Rejecting Foreign Elements in Pattern Recognition Problem
Reinforced Training of Rejection Level

Wladyslaw Homenda[1,2] and Agnieszka Jastrzebska[1] and Witold Pedrycz[3,4]
[1]*Faculty of Mathematics and Information Science, Warsaw University of Technology*
ul. Koszykowa 75, 00-662 Warsaw, Poland
[2]*Faculty of Economics and Informatics in Vilnius, University of Bialystok*
Kalvariju G. 135, LT-08221 Vilnius, Lithuania
[3]*System Research Institute, Polish Academy of Sciences, ul. Newelska 6, 01-447 Warsaw, Poland*
[4]*Department of Electrical & Computer Engineering, University of Alberta, Edmonton T6R 2G7 AB Canada*

Keywords: Pattern Recognition, Rejection Option, Native and Foreign Elements, Support Vector Machines.

Abstract: Standard assumption of pattern recognition problem is that processed elements belong to recognized classes. However, in practice, we are often faced with elements presented to recognizers, which do not belong to such classes. For instance, paper-to-computer recognition technologies (e.g. character or music recognition technologies, both printed and handwritten) must cope with garbage elements produced at segmentation level. In this paper we distinguish between elements of desired classes and other ones. We call them *native* and *foreign* elements, respectively. The assumption that we have only native elements results in incorrect inclusion of foreign ones into desired classes. Since foreign elements are usually not known at the stage of recognizer construction, standard classification methods fail to eliminate them. In this paper we study construction of recognizers based on support vector machines and aimed on coping with foreign elements. Several tests are performed on real-world data.

1 INTRODUCTION

In the standard attempt to pattern recognition an object is classified to one of given classes. The set of classes is either fixed a priori (supervised problem) or is determined at the stage of recognizer construction (unsupervised problem). In these cases, it is assumed that each classified element belongs to one of the given classes. However, in practice, this assumption is often too optimistic. It is proven in important practical applications that not only elements of the fixed set of classes but also ones not belonging to these classes are processed. Since elements not belonging to any of the given classes are usually not known a priori, i.e. at the stage of recognizer construction, they cannot be assumed to create their own class(es) and cannot be used at this stage. To distinguish between these two types of elements the following terms are used:

- *native* elements for elements of recognized classes and
- *foreign* elements for ones not belonging to any given class.

Recognizing printed texts, manuscripts, music notations, biometric features, voice, speaker, recorded music, medical signals, images, etc. are examples of problems dealing with foreign elements, (Zhang, 2011; Weber, 1993). In recognition of printed texts, foreign elements (blots, grease, or damaged symbols) appear in a negligible scale due to regular placement of printed texts' elements (letters, numbers, punctuation marks) and due to their good separability. These features of printed texts allow employing effective segmentation methods and filtering foreign elements. However, in recognition of such sources as recorded voice, biometric features, medical images, geodetic maps or music notation, the foreign elements problem is more noticeable and important. Unlike printed text, such sources contain symbols placed irregularly and overlapping with each other. Such elements are hardly distinguishable by size and shape analysis. Due to weak separability of foreign and native elements of recognized sources, segmentation criteria must be more tolerant than in the case of printed texts in order not to reject native elements at the stage of segmentation. In consequence, more foreign elements

Homenda W., Jastrzebska A. and Pedrycz W..
Rejecting Foreign Elements in Pattern Recognition Problem - Reinforced Training of Rejection Level.
DOI: 10.5220/0005207900900099
In *Proceedings of the International Conference on Agents and Artificial Intelligence* (ICAART-2015), pages 90-99
ISBN: 978-989-758-074-1

are subjected to stages of recognition next to segmentation one and, in consequence, they should be eliminated then, (Homenda et al., 2013; Homenda et al., 2014).

Also, the problem of rejecting foreign elements (in pattern recognition tasks) is rarely present in research papers on pattern recognition. Alike, papers describing practical applications of pattern recognition methods ignore the problem of foreign elements, what may come from insufficient theoretical investigations on this subject and limited abilities of existing rejection methods. There are significant exceptions which show that the rejection problem cannot be disregarded, (Pillai et al., 2011; Stefano et al., 2000).

