ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

Toward parallel feature selection from vertically partitioned data

Verónica Bolón-Canedo, Noelia Sánchez-Maroño and Joana Cerviño-Rabuñal *

Department of Computer Science - University of A Coruña
Campus de Elviña s/n 15071 - A Coruña, Spain

Abstract.
Feature selection is often required as a preliminary step for many pattern recognition problems. In recent years, parallel learning has been the focus of much attention due to the advent of high dimensionality. Still, most of the existing algorithms only work in a centralized manner, i.e. using the whole dataset at once. This paper proposes a parallel filter approach for vertically partitioned data. The idea is to split the data by features and then apply a filter at each partition performing several rounds to obtain a stable set of features. Later, a merging procedure is carried out to combine the results into a single subset of relevant features. Experiments on three representative datasets show that the execution time is considerably shortened whereas the performance is maintained or even improved compared to the standard algorithms applied to the non-partitioned datasets. The proposed approach can be used with any filter algorithm, so it could be seen as a general framework for parallel feature selection.

1 Introduction

In the last decade, feature selection (FS), which consists of detecting the relevant features and discarding the irrelevant ones [1], has been an active research area due to the large size of the datasets. Feature selection methods usually come in three flavors: filters, wrappers and embedded methods. While wrapper models involve optimizing a predictor as part of the selection process, filter models rely on the general characteristics of the training data to select features with independence of any predictor. The embedded methods generally use machine learning models for classification, and then an optimal subset of features is built by the classifier algorithm. As stated in [2], even when the subset of features is not optimal, filters are preferable due to their computational and statistical scalability.

Feature selection is usually applied in a centralized manner, i.e. using the whole set of features to determine the relevant ones. However, if the data is distributed, FS may take advantage of processing multiple subsets in sequence or concurrently. The need to use parallel FS can be two-fold. On the one hand, with the advent of network technologies data is sometimes distributed in multiple locations and often with multiple parties. On the other hand, most existing FS algorithms do not scale well and their efficiency significantly deteriorates or even becomes inapplicable when dealing with large-scale

*This research has been economically supported in part by the Secretaría de Estado de Investigación of the Spanish Government through the research project TIN 2012-37954; and by the Consellería de Industria of the Xunta de Galicia through the research projects CN2011/007 and CN2012/211; all of them partially funded by FEDER funds of the European Union. V. Bolón-Canedo acknowledges the support of Xunta de Galicia under *Plan I2C* Grant Program.

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

data. In order to increase efficiency, learning can be parallelized by distributing the subsets of data to multiple processors, learning in parallel and then combining them. There are two main techniques for partitioning and distributing data: vertically, i.e. by features, and horizontally, i.e. by samples. Distributed learning has been used to scale up datasets that are too large for batch learning in terms of samples [3, 4, 5]. While not common, there are some other developments that distribute the data by features [6, 7]. In this research we will distribute the data vertically in order to have the FS process distributed. After having the data distributed in small feature subsets and selecting the relevant features from each subset, a merging procedure is performed which updates the feature subset according to improvements in the classification accuracy, obtaining a unique set of features with a good generalization capability. The experimental results on three different databases demonstrate that our proposal can improve the performance of original FS methods and show important savings in running times.

2 Parallel feature selection

This paper proposes a parallel framework for FS by partitioning the data vertically. In this manner, a filter is applied over different partitions of the data and the results are combined in the final step into a single subset of features. There are three main stages: (i) partition of the data; (ii) application of filter to the partitions; and (iii) combination of the results. This parallel framework can be applied using different filters. Besides, the two first steps are repeated several rounds (r) to ensure capturing enough information for the combination step.

For each round, the first step is to split the data without replacement by randomly assigning groups of k features to each subset. Then, the chosen filter is applied to each partition separately and the features selected to be removed receive a vote (notice that this step can be done in parallel). At that point, a new round is performed leading to a new partition so as to allow the features to update their votes. This process goes on until reaching the predefined number of rounds. Finally, the features that have received a number of votes above a certain threshold are removed. A unique set of features is obtained to train a classifier C and to test its performance over an unseen test dataset.

