Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 35 (2014) 388 – 396

Procedia Computer Science

18th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems - KES2014

Ensembles of instance selection methods based on feature subset

Marcin Blachnik[a,b]

[a] *Silesian University of Technology, Department of Industrial Informatics, Akademicka 2A, Gliwice, 44-100, Poland*
[b] *National Centre for Nuclear Research, ul. Andrzeja Sołtana 7, Otwock-Świerk, 05-400, Poland*

Abstract

In this paper the application of ensembles of instance selection algorithms to improve the quality of dataset size reduction is evaluated. In order to ensure diversity of sub models, selection of a feature subsets was considered. In the experiments the Condensed Nearest Neighbor (CNN) and Edited Nearest Neighbor (ENN) algorithms were evaluated as basic instance selection methods. The results show that it is possible to obtain various trade-offs between data compression and classification accuracy depending on the *acceptance threshold* and *feature ratio* parameters. In some cases it was possible to achieve both: higher compression and higher accuracy than those of an individual instance selection algorithm.

Keywords: Instance selection; model ensembles; machine learning

1. Introduction

One of the most popular and intuitive machine learning methods used for both classification and regression problems is the nearest neighbor algorithm (kNN)[1]. In typical classification tasks the goal is to find a function $f(\cdot)$, which performs the mapping $f(\mathbf{z}) \rightarrow c$, where $\mathbf{z}$ is called a test vector and c is a symbol from a set of l possible class labels $c \in \{s_1, ..s_l\}$. In other words the function $f(\cdot)$ should assign c value to vector $\mathbf{z}$. To determine the mapping $f(\cdot)$ the kNN classifier remembers the training set $\mathbf{T}$ consisting of n-pairs $\{\mathbf{x}, c\}$, $\mathbf{T} = [\{\mathbf{x}_1, c_1\}, \{\mathbf{x}_2, c_2\}, ..., \{\mathbf{x}_n, c_n\}]$ where each vector $\mathbf{x}$ is determined in an m - dimensional space (the same space as $\mathbf{z}$) and c is a symbol $c \in \{s_1, ..s_l\}$. The mapping is obtained by the majority vote of the class labels c of the k instances $\mathbf{x}$ that are closest to the query instance $\mathbf{z}$ according to some distance function $D(\cdot)$ (usually Euclidean distance).

The quality and the prediction speed of the kNN classifier depends on the training set $\mathbf{T}$. The general rules say:

- The larger the training set is, the more distances should be calculated to determine the nearest neighbors, thus decreasing the speed of decision making.

* Corresponding author. Tel.: +48-32-603-4270; fax: +48-32-603-4280.
E-mail address: marcin.blachnik@polsl.pl

- The more noisy the training set is (with higher number of outliers), the less accurate the prediction is.

Instance selection methods were developed to overcome these limitations. The instance selection algorithms try to reduce the training set size, so to decrease the value of n in this way that the new size n' of the reduced training set (often called a prototype set) $\mathbf{P}$ is $n' \ll n$. The relation of n' to n is called compression - which is discussed in section 2. The second aim of using instance selection is to improve the prediction accuracy of the classifier. To achieve that goal these algorithms are designed to remove outliers and noisy instances.

During the last five decades many new algorithms in the area of instance selection were developed and an overview of them can be found in[2,3,4]. Nowadays, these algorithms can bring many benefits to the community by their applications to the Big Data challenge. Their application to massive datasets can lead to the reduction of the dataset size by discovering important patters and rejecting noisy and redundant samples. However individual algorithms have to be chosen carefully, considering its computational complexity. The idea of using instance selection as automatic data filters and running different machine learning methods on the top of the selected instances was already presented and initial results can be found in[5]. In[6,7] Kordos and Rusiecki have shown that using noise filters like ENN algorithm (which will be discussed later) leads to more accurate results in comparison to other advanced methods dedicated to dealing with noise in MLP neural networks training.

In the last decade also the ensemble learning has become one of the most promising machine learning approaches. It takes advantage of combining several models, which grouped together are able to outperform each individual method with only a linear increase in computational complexity.[8].

