Some properties of the classification algorithms using ensemble kernels

N Odinokikh\textsuperscript{1} and V Berikov\textsuperscript{1,2}

\textsuperscript{1} Novosibirsk State University, Novosibirsk, Russia
\textsuperscript{2} Sobolev Institute of Mathematics SB RAS, Novosibirsk, Russia

E-mail: n.odinokikh@gs.nsu.ru, berikov@math.nsc.ru

Abstract. In this paper, we study the properties of the KCCE algorithm (Kernel-based Classification with Cluster Ensemble) proposed in our previous publication. Different strategies for choosing the ensemble weights are investigated from the point of view of their influence on the quality and speed of the algorithm. We perform numerical experiments with the algorithm on real datasets from the UCI repository under conditions of artificially added noise at different levels.

1. Introduction

In the past decades, a large number of methods for solving the classification problems were proposed [1]. However, there is no method that could be used to effectively get a solution to all types of problems, especially, when the data are subject to noise distortions, outliers; with the presence of complex data structures.

One of the main tools used to solve the classification problems is a type of algorithms based on a linear partition of the feature space into the areas containing objects of the same class. In the case when the classes are linearly inseparable, a kernel trick is used for the transition into space in which objects become linearly separable. However, there are many different ways to choose a function for this transformation [2].

The classification problem in question can be formalized as follows. Suppose we are given a data set

\[ \mathbf{X} = \{x_1, \ldots, x_N\}, \]

where \( x_i \in \mathbb{R}^d \) is the feature vector for the \( i \)th observation taken at random from some statistical population following \( iid \) assumption, \( i = 1, \ldots, N \), \( N \) is the total sample size, and \( d \) is the dimensionality of the feature vector.

Let an additional set

\[ \mathbf{Y} = \{y_1, \ldots, y_M\} \]

of the class labels be given. The class labels are known only for the training set \( \mathbf{X}_{\text{train}} = \{x_1, \ldots, x_M\} \), where \( M \) is the training sample size, \( M < N \). It is necessary to predict in some optimal way the class labels for observations from the test set \( \mathbf{X}_{\text{test}} = \{x_{M+1}, \ldots, x_N\} \).
In our recent paper [3], a method for a classifier learning was proposed. It is based on the properties of the weighted averaged co-association matrix (WACM). This matrix can be obtained in several ways. Often it is constructed before the main steps of the algorithm from the combining results of different clustering algorithms [4]. This approach is explained below.

Let us assume we have obtained a collection of the base clustering partitions \( \{ P_l \}_{l=1}^L \), where \( L \) is the total number of partitions, \( P_l = \{ C_l,1, \ldots, C_l, K_l \} \), \( C_l,k \subset X, C_l,k \cap C_l,k' = \emptyset, K_l < N \). For each \( P_l \), we determine \( N \times N \) matrix
\[
H_l = (h_l(i,j))_{i,j=1}^N
\]
where
\[
h_l(i,j) = I[c_l(x_i) = c_l(x_j)]
\]
the entry \( h_l(i,j) \) of this matrix indicates if a pair of the points \( x_i, x_j \) belong to the same cluster: \( c_l(x_i) = c_l(x_j) \), where \( I(\cdot) \) is the indicator function \((I[true] = 1, I[false] = 0)\), \( c_l(x) \) is a label assigned to \( x \). Further, let us define the weighted averaged co-association matrix
\[
H = (H(i,j))_{i,j=1}^N
\]
as
\[
H(i,j) = \sum_{l=1}^L w_l H_l(i,j)
\]
where \( w_1, \ldots, w_L \) are the weights of partition versions, \( w_l \geq 0, \sum w_l = 1 \). Practically, the weights must reflect the “importance” of each clustering result in the ensemble and be dependent on some evaluation functions (cluster validity indices, diversity measures) \( \Gamma \): \( w_l = \gamma_l / \sum_{l'=1}^L \gamma_{l'} \), where \( \gamma_l = \Gamma(l) \) is an estimate of the clustering quality evaluated with \( \Gamma \) for the \( l \)th partition (we assume that the value of \( \Gamma \) increases with the improvement in the clustering quality). The procedure of choosing \( \Gamma \) can affect the overall quality of the algorithm.