The motivation of this study arises from discussion on classification with rejection option. As outlined above, up–to–date research and practice still need conducting further studies on new aspects in the domain of pattern recognition. It is expected that research in this area will overcome technological barriers and will increase effectiveness in areas mentioned above.

In this paper we investigate the problem of rejecting foreign elements assuming that classes of native elements are given but the foreign ones are not known a priori. Sets of (semi-synthetic) foreign elements are constructed from real-world sets of native elements (handwritten digits). Then the following is accomplished:

1. a standard classifier for classes of native elements is constructed,
2. the standard classifier is used for classification of native elements (to estimate its accuracy),
3. in order to reject foreign elements from native classes, additional classifiers are used for every native class. These extra classifiers are constructed and used as follows:
   (a) no training set of foreign elements is used while constructing these extra classifiers and sets of semi-synthetic foreign elements are used as testing sets,
   (b) one set of semi-synthetic foreign elements is used as training set and both sets of semi-synthetic foreign elements are used as testing sets.
4. results of both methods of rejection are compared.

Semi-synthetic foreign elements, i.e. constructed from native elements (this is why we call them *semi*-synthetic), are used as a substitute of real foreign elements unknown at the stage of classifier's construction. Elements constructed in such a way are expected to have characteristics similar to real foreign elements, but obviously not exactly the same. Semi-synthetic foreign elements may help in increasing the quality of rejection when using them at the construction stage of rejecting classifiers as in point 3 (b) above.

The paper is structured as follows. Related research and introductory remarks are presented in Section 2. In Section 3, classifiers with rejection option are tested. The discussion includes evaluation criteria of rejection option. Results of experiments are discussed in Section 4. Conclusions and directions of further research are presented in Section 5.

2 PRELIMINARIES

In this Section the concepts and methodologies crucial for the paper are briefly described: Support Vector Machines (SVMs) as binary classifier, a methodology of constructing SVM-based multi-class classifiers, concepts of rejecting foreign elements and parameters used in evaluation of recognition with rejection.

2.1 Support Vector Machines (SVMs)

Here we briefly present two-classes and one class support vector machines as classification tools. Both concepts are then used as basic tools for rejecting foreign elements. Two-classes SVMs are also employed in construction of multi-class classifier.

2.1.1 Two-classes Support Vector Machines

Support Vector Machines (SVMs) in their pure form are non-probabilistic binary linear classifier models used in supervised machine learning in order to separate two sets of elements, (Cortes and Vapnik, 1995). In SVM's mathematical definition the two classes are denoted as integers -1 and 1. In this paper we assume that the considered sets of elements are points of the Euclidean space $\mathbb{R}^n$, which are also referred as vectors of this space. SVMs training corresponds to the problem of finding the maximum-margin hyperplane that divides the samples from two given classes, compare Figure 1. The following formula with the requirement of margin maximization describes the hyperplane:

$$\mathbf{w} \cdot \mathbf{x} - b = 0 \tag{1}$$

where $\mathbf{w}, \mathbf{x} \in \mathbb{R}^n$, $b \in \mathbb{R}$ and $\cdot$ is an operator of the scalar product of two vectors. Vector $\mathbf{w}$ is a normal vector to the hyperplane and it can be obtained as a linear combination of training vectors $\mathbf{x}_i$ lying at

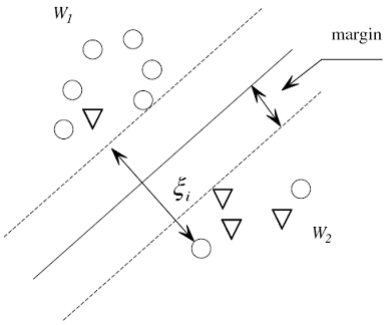

Figure 1: The concept of a linear SVM.

borders of the margin:

$$\mathbf{w} = \sum_i \alpha_i \mathbf{x}_i \tag{2}$$

Formally *support vectors* are such $\mathbf{x}_i$ that satisfy the following condition:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i - b) = 1 \tag{3}$$

Support vectors, i.e. those lying on the margin's border, have their corresponding $\alpha_i \neq 0$.

SVM learning algorithms finds $\mathbf{w}$ and b such that $||\mathbf{w}||^2$ is minimized subject to the following condition for all training vectors $\mathbf{x}_i$:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i - b) \geq 1 \tag{4}$$

where $y_i \in \{-1, 1\}$ is the class of $\mathbf{x}_i$.