These group of algorithms includes several state of the art approaches such as:

- Voting - where a set of c independent models of different type, all trained on the same training set $\mathbf{T}$ vote for the output of the instance being classified $\mathbf{t}$
- Stacking - is an extension of the voting approach, where c independent models of different type are trained on the same dataset $\mathbf{T}$, but the output of each of the models are combined using one extra model. This final model trained on the outputs of the submodes is used to make the final prediction.
- Bagging - which consists also of c models of the same type, where each of them is trained on the dataset $\mathbf{T}^*$ which is obtained from $\mathbf{T}$ by sampling with replacement. The final prediction is obtained by voting.
- Boosting - a similar approach to bagging, but the sampling probability of selecting an instance from $\mathbf{T}$ depends on the classification error of the previously built models, so that it is more likely to select an instance, which was incorrectly classified by the current models.

The general rule of achieving success with ensemble modeling is to ensure diversity of the obtained results of each individual model, which are then combined into the final model. The diversity of the results can be obtained in several ways

- by ensuring diversity of the models - in this approach usually several different algorithms are used or the same algorithms but with different hyper-parameters. An example of this approach is *Voting*
- by ensuring diversity of the data. In this case the model can stay the same, but the data used for training each of submodels generates the diversity. This can be achieved by:
  - manipulating the example set - an example of this approach is *Bagging*, where samples are sampled from the training data.
  - manipulating features - an example of this approach is sampling from the feature space, so that each model retrieves vectors in m_i^* dimensional space instead of m so that $m_i^* \leq m$, and each feature subspace is different.
  - manipulating class labels - an example of that is an application of error correcting output codes (ECOC) so that in each iteration different class labels are used during selection.
  - adding randomness - in this approach a small level of noise is added to the input data in each iteration loop. This approach was often used in MLP networks to prevent over-fitting.

Ensemble methods have not been wildly used in conjunction with instance selection, but recently this topic is becoming much more popular. A good example is a paper written by N. Garcia-Pedrajas[9] which addresses the concept of using instance selection to diverse the dataset, where the pair of instance selection and prediction models are all combined together. This topic has been also studied in[10].

In this paper we address the issue of using ensemble only for instance selection, so that only the instance selection is wrapped by the meta model and the diversity is obtained by manipulating the feature space. Our theses is that the ensemble of instance selection algorithms allows us to manipulate the trade-off between compression and prediction accuracy or even improve both.

The paper is organized as follows: the next section describes the basic algorithms used in the experiments and the ensemble models. The following section describes the testing environment and presents the numerical experiments. The last section summarizes the results and presents their interpretation.

2. Model description

Our experiments were based on two instance selection algorithms: Edited Nearest Neighbor (ENN)[11] and Condensed Nearest Neighbor (CNN)[1], which were wrapped by the ensemble algorithm. These two algorithms were selected because they are the most popular and simplest ones. Moreover, they represent two different group of instance selection methods. The first one - ENN is an example of noise filters, with computational complexity $O(n^2)$, where each instance is evaluated examining its k nearest neighbors. If voting of the k neighbors disagree with the class label of that instance, the instance is marked for removal (see Algorithm 2). ENN is useful for removing noise samples and thus by default its compression ratio (1) is low. Compression is a measure of the ability to reduce the size of the training set, so that it takes values close to 1 (100 %) if only a small subset of samples remain after removal, and values close to 0 if the size of the dataset after instance selection is similar to the original dataset.

$$C = \frac{|\mathbf{T}| - |\mathbf{P}|}{|\mathbf{T}|} \tag{1}$$

where C denotes compression, $|\mathbf{T}|$ - size of the original training set and $|\mathbf{P}|$ - size of the dataset after instance selection.

The CNN algorithm belongs to the condensation methods and tries to keep the smallest possible set of examples to preserve decision border of the nearest neighbor classifier. Its computational complexity is also $O(n^2)$. It starts with an empty set $\mathbf{P}$ and adds to it each instance, which is incorrectly classified by the current prototypes (reference vectors) collected in $\mathbf{P}$. This algorithm usually has high compression, what is its huge advantage. However, the problem, which can be deduced from its sketch (Algorithm 1) is its instability, so that the output set of prototypes $\mathbf{P}$ depends on the order of examples. Another important limitation of CNN is its sensitivity to the noise in the data; the noise always appears in the set of prototypes $\mathbf{P}$. To overcome this limitation CNN is usually preceded by a noise filter such as ENN algorithm[2]

As already described individual instance selection methods has some drawbacks. If they are designed to remove outliers they have small compression ratio, and if they are designed to preserve high compression, then they usually have poor accuracy.