Determining the threshold of votes is not an easy-to-solve question, since it depends on each dataset. Therefore, we use an automatic method to calculate it. This threshold is estimated from its effect on the 10% of the training set, to speed up the process. Following the recommendations in [8], the selection of the number of votes must take into account two different criteria: the training classification error and the percentage of features retained. Both values are desirable to be minimized to the extent possible which results in minimizing the following fitness criterion $e[v] \leftarrow \gamma \times error + (1 - \gamma) \times featPercentage$, where a term γ is introduced to measure the relative relevance of both values and was set to $\gamma = 0.75$ as suggested in [8], giving thus more importance to the classification error. Notice that the maximum number of votes is the number of rounds r, so all the threshold values from 1 to r are evaluated. At the end, the features with number of votes above the obtained threshold are removed from the final subset of features. This subset will be applied to the training and test sets in order to obtain the ultimate classification accuracies.

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

3 Experimental results

In order to test our vertically parallel filter approach, 3 problems were selected. Their properties can be consulted in Table 1, such as number of features, samples, classes and packets. These datasets can be considered representative of problems from medium to large size and can be free downloaded from the UCI Machine Learning Repository [9].

Table 1: Dataset description

Dataset	Features	Samples	Classes	Packets
Isolet	617	7474	26	5
Madelon	500	2400	2	5
Mnist	717	60000	2	5

The parallel approach proposed herein can be used with any filter method. In this work, five well-known filters, based on different metrics, were chosen, all of them available in the Weka tool [10]. While three of the filters return a feature subset (Correlation based Feature Selection - CFS, Consistency-based and INTERACT), the other two (ReliefF and Information Gain) are ranker methods, so it is necessary to establish a threshold in order to obtain a subset of features. In this research we have opted for retaining the c top features, being c the number of features selected by CFS. Notice that Information Gain is the only univariate method, so the complexity of the remaining multivariate filters are expected to highly depend on the number of features. A discretization step is also performed by default by Weka. For testing the adequacy of our proposal, four well-known supervised classifiers, of different conceptual origin, were selected: C4.5, naive Bayes, IB1 and SVM. Note that the experiments were performed on an Intel ®Core™2 Duo CPU E6750 @ 2.66GHz with RAM 4 GB.

This section presents the results over the three benchmark dataset, comparing our parallel approach with the centralized standard approach of each method. To distinguish between the two concepts, a "C" (centralized) or a "P" (parallel) was added to the name of the filter. In the case of the parallel approach, the number of rounds r was set to 5. To ensure reliable results, a hold-out validation was applied and repeated 5 times. This technique consists of randomly splitting the available data into a disjoint pair train-test. In this work the common partition 2/3 for training and 1/3 for testing was used. After the 5 repetitions, a Kruskal-Wallis test was applied to check if there were significant differences for a level of significance $\alpha = 0.05$. Then, a multiple comparison procedure (Tukey's) was applied to find the simplest approach whose accuracy is not significantly different from the approach with the best accuracy (labeled with a cross in Table 2). Notice that for each one of the 5 repetitions, 5 rounds of the partitioning and filtering steps are executed.

Table 2 shows the average test accuracy and the execution time required by the FS process on each dataset. In the parallel approach, considering that all the subsets can be processed in parallel, the time displayed is the maximum time required by the filter in each subset generated in the partitioning stage, plus the time required to find the

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

threshold of votes. Notice that in the table, the best result for each dataset and classifier is highlighted in bold face, while the best result for dataset is also shadowed.

Table 2: Test classification accuracy. Best results are highlighted.

