ban. "G-means: a clustering algorithm for intrusion detection", *Proceedings of the 15th international conference on Advances in neuro-information processing*, pp. 563-570, 2008.

[21] M. Ester, H. Kriegel, J. Sander, X. Xu. "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise", *KDD-96 Proceedings*, 1996.

[22] W. Rand. "Objective Criteria for the Evaluation of Clustering Methods.", *Journal of the American Statistical Association*, vol. 66, no. 336, pp. 846-850, 1971

[23] R. Reichart, and A. Rappoport. "The NVI Clustering Evaluation Measure", *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pp. 165-173, 2009