An interesting approach to overcoming the limitations is the application of ensemble learning. The idea of stabilizing the instance selection process can be achieved by placing instance selection method in a loop, where at each iteration the algorithms work on a different subset of the original dataset. An example of such an approach is bagging, as we presented in[12]. In bagging the diversity is obtained by manipulating the input examples by sampling a subset of examples from the original input data $\mathbf{T}$. As described in the first section, another source of diversity can be obtained by the manipulation in the feature space, where in each iteration a subset of features $\mathbf{A}^{(i)}$ is sampled from the initial set of all features $\mathbf{A}$, where $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_m]$.

Algorithm 1 Schema of the CNN algorithm

```
Require: $\mathbf{T}$
  $n \leftarrow |\mathbf{T}|$
  $k \leftarrow 1$
  $\mathbf{p}_1 \leftarrow \mathbf{x}_1$
  $flag \leftarrow$ true
  while flag do
    $flag \leftarrow$ false
    for $i = 1 \ldots n$ do
      $\bar{C}(\mathbf{x}_i) =$1NN$(k, \mathbf{P}, \mathbf{x}_i)$
      if $\bar{C}(\mathbf{x}_i) \neq C(\mathbf{x}_i)$ then
        $\mathbf{P} \leftarrow \mathbf{P} \cup \mathbf{x}_i$;
        $\mathbf{T} \leftarrow \mathbf{T} \setminus \mathbf{x}_i$
        $flag \leftarrow$ true
      end if
    end for
  end while
  return $\mathbf{P}$
```

Algorithm 2 Schema of the ENN algorithm

```
Require: $\mathbf{T}, k$
  $n \leftarrow |\mathbf{T}|$;
  $\mathbf{P} \leftarrow \mathbf{T}$;
  for $i = 1 \ldots n$ do
    $\bar{C}(\mathbf{x}_i) =$kNN$(k, (\mathbf{T} \setminus \mathbf{x}_i), \mathbf{x}_i)$;
    if $C(\mathbf{x}_i) \neq \bar{C}(\mathbf{x}_i)$ then
      $\mathbf{P} \leftarrow \mathbf{P} \setminus \mathbf{x}_i$
    end if
  end for
  return $\mathbf{P}$
```

Algorithm 3 Schema of the ensemble algorithm

```
Require: $\mathbf{T}, samplingRatio, threshold$
  for $i = 1 \ldots t$ do
    $\mathbf{A}^* \leftarrow SampleFeatureSubspace(\mathbf{A}, r)$
    $\mathbf{T}^* \leftarrow AdjustDataSet(\mathbf{T}, \mathbf{A}^*)$
    $\mathbf{P} \leftarrow InstanceSelection(\mathbf{T}^*)$
    $\mathbf{v} \leftarrow CollectVotes(\mathbf{P}, \mathbf{v})$
  end for
  $\mathbf{P} \leftarrow SelectInstancesByVotes(\mathbf{T}, \mathbf{v}, threshold)$
```

The diagram of this method is presented in Algorithm 3. This algorithm includes two hyper-parameters. The *feature ratio* describes the ratio of the feature space delivered to the input of the embedded instance selection method. This parameter should be chosen carefully because too small values of the *feature ratio* may radically change the data distribution, therefore making the method useless and unpractical. The second parameter is the *acceptance threshold*, which expresses the percentage of votes required by a data sample to pass the selection process.