		Isolet		Madelon		Mnist		Avg
		Acc.	Time	Acc.	Time	Acc.	Time	Acc.
C4.5	CFS-C	80.61 †	00:03:52	75.34	00:00:49	85.36	00:32:39	80.44
	CFS-P	**80.85** †	00:02:54	75.08 †	00:00:32	86.16 †	00:07:48	80.70
	IG-C	79.00 †	00:02:34	79.63 †	00:00:47	86.69 †	00:25:24	81.77
	IG-P	79.54 †	00:02:41	78.27 †	**00:00:29**	**88.85** †	**00:07:41**	82.22
	ReliefF-C	79.02 †	00:08:55	**82.93** †	00:01:15	85.59 †	06:22:00	82.51
	ReliefF-P	80.04 †	00:04:20	81.33 †	00:00:39	88.57 †	01:36:01	**83.31**
	INT-C	76.76	00:03:06	77.94 †	00:00:49	84.77 †	00:31:01	79.82
	INT-P	77.59 †	00:01:57	78.09 †	**00:00:29**	87.06 †	00:09:43	80.91
	Cons-C	54.04	00:03:57	78.39 †	00:01:05	85.40	01:21:16	72.61
	Cons-P	74.99	**00:01:31**	77.3 †	00:00:30	85.01	**00:07:41**	79.10
NB	CFS-C	72.44 †	00:03:52	**69.94** †	00:00:49	72.18 †	00:32:39	71.52
	CFS-P	**74.07** †	00:02:10	69.57 †	00:00:34	73.26 †	00:06:50	**72.30**
	IG-C	69.62 †	00:02:34	69.78 †	00:00:47	68.80	00:26:24	69.40
	IG-P	70.93 †	00:02:22	69.55 †	**00:00:29**	71.38 †	00:06:37	70.62
	ReliefF-C	64.67	00:08:55	69.64 †	00:01:15	69.58	06:22:00	67.96
	ReliefF-P	69.53 †	00:04:53	69.32 †	00:00:40	69.27	01:35:32	69.37
	INT-C	67.17 †	00:03:06	69.73 †	00:00:49	69.87	00:31:01	68.92
	INT-P	65.21 †	00:01:42	69.46 †	00:00:32	**76.08** †	00:08:14	70.25
	Cons-C	40.19	00:03:57	69.81 †	00:01:05	71.78 †	01:21:16	60.59
	Cons-P	60.22	**00:01:27**	69.74 †	00:00:32	73.84 †	**00:06:01**	67.93
IB1	CFS-C	55.63 †	00:03:52	68.86	00:00:49	86.34	00:32:39	70.28
	CFS-P	57.90 †	00:01:55	71.75	00:00:31	91.25 †	00:11:01	73.63
	IG-C	53.94	00:02:34	78.07 †	00:00:47	89.35 †	00:26:24	73.79
	IG-P	55.62 †	00:02:08	78.76 †	**00:00:28**	94.02 †	00:10:36	76.13
	ReliefF-C	56.69 †	00:08:55	**89.32** †	00:01:15	89.27 †	06:22:00	78.43
	ReliefF-P	**59.36** †	00:04:37	88.76 †	00:00:38	95.80 †	01:39:27	**81.31**
	INT-C	46.88	00:03:06	72.95 †	00:00:49	85.04	00:31:01	68.29
	INT-P	53.10 †	00:01:37	72.64 †	00:00:29	94.36 †	00:13:20	73.37
	Cons-C	53.05	00:03:57	75.04 †	00:01:05	85.42	01:21:16	71.17
	Cons-P	57.39 †	**00:01:24**	73.91 †	00:00:29	**96.03** †	**00:09:19**	75.78
SVM	CFS-C	82.36 †	00:03:52	65.23 †	00:00:49	80.15 †	00:32:39	75.91
	CFS-P	83.55 †	00:07:21	64.92 †	00:00:32	**81.58** †	**00:23:53**	**76.68**
	IG-C	81.09 †	**00:02:34**	65.9 †	00:00:47	78.6 †	00:26:24	75.20
	IG-P	82.61 †	00:07:41	65.16 †	**00:00:28**	80.02 †	00:30:59	75.93
	ReliefF-C	81.92 †	00:08:55	66.4 †	00:01:15	74.93	06:22:12	74.42
	ReliefF-P	**83.98** †	00:10:16	**66.33** †	00:00:40	76.82 †	02:34:22	75.71
	INT-C	72.53	00:03:06	65.01 †	00:00:49	78.44 †	00:31:01	71.99
	INT-P	80.38 †	00:04:46	64.97 †	00:00:30	78.46 †	00:41:53	74.60
	Cons-C	29.29	00:03:57	65.04 †	00:01:05	74.42	01:26:16	56.25
	Cons-P	76.45	00:02:59	65.73 †	00:00:31	80.17 †	00:40:47	74.12

In terms of classification accuracy, the best results were obtained by the parallel approach for Isolet and Madelon, as well as in average for all datasets and classifiers. However, as can be seen in Table 2, in general there are no significant differences in

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

terms of accuracy, as expected. When proposing a parallel approach, the goal is to maintain the classification performance whilst reducing the running time. In this research, indeed, the execution time is drastically shortened, in some cases from six hours to one hour (Mnist dataset with ReliefF filter). Notice that the larger the dataset, the larger the reduction in time. It is also worth mentioning that SVM is the classifier which requires the largest time to find the optimal threshold of votes, so the user must select another classifier if the main objective is to reduce notably the time.

