This is the published version:


Available from Deakin Research Online:

http://hdl.handle.net/10536/DRO/DU:30034406

©2010 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.

Copyright: 2010, IEEE
Meta-Learning for Data Summarization Based on Instance Selection Method

Kate Smith-Miles, Senior member IEEE and Rafiqul Islam

Abstract – The purpose of instance selection is to identify which instances (examples, patterns) in a large dataset should be selected as representatives of the entire dataset, without significant loss of information. When a machine learning method is applied to the reduced dataset, the accuracy of the model should not be significantly worse than if the same method were applied to the entire dataset. The reducibility of any dataset, and hence the success of instance selection methods, surely depends on the characteristics of the dataset, as well as the machine learning method. This paper adopts a meta-learning approach, via an empirical study of 112 classification datasets from the UCI Repository [1], to explore the relationship between data characteristics, machine learning methods, and the success of instance selection method.

I. INTRODUCTION

When dealing with very large-scale datasets, it is often a practical necessity to seek to reduce the size of the dataset, acknowledging that in many cases the patterns that are in the data would still exist if a representative subset of instances were selected. Further, if the right instances are selected, the reduced dataset can often be less noisy than the original dataset, producing superior generalization performance of classifiers trained on the reduced dataset. The goal of instance selection is to select such a representative subset of instances, enabling the size of the new dataset to be significantly reduced without compromising the accuracy of a particular classification algorithm, and cleaning it of noisy data that may otherwise affect the performance of classification algorithms [2].

The aim of this paper is to demonstrate how a meta-learning approach can be used to identify which datasets lend them to reducibility. We present a methodology that can be used to learn the relationship between the characteristics of the data and the performance of instance selection methods. We explore the statistical properties of a classification datasets that enable a significant reduction in the size of the training data to be achieved without compromising classification accuracy. We consider an extensive set of 112 classification problems from the UCI repository [1], and characterize each dataset with a clear set of statistical metrics. Using a instance selection method together with a variety of classifier, we evaluate the reducibility that can be achieved for each dataset, by finding the smallest subset of the training data that enables the accuracy to be not significantly worse than the accuracy obtained using variety of classifiers on the original training data. The relationship between the data characteristics and the performance of this naïve instance selection approach is then explored using both supervised and unsupervised learning methods. The proposed methodology can be readily extended to consider other instance selection methods from the major classes shown in Figure 1, and other classifiers.

The remainder of this paper is as follows: Section 2 discusses the instance selection problem in more detail, and reviews some of the existing approaches. The naïve method used in this paper is presented in the context of these other methods. Section 3 presents the methodology used in this study, including a description of the datasets used, and how their features are measured. Section 4 presents the experimental results, where the relationships in the meta-data are learned using both supervised and unsupervised learning approaches. Section 5 concludes the paper with a summary of the insights generated by this meta-learning process, and identifies opportunities to extend these ideas for future research.

II. INSTANCE SELECTION METHODS

There have been many instance selection methods proposed over the last four decades or so [2-4]. With the existence of so many different approaches and algorithms for instance selection though, it is natural to wonder which method is likely to perform best for a given dataset. The No-Free-Lunch Theorem [5] suggests that it is unlikely that there is a single method that will perform best on all datasets regardless of their characteristics or properties. Indeed, the comparison on instance selection algorithms performed by Grochowski and Jankowski [6] confirms that even the average performance of instance selection methods across a group of 6 UCI classification problems varies considerably, and also depends on which classification algorithm is applied to the reduced dataset.

Jankowski and Grochowski [3] classify instance selection approaches into three main types: i) noise filters [15-17] are decremental algorithms which remove instances whose class labels do not agree with the majority of their neighbours; ii) condensation algorithms [18-22] are typically incremental algorithms that add instances from the training data to a new dataset if they add new information, but not if they have the same class label as their neighbours; iii) prototype
construction methods [23-25] do not focus on selecting which instances of the training data to include in the reduced dataset, but create new instances which are representative of the whole dataset via data squashing [26] or clustering methods [27, 28].

Reinartz [4] provides a brief summary of some experiments performed to explore how the performance of various simple sampling methods varied with changing data characteristics. According to Reinartz’s unifying view on instance selection [4] the first two types of instance selection methods (noise filters and condensation algorithms) can also be considered prototype selection methods (deciding which instances in the training data to include in the reduced dataset, using either incremental or decremental methods), while the third type are basically prototype construction approaches which seek to find new instances that can represent the whole dataset more compactly. Added to the group of prototype selection methods are those based on random sampling [29, 30] which randomly select instances at first and then identify instances to swap based on goodness measures. Figure 1 provides a taxonomic summary of the related literature and the various approaches to instance selection.

