ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

A Distributed Wrapper Approach for Feature Selection

Verónica Bolón-Canedo, Noelia Sánchez-Maroño and Amparo Alonso-Betanzos *

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

Abstract. In recent years, distributed learning has been the focus of much attention due to the proliferation of big databases, usually distributed. In this context, machine learning can take advantage of feature selection methods to deal with these datasets of high dimensionality. However, the great majority of current feature selection algorithms are designed for centralized learning. To confront the problem of distributed feature selection, in this paper we propose a distributed wrapper approach. In this manner, the learning accuracy can be improved, as well as obtaining a reduction in the memory requirements and execution time. Four representative datasets were selected to test the approach, paving the way to its application over extremely-high data which prevented previously the use of wrapper approaches.

1 Introduction

Feature selection is a dimensionality reduction technique which consists of detecting the relevant features and discarding the irrelevant and the redundant ones. A correct selection of the features can lead to an improvement of the inductive learner, either in terms of learning speed, generalization capacity or simplicity of the induced model [1]. Feature selection, since it is an important activity in data preprocessing, has been an active research area in the last decade, finding success in many different real world applications [2, 3, 4, 5].

There are three main models which can be distinguished regarding the relationship between the feature selection algorithm and the inductive learning method user to infer a model. The *filter* model relies on the general characteristics of training data and carry out the feature selection process as a pre-processing step with independence of the induction algorithm. On the other hand, *embedded* methods perform feature selection in the process of training and are usually specific to given learning machines. Finally, the *wrapper* model involves a learning algorithm as a black box and consists of using its prediction performance to assess the relative usefulness of subsets of variables. In other words, the feature selection algorithm uses the learning algorithm as a subroutine with the computational cost that comes from calling the learning algorithm to evaluate each subset of features. However, this interaction with the classifier tends to give better performance results than filters and embedded methods [6]. This is probably due to the fact that the relevant feature subset could not reflect the classifier's specific characteristics. For this reason, the wrapper approach will be the focus of this research. Nevertheless,

*This research has been economically supported in part by the Secretaría de Estado de Investigación of the Spanish Government through the research projects TIN2009-10748 and 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 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

it is a very computationally expensive model and in this work we will try to reduce the time and effort required by the machine learning methods.

Thus, we will investigate how wrapper approaches can take advantage of distributed learning to reduce the computational cost and speed up the feature selection process in high dimensional datasets. Distributed learning methods learn from multiple subsets of data processed concurrently. In order to increase efficiency, learning can be parallelized by distributing the subsets of data to multiple processors, learning in parallel and then combining the obtained results. This research will introduce a distributed wrapper approach which consists of several rounds of feature selection processes whose outputs are combined into a single subset of relevant features. In this manner, the computational cost and therefore the time required by the wrapper approach will be significantly reduced, allowing its use over some datasets where it was previously unfeasible.

2 Distributed feature selection

As mentioned in the Introduction, the wrapper model will be the focus of our attention. The idea of the wrapper approach is to select a feature subset using a learning algorithm as part of the evaluation function. Instead of using subset sufficiency, entropy or another explicitly defined evaluation function, a kind of "black box" is used to guide the search. The evaluation function for each candidate subset returns an estimate of the quality of the model that is induced by the learning algorithm. The search strategy usually comes in two flavors [7]: *forward selection* and *backward elimination*. The former starts with an empty set of features and adds features one by one, whereas the latter begins with a full set and removes features one by one; both techniques add or remove according to improvements in the performance results obtained by the evaluation function. Forward selection is far less time-consuming than backward elimination [8] so it will be used in the experiments of this research.

Our proposal is a distributed wrapper method which performs several fast feature selectors over several partitions of the data, combined afterwards into a single subset of features. More specifically, we divide each dataset D into several small disjoint datasets D_i. The wrapper algorithm is applied to each one of these subsets, and a selection S_i is generated for each subset of data. After all the small datasets D_i were used, (which could be done in parallel, as all of them are independent from each other), the combination method constructs the final selection S as the result of the feature selection process. To sum up, there are three main stages: (i) partition of the datasets; (ii) application of feature selection to the subsets; and (iii) combination of the results.

The partition of the dataset consists of dividing the original training dataset into several disjoint subsets of approximately the same size that cover the full dataset. In this research, the partition will be done vertically, i.e. by features. Instead of performing a randomly partition, a first step was added in order to rank the features. For this sake, the well-known *Information Gain* [9] filter was chosen, which has proven to perform better than the randomly partition. This univariate filter provides an ordered ranking of all the features where the worth of an attribute is evaluated by measuring the information gain with respect to the class. After obtaining this ranking, the data is split by assigning groups of k features to each subset, sequentially over the ranking. The number of

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

features k in each subset is assigned ad-hoc for the datasets employed in this work, trying to achieve a good balance between number of samples and number of features.

