INCREMENTAL LEARNING WITH SEMI-SUPERVISED K-MEANS

Frederico Damasceno Bortoloti* Patrick Marques Ciarelli†

*Department of Production Engineering
Federal University of Espirito Santo
Vitória, Espírito Santo, Brasil

†Department of Industrial Technology
Federal University of Espirito Santo
Vitória, Espírito Santo, Brasil

Email: frederico.bortoloti@ufes.br, patrick.ciarelli@ufes.br

Abstract— In many real-world tasks a lot of unlabeled data are collected over time and, although they may be useful to improve the quality of classification models, they are usually ignored. Semi-supervised learning techniques combine unlabeled and labeled data to capture more useful information about a particular task. On the other hand, an incremental learning technique can incorporate new information to an existing model, so that it can dynamically adapt its structure to follow the environment changes. K-means is a well known technique for unsupervised learning, which has adapted versions to supervised learning. K-means works by clustering information providing a node based structure. In order to unify the characteristics of both approaches, in this paper is proposed an incremental semi-supervised learning method called SSK-means, which is based on a supervised k-means method. Experiments were conducted using publicly available benchmark datasets. Results show the proposed method is promising.

Keywords— Semi-supervised learning, Incremental learning, K-means, k nearest neighbors.

1 Introduction

Traditional classifiers often use for training only datasets previously labeled. The great problem is that the amount of labeled data may be quite limited. Moreover, sometimes, it is not easy to obtain labeled data because the task of categorizing data is expensive, is time consuming and must be performed by experts. On the other hand, there are vast amounts of unlabeled data that are relatively easy to be collected and may be very useful for classification purposes, but they are normally neglected. Semi-supervised learning techniques treat this problem using a huge number of unlabeled data along with labeled data to create a classification or prediction model.

Another interesting method of learning is incremental learning. Incremental learning allows to incorporate new data to an already trained classifier, without the requirement to access the original data. An incremental technique does not need to be retrained from scratch to insert new data. In such techniques, learning occurs continuously over time and does not cease once available data have been exhausted (Giraud-Carrier, 2000). Incremental learning models are efficiently updated as new information comes.

An useful approach is to combine incremental learning with semi-supervised learning, so that they can complement each other. Although incremental supervised learning techniques learn as new data are collected, such methods need human intervention to label the data. On the other hand, semi-supervised learning methods are able to label the examples and use them for training, but they normally need to store the whole training data to obtain a new model. The combination of both techniques allows to obtain a system with a fast learning, self-adaptive, able to supply labels to examples, with the additional advantage of no need of human interference to perform the task of labeling the examples.

The objective of the present work is to obtain a method of incremental semi-supervised learning that efficiently adapts its structure. For this purpose, in this work is used a modified version of k-means. K-means is an unsupervised learning method that uses a very simple structure of clusters that represents groups of examples. The k-means structure can be easily updated with new data, so that it is able to dynamically adapt its knowledge to follow the environment changes.

Many supervised k-means have been proposed to confer a class to each cluster, and, to that effect, a set of weights is used for the features. In this work, the weights are initialized by a feature selection method using a chi-square metric. During training, the weights are optimized by a meta-heuristic technique. As distance metric a combination of weighted distance and Mahalanobis metric is used. In this paper were proposed additional procedures to k-means in order to obtain an incremental semi-supervised technique.

The proposed method, called semi-supervised k-means (SSK-means), was tested with public datasets and its results were compared to a decision tree classifier. Results of SSK-means with two different set of settings were also compared. Experimental results show the proposed method is promising.

This work has the following structure. In Sec-

tion 2 chi-square feature selection and supervised k-means are explained. The method proposed is described in Section 3. The experiments and results are presented in Section 4. Finally, the conclusions are exposed in Section 5.