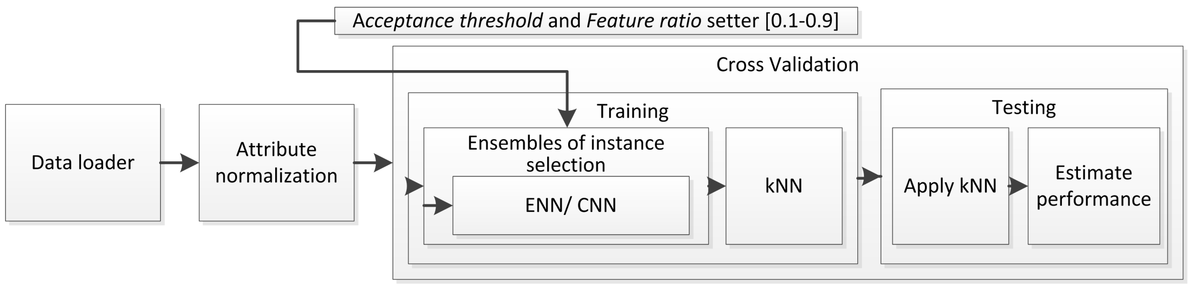


Fig. 1. Data flow diagram

Table 1. Datasets used in the experiments aaa

dataset	# features	# samples	# class labels
Heart Disease	13	303	2
Ionosphere	34	351	2
Pima Indian Diabetes	8	768	2
Sonar	60	208	2
Vehicle	18	846	4
Wisconsin Brest Cancer	9	699	2

3. Numerical experiments

Considering the discussion presented above we have decided to conduct the experiments, which empirically verify the influence of the ensemble learning method on the quality of the instance selection. In the experiments we examine the influence of different parameters on the compression of the training data and on the accuracy of the final prediction model. Because of the limitations of the size of the paper, we evaluated only the accuracy of the nearest neighbor classifier.

In our study we used the RapidMiner software with the Instance Selection and Prototype Based Rules (ISPR) extension[13], which we have developed. The scheme of the experiments are presented in Fig. 3 It starts by loading the data, then the numerical attributes of the dataset are normalized to range $[0, 1]$ and the rest of the process is wrapped by the 10-fold cross-validation. Inside the cross-validation the ensemble of instance selection algorithms is used to reduce the size of the training set, and finally the nearest neighbor classifier is built. The testing side consists of model application and evaluation of the accuracy of the model built on the training side. In the experiments some basic hyperparameters were fixed such as number of iterations of the ensemble algorithm $t = 10$, so each time 10 instance selection submodes were created and the k value of the ENN was set to $k = 3$. For the kNN classifier used for prediction the value of k was set to 1.

All of the experiments were performed on a six popular datasets obtained from the UCI[14]. These datasets are: Heart Disease, Ionosphere, Pima Indian Diabetes (Diabetes), Sonar, Vehicle, Wisconsin Brest Cancer (WBC).

Both of the embedded algorithms; ENN and CNN were tested independently and the results are presented using the *accuracy-gain plot*. In that kind of plots the values of both axes are obtained by the formulas $acc_{ML} - acc_{SM}$ and $comp_{ML} - comp_{SM}$ where the symbol acc_{ML} and $comp_{ML}$ represents the accuracy and compression of the ensemble learner respectively, and the acc_{SM} and $comp_{SM}$ the accuracy and compression of the single method (without ensemble). The plot interpretation is based on the analysis of the graphs with respect to the $[0, 0]$ coordinates, because they represent the performance of instance selection obtained without ensemble. A positive values on the compression or accuracy axis represents the gain (the benefit), and negative values represent the loss. Because the plot represent the values in relation to the reference point ([0,0] coordinates), the acc_{SM} and $comp_{SM}$ values of the reference point are presented in Table 2.

Table 2. Reference results of 1NN classifier obtained using instance selection methods without ensemble learning

Dataset	Algorithm	Accuracy	Compression
Sonar	ENN	81.28±8.59	16.83±1.29
	CNN	86.52±6.01	68.75±1.98
Ionosphere	ENN	84.89±3.63	14.28±0.51
	CNN	87.17±4.48	77.34±1.91
WBC	ENN	95.99±2.00	4.04±0.49
	CNN	91.41±3.38	86.95±0.80
Diabetes	ENN	74.60±3.88	26.28±0.55
	CNN	65.22±4.19	50.77±1.59
Heart Disease	ENN	81.87±5.31	18.38±0.91
	CNN	72.94±7.31	58.5±2.28
Vehicle	ENN	69.50±3.71	29.36±0.78
	CNN	66.90±4.84	50.22±1.08

During the first stage of the experiments the *feature ratio* representing the relative number (or percentage) of the features delivered to the input of the instance selection method was fixed and equal 0.8, and different values of *acceptance threshold* were being evaluated within the range $[0.1, 0.9]$. The obtained results are presented in Fig.3. In order to keep the figures simple and readable, values of the accuracy gain less than -10% were rejected.