Algorithm 1 Pseudo-code for distributed wrapper

$D_{(m \times s)}$:= training dataset with m samples and s features
n_s := number of subsets of k features

1. Apply InfoGain over D and obtain a ranking R of the features
2. for $i = 1$ to n_s do
   (a) R_i = first k features in R
   (b) $R = R \setminus R_i$
   (c) $D_i = D_{(m \times R_i)}$
3. for $i = 1$ to n_s do
   (a) S_i = subset of features obtained after applying wrapper over D_i
4. $S = S_1$
5. $baseline$ = accuracy classifying subset $D_{(m \times S_i)}$ with classifier C
6. for $i = 2$ to n_s do
   (a) $S_{aux} = S \cup S_i$
   (b) $accuracy$ = classifying subset $D_{(m \times S_{aux})}$ with classifier C
   (c) if $accuracy > baseline$
      i. $S = S_{aux}$
      ii. $baseline = accuracy$
7. Build classifier C with $D_{(m \times S)}$
8. Obtain prediction P

Once we have several small disjoint datasets D_i, the wrapper will be applied to each one of them, returning a selection S_i for each subset of data. Finally, to combine the results, the first selection S_1 is taken to calculate the classification accuracy, which will be the *baseline*, and the features in S_1 will become part of the final selection. For the remaining selections S_j, they will be incorporated to the final selection S if they improve the baseline accuracy, as can be seen in more detail in Algorithm 1. At the end, this final selection S is applied to the training and test sets in order to obtain the ultimate classification accuracies. With this final step we try to improve the performance by removing irrelevant and redundant features.

3 Experimental setup

In order to test our distributed wrapper proposal, we have selected 4 binary problems which can be consulted in Table 1, depicting their properties (number of features, training samples, test samples and distributions between the positive and the negative

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

classes). These datasets can be considered representative of problems from medium to large size and can be free downloaded from the UCI Machine Learning Repository [10]. Some of the datasets come originally with training and test samples that were drawn from different conditions. For the sake of comparison, datasets with only training set were randomly divided using the common rule 2/3 for training and 1/3 for testing.

Four well-known supervised classifiers, of different conceptual origin, where chosen for forming part of the evaluation function of the wrappers, as well as for performing the class prediction once the feature selection was accomplished. All the classifiers (C4.5, naive Bayes, IB1 and SVM) were executed using the Weka tool [11]. Experimentation was performed on an Intel(R) Xeon(R) CPU W3550 @ 3.07 GHz with RAM 12 GB.

Table 1: Dataset description

Dataset	Attributes	Samples		Train	Test
		Train	Test	distribution	distribution
Madelon	500	1600	800	49% - 51%	52% - 48%
Spambase	57	3068	1533	40% - 60%	39% - 61%
Mushroom	112	5416	2708	47% - 53%	50% - 50 %
Adult	122	16100	16461	24% - 76%	24% - 76%

4 Results and discussion

Our goal is to test the previously described distributed wrapper. By using a distributed approach, we will be able to execute the wrapper model in scenarios where it was unfeasible or took a very long time before. To show the adequacy of the proposed wrapper (*Distributed*), it will be compared with the performance of the wrapper in a centralized manner (*Centralized*), i.e. when applying the wrapper over all the whole set of features directly. When testing a distributed approach, machine learning researchers are not only interested in classification accuracy but also in execution time. Table 2 shows the train and test accuracy, the number of features and the execution time required by each method on each dataset.

A slight decrease in accuracy is acceptable when the processing time is significantly reduced, as happens with Madelon dataset. Applying the classifier IB1 over this dataset, the maximum accuracy was obtained (91.50%) with the centralized approach. However, the time required by the centralized wrapper was 13 hours against 14 minutes required by each packet in the distributed approach, while the accuracy only decreased in 1%. For some cases, such as Madelon and Spambase with NB and SVM, or Mushroom with IB1 and SVM, the accuracy when using the distributed wrapper is improved at the same time that an important reduction in the execution time is achieved. It is worth to note that for Mushroom dataset, our proposed wrapper achieves a 100% of accuracy.

On Adult dataset with IB1 classifier our method is able to work in cases where the centralized approach was unfeasible to run. Plus, for SVM classifier, the time is reduced from the order of days to the order of hours when employing our proposed method, with

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

a slight decrease in classification accuracy (under 1%) and using a smaller number of features.

Table 2: Classification results for both implementations of wrapper for each dataset. N/A stands for *Not Applicable*.