There are other instance selection methods which combine elements of clustering and prototype selection. Leader sampling [4] identifies prototypes (leaders) based on clustering, and these prototypes represent a set of instances that are close enough to the leader (within a similarity threshold). New leaders are identified to represent any instances which are not close enough to a leader.

We adopt in this paper another approach, related to leader sampling, but quite simpler. Prototype points (leaders) are identified through the k-means clustering algorithm [31]. The prototypes are not used for constructing new instances, but form the basis of a prototype selection process. From each cluster we select the closest (100-%) of the cluster size measured as the Euclidean distance from the cluster centroid (leader). This is a form of stratified sampling based on the similarity of the instances, rather than the class labels, and thus is quite naïve since priori knowledge about class probabilities is not being used. Of course, this strategy means that it is not being used as a noise filter based on class membership, and so it is closer to a condensation algorithm. The data reduction achieved is β%. We vary the value of β to explore the effectiveness of a classification algorithm on the reduced dataset, compared to the performance of the classification algorithm on the original dataset. Naturally, any of the instance selection methods discussed in this section could have been selected, but for the sake of demonstrating the meta-learning methodology, we have selected to focus on this instance selection method working in partnership with a range of classifiers. There is no doubt that many of the more sophisticated instance selection methods would yield improved accuracy for the classifier, but the point here is to explore how the performance on a given instance selection method varies with instance characteristics. The methodology is broadly applicable and extendable to other instance selection methods and classification algorithms.

III. METHODOLOGY

In this section we describe the experimental meta-data used for learning the relationships between classification problem features and the reducibility achieved by the instance selection method coupled with a range of different classifier algorithms. We also provide a description of the machine learning algorithms applied to the meta-data to produce rules and visualizations of these relationships.

A. Generating Meta-Data

Using the notation of Rice [7], the meta-data used in this study comprises a set of 112 classification problems (P) selected from the UCI Repository [1] (see Appendix A); the set of algorithms (A) in this study comprises the combination of the naïve instance selection method described in Section 2 implemented with various values of β, with each one followed by the selected classifier; the performance metric (Y) is the maximum percentage reduction in data size possible while maintaining a classification accuracy that is not statistically significantly worse than the accuracy obtained on the original (complete) dataset; and the set of features (F) used to characterize the classification problems comprises a set of statistical metrics described below.

B. Statistical Features

Each data set can be described by simple, distance and distribution-based statistical measures [32]. Let \( X_{ikj} \) be the value of the \( j_{th} \) attribute (column) in the \( k_{th} \) instance (row) of dataset \( i \). These three types of measures characterise the data set in different ways. Firstly, the simple classical statistical measures identify the data characteristics based on attribute to attribute comparisons (i.e. comparisons between columns of the dataset). Then, the distance based measures identify the data characteristics based on instance to instance comparisons (between rows of the data set). Finally, the density-based measures consider the relationships between single data points and the statistical properties of the entire data matrix to identify the data set features. All of these statistical measures have been briefly summarised in [32]. The simple statistical measures are calculated within each column, and then averaged over all columns to obtain global measures of the data set. Likewise, the distance measures are averaged over all pair wise comparisons, and the density based measures are averaged across the entire matrix. For each dataset \( i \), a total of 29 measures are calculated (11 statistical, 3 distance-based, 15 density-based). The data set feature matrix is then assembled with the columns comprising the 29 features, and the rows comprising the 112 datasets.
C. Algorithm Implementation

The algorithm used in this experimental study is a combination of the instance selection method to build a reduced dataset coupled with the selected classification algorithms. Bayesian classifiers are derived from Bayesian Decision Theory [10]. The decision tree (DT) algorithm is a simple rule-based algorithm based on a set of rules which takes advantage of the sequential structure of decision tree branches so that the order of checking rules and corresponding actions is immediately noticeable. Support vector machine (SVM) is a powerful, state-of-the-art algorithm with strong theoretical foundations based on Vapnik’s theory [13]. SVM has a strong data regularization property and can easily handle high dimensional feature spaces. Random Forest (RF) is a classifier that is based on a combination of many decision tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. RF has excellent accuracy among current classifier algorithms [11]. The IB1 (Instance Based 1) algorithm is the simplest instance-based learning algorithm; it is a nearest neighbour algorithm which in addition normalizes its attributes’ ranges, processes instances incrementally, and has a simple policy for tolerating missing values [9].