Table 3 reports a comparison of the results obtained in this research with other results found in the literature. In [11], we have evaluated a horizontal distribution over the same datasets, which will be referred in the table as "PH" (parallel horizontally) in contrast to "PV" (parallel vertically) which refers to the results in this research. Moreover, we have included other results found in the literature which deal with the same datasets, although not with a parallel approach. Notice that this comparative is not completely fair since different validation might have been applied. For Isolet dataset, our vertical parallel approach outperforms both the horizontal parallel method and the combination of the Fast Correlation-Based Filter (FCBF) with C4.5 classifier. As for Madelon, the best results were obtained with the horizontal partition, improving our result with the vertical partition in only 1.12%. Both parallel approaches surpass by far the results achieved in [12], where an ensemble of filters is used combined with the IB1 classifier. Finally, on Mnist dataset our proposed method for vertically distributed data gets the highest accuracy ahead of the horizontal partitioning and the combination of consistency-based filter with an artificial neural network [13].

Table 3: Comparison with other results in the literature. Best results are highlighted

Dataset	Method	Accuracy
Isolet	FCBF+C4.5 [14]	75.77
	CFS-PH+SVM [11]	81.10
	ReliefF-PV+SVM	**83.98**
Madelon	E1-ns+IB1 [12]	66.75
	ReliefF-PH+IB1 [11]	**89.88**
	ReliefF-PV+IB1	88.76
Mnist	Consistency+ANN [13]	63.00
	Cons-PH+IB1 [11]	94.85
	Cons-PV+IB1	**96.03**

4 Conclusions

In this research we have proposed a new method for scaling up feature selection: a vertical parallel filter approach. The main goal was to design a method that would be able to successfully distribute the feature selection process, expecting that the execution time would be considerably shortened and the classification accuracy would not deteriorate.

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

The experiments on three datasets considered representative of problems from medium to large size showed that our proposal was able to reduce the running time significantly with respect to the standard (centralized) filtering algorithms. In terms of execution time, the behavior is excellent, being this fact the most important advantage of our method. Moreover, regarding the classification accuracy, our vertical parallel approach was able to match and in some cases even improve the standard algorithms applied to the non-partitioned datasets. Finally, it is worth mentioning that our proposal can be used with any filter algorithm without any modifications, so it could be seen as a general framework for parallel feature selection.

References

[1] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

[2] Y. Saeys, I. Inza, and P. Larrañaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, 2007.

[3] P.K. Chan, S.J. Stolfo, et al. Toward parallel and distributed learning by meta-learning. In *AAAI workshop in Knowledge Discovery in Databases*, pages 227–240, 1993.

[4] V.S. Ananthanarayana, D.K. Subramanian, and M.N. Murty. Scalable, distributed and dynamic mining of association rules. *High Performance Computing HiPC 2000*, pages 559–566, 2000.

[5] G. Tsoumakas and I. Vlahavas. Distributed data mining of large classifier ensembles. In *Proceedings Companion Volume of the Second Hellenic Conference on Artificial Intelligence*, pages 249–256, 2002.

[6] S. McConnell and D.B. Skillicorn. Building predictors from vertically distributed data. In *Proceedings of the 2004 conference of the Centre for Advanced Studies on Collaborative research*, pages 150–162. IBM Press, 2004.

[7] D.B. Skillicorn and S.M. McConnell. Distributed prediction from vertically partitioned data. *Journal of Parallel and Distributed computing*, 68(1):16–36, 2008.

[8] A. de Haro García. *Scaling data mining algorithms. Application to instance and feature selection*. PhD thesis, Universidad de Granada, 2011.

[9] A. Asuncion and D.J. Newman. UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences. `http://mlearn.ics.uci.edu/MLRepository.html`. Last accessed: November 13.

[10] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten. The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.

[11] V. Bolón-Canedo, N. Sánchez-Marono, and J. Cerviño-Rabuñal. Scaling up feature selection: A distributed filter approach. In *Advances in Artificial Intelligence*, pages 121–130. Springer, 2013.

[12] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos. Data classification using an ensemble of filters. *Neurocomputing*, 2013. In press.

[13] D. Peteiro-Barral, V. Bolón-Canedo, A. Alonso-Betanzos, B. Guijarro-Berdiñas, and N. Sánchez-Maroño. Toward the scalability of neural networks through feature selection. *Expert Systems with Applications*, 40(8):2807–2816, 2012.

[14] L. Yu and H. Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *Machine Learning International Workshop*, volume 20, page 856, 2003.