Full and Semi-Supervised k-Means Clustering Optimised by Class Membership Hesitation.

Piotr Płoński and Krzysztof Zaremba

Institute of Radioelectronics, Warsaw University of Technology,
Nowowiejska 15/19,00-665 Warsaw, Poland,
{pplonski,zaremba}@ire.pw.edu.pl

Abstract. K-Means algorithm is one of the most popular methods for cluster analysis. K-Means, as the majority of clustering methods optimise clusters in an unsupervised way. In this paper we present a method of cluster's class membership hesitation, which enables k-Means to learn with fully and partially labelled data. In the proposed method the hesitation of cluster during optimisation step is controlled by Metropolis-Hastings algorithm. The proposed method was compared with state-of-art methods for supervised and semi-supervised clustering on benchmark data sets. Obtained results yield the same or better classification accuracy on both types of supervision.

Keywords: k-Means, Semi-supervised clustering, Supervised clustering, Classification, Metropolis-Hastings algorithm

1 Introduction

Cluster analysis is one of the most used data mining techniques [6]. Clustering assigns similar data points into the same cluster, whereas separate different data points by assigning them to different clusters. Clustering is NP-hard problem and among clustering algorithms the most popular is k-Means [14], [10]. It is a simple algorithm that requires tuning of the k number of clusters and selecting optimal distance metric. There have been proposed many heuristic improvements to find optimal k [7], [13] and to designate distance metric [20], [5], [16]. The drawback of k-Means algorithm is its sensitivity to initial cluster centre values. Therefore, a careful seeding of k-Means is needed [3].

K-Means is originally an unsupervised learning algorithm. However, there were proposed several techniques to use it in a supervised and semi-supervised manner. They can be divided into three groups. The first group of methods upon labelled data is optimising the distance metric, which gives small distance for samples from the same class and separates by a large distance samples from different classes. After metric optimisation step, the k-Means in an unsupervised way is performed with learned metric [1], [20]. The other group of methods uses labelled data to compute initial centre values to k-Means algorithm [22], [4]. The last group of methods used labelled data to generate constrains. They control which points can be in the same cluster and which should be separated by assigning into different clusters. The constrains are used during k-Means learning [21], [5].

Herein, we present the method for controlling supervision process in k-Means. It is based on Metropolis-Hastings (MH) algorithm [15], [8] well known from Simulated Annealing (SA) method [11]. MH algorithm is used to simulate a hesitation of cluster's class membership during k-Means learning. In the supervision process both fully and partially labelled data can be used. SA was already used in k-Means algorithm focused on seed selection [17] or parameters tuning [23] rather than supervision control. Recently, we proposed a similar method for controlling learning of neurons in Self-Organising Maps [19].

2 Methods

Let's denote data set as $D = \{(\boldsymbol{x}_i, c_i)\}$, where $\boldsymbol{x}_i$ is an attribute vector, $\boldsymbol{x} \in \mathcal{R}^d$ and c_i is a discrete class number of i-th sample, $i = [1, 2, ..., N]$ and $c = [1, 2, ..., C]$.

2.1 K-Means Algorithm

K-Means is an unsupervised learning algorithm. However, it can be used for classification. The two methods that enable handling class labels in k-Means are described below. The original K-Means algorithm can be described in four steps:

1. Initialize k cluster's centre $\{\mu_1, \mu_2, ..., \mu_k\}$ with randomly selected values $\boldsymbol{x}_i \in D$.
2. Assign each data point $\boldsymbol{x}_i$ to the closest cluster ζ_h, $h = \underset{h}{\text{argmin}} ||\boldsymbol{x}_i - \mu_h||^2$.
3. Update each cluster centre μ_i as mean of the assigned points, $\mu_i = 1/|\zeta_i| \sum_{\boldsymbol{x}_i \in \zeta_i} \boldsymbol{x}_i$.
4. Repeat steps 2 and 3 until convergence.

In k-Means algorithm information about sample's class label is not used. However, after unsupervised learning for each cluster can be assigned class label based on major vote of sample's class, which belongs to the cluster. In testing phase, for input sample is designated a class label from the closest cluster. We will call this method vote-k-Means.

Another method for using unsupervised k-Means for classification is to assign clusters to the classes arbitrarily - usually the same number of clusters for each class. During learning for each sample only clusters with the same class as the sample's class are considered [9]. Therefore, clusters are updated only with samples from the same class. For a testing sample a label the same as class of the closest cluster is given. We will call this method a class-k-Means.

