Comparison of Instances Seletion Algorithms I. Algorithms Survey

Norbert Jankowski and Marek Grochowski

Department of Informatics, Nicholaus Copernicus University
ul. Grudziądzka 5, 87-100 Toruń, Poland, http://www.phys.uni.torun.pl/kis
{norbert|grochu}@phys.uni.torun.pl

Abstract. Several methods were proposed to reduce the number of instances (vectors) in the learning set. Some of them extract only *bad* vectors while others try to remove as many instances as possible without significant degradation of the reduced dataset for learning. Several strategies to shrink training sets are compared here using different neural and machine learning classification algorithms. In part II (the **accompanying paper**) results on benchmarks databases have been presented.

1 Introduction

Most algorithms to train artificial neural networks or machine learning methods use all vectors from the training dataset. However, there are several reasons to reduce the original training set to smaller one. The first of them is to reduce the noise in original dataset because some learning algorithms may be noise-fragile (for example, plain linear discrimination methods [1]). The second reason to shrink the training set is to reduce the amount of computation, especially for instance-based learning (or lazy-learning) algorithms [2] such as the k-nearest neighbors [3], or for huge training sets. The third and relatively new reason to use vector selection appeared together with new prototype selection algorithms. These algorithms shrink training sets sometimes even below 1% of original size keeping the accuracy for unseen vectors high. As the results of shrinking good prototype vectors are selected. Such prototypes may be seen as knowledge representation — each prototype represent a cluster in simple[1] Voronoi diagram.

Probably the first instance selection algorithm was proposed by Hart in the Condensed Nearest Neighbor Rule (CNN) [4]. As will be shown below CNN condenses on the average the number of vectors three times. The performance of CNN algorithm is not good, but this model inspired construction of new methods such as SNN by Ritter et al. [5], RNN by Gates [6] or ENN by Wilson [7]. A group of three algorithms were inspired by *encoding length* principle [8]. Other algorithms were derived from graph theory [9], sets theory [10] or Monte Carlo sampling [11].

Typically the performance of selection methods has been tested usually only on the k-nearest neighbors model. Tests of the performance of instance selection

---

[1] Simple because of few prototypes.

methods is carried in this paper on machine learning algorithms and neural network algorithms. From the ML group of algorithms the k-nearest neighbor, support vectors machine [12] and SSV decision tree [13] has been chosen. From the artificial neural network domain the NRBF (a normalized version of RBF network), FSM model [14] and IncNet [15] algorithms have been selected.

2   Short Survey of Instance Selection Algorithms

Algorithms for selection of instances may be divided in three application-type groups: noise filters, condensation algorithms and prototype searching algorithms. Because of space limitation full description cannot be given here but details of algorithms presented below may be found in the bibliographical links. Let's assume that there is a training set $\mathcal{T}$ which consists of pairs $\langle \mathbf{x}_i, y_i \rangle, i = 1, \ldots, n$, where $\mathbf{x}_i$ defines input vector of attributes and y_i defines the corresponding class label.

2.1   Noise Filters

Edited Nearest Neighbor (ENN) algorithm was created in 1972 by Wilson [7]. The main idea of ENN is to remove given instance if its class does not agree with majority class of its neighbors. ENN starts from original training set.

Repeated ENN was also proposed by Wilson. The only difference is that the process of ENN is repeated as long as any changes are made in the selected set.

All k-NN presented by Tomek in [16] is another modification of ENN algorithm: the ENN is repeated for *all k* (k=1,2, ... ,l).

ENRBF is an *Edited* version of NRBF[17,18]. NRBF is defined as normalized version of RBF. NRBF estimates probability of k-th class given vector $\mathbf{x}$ and training set $\mathcal{T}$:

$$P(k|\mathbf{x}, \mathcal{T}) = \sum_{i \in I^k} \bar{G}_i(\mathbf{x}; \mathbf{x}_i), \tag{1}$$

where $I^k = \{i : \langle \mathbf{x}_i, y_i \rangle \in \mathcal{T} \wedge y_i = k\}$, and $\bar{G}_i(\mathbf{x}; \mathbf{x}_i)$ is defined by

$$\bar{G}_i(\mathbf{x}; \mathbf{x}_i) = \frac{G(\mathbf{x}; \mathbf{x}_i, \sigma)}{\sum_{j=1}^{n} G(\mathbf{x}; \mathbf{x}_j, \sigma)}, \tag{2}$$

and $G(\mathbf{x}; \mathbf{x}_i, \sigma)$ (σ is fixed) is defined by $G(\mathbf{x}; \mathbf{x}_i, \sigma) = e^{-\frac{||\mathbf{x} - \mathbf{x}_i||^2}{\sigma}}$.