2 Methods

2.1 Chi-square Feature Selection

Filter methods use the relationship between input and class attributes in examples to select important features. One method to assess the degree of involvement of attributes in classification is the chi-square metric (Liu and Setiono, 1995). The input attribute that has the largest chi-squared associated with the class attribute is the most correlated, therefore the most important. Attributes must be discrete, because this metric depends on a contingency cross-table of input attribute value and class value. Chi-square (χ^2) is calculated using the formula in Eq. 1.

$$\chi^2 = \sum_{i=1}^{V}\sum_{j=1}^{C}\frac{(O_{ij}-E_{ij})^2}{E_{ij}} \tag{1}$$

where V is the number of distinct attribute values, C is the number of classes, O_{ij} is the observed value in a cell contingency table and E is the expected value for that cell. To calculate the value E_{ij} of a cell, the formula in Eq. 2 is used.

$$E_{ij} = \frac{\left(\sum_{k=1}^{C} O_{ik} \times \sum_{l=1}^{V} O_{lj}\right)}{\sum_{k=1}^{C}\sum_{l=1}^{V} O_{lk}} \tag{2}$$

Iteratively, one can obtain the chi-square of all attributes, and put them in order of importance. The result of this procedure are the features in order of relevance.

2.2 Supervised k-means

K-means is a typical algorithm for unsupervised learning that takes examples and groups them in k groups or clusters. Each example is assigned to the closest cluster centroid based on the Euclidean distance to the cluster centroid. Initially, the cluster centroids are randomly obtained. The attributes are considered coordinates and each cluster centroid receives the mean values of the attributes of those examples assigned to it. The centroids are updated at each iteration, then, in the next iteration, the examples are reassigned to the closest cluster centroid. The algorithm converges when a given number of iterations is reached or the centroids are considered unchanged.

One adaptation of k-means for supervised learning is proposed in (Al-Harbi and Rayward-Smith, 2006). Changes applied to the traditional k-means to accomplish supervised learning are presented below.

Firstly, it is natural to assign a class to each cluster. One class may comprehend one or multiple clusters. Thus, members of each cluster have labels that can be compared to the class label assigned to the cluster, and an error can be computed. In order to assign a class to each cluster, the predominant class of the corresponding examples of each cluster is used.

Secondly, the metric to estimate the distance between examples and clusters is changed to weighted Euclidean distance. The weighted Euclidean distance is obtained by multiplying a weight w_i to each attribute $x^{\{i\}}$ squared difference to its mean value, as in Eq. 3. The purpose of the weights is to direct the example to the cluster with same class label, by altering the relevance of the attributes in the classification task. Therefore, they work as a feature selection mechanism.

$$\delta_w(x,y) = \sqrt{\sum_{i=1}^{n} w_i (x^{\{i\}} - \mu_i)^2}. \tag{3}$$

The modified k-means procedure with class label assignment and weighted Euclidean distance is presented in algorithm Alg. 1. The algorithm takes a set of examples X, their real classes t, the number of clusters k, the attribute weight vector w and the number of iterations n_{it} as inputs, and returns a set of clusters c, classes assigned to each example in y and the classification error e.

Algorithm 1 Pseudocode of modified k-means with class assignment and weighted distance

input: X, t, k, w, n_{it}
output: c, y, e
top:
Initialize $c \leftarrow$ random vector list
loop:
for $r \leftarrow 1$; $r \leq n_{it}$ **do**
  for $x_i \in X$ **do**
    calculate weighted distance to each cluster $c_j \in c$
    assign to the closest cluster
  end for
  for $c_j \in c$ **do**
    $Xc_j \leftarrow$ set of $x_i | x_i$ is closest to c_j
    recalculate cluster $c_j \leftarrow \frac{\sum_{i=1}^{m} x_i}{m} | x_i \in Xc_j$
  end for
end for
for $c_j \in c$ **do**
  $Xc_j \leftarrow$ set of $x_i | x_i$ is closest to c_j
  $m_j \leftarrow |Xc_j|$
  $tc_j \leftarrow$ set of $t_i | x_i$ is closest to c_j
  $class_j \leftarrow mode(tc_j)$
  $Xl_j \leftarrow$ set of $Xc_{ji} | tc_{ji} = class_j$
  $l_j \leftarrow |Xl_j|$
  $e_j \leftarrow m_j - l_j$
end for
$e \leftarrow \sum e_j$
for $x_i \in X$ **do**
  $y_i \leftarrow class_j$
end for
return c, y, e

Choosing the best weights can be seen as an optimization problem that can be solved by a

metaheuristic technique. In this case, the optimization is carried on by the Simulated Annealing algorithm. The task is to minimize a fitness function (Alg. 2) that returns the classification error of a k-means clustering of a candidate weight vector.