2.2 Seeded k-Means

The next method that uses labels is seeded-k-Means [4]. It uses labelled data to compute better initial values of cluster centres. In seeded-k-Means we assume that as algorithm input we have k disjoint sets $S = \{S_1, S_2, ..., S_k\}$, $S \subseteq D$, on which supervision is provided - for each $\boldsymbol{x}_i \in S$ is known a cluster ζ_l to which it belongs. The cluster centres are initialized as follows:

$$\mu_i = \frac{1}{|S_i|} \sum_{\boldsymbol{x}_j \in S_i} \boldsymbol{x}_j. \tag{1}$$

In [4] there is also an assumption that each cluster has at least one seed point. However, this condition is hard to be satisfied, and for clusters without any seed point a randomly selected value $\boldsymbol{x}_i$ can be used [22].

2.3 Proposed Method (MH-k-Means)

In the proposed method, for each cluster we compute a class membership in each iteration. Let's note it as $P_l(j)$, where l is cluster index and j is class number. For each sample a group of clusters is selected, which will take part in the closest cluster finding. Selection is described by a matrix T, where $T_l^i = 1$ means that cluster ζ_l will participate in learning, using i-th sample, $T_l^i = 0$ otherwise. Clusters for training are selected in two steps. At first we choose clusters having maximum probability for the class matching the class c_i of the input sample:

$$T_l^{i(1)} = \begin{cases} 1 & \text{if } \arg\max_j (P_l(j)) = c_i; \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

In the second step, remaining clusters with $T_l^{i(1)} = 0$ are considered. The decision on joining into the training with i-th sample is taken upon MH algorithm. The probability of joining is computed upon the following equation:

$$J_l^i = 1 - exp(-\rho P_l(c_i) t_{stop}/t), \tag{3}$$

where ρ is the parameter that controls the intensity of hesitation, $\rho \in [0, 1]$. The greater ρ, the more clusters are selected additionally to learning in the MH step. In eq.(3) the parameter t is a number of current algorithm iteration and t_{stop} is an overall number of algorithm iterations. The fact that t is presented in (3) ensures that clusters added during MH step will be selected less frequently at the end of learning process than at its beginning. This can be interpreted as a hesitation of the cluster, which decreases during the learning. To decide whether the MH decision is positive, we draw a random number a from a uniform distribution, $a \in [0, 1]$. The cluster will be added to the training group if a is smaller than J_l^i:

$$T_l^{i(2)} = \begin{cases} 1 & \text{if } a < J_l^i; \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

This procedure is repeated for each sample. We called $T_l^{i(2)} = 1$ as a positive MH decision. The final decision on cluster selection is a logical 'or' of the decisions $T_l^i = T_l^{i(1)} \vee T_l^{i(2)}$. For i-th sample the closest cluster ζ_h is computed as follows:

$$h = \underset{h}{\operatorname{argmin}} ||\boldsymbol{x}_i - \mu_h||^2 \wedge T_h^i = 1. \tag{5}$$

After the presentation of all samples new class membership probabilities are computed for each cluster:

$$P_l(h) = \frac{|\boldsymbol{x}_i \in \zeta_l \wedge \boldsymbol{x}_i \in c_h|}{|\boldsymbol{x}_i \in \zeta_l)}. \tag{6}$$

The procedure described above is repeated till algorithm's convergence. We have called the proposed algorithm MH-k-Means. The MH-k-Means algorithm described above works on fully labelled data. In case of partially labelled data, for samples without class label we assume that $T_h^i = 1$ for all clusters. Thus, all clusters participate in training. For labelled samples the procedure described above is used.

3 Results

To test performance of MH-k-Means method on fully labelled data, we will compare it to the Learning Vector Quantization algorithm (LVQ) [12], vote-k-Means, class-k-Means and seeded-k-Means. On partially labelled data sets, we will compare MH-k-Means to seeded-k-Means and vote-k-Means[1]. The comparison is made on 6 real data sets. We used data sets 'Wine', 'Ionosphere', 'Iris', 'Sonar', 'Spam' from the 'UCI Machine Learing Repository' [2] [2], and set 'Faces' are from the 'The ORL Database of Faces'[3]. Data sets are described in Table 1.