Having in mind formula (2), the linear decision (classification) function $I(\mathbf{x})$ is expressed as follows:

$$I(\mathbf{x}) = \text{sgn}\left(\sum_i \alpha_i \mathbf{x_i} \cdot \mathbf{x} - b\right) \tag{5}$$

SVMs efficiency can be enhanced by using a so called kernel functions other than linear dot product so that they may be successfully applied to non-linearly-separable problems. The generalized decision function with a kernel K is written as:

$$I(\mathbf{x}) = \text{sgn}\left(\sum_i \alpha_i K(\mathbf{x_i}, \mathbf{x}) - b\right) \tag{6}$$

where polynomials and Gaussian functions are examples of kernels.

All the SVMs in this work use kernels that are based on a Gaussian radial basis function:

$$K(\mathbf{x_i}, \mathbf{x}) = \exp\left(-\gamma ||\mathbf{x_i} - \mathbf{x}||^2\right) \tag{7}$$

where γ is set to the inverse of the number of features used in classification as in (Homenda et al., 2014).

Yet, sometimes classes are not completely separable what leads to misclassification errors. The concept of allowing only partial separation is known as the soft margin. The C parameter, also called the regularization parameter, controls the shape of the margin by adjusting the penalty of misclassification. The

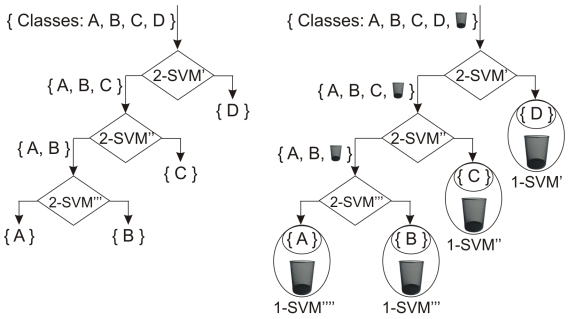

Figure 2: SVM based multi-class classifier without Rejection (left part) and with local rejection by one-class SVMs. Presented is the *cascade* architecture of the classifiers.

C parameter is a positive real number, its high values might cause over-fitting. In the soft margin case the optimization problem becomes a problem of minimizing the following expression:

$$\frac{1}{2}||\mathbf{w}||^2 + C\sum_i \xi_i \tag{8}$$

where ξ_i are the deviations (of wrongly classified vectors) and C is the regularization constant. The goal is to minimize the number of errors in the training set subject to the following ξ_i based prerequisites:

$$y_i(\mathbf{w} \cdot \mathbf{x_i} - b) \geq 1 - \xi_i \qquad \xi_i \geq 0 \tag{9}$$

2.1.2 One-class Support Vector Machines

One-class SVMs are a little different from regular two-classes SVMs as they use the ν parameter in place of the C parameter[1], (Schölkopf et al., 1999). The ν parameter has a similar role but its range is from 0 (exclusive) to 1 (inclusive) and is effectively the upper bound on the fraction of outliers in the training set, i.e. elements allowed to be misclassified. It is also a lower bound on the fraction of training samples used as support vectors. This time it is necessary to minimize the following expression:

$$\frac{1}{2}||\mathbf{w}||^2 + \frac{1}{\nu n}\sum_i \xi_i - b \tag{10}$$

where n is the number of samples present in the training set.

All the SVMs used in this work were implemented using machine learning library in Python, so called scikit-learn library, see (Pedregosa et al., 2011) for details. It internally uses LIBSVM, (Chang and Lin, 2011).

[1]There are also two-classes SVMs which use parameter ν instead of C and those are called ν-SVMs, (Schölkopf et al., 2000).

2.2 SVM-based Multi-class Classifiers

Since SVMs are able to separate only two distinct areas of an input space, i.e. two classes of presented elements, it is necessary to design ensembles of SVMs in order to cope with multi-class problems. There are several methods leading to SVM-based multi-class classifiers. In this Section we briefly describe the construction of SVM-based multi-class classifiers. A possible architecture of such SVM-based ensemble is indicated with no intention of a thorough discussion on this topic. On the other hand, expansion of the presented method on other architectures is rather straightforward, (Homenda et al., 2013).