The analysis of the literature on different classification algorithms led to our choice of five algorithms, SVM, NB (Naïve Bayes), DT, RF, IB1 representing the spectrum of major classification techniques available. The data we obtained shows that not all algorithms performed equally well; however, in each case, SVM and RF showed improved performance compared to others.

The pseudo-code for the algorithm is as follows:

Step 1: Randomly divide original data set (containing N instances) into 50% for training (D_t) and 50% for testing or evaluation (D_e). Let the size of each dataset be denoted by N_t=N_e=0.5N;

Step 2: Cluster the training data (D_t) using the k-means algorithm, for a given value of k, selecting 1≤k≤N_t. Cluster j contains N_j instances, for 1≤j≤k;

Step 3: Identify each of the k cluster centroids as a “leader”, and build a new dataset by selecting for each cluster the closest η instances (in Euclidean space) around leader j, where η = N_j*(100-β)%, for a selected value of β. The new reduced dataset D_r(β) is β% smaller than the original training data D_t;

Step 4: The reduced dataset D_r(β) is used to train a classifier (using the default parameter settings in Weka [33])

Step 5: The evaluation dataset D_e (50% of the original data) is applied to the classifier, and the accuracy of the classification is recorded as γ_r(β);

Step 6: Repeat Steps 3-5 for varying values of β (0≤β≤100 in steps of 10), recording the accuracy of the classifier when using the entire dataset D (when β=0) as γ;

Step 7: Identify the value of β, denoted by β*, when the difference between γ and γ_r(β) first becomes statistically significant (using a t-test with p=0.05). Clearly, this value of β* depends on the value of k, but also on the features of the dataset.

Step 8: Repeat Steps 2-7 for varying values of k.

Based on an initial small study of randomly selected UCI repository problems, we found this approach to be quite robust to the value of k, with all problems following a similar contour when plotting how the test set accuracy varied with β (see Figure 2). Regardless of the number of clusters (k-value), the point at which data reduction was no longer possible seems similar, but as expected, the rate of reduction in accuracy is faster for small numbers of clusters if the sampling around the cluster centre is more limited. The pattern shown in Figure 2 was similar for several randomly chosen datasets. Since we are only interested in identifying the smallest subset that retains the original accuracy of the model trained on all of the data, we arbitrarily set the number of clusters as 10% of the size of the training data, so that k=0.1N_t. Naturally, different values of k can be tested within this methodology though.
Figure 3 shows the impact of the data reduction based on prototype points (leaders) through the k-means clustering algorithm. It is clear that the rate of data reduction is direct proportional to the distance of the prototype points, however, the reduction rate to some extent depends of the data characteristics.

Figure 4 shows the performance of several of the randomly selected subset of datasets, using \( k = 0.1N \). It is clear that even fixing the value of \( k \) based on the size of the dataset still reveals much differentiation in performance of the algorithm, and different optimal \( \beta \) values for each dataset due to the different characteristics of the datasets. It is the relationship between these characteristics and the kind of performance results we observe in Figure 4. We have extended across all 112 problems shown in Figure 5 that we now seek to learn based on the meta-data.
Figure 5 demonstrates the impact of various classifiers into our data reduction process. As mentioned earlier, we have chosen five base classifiers in our research to observe the data characteristics across all 112 data set. It has been shown from our empirical study that the reduction rate varies from classifier to classifier. It is very difficult to select a particular classifier which will always show better performance, although SVM and RF tend to perform well as indicated in Figure 5.

In Figure 5, the vertical axis shows the % of data reduction and the horizontal line shows the sorted 112 dataset (largest to smallest) based on the number of samples. It can be seen from the classification results that the reduction rate depends on the number of samples of the data sets. Higher sample gives higher reduction rate and vice versa, as we would intuitively expect, but it may be that other features besides size affect the reducibility of a dataset. The less-obvious relationships are what we seek when applying knowledge discovery processes to the meta-data.