	Train examples	Test examples	Attributes	Classes	# clusters	MH ρ
Faces	320	80	50*	40	80	1
Sonar	166	42	60	2	36	0.5
Spam	3680	921	57	2	72	0.25
Iris	120	30	4	3	12	1
Ionosphere	280	71	34	2	24	0.25
Wine	142	36	13	3	12	1

Table 1: Description of data sets used to test performance, number of clusters used to each data set and optimal ρ in MH-k-Means. (*In 'Faces' data set, the number of attributes was reduced with PCA.)

In all experiments we train algorithms with number of iterations $t_{stop} = 200$. For LVQ we use learning rate $\eta_1 = 0.1$, exponentially decreasing to $\eta_{200} = 0.001$. All algorithms, except seeded-k-Means, were initialized with random samples. For seeded-k-Means we assigned available labelled samples to appropriate clusters reusing clustering from class-k-Means, in case of incomplete seeding of the cluster we used initialization with random sample [22]. For each data set, we

[1] We used only labelled data for cluster's class voting.
[2] http://archive.ics.uci.edu/ml/
[3] http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html

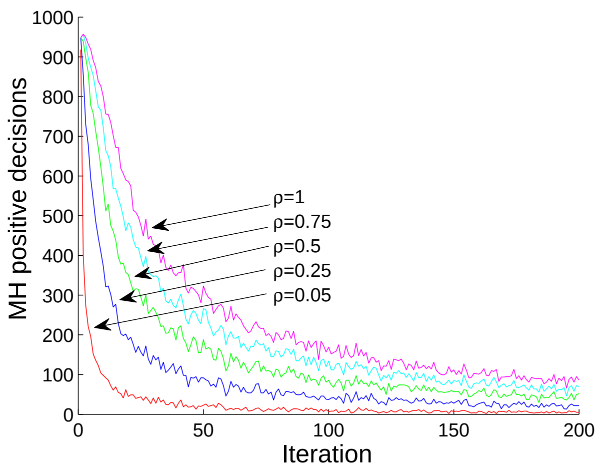

Fig. 1: Number of positive MH decisions in MH-k-Means algorithm taken in each training iteration for different ρ values on 'Iris' data set.

	LVQ	vote-k-Means	class-k-Means	seeded-k-Means	MH-k-Means
Faces	8.25±3.34	28.0±4.5	6.5±3.72	6.88±4.09	**4.0±2.34**
Sonar	**14.52±7.48**	26.9±8.1	15.48±4.92	17.38±5.73	**14.52±5.32**
Spam	**13.34±1.16**	18.43±1.25	14.18±1.77	17.13±2.19	14.47±1.3
Iris	4.0±2.11	3.33±3.51	4.33±2.74	4.67±3.58	**3.0±1.89**
Ionosphere	10.99±2.95	9.58±3.31	12.82±2.61	10.0±3.22	**8.73±2.38**
Wine	5.0±3.66	7.5±3.72	4.72±3.72	6.39±5.08	**4.44±2.68**

Table 2: Percent of incorrect classification on fully labelled testing subsets. Results are mean and σ over 10 runs.

arbitrarily chose the cluster number (selecting optimal cluster size is not in the scope of this paper), selected values are presented in Table 1. The total cluster number for each algorithm type is equal. For MH-k-Means the parameter ρ must be tuned. We checked several values of ρ, $\rho = \{0.05, 0.25, 0.5, 0.75, 1\}$ and for each data set an optimal value was selected by cross-validation. Selected ρ values are presented in Table 1. The impact on number of positive MH decision, depending on ρ value is demonstrated on 'Iris' set in the Fig.1. The greater ρ value is, the more positive MH decisions are made and the more frequently cluster takes part in training with the sample, of which class is different than its major class. For each data set we made 10 repetitions to avoid effect of local minima, each time training and testing subsets were redrawn. As an accuracy measure of clustering, we take a percentage of incorrect classifications - for each testing sample we compute the closest cluster and compare the labels. The mean results