As mentioned, since pure SVMs are two-class (binary) classifiers, SVM-based multi-class classifiers have to be designed to cope with more complex classification problems. One of the approaches is presented in the left part of Figure 2. The so called *cascade* architecture is implemented to deal with four classes. The classifier consists of three two-classes SVMs. Each SVM separates one class from the rest presented at its input.

There are several attempts to rejecting foreign elements, e.g. they might be identified prior to classification to native classes (global rejection), after classification to native classes (local rejection) or even at different stages of recognition process, (Homenda et al., 2013).

Models with local rejection can be built based on the no rejection classifier, as for instance the one described above. An example of classification with local rejection is presented in the right part of Figure 2. It essentially reuses the SVM-based classifier structure (without any change) and adds SVMs, which are responsible for rejecting foreign elements, see the right part of this Figure. Assuming that foreign elements are presented to no rejection model, they are (incorrectly) classified to native classes. Therefore, output of such classification (native classes with foreign elements incorrectly included) is processed by additional SVMs in order to reject foreign elements.

2.3 Spectral Clustering

Clustering (cluster analysis) is a category of methods in machine learning which are used to group similar samples together. Different clustering methods require different inputs and give different results. In our work we use *spectral clustering* to build the architecture of our classifier.

Spectral clustering is a clustering method that, as its input, takes an affinity matrix W, i.e. a weighted adjacency matrix of a similarity graph, and the desired number of clusters to create k, refer to (Shi and Malik, 2000) and (Ng et al., 2001). For clustering of n samples the affinity matrix has size (n,n). It is always symmetrical. One way to compute this matrix is to use a similarity function such as a Gaussian similarity function as in Equation (7) on pairs of distinct samples and put zeros on the diagonal.

The steps of the spectral clustering algorithm are as follows, refer to (von Luxburg, 2007). Given an affinity matrix W of the similarity graph and a clusters count k:

1. Compute the normalized Laplacian
$$L_{sym} = I - D^{-1/2} W D^{-1/2} \tag{11}$$
of the similarity graph, where I is the unit matrix and D is the diagonal degree matrix of the similarity graph.
2. Compute the k largest eigenvectors of L_{sym}, i.e. eigenvectors with the largest eigenvalues, $u_1, u_2, \ldots, u_k$ (orthogonal to each other, if possible, to have them maximally independent of each other).
3. Let $U \in R^{n \times k}$ be the matrix containing the vectors $u_1, u_2, \ldots, u_k$ as columns in this order.
4. Form the matrix $T \in R^{n \times k}$ from U by normalizing the rows to Euclidean norm, i.e. dividing each value in a row by the value of this norm of the row.
5. Treating each row of T as a point in R^k (in fact $[-1,1]^k$), cluster them into k clusters using K-means or any other algorithm (e.g. the optimal discretization).

In our work we use the optimal discretization algorithm which is described in (Yu and Shi, 2003). We did not include its definition here due to space constraints and the fact that its definition includes several complex mathematical concepts that would need to be described as well.

Scikit-learn library, refer to (Pedregosa et al., 2011), provides the implementation of the spectral clustering algorithm.

2.4 Evaluations

In this subsection, factors and measures are defined in order to provide ways to reliably evaluate the quality of recognition with rejection.

Native elements are all the elements that are considered to be included in designed classes and need to be assigned a corresponding class (i.e. the class they present). *Foreign* elements are everything else — the garbage. We would like our classifiers to be able to:

- identify all native elements as native (i.e. *accept* them) and all foreign ones as foreign ones (i.e. *reject* them),
- classify native elements to respective classes.

Quality evaluation of classification with rejection requires non-standard measures. Intuitively, it is important to measure how exact rejection procedure, i.e. how many foreign elements are accepted and, vice versa, how many native elements are rejected. Of course, measuring classification's quality understood as assigning native elements to proper classes is still of great importance, (Homenda et al., 2014).