The ENRBF eliminates all vectors if only:

$$\exists_{k \neq y_i} \quad P(y_i|\mathbf{x}, \mathcal{T}^i) < \alpha P(k|\mathbf{x}, \mathcal{T}^i), \tag{3}$$

where $\mathcal{T}^i = \mathcal{T} - \{\mathbf{x}_i, y_i\}$, and $\alpha \in (0, 1]$.

2.2   Condensation Algorithms

Condensed Nearest Neighbor Rule (CNN) was made by Hart [4]. The CNN algorithm starts new data set from one instance per class randomly chosen from training set. After that each instance from the training set that is wrongly classified using the new dataset is added to this set. This procedure is very fragile in respect to noise and the order of presentation.

Reduced Nearest Neighbor (RNN) described in [6] by Gates was based on the same idea as CNN. However RNN starts from original training set and rejects only those instances that do not decrease accuracy.

IB3 was described by Aha et al. in [2]. IB3 is an incremental algorithm. Instance x from the training set is added to a new set $\mathcal{S}$ if the nearest *acceptable* instance in $\mathcal{S}$ (if there are no acceptable instance a random one is used) has different class than x. Acceptability is defined by the confidence interval

$$\frac{p + \frac{z^2}{2n} \pm z\sqrt{\frac{p(p-1)}{n} + \frac{z^2}{2n^2}}}{1 + \frac{z^2}{n}} \tag{4}$$

z is confidence factor (in IB3 0.9 is used to accept, 0.7 to reject). p is the classification accuracy of a given instance (while added to $\mathcal{S}$). n is equal to a number of classification-trials for given instance (while added to $\mathcal{S}$). See [2] for more details.

Gabriel Editing (GE) and Relative Neighborhood Graph Editing (RNGE) — two algorithms based on graph theory – were constructed by Bhattacharya et al. [9]. Decision surface of the 1-NN algorithm creates Voronoi diagram. It can be observed that instances on the border between classes are important in classification process. The complexity of building Voronoi diagrams is $O(N^{d/2})$, that is too expensive for real datasets. Because of that authors decided to use Gabriel graphs. The complexity of this algorithm is $O(n^3)$. Stronger instance–shrinking can be obtained using a modification of GE method called RNGE.

Iterative Case Filtering (ICF) was proposed by Brighton & Mellish in [10]. ICF defines *local set* $\mathcal{L}(\mathbf{x})$ which contain all cases inside largest hypersphere centered in $\mathbf{x}$ such that the hypersphere contains only cases of the same class as instance $\mathbf{x}$. Authors define two properties, *reachability* and *coverage*:

$$Coverage(x) = \{\mathbf{x}' \in \mathcal{T} : \mathbf{x} \in \mathcal{L}(\mathbf{x}')\}, \tag{5}$$
$$Reachability(x) = \{\mathbf{x}' \in \mathcal{T} : \mathbf{x}' \in \mathcal{L}(\mathbf{x})\}. \tag{6}$$

In the first phase ICF uses ENN algorithm to remove the noise from the training set. In the second phase ICF algorithm removes each instance $\mathbf{x}$ for which the *Reachability(x)* is bigger than the *Coverage(x)*. This procedure is repeated for each instance in $\mathcal{T}$. After that ICF recalculates *reachability* and *coverage* properties and restarts the second phase (as long as any progress is observed).

ENRBF2 is also based on NRBF defined by Eq. 1. ENRBF2 removes given instance $\mathbf{x}_i$ from the training set if the criterion below is satisfied:

$$P(y_i|\mathbf{x}_i; \mathcal{T})\beta < P(y_i|\mathbf{x}_i; \mathcal{T}^i) \tag{7}$$

$\beta \in (0, 1]$. This means that if removing instance $\mathbf{x}_i$ probability that this instance belongs to the class y_i is not significantly reduced then such instance can be removed.