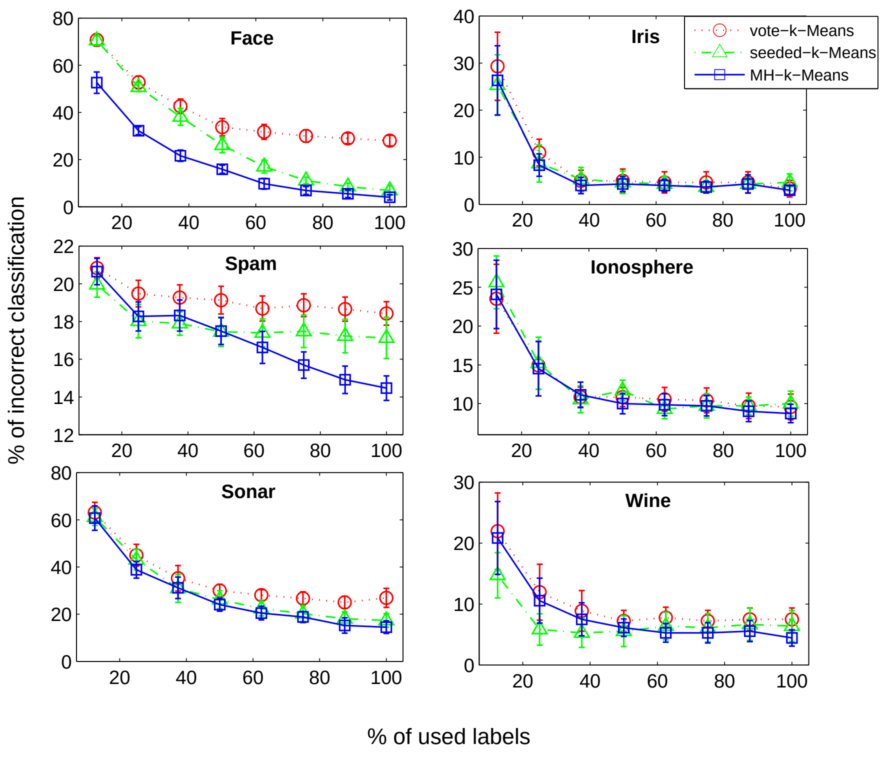

Fig. 2: Performance of vote-k-Means, seeded-k-Means and MH-k-Means on partially labelled data. Results are mean and σ over 10 runs.

on testing subsets for all the methods on fully labelled sets are presented in Table 2. With one exception the MH-k-Means achieves lower error rates than other methods. On 'Spam' data sets the best method was LVQ. The MH-k-Means obtained the greatest improvement over 'vote-k-Means' on 'Faces' data set. There are 40 classes (persons) in 'Faces' set, therefore it is difficult to compute good clusters without any information about class labels during learning. On this set, MH-k-Means uses the largest ρ, which means that cluster's class membership hesitates the most. On 'Sonar' data set, LVQ and MH-k-Means gain the same average of incorrect classification. However, MH-k-Means has lower standard deviation value. On 'Iris' set all methods give similar results, with slightly lower error of MH-k-Means. On this set and 'Ionosphere' the vote-k-Means accuracy is better than LVQ. They are simple sets, therefore without any supervision a good minima can be obtained. On 'Wine' set, class-k-Means is better than LVQ. The poorest accuracy on all data sets was obtained by vote-k-Means method. This was expected, as this method does not use the information about sample's class during the cluster's centre tuning. Seeded-k-Means have results slightly worse than class-k-Means, on all sets, except 'Ionosphere'. This is due to the fact that clusters obtained by class-k-Means were used in initialization of seeded-k-Means.

To test performance of the proposed MH-k-Means method on partially labelled data, we used only part of available labels in training subsets, in per cent $r = \{12.5, 25, 37.5, 50, 75, 87.5, 100\}$. The results of comparison are presented in Fig.2. The MH-k-Means is significantly better than other methods on 'Faces', 'Spam' and 'Sonar' data sets. On 'Iris', 'Ionosphere' and 'Wine' all methods seem to give similar results. The 'Faces', 'Spam' and 'Sonar' seem to be more complex in classification than other sets, therefore we observe that using information about class labels during learning gives better clustering. For simple data sets semi-supervised and unsupervised methods gain similar minima.