For better understanding of how quality of classification with rejection should be measured we adopt parameters and quality measures used in signal detection theory. Since these parameters are widely utilized, we do not refer to original sources here. The following parameters were used in defining several factors, which outline classification's quality:

- TP (True Positives) – the number of native elements classified as native elements (no matter, if classified to correct class, or not),
- FN (False Negatives) – the number of native elements incorrectly classified as foreign ones,
- FP (False Positives) – the number of foreign elements incorrectly classified as native ones,
- TN (True Negatives) – the number of foreign elements correctly classified as foreign ones.
- CC (Correctly Classified) – the number of correctly classified elements, i.e. foreign elements classified as foreign and native elements classified as native with the correct class

These notions can be used to construct the following seven characteristics:

$$\text{Strict Accuracy} = \frac{CC}{TP+FN+FP+TN}$$

$$\text{Accuracy} = \frac{TP+TN}{TP+FN+FP+TN}$$

$$\text{Native Precision} = \frac{TP}{TP+FP}$$

$$\text{Foreign Precision} = \frac{TN}{TN+FN}$$

$$\text{Native Sensitivity} = \frac{TP}{TP+FN}$$

$$\text{Foreign Sensitivity} = \frac{TN}{TN+FP}$$

$$\text{F–measure} = 2 \cdot \frac{\text{Precision} \cdot \text{Sensitivity}}{\text{Precision} + \text{Sensitivity}}$$

These characteristics were described in more detail in (Homenda et al., 2014). Strict accuracy was added to have a characteristic corresponding to accuracy of a model without rejection. The following relation holds: Strict Accuracy $\leqslant$ Accuracy and in the case of only one class of native elements it is obviously turned into equality. The following comments help to understand emaning of characteristics:

- *Strict accuracy* is the absolute measure of the classifier's performance. It is the ratio of the number of all *correctly* classified elements, i.e. foreign classified as foreign (rejected), and native classified to their respective classes, to the number of all elements being classified.
- *Accuracy* is a characteristic derived from strict accuracy by ignoring the need to classify native elements to their respective classes: in other words, it is sufficient to correctly identify whether elements are native or foreign. This measure describes the ability to distinguish between native and foreign elements. Of course, the higher the value of this measure, the better the identification.
- *Native Precision* is the ratio of the number of elements *correctly* classified as native to the number of all ones classified as native. Native Precision evaluates the ability of the classifier to distinguish foreign elements from native ones. The higher the value of this measure, the better ability to distinguish foreign elements from native ones. Native Precision does not evaluate how effective identification of native elements is.
- *Foreign Precision* corresponds to Native Precision. It is the ratio of the number of elements *correctly* classified as foreign ones to the number of all ones classified as foreign.
- *Native Sensitivity* is the ratio of the number of elements *correctly* classified as native to the number of all that *should be* classified as native, i.e. all that are in fact native ones. This measure evaluates the ability of the classifier to identify native elements. The higher the value of Native Sensitivity, the more effective identification of native elements. Unlike the Native precision, this measure does not evaluate the effectiveness of separation between native and foreign elements.
- *Foreign Sensitivity* corresponds to Native Sensitivity. It is the ratio of the number of elements *correctly* classified as foreign ones to the number of all that *should be* classified as native ones, i.e. all that are in fact native ones.
- Precision and Sensitivity are complementary and there exists yet another characteristic that combines them: the *F–measure*. It is there to express the balance between precision and sensitivity since, in practice, these two affect each other.

Increasing sensitivity can cause a drop in precision since, along with correctly classified elements, there might be more incorrectly classified.

It is the goal of classification with rejection to maximize all of the above measures. However, it is not that simple since, in practice, increase in one can lead to decrease in another. In practice, depending on real application, some measures might be more important than others. For instance, if importance is given to minimizing the number of foreign elements identified as native, then Native Precision gets higher importance than other measures and it should be maximized. On the other hand, if the highest priority is given to minimize loss of native elements, then focus should be on Native Sensitivity, etc.

3 EXPERIMENT

In this Section the datasets and classifiers architectures used in this paper's experiment are introduced and briefly described. The datasets were constructed based on the MNIST database of handwritten digits, (LeCun et al., 1996). This database was directly used in construction of the classifier without rejection. Then, based on this database, two sets of foreign elements were constructed and used for (reinforcement) training of rejecting SVMs attached at local level to the recognizer without rejection.

3.1 Datasets

The complete data consists of 10000 samples with 106 features. This data was split into two sets with ratio approximately equal 7:3 - the training set with 6999 samples and the test set with 3001 samples (numbers of samples from each class are not equal but approximately the same). These samples represent handwritten decimal digits (i.e. from 0 to 9) and their classes correspond directly to represented digits. All 106 features were used for classification. Their values were scaled to the standard normal distribution in order to avoid dominance of features with greater values.