The elements of WACM can be considered as degrees of similarity between objects in the feature space obtained after some implicit transformation of input data. The learned metric is used for a more accurate classification using a kernel-based methodology. The low-rank representation of the WACM allows for increasing the processing rate and memory saving.

There are several different approaches to the generation of ensemble elements and a combination of base clusterings [5, 6]. Individual clustering partitions can be obtained for a single algorithm by random sampling of its parameters (a homogeneous ensemble) or by the usage of several completely different algorithms in their parameter domains (a heterogeneous ensemble). Quite often, further selection of the most “diverse” partitions allow one to obtain better results in comparison with a full ensemble pool.

The final clustering partition can be found with the use of such methodologies as consensus clustering, graph-theoretic algorithms, evidence accumulation approach [5, 6]. To find the final partition, one may use a procedure based on a pairwise similarity, for example, the hierarchical agglomerative clustering, spectral clustering or minimum spanning tree methodology.

In this paper, we compare different methodologies to construct the WACM and their effect on the final results.

The rest of the paper is organized as follows. Section 2 considers the classification problem using the kernel-based learning method with the WACM as a kernel. Some theoretical properties of the considered methodology are given. The algorithm KCCE is presented. In Section 3, we describe several different methods of choosing the weights for the KCCE and analyze the characteristics of those methods. Section 4 describes the results of several numerical experiments that demonstrate the difference between these methods. The final section summarizes this study and discusses its findings.

### 2. Properties of the WACM

In this section, we consider some properties of the WACM related to its usage in the kernel-based methods. We also describe the KCCE algorithm, which is presented in [3].
Property 1. The WACM admits a low-rank representation in the form:

\[ H = \sum_{l=1}^{L} w_l A_l A_l^T, \]  

(2)

where \( L \) is the total number of clustering partitions, \( A_l \) is \( N \times K_l \) cluster assignment matrix for the \( l \)th partition: \( A_l(i,k) = 1 \) if \( c(x_i) = k \), \( i = 1,\ldots,N \), \( k = 1,\ldots,K_l \).

Property 2. The entries of the matrix \((1 - H) = (1 - H(i,j))_{i,j=1}^{N}\) satisfy the properties of semi-metrics.

Property 3. The WACM is symmetric positive definite.

The proofs of the presented properties can be found in [3]. According to those properties and the Mercer Theorem, the WACM can be used as a kernel matrix [2]. The algorithm KCCE (Kernel-based Classification with Cluster Ensemble) can be presented as follows:

Algorithm KCCE

Input:
– dataset \( X \), consisting of the training part \( X_{\text{train}} \) and the testing part \( X_{\text{test}} \); a set of the class labels \( Y \).
– \( L \): the number of base clustering partitions obtained with the clustering algorithm \( \mu \);
– \( \Omega \): a set of allowable parameters (algorithm settings) of \( \mu \);
– evaluation function (quality/diversity measure) \( \Gamma \).

Output:
– Predicted class labels for \( X_{\text{test}} \).

Steps:
1. Create \( L \) versions of clustering partitions of \( X \) with the algorithm \( \mu \) with parameters randomly chosen from \( \Omega \); calculate the cluster assignment matrices \( A_1,\ldots,A_L \).
2. Determine the weights \( w_1,\ldots,w_L \) of partition variants using \( \Gamma \);
3. Find a classifier using any chosen kernel-based classifier with the matrix \( H = \sum_{l=1}^{L} w_l A_l A_l^T \) as input kernel matrix;
4. Classify the test sample \( X_{\text{test}} \) using the found kernel classifier and the kernel matrix \( H \);
end.

Let us note that any clustering algorithm, for example, k-means, can be used as an algorithm \( \mu \) in KCCE. At step 3, examples of the used kernel-based classifiers are SVM or nearest neighbour (1NN) classifier, in which similarities between the data points are defined using the kernel matrix \( H \).

Let us define \( z(x_i,x_j) \) to be equal 1 iff \( x_i \) and \( x_j \) belong to the same class, otherwise \( z(x_i,x_j) = 0 \). Then the following statement is valid.

**Theorem 1.** Let \( \forall l \in \{1,\ldots,L\} \) and \( \forall i,j \in \{1,\ldots,N\} \) the inequalities hold:

\[ P[H_l(i,j) = 1|z(x_i,x_j) = 1] > \frac{1}{2} \]
\[ P[H_l(i,j) = 0|z(x_i,x_j) = 0] > \frac{1}{2} \]

Then for any object \( x \), the probability of a wrong classification with the KCCE in combination with 1NN converges to zero as \( L \to \infty \). The proof of this theorem can be found in [3].