D. Knowledge Discovery on the Meta-Data

When exploring any data-set to discover knowledge, there are two broad approaches. The first is supervised learning (aka directed knowledge discovery) and second is unsupervised learning (aka undirected knowledge discovery). The first approach is useful for building models to predict which algorithm or heuristic is likely to perform best given only the feature vector as inputs. The second approach is useful for our goal of seeking greater insight into why certain algorithms (combinations of instance selection methods and classification algorithms) might be better suited to certain datasets, rather than just building predictive models of algorithm performance. In this section we briefly summarise the methods we have used for knowledge discovery on the meta-data. The machine learning algorithm used in this section is NB. In order to determine our experimental study, we have selected 14 features out of the 29 features based on a correlation analysis

E. Neural Networks.

As a supervised learning method [35], neural networks can be used to learn to predict the data reduction capability ($\beta^*$) of a dataset using a certain algorithm (instance selection method and classifier). In the case of multiple competing algorithms, the neural network can be used to predict the relative performance of the algorithms, thus solving the Algorithm Selection Problem [7] via supervised learning. A training dataset is randomly extracted (80% of the 112 problems) and used to build a non-linear model of the relationships between the input set (features F) and the output (metric Y). Once the model has been learned, its generalisation on an unseen test set (the remaining 20% of the datasets) is evaluated and recorded as percentage accuracy in predicting the performance of the algorithm. This process is repeated five times for different random extractions of the training and test sets, to ensure that the results were not simply an artifact of the random number seed.

For our experimental results, the neural network implementation within the Weka machine learning platform [33] was used with 14 input nodes (excluding ResErr), 18 hidden nodes, and a single output node. The transfer function for the hidden nodes was a sigmoidal function, and the neural network was trained with the backpropagation (BP) learning algorithm with learning rate $= 0.3$, momentum $= 0.2$. The BP algorithm stops when the number of epochs (complete presentation of all examples) reaches a maximum training time of 500 epochs or the error on the test set does not decrease after a threshold of 20 epochs.

F. Decision Tree

A decision tree [35] is also a supervised learning method, which uses the training data to successively partition the data, based on one feature at a time, into classes. The goal is to find features on which to split the data so that the class membership at lower leaves of the resulting tree is as “pure” as possible. In other words, we strive for leaves that are comprised almost entirely of one class only. The rules describing each class can then be read up the tree by noting the features and their splitting points. Five-fold cross validation is also used in our experiments to ensure the generalisation of the rules.

In order to apply a decision (classification) tree to our problem, we first discretize the $\beta^*$ values into three categories: LOW corresponding to a $\beta^*$ value under 20%, REDUCIBILITY, HIGH corresponding to a $\beta^*$ value greater than 40%, and MEDIUM corresponding to a $\beta^*$ value between 20% and 40%. These bins were determined based on an examination of the frequency distribution to ensure a reasonable distribution of the datasets based on their relative reducibility. The J4.8 decision tree algorithm, implemented in Weka [33], was used for our experimental results based on the 14 features (excluding ResErr), with a minimum leaf size of 10 datasets. The generated decision tree is pruned using subtree raising with confidence factor $= 0.25$.

G. Self-Organizing Maps

Self-Organizing Maps (SOMs) are the most well-known unsupervised neural network approach to clustering. Their advantage over traditional clustering techniques such as the k-means algorithm lies in the improved visualization capabilities resulting from the two-dimensional map of the clusters. Often patterns in a high dimensional input space have a very complicated structure, but this structure is made more transparent and simple when they are clustered in a lower dimensional feature space. Kohonen [36] developed SOMs as a way of automatically detecting strong features in data sets. SOMs find a mapping from the high dimensional input space to low dimensional feature space, so any clusters that form become visible in this reduced dimensionality. The architecture of the SOM is a multi-dimensional input vector connected via weights to a 2-dimensional array of neurons. When an input pattern is presented to the SOM, each neuron calculates how similar the input is to its weights. The neuron
whose weights are most similar (minimal distance in input space) is declared the winner of the competition for the input pattern, and the weights of the winning neuron, and its neighbours, are strengthened to reflect the outcome. The final set of weights embeds the location of cluster centres, and is used to recognize to which cluster a new input vector is closest.

IV. EXPERIMENTAL EVALUATION

For our experiments we randomly split the 112 problems into training data (80%) and test data (20%). We use the Viscovery SOMine software (www.eudaptics.com) to cluster the instances based only on the 14 features (excluding ResErr) as inputs. A map of 2000 nodes is trained for 41 cycles, with the neighbourhood size diminishing linearly at each cycle. After the clustering of the training data, the distributions of β* and ResErr values are examined within each cluster, and knowledge about the relationships between problem structure and algorithm performance is inferred and evaluated on the test data.

The neural network results demonstrate that the relationships between the features of the datasets and the reducibility of each dataset using the selected algorithm can indeed be learned to a very high accuracy. Based on the five-fold cross validation testing procedure, an R squared value of 0.941 was obtained. These prediction results outperform the original regression model’s R squared value of 0.87. While the neural network can be expected to learn the relationships in the data more powerfully, due to its nonlinearity, its limitation is the lack of insight and explanation of those relationships.