These features were obtained by computing them from binary (black and white) images. The following vector features were computed:

- projections (vertical and horizontal) — for each column (row) black pixels in that column (row) were counted
- histograms of projections — for each possible amount of black pixels, columns (rows) in which exactly this amount of black pixels is present were counted
- transitions (vertical and horizontal) — for each column (row) color changes from black to white were counted (i.e. how many times a black pixel precedes a white one)
- offsets (from left and right) — for each row the distance between image's border and the first black pixel in that row (counting from left or right respectively) was computed
- differentials of the above

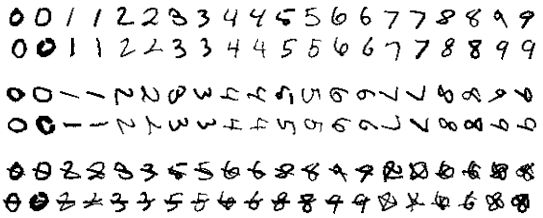

Figure 3: Elements of native classes (upper part) and foreign elements of set A (middle part) and set B (bottom part).

And from these vectors the following scalar features were extracted:

- minimum value and its position
- maximum value and its position
- mean
- first absolute moment
- peaks count

Apart from the scalars obtained from vector features, the following scalar features were computed directly from the matrix representations of images:

- for each of 4 directions $(0°, 45°, 90°, 135°)$ the length of the longest black segment in that direction. Notice that such segments were found for each pixel of the image and, furthermore, a given segment of black pixels is the common direction for all its pixels
- first raw moments
- height to width ratio

The rejection models were tested with two sets of foreign elements. The set of all 10000 native elements is their common core:

- the first set (A) of (semi-synthetic) foreign elements includes rotated native elements,
- the second set (B) of (semi-synthetic) foreign elements includes chosen native elements overlapping chosen rotated ones.

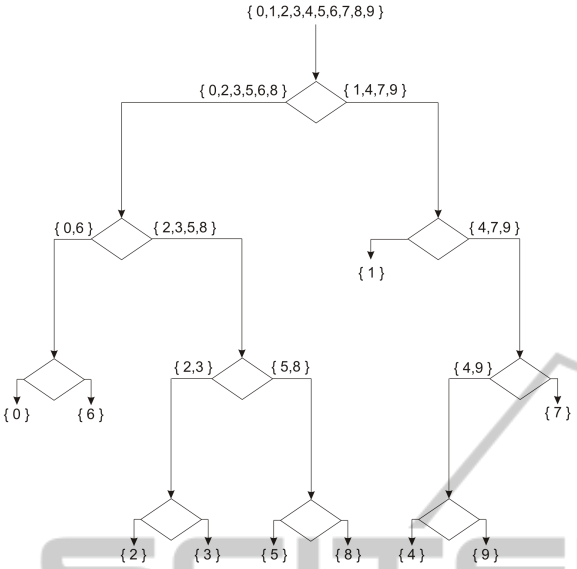

Figure 4: The basic architecture of a SVM-based classifier for native elements without rejection.

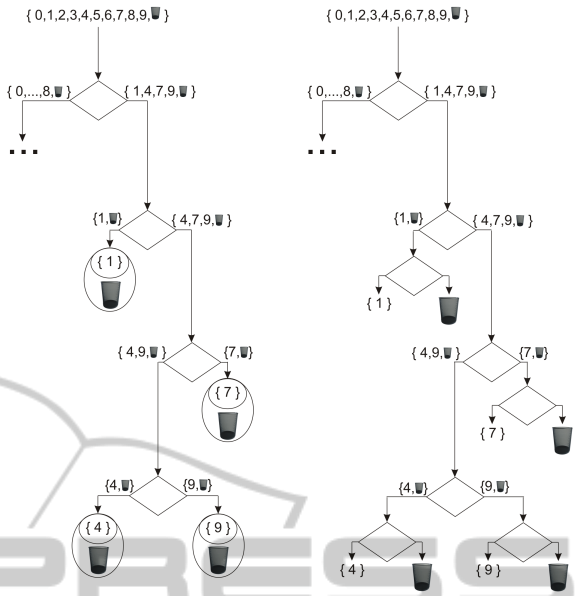

Figure 5: SVM-based classifiers with rejection based on the classifier without rejection. The classifier presented in Figure 4 is enhanced by applying one-class SVMs (left part) and two-classes SVMs (right part) to reject foreign elements.