In this paper, we compare the classification results obtained only with equal weights [3], with the results obtained by other different strategies of choosing weights for partition versions.
3. The weight selection strategies

In the proposed algorithm, there are many different parameters that may affect the quality of the results. In particular, it is possible to choose different clustering algorithms, as well as their parameters, the number of elements in the ensemble, a kernel classification algorithm, and the weights with which various clustering algorithms are averaged [7]. In this section, we will consider various strategies for choosing weights for the averaged matrix.

As a default method for comparison, we will always use 
\[ w_l = \frac{1}{L}, \quad l = 1, \ldots, L. \]

3.1. The Silhouette score

There are many different measures that determine the quality of machine learning algorithms for supervised learning, such as logloss, precision, or recall. However, the task of assessing the quality of algorithms for unsupervised learning problems is much harder since this task has no objectively correct solution. It is necessary to find an approach for evaluating the results without knowledge of the true classes of objects, only using information about a relative position of objects in a given space.

One of the known quality measures for the clustering algorithms is the Silhouette score (hereinafter referred to as silhouette) [8]. It is based on a relative difference between the distance inside clusters and the distance between clusters.

**Intracluster distance** is the average distance between all the points that lie in the same cluster with each other:

\[ ID = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{|C_i| - 1} \sum_{j \neq i, j \in C_i} d(i, j), \]

where \( d(i, j) \) is the distance between the points \( x_i \) and \( x_j \); \( C_i \) is the set of objects in one cluster with the point \( x_i \). In the case of isolated points (only one point lies in the cluster), it gives a zero contribution to the \( ID \) and will be considered as 0. This value can be considered to be an indicator of the proximity of points inside each cluster.

**Intercluster distance** is the average distance between the points lying in the nearest different clusters:

\[ ED = \frac{1}{N} \sum_{i=1}^{N} \min_{i \neq j} \frac{1}{|C_j|} \sum_{j \neq i, j \in C_j} d(i, j). \]

In the case when there is only one cluster, the intercluster distance is considered to be 0. This value can be interpreted as an indicator of how clusters are well separated from each other.

The clustering **silhouette** is the quantity:

\[ S = \frac{ED - ID}{\max(ED, ID)}. \]

In general, the silhouette can be considered as an indicator of the quality of clustering. This value lies on the interval \(-1 \leq S \leq 1\). The closer the value to 1, the better clustering. A negative value of the silhouette can be interpreted as a case of strongly intersecting clusters.

We take weights to be positive values between 0 and 1 that will sum up to 1, so we will construct weights for partition versions as

\[ s_l = \frac{1 + S_l}{\sum_{l'=1}^{L}(1 + S_{l'})}, \]
where $S_l$ is the Silhouette score for the $l$th partition version.

3.2. The Calinski-Harabasz Index

The Calinski-Harabasz Index is another way of evaluating the quality using the relative position of clusters among themselves, based on the intracluster and intercluster covariance matrices.

Further, we introduce the following notation: let $K$ be the number of clusters obtained as a result of the algorithm. Then for the cluster $k$, we define the total number of points in the cluster as $n_k$. Let $c_k$ be the centroid of the cluster, and $C_k$ be the set of all objects in the cluster $k$.

**Intracluster covariance matrix** is defined as

$$ W = \frac{1}{N - K} \sum_{k=1}^{K} \sum_{x \in C_k} (x - c_k)(x - c_k)^T, $$

and **Intercluster covariance matrix** as

$$ B = \frac{1}{K - 1} \sum_{k=1}^{K} n_k(c_k - c)(c_k - c)^T, $$

where $c$ is the common centroid, i.e., the centroid for all objects in the dataset.

Then the Calinski-Harabasz index is [9]:

$$ I = \frac{tr B}{tr W}. $$

This index shows how much the spread between clusters is larger than the scatter of points inside them. The index $I$ can take values between 0 and $+\infty$.

In the case of a single cluster, we consider this index to be equal to zero.

Values for each Calinski-Harabasz index may be unbounded, so we normalize it and construct weights for partition versions as

$$ i_l = \frac{\arctg(I_l)}{\sum_{l'=1}^{L} \arctg(I_{l'})}, $$

where $I_l$ is the Calinski-Harabasz index for the $l$th partition.