Algorithm 2 Fitness function pseudocode

input: w
output: e
$e \leftarrow$ k-means with weighted distance metric
return e

The pseudocode for supervised k-means is shown in Alg. 3.

Algorithm 3 Supervised k-means pseudocode

input: X, t, k
output: c, w
top:
Initialize $w \leftarrow$ random vector
e, $c \leftarrow$ k-means with weighted distance metric
loop:
while $e <$ *tolerance* **do**
  $w \leftarrow$ optimize weights
  e, $c \leftarrow$ k-means with weighted distance metric
end while
return c, w

3 Proposed methods

In this section, we describe the use of Chi-square to initialize the weights and Mahalanobis distance to measure the closest cluster to each example. Then, we show how to update the cluster structure to incrementally treat unlabeled examples.

3.1 Chi-square metric to initialize weights

Instead of random weights being assigned at the beginning of the supervised k-means, weights can be already initialized with values indicative of each attribute relevance involved in the classification task. This is often accomplished by feature selection methods, but in this case, it is necessary to obtain a real-valued number associated with each attribute, that can represent the relevance of each attribute in obtaining the correct label for the example in question, without discarding any attribute.

Therefore, a metric chosen for this assignment is the Chi-square metric, calculated between one discrete attribute and the class label. Each attribute of the training dataset is tested against the class label, and their respective Chi-squares are calculated. Furthermore, in this work, the Chi-square values are normalized and assigned to a weight vector. This will be the initial weight vector for the supervised k-means.

3.2 Mahalanobis distance metric to measure distance between examples and cluster centroids

Besides the weight assigned to each attribute of the training dataset, it is interesting to characterize the variance and covariance of the attributes and use them to inform the distance measurements between examples and clusters. This can be accomplished by combining the weighted method with the Mahalanobis distance metric (Eq. 4)

$$\delta_{wm}(x, y) = \sqrt{w(x-\mu)^T S^{-1}(x-\mu)}, \quad (4)$$

where, w is the weight vector, x is the example, μ is the cluster and S is the covariance matrix. The covariance matrix S is obtained using only the batch of labeled examples to be used by k-means, before the iteration phase takes place in the supervised k-means.

3.3 Updating the structure incrementally using new unlabeled examples

The main purpose of this study is to adapt the supervised k-means to perform semi-supervised learning using the improvements described in the first two subsections, and by adding a new method to take unlabeled data, updating the previous cluster structure. The method is denominated here as semi-supervised k-means (SSK-means). The new set of online incremental methods was proposed in (Ru et al., 2014).

At first, the supervised k-means algorithm will obtain the clusters and weights using only the labeled data. In this case, the procedures presented in Alg. 3 with the changes described in Subsections 3.1 and 3.2 are applied.

When dealing with unlabeled data, the aim is to still learn without degrading the prediction power of the model. A nearest neighbor classifier can be modified to incorporate learning mechanisms into the k-means structure. Because k-means has a cluster structure, new data examples can be classified and clusters can be altered with relative ease to change the model. The proposed method is shown in Alg. 4. It accepts chunks of unlabeled data as well as labeled data, for online learning purposes. Therefore, due to unlabeled data, majority class cannot be used at incremental phase to account for classification errors.

3.4 Clustering features

The cluster structure can be inefficient when, for example, using Mahalanobis distance metric, as the covariance matrix cannot be easily updated. To solve this issue, Ru et al. (2014) proposed to use Clustering Features (CF), originally developed by Alsabti et al. (1997) for the BIRCH algorithm. CF is 3-tupple that stores summarized information about the cluster, containing an integer and

Algorithm 4 Semi-supervised k-means update method pseudocode

input: X, c, w
output: c
top:
e, $c \leftarrow$ k-means with weighted distance metric
loop:
for $x_i \in X$ **do**
  calculate weighted distance to each cluster $c_j \in c$
  assign to the closest cluster
end for
for $c_j \in c$ **do**
  $Xc_j \leftarrow$ set of $x_i | x_i$ is closest to c_j
  recalculate cluster $c_j \leftarrow \frac{\sum_{i=1}^{m} x_i}{m} | x_i \in Xc_j$
end for
return c, w

two real vectors, representing the number of examples N_j, the sum of examples per attribute LS_j, and the squared sum of examples per attribute SS_j. CF information for each cluster j is defined by Eq. 5.