In the second stage of the experiment, the *feature ratio* was being evaluated also in the range $[0.1, 0.9]$ keeping the *acceptance threshold* constant. However, in that experiment the *acceptance threshold* was chosen individually based on the results obtained in the first stage, taking the value of acceptance threshold, which maximizes the accuracy. The obtained results are presented in Fig. 3

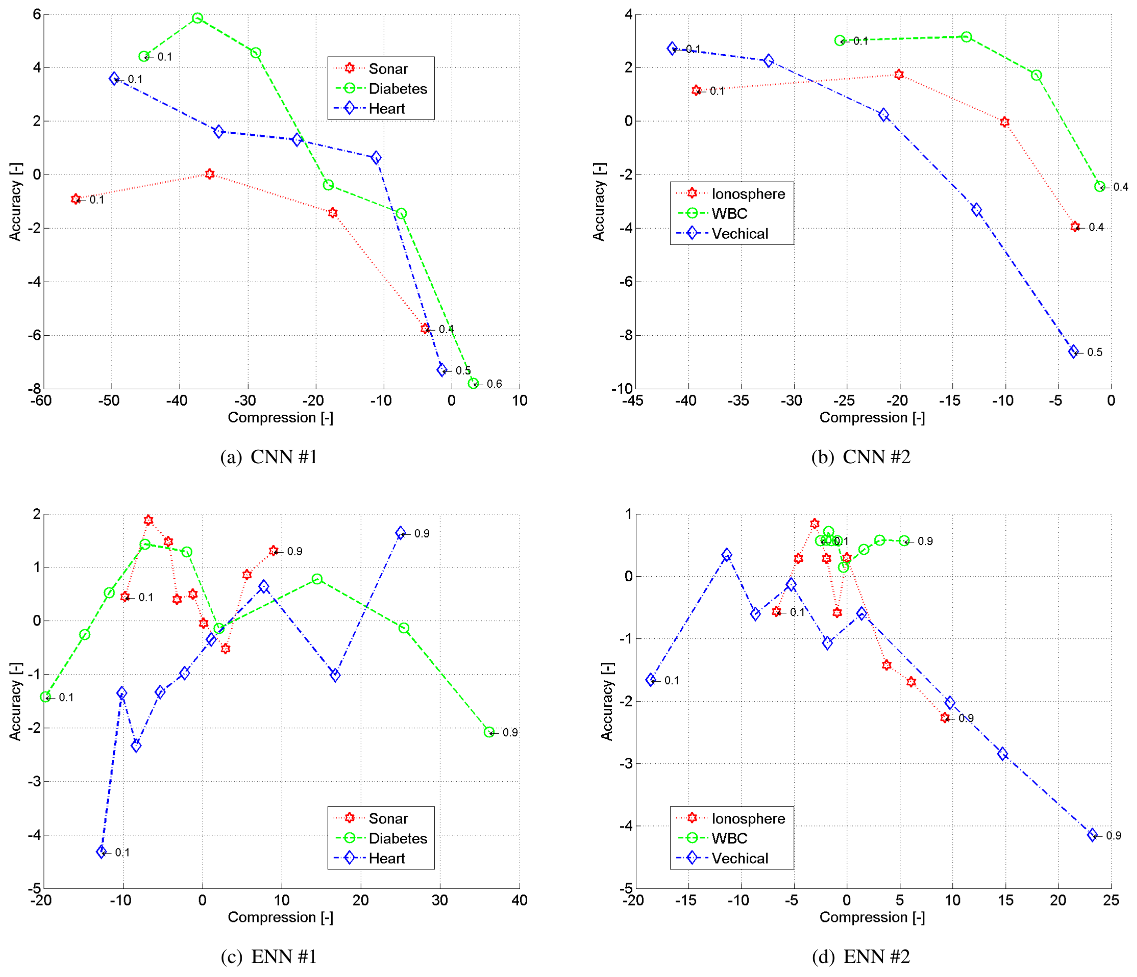
The obtained results show that the CNN algorithm is very sensitive for the threshold value. Choosing too high value increases the compression but dramatically reduces the accuracy making it drop even below 10% in comparison to the CNN used without ensemble. Only values close to $0.1, 0.2, 0.3$ are applicable, and for those values we were able to outperform both individual methods in terms of accuracy (even up to 6%). The drawback was reduction of the compression. This reduction was rather significant reaching the level of 40% also in comparison to the CNN without ensemble learning. This means that when using ensemble learning the accuracy of the *k*NN model trained on the dataset **P** can be improved but requires significant reduction of the compression. The algorithm behaves similarly when modifying *feature ratio*; the accuracy increases at the expanse of compression. Modification of the *feature ratio* also allowed to improve the accuracy, which is able to rise up to 3%.