3.3. The Density Based Clustering Validation (DBCV)

The quality measures based on the distances between objects inside and between clusters can show good results in the tasks where one can divide data into clusters whose convex combinations do not intersect and are at a fairly large distance from each other. However, in the cases of more complex partitions, such methods perform rather poorly. It can be seen in Figure 1, where both Silhouette and the Calinski-Harabasz indices gave poor results.

In this case, a different approach might be helpful. One of such quality measures is a measure called Density-Based Clustering Validation (DBCV) [10]. A way to evaluate this measure is based on the DBSCAN algorithm, which allows one to get a good result even with clusters of complex shapes. On the other hand, the evaluation using this method takes more time than the previously given analogs. A more in-depth description of the DBCV metrics can be found in [10]. It is important to note that the DBCV index can be a number between $-1$ and 1, so we construct weights for partition versions similar to the Silhouette score

$$ d_l = \frac{1 + D_l}{\sum_{l=1}^{L} (1 + D_l)}, $$
where \( D_l \) is the DBCV measure for the \( l \)th partition.

A comparison between these quality evaluation strategies can be seen in Figures 1 and 2.

**Figure 1.** Different quality measures on linearly unseparable data. Two versions of partition are shown; clusters are in different colors. For each version, values of the Silhouette score, the Calinski-Harabasz index and the DBCV measure are given.

**Figure 2.** Different quality measures on linearly separable data. Three versions of partition are shown; clusters are in different colors. For each version, values of the Silhouette score, the Calinski-Harabasz index and the DBCV measure are given.

4. The Experimental Comparison of Quality Measures

Several experiments with KCCE on different datasets were conducted to find out how the choice of a quality measure in different strategies of selecting weights affects the final classification accuracy.

In these experiments, we have also added a normally distributed noise. The tests were carried out without noise, and then artificially generated normal noise was added to later experiments. The experiments were performed with 10-fold cross-validation. The base classification algorithm for the KCCE was SVM. Clustering algorithms were k-means with different initializations and the number of clusters from the interval \( 2 \leq L \leq 30 \). The resulting accuracy and computing time are presented in Tables 1-3. Datasets and their characteristics can be found in the UCI
repository [11]. Identifiers for the datasets are given in Tables 1-3. Different columns represent different algorithms, and different rows describe different noise levels. The best values in each category are highlighted.

As is seen from the tables, the choice of the best measure strongly depends on the task. Most of the time, it is better to use a special strategy rather than equal weights, even if sometimes it takes an additional computing time. In particular, the Silhouette index works pretty well on noisy data. However, on more complex datasets, the Calinski-Harabasz Index and the Density-Based Clustering Validation measure show a better overall quality.

Table 1. Results of experiments on public dataset Wine [12].

| Noise std | Equal | Silhouette | CHI   | DBCV   |
|-----------|-------|------------|-------|--------|
| 0         | 0.9832| 0.9662     | 0.9664| 0.9775 |
| 0.25      | 0.9775| 0.9271     | 0.9549| 0.9464 |
| 0.5       | 0.9217| 0.9216     | 0.9212| 0.926  |
| 0.75      | 0.8314| 0.8595     | 0.8032| 0.8437 |
| 1.0       | 0.7916| 0.747      | 0.725 | 0.803  |
| 1.25      | 0.6743| 0.7246     | 0.7021| 0.702  |
| 1.5       | 0.5559| 0.6229     | 0.5733| 0.5983 |
| 1.75      | 0.5007| 0.5959     | 0.5566| 0.5358 |
| AvgTime (s)| 7.282 | 7.903      | 8.175 | 36.522 |

Table 2. Results of experiments on public dataset Iris [13].

| Noise std | Equal | Silhouette | CHI   | DBCV   |
|-----------|-------|------------|-------|--------|
| 0         | 0.94  | 0.9467     | 0.9533| 0.9467 |
| 0.25      | 0.9067| 0.94        | 0.94  | 0.9333 |
| 0.5       | 0.8333| 0.78       | 0.8133| 0.8933 |
| 0.75      | 0.7733| 0.7467     | 0.72  | 0.78   |
| 1.0       | 0.6533| 0.7067     | 0.6133| 0.6533 |
| 1.25      | 0.6333| 0.58       | 0.6067| 0.64   |
| 1.5       | 0.6333| 0.5467     | 0.5467| 0.6333 |
| 1.75      | 0.5133| 0.5333     | 0.48  | 0.5267 |
| AvgTime (s)| 6.469 | 7.185      | 7.051 | 28.915 |