$$CF_j = (N_j, LS_j, SS_j), \tag{5}$$

where,

$$LS_j = \sum_{i=1}^{N_j} x_i, \tag{6}$$

$$SS_j = \sum_{i=1}^{N_j} x_i^2. \tag{7}$$

When each example is merged with the cluster structure, CF_j is modified by Eq. 8.

$$CF_j = (N_j + 1, LS_j + x_i, SS_j + x_i^2). \tag{8}$$

For online classification and learning, in (Ru et al., 2014) is trained the model with examples when the prediction has high confidence. High confidence of an example x_i is qualified when:

1. Two nearest clusters (μ_k and μ_l) have the same class;

2. The distance to the nearest cluster is within radius R (Eq. 9).

$$R(\mu_j) = \sqrt{\frac{\sum_{i=0}^{n} SS_{\mu_j}^{\{i\}} - \frac{\sum_{i=0}^{n} LS_{\mu_j}^{\{i\}}}{N_{\mu_j}}}{N_{\mu_j}}} \tag{9}$$

If the class of an example is known, the algorithm also retrains the model by merging the new example x_i, or creating a new cluster in the case of a new class or if the distance to the nearest cluster is greater than R. The retraining algorithm is shown in Alg. 5.

The training algorithm is shown in Alg. 6.

Algorithm 5 Retraining algorithm pseudocode

input: μ_k, μ_l, x_i
output: CF
if $y(x_i) = y(\mu_k)$ **then**
  if $\delta(x_i, \mu_k) < R(\mu_k)$ **then**
    Merge x_i to μ_k
  else
    Create new cluster
  end if
else
  if $\delta(x_i, \mu_l) < R(\mu_l)$ **then**
    Merge x_i to μ_l
  else
    Create new cluster
  end if
end if
return CF

Algorithm 6 Training algorithm pseudocode

input: CF, x_i, y_i
output: CF, $y_i^{'}$
while new x_i **do** $y_i^{'} \leftarrow$ Classify(CF, x_i)
  if high confidence **then**
    $CF \leftarrow$ Train$(CF, x_i, y_i^{'})$
  end if
  if exists y_i **then**
    $CF \leftarrow$ Retrain(CF, x_i, y_i)
  end if
end while
return CF, $y_i^{'}$

4 Results

4.1 Datasets

Datasets from UCI (Lichman, 2013), SSL-Book (Chapelle et al., 2006) and KEEL (Alcalá-Fdez et al., 2011) benchmarks were used to train and test the algorithms. A total of 24 data sets of different origins were used for the experiments. These data sets were selected to provide a diversity of classification problems and are thus not limited to a distinct subset of problems. The datasets names, number of examples (m), features (n), classes (n_c) and source are shown in Tab. 1.

4.2 Experiments

In our experiments, we compare semi-supervised k-means (SSK-means) with different parameters. The parameter settings are the distance metric (weighted Euclidean or weighted Mahalanobis), weight initialization (random or chi-square), the optimization method (Simulated Annealing or PSO), the use of Clustering Features (CF), the number of clusters k, and the error tolerance. The value used for k in SSK-means varied around multiples of the number of classes, depending on the dataset. The tolerance chosen for convergence was also dataset-based.

Each dataset was divided in three parts: 20% labeled examples, 60% unlabeled examples and 20% test examples. SSK-means is initially trained with the labeled examples (similar to supervised k-means), and after that, it is trained with un-

Table 1: UCI, SSL-Book and KEEL benchmarks.