4 Conclusions

We present a novel method MH-k-Means for learning k-Means with fully and partially labelled data. In each iteration for every cluster a class membership is computed. Upon this, for each sample for the closest cluster finding a group of clusters with the same as sample's class is selected. What is more, the hesitation mechanism is introduced, which enables clusters with different class to take part in the closest cluster finding. The hesitation is based on Metropolis-Hastings algorithm, with hesitation intensity controlled by ρ parameter and current iteration number. The number of MH positive decisions decrease during learning, which can be interpreted as making clusters more confident. In case of partially labelled data, the clusters selection for learning is made only for labelled samples. For unlabelled samples all clusters participate in training. The proposed MH-k-Means method was compared to other state-of-art methods on classification tasks. The results confirm that proposed method obtains better or similar accuracy than other methods. Matlab implementation of the MH-k-Means algorithm is available at `http://home.elka.pw.edu.pl/~pplonski/mh_kmeans`. Future work will be focused on two aspects: testing what impact on MH-k-Means accuracy has the noise of class labels and the use of distance metric learning method for classification accuracy improvement[18].

References

1. Al-Harbi, S.H., Rayward-Smith, V.J.: Adapting k-means for supervised clustering, Applaied Intelligence vol.24, pp 219-226 (2006)
2. Asuncion, A., Newman, D.J.: UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences (2007)
3. Arthur, D., Vassilvitskii, S.: K-means++: The Advantages of Careful Seeding. In: Symposium on Discrete Algorithms (2007)
4. Basu, S., Banerjee, A., Mooney, R.J.: Semi- supervised clustering by seeding, In: Proceedings of the 19th International Conference on Machine Learning, pp 19–26 (2002)

5. Bilenko, M., Basu, S., Mooney, R.J.: Integrating constraints and metric learning in semi-supervised clustering, In: Proceedings of the 21th International Conference on Machine Learning, pp 81-88 (2004)
6. Du, K.-L.: Clustering: A neural network approach, Neural Networks vol.23, pp 89-107 (2010)
7. Hamerly, G., Elkan, C.: Learning the k in k-means, In: Neural Information Processing Systems Conference (2003)
8. Hastings, W.K.: Monte Carlo Sampling Methods Using Markov Chains and Their Applications. Biometrika vol.57, pp 97–109 (1970)
9. Hinton, G.E., Dayan, P., Revow, M.: Modeling the manifolds of images of handwritten digits, IEEE Transactions on Neural Networks vol.8, pp 65-74 (1997)
10. Jain, A.K.: Data clustering: 50 years beyond k-means, Pattern Recognition Letters vol.31, pp 651-666 (2010)
11. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by Simulated Annealing. Science vol.220, pp 671–680 (1983)
12. Kohonen, T.: The Self-Organizing Map. Proceedings of the IEEE, vol.78, pp 1464-1480 (1990)
13. Likas, A., Nikos, V., Verbeek, J.J.: The global k-means clustering algorithm, Pattern Recognition vol.36, pp 451-461 (2003)
14. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In: Proceeding of the 5th Berkeley symposium, pp 281–297 (1967)
15. Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E.: Equations of State Calculations by Fast Computing Machines. Journal of Chemical Physics vol.21, pp 1087–1092 (1953)
16. Olszewski, D.: Asymmetric k-Means Algorithm, Lecture Notes in Computer Science vol.6594, pp 1–10 (2011)
17. Perim, G.T, Wandekokem, E.D., Varejão, F.M.: K-Means Initialization Methods for Improving Clustering by Simulated Annealing, Lecture Notes in Artificial Intelligence vol.5290, pp 133–142 (2008)
18. Płoński, P., Zaremba, K.: Improving Performance of Self-Organising Maps with Distance Metric Learning Method, Lecture Notes in Computer Science vol.7267, pp 169-177 (2012)
19. Płoński, P., Zaremba, K.: Self-Organising Maps for Classification with Metropolis-Hastings Algorithm for Supervision, Lecture Notes in Computer Science vol.7665, pp 149–156 (2012)
20. Siriseriwan, W., Sinapiromsaran, K.: Attributes Scaling for K-means Algorithm Controlled by Misclassification of All Clusters, In: Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining, pp 220-223 (2010)
21. Wagstaff, K., Cardie, C., Rogers, S.: Constrained k-means clustering with background knowledge, In: Proceedings of the 18th International Conference on Machine Learning, pp 577–584 (2001)
22. Wang, X., Wang, C., Shen, J.: Semi-supervised k-means clustering by optimizing initial cluster centers, Lecture Notes in Computer Science vol.6988, pp 178-187 (2011)
23. Yang, W., Rueda, L, Ngom, A.: A Simulated Annealing Approach to Find the Optimal Parameters for Fuzzy Clustering Microarray Data, In: Proceedings of the 25th International Conference of Chilean Computer Science Society, pp 45-55 (2005)