A different situation was observed for the ENN algorithm. In that case the model ensemble allows for compression improvement, what is associated with the increase of the *acceptance threshold* parameter and corresponds to the mid values of *feature ratio*. For that parameters of the models we are able to improve the compression up to 40% without any loss of accuracy.

Unfortunately it is not possible to define a universal and optimal set of parameters. They must be chosen in each case independently, their wrong choice can cause significant deterioration in performance.

4. Conclusions

Nowadays new possibilities open for the use of instance selection methods, in particular for the analysis of large data sets. These types of applications have two objectives: to improve or maintain the accuracy of the prediction model created on the selected data and to achieve the compression as high as possible. The aim of this study was to analyze the possibility of using ensemble learning methods to improve the efficiency of instance selection methods. The empirical experiments were performed with one of the possible approaches, which was based on manipulating the feature subspace. The results indicate that it is possible to improve both the accuracy and compression. The tests include two base algorithms from two different families of instance selection approaches. Wrapping the ENN algorithm by the ensemble model allows both: improving the accuracy of the prediction model and increasing of



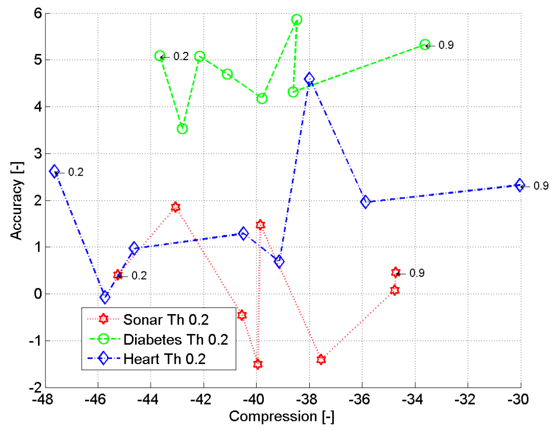
(a) CNN #1 (b) CNN #2 (c) ENN #1 (d) ENN #2

Fig. 2. Comparison of gain of the accuracy and compression of the ensemble of instance selection methods achieved by the feature manipulation for different values of *acceptance threshold*

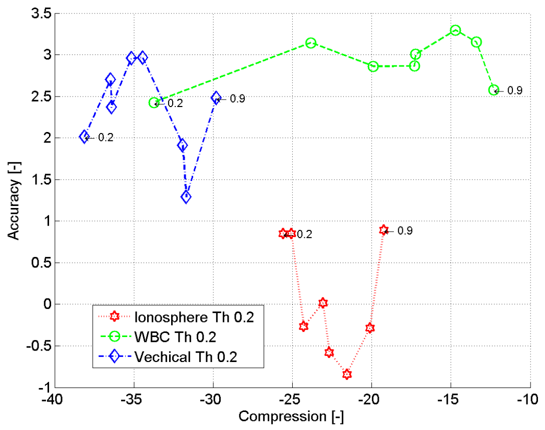
compression ratio. Especially good results were achieved in terms of compression. But this should not be surprising, since the ENN algorithm is designed to remove noise from the data, so when working individually usually does not reach the high values of compression.

In the case of the CNN algorithm, which is a condensation method, it is easy to observe a significant decrease of relative compression, while increasing the accuracy. This can be explained by the fact that one of the properties of the CNN algorithm is its high compression, which means that the selected subset of vectors is very small. That can be explained by the fact that only a few instances are usually required to determine the boundaries between different classes. These are the instances situated close to the boundaries. While the most of the instances are situated further from the decision boundaries and since they do not take an active role in classification, they get removed by CNN. In that situation, the probability of multiple selections of the same instance decreases, so it is very likely that in the following iterations of the ensemble model different instances will be selected, leading to the compression drop.