DROP1–5 models were developed by Wilson & Martinez [19]. Let's define $\mathcal{A}(\mathbf{x})$ as a set of instances for which instance $\mathbf{x}$ is one of the k nearest neighbors. DROP1 removes instance $\mathbf{x}$ from the training set if it does not change classification of instances from $\mathcal{A}(\mathbf{x})$ (only those instances depend on $\mathbf{x}$). The performance of DROP1 was really bad. The second version — DROP2 — starts the process from sorting instances according to their distances from the nearest opposite class instance. The DROP3 additionally run the ENN algorithm before starting the DROP2 algorithm. DROP4-5 are another version of DROP2 – see [19] for details.

2.3   Prototype Selection

Prototypes methods are very interesting because original training set may be transformed even to a few prototypes, therefore it can be treated as an approach for knowledge representation. Each of the prototypes represents one field covering its Voronoi diagram. Such field in highly compressed dataset (few vectors) corresponds to a cluster. If prototypes are used with 1NN the prototypes may be seen as prototype rules because each prototype is assigned to a single class. For example if $\mathcal{S}$ contains prototype vectors $\mathbf{p}_i$ and its corresponding classes are c_i than the decision process of instance $\mathbf{x}$ is simplified to find i:

$$i := \arg \max_{\mathbf{p}_j \in \mathcal{S}} ||\mathbf{p}_j - \mathbf{x}|| \tag{8}$$

which points to the winner class c_i.

Learning Vectors Quantization (LVQ) is well known model proposed by Kohonen in [20]. In contrary to all previous algorithms (except CA) LVQ changes the positions of codebook vectors (neurons) during learning and finally neurons have different values than original instances of the training set. See [20] for more.

Monte Carlo 1 (MC1) and Random Mutation Hill Climbing (RMHC). These two methods described in [11] by Skalak are based on stochastic behavior. MC1 in each iteration use Monte Carlo to draw new set of instances and to test the accuracy. Only the best drawn set of instances is remembered. RMHC use mutation in place of Monte Carlo.

Encoding length — ELH, ELGrow and Explore: these three algorithms (Cameron-Jones [8]) use cost function defined by:

$$J(m, n, x) = F(m, n) + m \log_2 c + F(x, n - m) + x \log_2(c - 1), \tag{9}$$

where n and m are numbers of instances in the training set and in the new data set $\mathcal{S}$ respectively. x defines the number of badly classified vectors (basing on $\mathcal{S}$), and $F(m, n)$ is defined by

$$F(m, n) = \log^* \left(\sum_{i=0}^{m} \frac{n!}{i!(n-i)!} \right), \tag{10}$$

$\log^* n = \arg\min_k F(k) \geq n$, k – is integer, and $F(0) = 1, F(i) = 2^{F(i-1)}$.

ELH starts from the empty set and adds instances if only they minimize the cost function $J(\cdot)$. ELGrow additionally tries to remove instances if it helps to minimize the cost function $J(\cdot)$. Explore extend the ELGrow by 1000 iterations of stochastic addition or deletion of instances if only it minimizes the costs. Those methods are very effective.

The algorithm DEL is another modification of ELH. It can be seen as decremental version of ELH [19].

2.4   Classification of Algorithms

Instance selection algorithms work in different ways; some of them belong to incremental or decremental family, while others try to mix both strategy.

Instance dataset creation strategy	
Incremental	CNN, IB3, ELH
Decremental	RNN, SNN, ENN, CA (Chang), ENRBF, DROP1-5, Del
Mixed	RENN, All k-NN, LVQ, MC1, RMHC, ELGrow(!), Explore

However, more important than strategy of the dataset building presented above is the complexity of presented algorithms. In the table below complexity comparison can be found.