Dataset	Method	Accuracy		No. feat	Time
		Train	Test		hh:mm:ss
Madelon	Centralized + C4.5	98.75	86.50	35	06:30:51
	Distributed + C4.5	94.56	82.63	28	00:01:08
	Centralized + NB	71.94	67.38	21	06:45:08
	Distributed + NB	71.50	69.00	20	00:00:29
	Centralized + IB1	100.00	91.50	14	13:51:40
	Distributed + IB1	100.00	90.13	17	00:04:13
	Centralized + SVM	68.06	66.38	16	15:06:40
	Distributed + SVM	67.00	66.75	7	00:04:13
Spambase	Centralized + C4.5	95.66	92.04	17	00:13:02
	Distributed + C4.5	97.65	91.65	35	00:01:34
	Centralized + NB	87.26	87.87	11	00:14:13
	Distributed + NB	88.07	88.71	15	00:00:08
	Centralized + IB1	98.60	90.99	16	02:36:30
	Distributed + IB1	99.90	90.54	14	00:34:50
	Centralized + SVM	90.45	90.08	25	02:50:14
	Distributed + SVM	90.78	90.80	50	00:01:59
Mushroom	Centralized + C4.5	99.93	99.85	6	00:03:18
	Distributed + C4.5	100.00	100.00	30	00:03:36
	Centralized + NB	98.54	98.49	3	00:04:19
	Distributed + NB	97.88	97.45	6	00:00:35
	Centralized + IB1	100.00	99.89	10	06:45:00
	Distributed + IB1	100.00	100.00	33	02:14:20
	Centralized + SVM	99.80	99.67	7	06:53:20
	Distributed + SVM	100.00	100.00	31	00:43:10
Adult	Centralized + C4.5	85.34	83.86	31	06:48:20
	Distributed + C4.5	84.14	80.18	16	00:08:53
	Centralized + NB	82.97	83.21	26	06:56:40
	Distributed + NB	83.30	79.07	19	00:02:32
	Centralized + IB1	N/A	N/A	N/A	N/A
	Distributed + IB1	78.78	79.58	14	18:20:00
	Centralized + SVM	82.83	83.25	14	43:53:21
	Distributed + SVM	82.80	82.69	7	01:16:40

It is also worth mentioning that further experiments (not included for the sake of brevity) revealed that our proposed distributed wrapper improved in average for all classifiers and datasets the classification accuracy in 3% when compared with the well-known Correlation-based Feature Selection filter [12]. As expected, the wrapper approach achieves better results than the filter approach.

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

5 Conclusions

In this work, the adequacy of a distributed approach for wrapper feature selection was tested over four datasets considered representative of problems from medium to large size. Our goal was to design a distributed wrapper which would led to a reduction in the running time as well as in the storage requirements while the accuracy would not drop to inadmissible values.

The experiments showed that our method is able to shorten the execution time impressively compared to the standard wrapper algorithms. Furthermore, our distributed wrapper achieved a similar performance to the original wrapper. In terms of test accuracy, our method is able to match and in some cases even to improve the standard results applied to the non-partitioned datasets.

As future work, we plan to apply our proposed method over datasets that prevent the use of standard wrappers, such as microarray datasets.

References

[1] I. Guyon, S. Gunn, M. Nikravesh, and L.A. Zadeh. *Feature extraction: foundations and applications*, volume 207. Springer, 2006.

[2] L. Yu and H. Liu. Redundancy based feature selection for microarray data. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 737–742. ACM, 2004.

[3] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos. Feature selection and classification in multiple class datasets: An application to kdd cup 99 dataset. *Expert Systems with Applications*, 38(5):5947–5957, 2011.

[4] G. Forman. An extensive empirical study of feature selection metrics for text classification. *The Journal of Machine Learning Research*, 3:1289–1305, 2003.

[5] P. Saari, T. Eerola, and O. Lartillot. Generalizability and simplicity as criteria in feature selection: application to mood classification in music. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(6):1802–1812, 2011.

[6] Mark Andrew Hall and Geoffrey Holmes. Benchmarking attribute selection techniques for discrete class data mining. *Knowledge and Data Engineering, IEEE Transactions on*, 15(6):1437–1447, 2003.

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

[8] Sanmay Das. Filters, wrappers and a boosting-based hybrid for feature selection. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 74–81. Morgan Kaufmann Publishers Inc., 2001.

[9] M.A. Hall and L.A. Smith. Practical feature subset selection for machine learning. *Computer Science*, 98:181–191, 1998.

[10] A. Frank and A. Asuncion. UCI Machine Learning Repository. `http://archive.ics.uci.edu/ml`, 2010. [Online; accessed 14-November-2012].

[11] 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.

[12] M.A. Hall. *Correlation-based feature selection for machine learning*. PhD thesis, The University of Waikato, 1999.