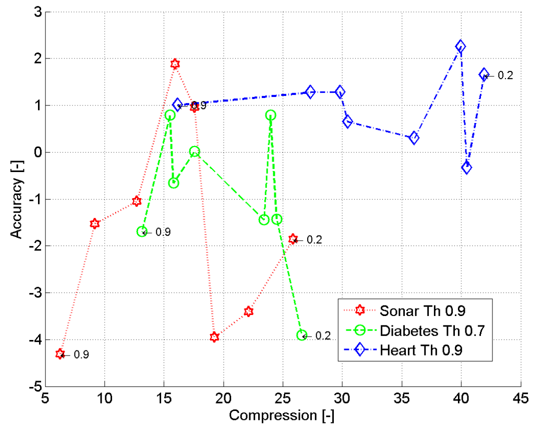
As we stated in the beginning, instance selection methods have great potential for mining large datasets. However, this requires reduction of the computational complexity of recent instance selection algorithms. The easiest way to achieve this is by reducing the number of samples, and the size of the feature space. An example application of the



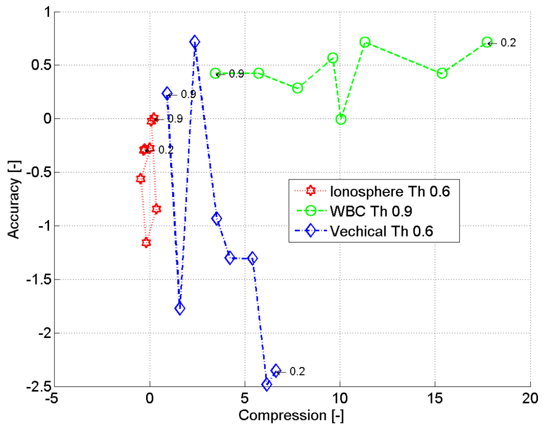
(a) CNN #1



(b) CNN #2



(c) ENN #1



(d) ENN #2

Fig. 3. Comparison of gain of the accuracy and compression with the ensemble of instance selection methods achieved by the feature manipulation with different values of *feature ration*

first approach have been shown in[12], and the feature space reduction is shown in this paper. So our further research will consider combination of the both approaches on a real world massive dataset problems.

References

1. P. Hart, The condensed nearest neighbor rule., IEEE Trans. on Information Theory 16 (1968) 515–516.
2. H. Liu, H. Motoda, Instance selection and construction for data mining, Vol. 608, Springer, 2001.
3. G. Salvador, D. Joaquin, C. Jose, Ramon, H. Francisco, Prototype selection for nearest neighbor classification: Taxonomy and empirical study, IEEE Transactions on Pattern Analysis and Machine Intelligence 34 (3) (2012) 417–435.
4. N. Jankowski, M. Grochowski, Comparison of instance selection algorithms. i. algorithms survey, LNCS 3070 (2004) 598–603.
5. M. Grochowski, N. Jankowski, Comparison of instance selection algorithms. ii. results and comments, LNCS 3070 (2004) 580–585.
6. M. Kordos, A. Rusiecki, Improving mlp neural network performance by noise reduction., Lecture Notes in Computer Science 8273 (2013) 133–144.
7. A. Rusiecki, M. Kordos, T. KamiĹ„ski, K. GreĹ„„, Training neural networks on noisy data, LNCS 8467 (2014) 131–142.
8. L. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, Wiley, 2004.

9. N. Garcia-Pedrajas, Constructing ensembles of classifiers by means of weighted instance selection., IEEE Transactions on Neural Networks 20 (2) (2009) 258–277.
10. Y. Caises, A. GonzĂˇlez, E. Leyva, R. PĂ©rez, Combining instance selection methods based on data characterization: An approach to increase their effectiveness, Information Sciences 181 (20) (2011) 4780–4798.
11. D. Wilson, Asymptotic properties of nearest neighbor rules using edited data, IEEE Trans. Systems, Man and Cybernetics 2 (1972) 408–421.
12. M. Blachnik, M. Kordos, Bagging of instance selection algorithms, LNCS 8468 (2014) 40–51.
13. M. Blachnik, M. Kordos, Instance selection in rapidminer, in: RapidMiner: Data Mining Use Cases and Business Analytics Applications, CRC Press, 2013.
14. A. Asuncion, D. Newman, UCI machine learning repository, http://www.ics.uci.edu/∼mlearn/MLRepository.html (2007).
URL `http://www.ics.uci.edu/\hbox{$\sim$}mlearn/MLRepository.html`