Table 3. Results on public dataset Breast Cancer [14].

| Noise std | Equal | Silhouette | CHI   | DBCV   |
|-----------|-------|------------|-------|--------|
| 0         | 0.9714| 0.9728     | 0.9714| 0.9714 |
| 0.25      | 0.9657| 0.9628     | 0.9685| 0.9635 |
| 0.5       | 0.9572| 0.9542     | 0.9571| 0.9628 |
| 0.75      | 0.9541| 0.9471     | 0.9471| 0.9522 |
| 1.0       | 0.9185| 0.9299     | 0.9313| 0.927  |
| 1.25      | 0.9027| 0.8984     | 0.8984| 0.9085 |
| 1.5       | 0.8712| 0.8813     | 0.8884| 0.8812 |
| 1.75      | 0.8226| 0.8441     | 0.8655| 0.8303 |
| AvgTime (s)| 10.63 | 14.265     | 12.273| 93.004 |
5. Conclusion and discussion
In this research, we have analyzed different approaches to the choice of weights for the kernel-based classifier with a cluster ensemble. We have also demonstrated and compared them on some publicly available datasets under the presence of normally distributed noise.

There are several arguments in favor of choosing different strategies depending on the problem. The selection of identical weights is often very naive because the algorithm does not see any difference between “good” and “bad” clustering results. In the end, those results can affect the overall quality of the algorithm, making it less reliable than choosing weights with a more complex strategy. Experimental results show that the choice of weights can improve the classification accuracy. As is shown in this paper, the noise distortions can also make an impact on the optimal choice of weights.

Of course, there are also arguments against a specific choice of weights as compared with the equal-weight distribution. The analysis of the best strategy can take a lot of time, and some of those methods (i.e. the DBCV) can also take a lot of time to finish even on relatively small datasets. On large datasets, such a procedure would be unacceptable in terms of given computing resources. It is possible that such a procedure would not grant a noticeable improvement, or sometimes (as is shown in some experiments), we can have better results with equal weights.

The authors are planning to continue studying the properties of the KCCE method. We are going to generalize it for other tasks and other machine learning algorithms (decision trees, neural networks).

Acknowledgement
The work was partly supported by RFBR grants 18-07-00600 and 19-29-01175. The study was carried out within the framework of the state contract of Sobolev Institute of Mathematics (project no 0314-2019-0015).

References
[1] Singaravelan S, Arun R, Shunmugam D, Soundar K, Mayakrishnan R and Murugan D 2018 Analysis of classification algorithms on different datasets. Review of Innovation and Competitiveness 4 41-54
[2] Shawe-Taylor J, Cristianini N 2004 Kernel methods for pattern analysis. Cambridge University Press
[3] Odinokikh N and Berikov V 2019 Cluster ensemble kernel for kernel-based classification. SIBIRCON 2019 pp 670–674
[4] Berikov V, Karaev N and Tewari A 2017 Semi-supervised classification with cluster ensemble. SIBIRCON 2017 pp 245–250
[5] Ghosh J and Acharya 2011 A Cluster ensembles. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 1 305–315
[6] Vega-Pons S and Ruiz-Shulcloper J 2011 A survey of clustering ensemble algorithms. IJPRAI 337–372
[7] Berikov V 2014 Weighted ensemble of algorithms for complex data clustering Pattern Recognition Letters 38 99–106
[8] Rousseeuw P J 1987 Silhouettes: a graphical aid to the interpretation and validation of cluster analysis Journal of Computational and Applied Mathematics 20 53–65
[9] Calinski T and Harabasz J 1974 A dendrite method for cluster analysis. Communications in Statistics 1–27
[10] Moulavi D, Jaskowiak P A, Campello R J G B, Zimek A and Sander J 2014 Density-based clustering validation
[11] https://archive.ics.uci.edu/
[12] https://archive.ics.uci.edu/ml/datasets/wine
[13] https://archive.ics.uci.edu/ml/datasets/iris
[14] https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)