If (N >=768) Then
If (KU >=1.04) Then
    If (Npdf >=12.6) Then HIGH (100%)
    Else MEDIUM (83%)
Else MEDIUM (89%)
Else LOW (96%)  

Fig. 6. Pseudo-code for the decision tree rule system, showing the accuracy of each rule

The decision tree results produced classification accuracy, on five-fold cross-validation, of 87.5%, with most of the errors due to misclassification of some HIGH datasets as MEDIUM. The arbitrariness of the discretization bins may be contributing to this performance. Figure 6 shows the resulting decision tree rules, with the confidence for each rule shown in brackets. These rules suggest that, based on the meta-data, if the number of instances in a dataset is too small (below 768 instances) then the reducibility of the dataset using the chosen algorithms is likely to be low. This makes sense given the way the algorithm works, particularly the chosen value of k in the k-means clustering algorithm being a fraction (10%) of the number of training instances available. For datasets with a higher kurtosis and higher Normal pdf value the reducibility of the datasets in the metadata tends to be higher, since more of the variance is due to infrequent extreme deviations which can be eliminated without as much impact.

The advantage of the Self-organizing Map is its visualization capabilities, enabling the exploration of visual correlations and patterns between features and clusters of datasets. After training the SOM based on the 14 input features (excluding ResErr and β* values), the converged map shows 7 clusters, each of which contains similar datasets defined by Euclidean distance in feature space (see Figure 7). Essentially, the 14-dimensional input vectors have been projected onto a two-dimensional plane, with topology-preserving properties. The clusters can be inspected to see which datasets are most similar, and a statistical analysis of the features within each cluster can be performed to understand what the datasets within each cluster have in common.

The distribution of the input features, and additional information including the distribution of β* values, can be visually explored using the maps shown in Figure 7 (not all features have been included). A k-nearest neighbour algorithm (with k=7) is used to distribute additional data instances (from the test set) or extra variables (β* values) across the map.

In addition to supporting the kind of rules generated by the decision tree (see Figure 8), this kind of visual exploration can be used to identify the group of datasets that are not very reducible using the chosen algorithm (clusters 1, 5 and 6), and to observe that it is not just the number of instances that creates this performance (although all analysis performed so far including the correlation analysis supports the number of instances as a primary determinant of the reducibility of the chosen algorithm). In fact there is significant variation within cluster 2 of the reducibility of the datasets, despite most datasets in cluster 2 being of similar size. The residual errors for these datasets are high suggesting that these datasets were not well modeled by the regression model, and may have some unique properties. The success of the SOM to assist with developing insights relies very heavily on the quality of the features selected to explore the relative differences between the datasets. A visual exploration of the distribution of the 14 features across these clusters suggests that none of the existing features can, on their own, explain well the behaviour of the datasets in the middle of map. The rules shown in Figure 7 are supported by the observation of cluster 3, 4, and 7. The benefit of the SOM is that if new features are derived, their values can easily be superimposed across the map to explore if they help to explain the performance of certain datasets.
Fig. 7. Self-Organizing Map showing 7 clusters, with classification datasets labeled within each cluster.

Fig. 8. The distribution of $\beta^*$ values (Fig. 8a) and several key features (Fig. 8b-8h) across the clusters. The colour scale shows each feature at its minimum value as blue, and maximum value as red.
V. CONCLUSIONS AND FUTURE RESEARCH

In this paper we have generalized the meta-learning process to consider what can be learned about important data pre-processing steps such as instance selection. We proposed a methodology that can be used to learn the relationship between the characteristics of the data and the performance of instance selection methods. The accuracy of a classifier depends strongly on which instances are selected as representatives of the dataset. In our experimental study, we have illustrated the methodology using instance selected method based on clustering, and a range of chosen classifiers. The performance metric is the level of reducibility this algorithm managed to achieve on a dataset.

It is clear from the experimental results that the selected features enabled the performance of the algorithm to be predicted to a high accuracy, and that rules can be generated to explain the conditions under which the algorithm performs well or poorly. We have utilized a subset of the features used in previous meta-learning studies of classification problems, but in future research it is advisable that these features be re-examined. The Self-organising map visualizations revealed that while some of the algorithm performance can be explained by the decision tree rules, and visual observations were able to confirm these rules, there are clusters of datasets whose performance was not readily explained by the selected features. The construction of features to suitably characterize a dataset, and capture the diversity of problem difficulty that may enable the relative strengths and weaknesses of a range of algorithms to be exposed, remains one of the main challenges in meta-learning.

REFERENCES