ENN	RENN All-kNN	CNN RNN	IB3	GE RNGE	ICF	EN-RBF(2)	DROP 1-5	LVQ	MC1 RMHC	ELH ElGrow Explore Del
$O(n^2)$	$O(in^2)$	$O(n^3)$	$O(n^2 \log_2 n)$	$O(n^3)$	$O(in^2)$	$O(n^2)$	$O(n^3)$	$O(in^2)$	$O(n^2)$	$O(n^2)$

Another feature which distinguish between groups of model is their "scene analysis". Some algorithms try to preserve the border points. This can be observed especially with algorithms based on graph theory – GE and RNGE. In contrary are algorithms which try to estimate cluster centers, like the LVQ. In the next group models which remove the noise can be placed. Instances that remained have clusters with smoother shapes. More sophisticated behavior can be observed in such algorithms as ICF, DROP3-5, Encoding Length or MC1.

References

1. Duda, R.O., Hart, P.E., Stork, D.G.: Patter Classification and Scene Analysis. 2 edn. Wiley (1997)
2. Aha, D.W., Kibler, D., Albert, M.K.: Aha,. Machine Learning **6** (1991) 37–66
3. Cover, T.M., Hart, P.E.: Nearest neighbor pattern classification. Institute of Electrical and Electronics Engineers Transactions on Information Theory **13** (1967) 21–27
4. Hart, P.E.: The condensed nearest neighbor rule. IEEE Transactions on Information Theory **14** (1968) 515–516
5. Ritter, G.L., Woodruff, H.B., Lowry, S.R., Isenhour, T.L.: An algorithm for a selective nearest neighbor decision rule. IEEE Transactions on Information Theory **21** (1975) 665–669
6. Gates, G.: The reduced nearest neighbor rule. IEEE Transactions on Information Theory **18** (1972) 431–433
7. Wilson, D.: Asymptotic properties of nearest neighbor rules using edited data. IEEE Transactions on Systems, Man, and Cybernetics **2** (1972) 408–421
8. Cameron-Jones, R.M.: Instance selection by encoding length heuristic with random mutation hill climbing. In: Proceedings of the Eighth Australian Joint Conference on Artificial Intelligence. (1995) 99–106
9. Bhattacharya, B.K., Poulsen, R.S., Toussaint, G.T.: Application of proximity graphs to editing nearest neighbor decision rule. In: International Symposium on Information Theory, Santa Monica (1981)
10. Brighton, H., Mellish, C.: Advances in instance selection for instance-based learning algorithms. Data Mining and Knowledge Discovery **6** (2002) 153–172
11. Skalak, D.B.: Prototype and feature selection by sampling and random mutation hill climbing algorithms. In: International Conference on Machine Learning. (1994) 293–301
12. Schölkopf, B., Smola, A.: Learning with Kernels. MIT Press, Cambridge, MA (2002)
13. Grąbczewski, K., Duch, W.: A general purpose separability criterion for classification systems. In: 4th Conference on Neural Networks and Their Applications, Zakopane, Poland, Polish Neural Networks Society (1999) 203–208
14. Adamczak, R., Duch, W., Jankowski, N.: New developments in the feature space mapping model. In: Third Conference on Neural Networks and Their Applications, Kule, Poland, Polish Neural Networks Society (1997) 65–70
15. Jankowski, N., Kadirkamanathan, V.: Statistical control of RBF-like networks for classification. In: 7th International Conference on Artificial Neural Networks, Lausanne, Switzerland, Springer-Verlag (1997) 385–390
16. Tomek, I.: An experiment with the edited nearest-neighbor rule. IEEE Transactions on Systems, Man, and Cybernetics **6** (1976) 448–452
17. Grochowski, M.: Wybór wektorów referencyjnych dla wybranych method klasyfikacji. Master's thesis, Department of Informatics, Nicholas Copernicus University, Poland (2003)
18. Jankowski, N.: Data regularization. In Rutkowski, L., Tadeusiewicz, R., eds.: Neural Networks and Soft Computing, Zakopane, Poland (2000) 209–214
19. Wilson, D.R., Martinez, T.R.: Reduction techniques for instance-based learning algorithms. Machine Learning **38** (2000) 257–286
20. Kohonen, T.: Learning vector quantization for pattern recognition. Technical Report TKK-F-A601, Helsinki University of Technology, Espoo, Finland (1986)