Dataset	m	n	n_c	**Source**
appendicitis	106	7	2	KEEL
australian	690	14	2	KEEL
balance	625	5	3	UCI
bci	400	118	2	SSL
bupa	345	6	2	KEEL
cancer	569	32	2	UCI
cleveland	297	13	5	KEEL
hayes-roth	160	4	3	KEEL
heart	267	23	2	UCI
hepatites	80	19	2	KEEL
horse	364	58	3	UCI
ionosphere	351	35	2	UCI
iris	150	5	3	UCI
led7digit	500	7	10	KEEL
lenses	18	6	3	UCI
mammographic	830	5	2	KEEL
monk-2	432	6	2	KEEL
newthyroid	215	5	3	KEEL
spectfheart	267	44	2	KEEL
tae	151	5	3	KEEL
vote	435	16	2	UCI
wine	178	14	3	UCI
wisconsin	683	9	2	KEEL
zoo	101	16	7	KEEL

labeled examples. Each algorithm runs 10 times, randomly selecting the examples, and computing the mean error rate, that is, the number of misclassified examples by number of examples.

4.3 Results

Error rates for SSK-means classifiers trained with the UCI, SSL-Book and KEEL benchmarks are shown in Tab. 2. The results obtained by a Decision Tree (DT) were also included as reference. $SSKM_1$ and $SSKM_2$ columns represent the results of the proposed SSK-means method. $SSKM_1$ is the SSK-means set with weighted Euclidean distance metric, initial random weights and Simulated Annealing as weight updating method. $SSKM_2$ is the SSK-means set with weighted Mahalanobis distance metric, initial weights by chi-square metric and PSO as weight updating method. k $SSKM_1$ and k $SSKM_2$ columns represent the number of clusters parameter to achieve the respective error rates. Best results for each dataset are in bold letters.

According to the results, $SSKM_2$ outperformed the $SSKM_1$ in 21 datasets, and the DT classifier in all datasets, except for *balance* dataset. For all datasets that $SSKM_2$ outperformed $SSKM_1$, the k $SSKM_2$ was equal or lesser than k $SSKM_1$. $SSKM_1$ outperformed $SSKM_2$ in only three datasets (*horse*, *ionosphere* and *tae*). Either way, the SSK-means outperformed a conventional supervised DT classifier in all datasets, except one (*balance*). In the case of *balance* dataset, even with the k value higher than the predominant value of $3n_c$, the proposed algorithm didn't converge to a better result. In some cases, the $SSKM_2$ performance was a lot better than $SSKM_1$, possibly indicating overfitting, but a validation subset could be used and the k value could be relaxed, improving generalization.

In terms of approximate run-time of the algorithms, offline training in $SSKM_2$ was six times faster than in $SSKM_1$, however, because of the optimization involved, offline training was overwhelmingly more time consuming than online training regardless. the offline run-time ranged from nearly 10 seconds to 4 hours, while the online run-time ranged from 0.4×10^{-2} to 16.0×10^{-2} seconds.

4.4 Hypothesis test

In order to compare the significance of the results, the Wilcoxon signed-ranks test (Demsar, 2006) was applied at a 5% significance level. Values of $z < -1.96$ mean there is a statistically significant improvement of one classifier over the other. The Tab. 3 shows the results of the Wilcoxon signed-ranks test for $SSKM_2$ over decision tree (DT) and $SSKM_1$.

According to Wilcoxon signed-ranks test, SSK-means versions are significantly different from DT with respect to error rates. The $SSKM_2$ version had statistically significant superiority over DT and the $SSKM_1$ version. Values of z for this setting were -3.55 and -2.81 over DT and $SSKM_1$, respectively. The Wilcoxon test for $SSKM_1$ versus DT obtained a value of z equal to -3.33, showing statistically significant difference.

5 Conclusions

The objective of the present work was to obtain a method of incremental semi-supervised learning using k-means that efficiently adapts its structure as new data are presented. Results of the proposed algorithm (SSK-means) showed statistically significant improvement over supervised k-means settings and decision tree.

Although the cluster updating methods applied for incremental learning with unlabeled examples were efficient, the initial model created with labeled examples using the supervised k-means had its drawbacks. One point for future investigation is the weight updating technique, because an optimization algorithm was too computationally demanding. Other heuristics algorithms, metrics like Minkowsky, may be interesting to explore. The Mahalanobis distance metric with weights could benefit of eigenvalue decomposition, so that the weights could be directly applied to the eigenvalues of the covariance matrix, without the need to recalculate it at each time new weights are selected.

New techniques like clouds are interesting pathways, because they allow irregular shaped

Table 2: Error rates for SSK-means.

Dataset	DT	$SSKM_1$	$SSKM_2$	$k\ SSKM_1$	$k\ SSKM_2$
appendicitis	21.88	16.47	**4.76**	$3n_c$	$3n_c$
australian	15.70	11.59	**9.42**	$3n_c$	$3n_c$
balance	**10.80**	14.20	13.60	$18n_c$	$11n_c$
bci	47.13	37.50	**28.75**	$4n_c$	$4n_c$
bupa	39.13	33.70	**5.80**	$3n_c$	$3n_c$
cancer	8.72	7.91	**7.89**	$2n_c$	$2n_c$
cleveland	46.63	29.41	**3.33**	$3n_c$	$3n_c$
hayes-roth	42.71	39.84	**12.50**	$3n_c$	$3n_c$
heart	18.18	17.76	**5.56**	$2n_c$	$2n_c$
hepatites	18.75	14.06	**6.25**	$3n_c$	$3n_c$
horse	38.53	**13.70**	20.55	$3n_c$	$3n_c$
ionosphere	26.76	**21.71**	24.29	$2n_c$	$3n_c$
iris	23.33	17.50	**10.00**	$2n_c$	$2n_c$
led7digit	32.33	30.00	**7.00**	$2n_c$	n_c
lenses	42.86	26.32	**20.00**	n_c	n_c
mammographic	24.70	14.61	**3.61**	$3n_c$	$3n_c$
monk-2	3.86	3.45	**1.15**	$5n_c$	$5n_c$
newthyroid	11.63	8.14	**6.98**	$3n_c$	$3n_c$
spectfheart	28.75	14.95	**5.56**	$3n_c$	$3n_c$
tae	56.04	**40.50**	43.33	$3n_c$	$3n_c$
vote	4.94	2.30	**1.15**	$2n_c$	$2n_c$
wine	24.74	15.49	**8.57**	$2n_c$	$2n_c$
wisconsin	9.27	3.66	**2.94**	$4n_c$	$3n_c$
zoo	24.59	22.22	**20.00**	$5n_c$	$5n_c$

Table 3: Results of Wilcoxon signed-ranks test.

	DT	$SSKM_1$
R+	230	225
R-	4	21
W	4	21
z	-3.55	-2.81

example clustering to take place. Strategies to remove, merge, split clusters from the structure might be investigated, and all these can be set in a cloud structure.

Acknowledgments

Patrick Marques Ciarelli thanks the partial funding of his research work provided by CNPq (grant 312032/2015-3).

The authors would like to thank the Graduate Program in Electrical Engineering (PPGEE) at UFES.

References

Al-Harbi, S. H. and Rayward-Smith, V. J. (2006). Adapting k-means for supervised clustering, *Applied Intelligence* **24**(3): 219–226.

Alcalá-Fdez, J., Fernandez, A., Luengo, J., Derrac, J., García, S., Sánchez, L. and Herrera, F. (2011). Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework, *Journal of Multiple-Valued Logic and Soft Computing* **17**: 2–3.

Alsabti, K., Ranka, S. and Singh, V. (1997). An efficient k-means clustering algorithm, *Electical engineering and Computer Science* .

Chapelle, O., Schölkopf, B. and Zien, A. (eds) (2006). *Semi-Supervised Learning*, MIT Press, Cambridge, MA. http://www.kyb.tuebingen.mpg.de/ssl-book.

Demsar, J. (2006). Statistical Comparisons of Classifiers over Multiple Data Sets, *Journal of Machine Learning Research* **7**(1): 1–30.

Giraud-Carrier, C. (2000). A note on the utility of incremental learning, *AI Communications* pp. 215–223.

Lichman, M. (2013). UCI machine learning repository. http://archive.ics.uci.edu/ml.

Liu, H. and Setiono, R. (1995). Chi2: Feature selection and discretization of numeric attributes, *Proceedings of the 7th International Conference on Tools with Artificial Intelligence*, Washington D.C., pp. 388–391.

Ru, L. H., Andromeda, T. and Marsono, M. N. (2014). Online Data Stream Learning and Classification with Limited Labels, *Proceeding of International Conference on Electrical Engineering, Computer Science and Informatics* (August): 161–164.