For a preview see Figure 3.

Notice that both sets of foreign elements are created from native ones, because we do not have real foreign elements. Such elements are called semi-synthetic since they are still handwritten symbols which might be similar to real ones, on one hand, but, on the other hand, they were artificially created.

The complete set of 10000 digits was processed as shown in Figure 3, i.e. digit's images were converted to 1 bit per pixel, black ink on white paper, and then rotated and overlapped.

3.2 Classifiers Architecture

A quasi-balanced tree obtained by using spectral clustering was used as a core of classifier's architecture.

First, average values of features were computed for each of 10 classes using the training dataset. These 10 vectors were used to build an affinity matrix using Gaussian similarity function and used as input to spectral clustering algorithm along with the number of clusters equal to 2. This process resulted in classes being grouped into two separate clusters. The first cluster contained classes: $0,2,3,5,6,8$, and the second one: $1,4,7,9$. The process was repeated for newly obtained sets of classes until they have at least two classes.

The final tree as was generated by the spectral clustering is presented in Figure 4.

The tested classifiers are based on this tree. There are essentially three classifier models:

1. the no rejection model constructed at the basis of the training/testing sets of native elements. Its architecture is presented in Figure 4. Internal nodes represent SVMs, however, these can be replaced by any classifier capable of at least two-class output (e.g. decision trees, randomized forests, etc.),
2. the model with local rejection based on one-class Support Vector Machines (1-SVMs), cf. Figure 5,
3. the model with local rejection based on two-classes Support Vector Machines (2-SVMs), cf. Figure 5.

The tree model can effectively be replaced by any classifier architecture capable of multi-class output (e.g. decision trees, randomized forests, etc.).

We also considered a global rejection model with one-class SVMs instead of doing local rejection. Namely, rejection of foreign elements was applied first and then non-rejected elements were subjected to classification to native classes instead of classifying all elements to native classes first and then rejecting foreign elements from every native class using specifically trained SVMs. However, it turned out that one-class SVM based global rejection was less effective than local rejection, i.e. Foreign Sensitivity was significantly lower in such a case.

It is worth to underline, that investigation on rejecting foreign elements was the main aim of this study rather than achieving outrageous recognition rates of native elements. Therefore, standard library functions implementing considered methods

Table 1: Accuracy matrix for native classes for the classifier shown in Figure 4. Values stand for percents. For the sake of clarity, zeros are not printed.

	0	1	2	3	4	5	6	7	8	9
0	99.29		0.20		0.10		0.20		0.10	0.10
1		99.03	0.26	0.09			0.35	0.26		
2	0.19		95.45	1.74	0.39	1.07	0.29	0.19	0.39	0.29
3		0.10	0.69	95.74		1.29		0.59	0.79	0.79
4			1.32	0.10	95.72	0.31	0.10	0.20	0.20	2.04
5			1.68	2.69		93.83	0.34	0.22	0.78	0.45
6			0.21		0.10	0.42	98.54		0.73	
7		0.10	0.19	0.49	0.58	0.39		94.94		3.31
8	0.41		0.92	1.13	0.21	1.13	0.21		95.89	0.10
9	0.10		0.20	0.69	1.59	0.50		1.29	0.40	95.24

were used without any effort put into increasing efficiency/quality of classifiers. Of course, detailed researches on the problem of recognition with rejecting foreign elements as well as practical application of such researches require more effort put on quality of every stage of recognition, but this is out of the scope of this study.

4 RESULTS

In this Section we present the results of the described experiment. The classifiers are built and evaluated and their characteristics presented and commented on.

4.1 The no Rejection Model

The first model is build upon the balanced tree in the following way: all internal nodes are assigned a two-class SVM which is trained to separate samples of classes from the set in the left child from samples of classes from the set in the right child. There are no classifiers rejecting foreign elements in the leaves of the tree and it means that this SVM-based classifier does not reject any sample, i.e. it to classifies all samples, native and foreign, to one of the 10 classes.

This classifier was used to estimate the right common regularization (C) parameter for all two-classes SVMs. The method utilized for estimation of the optimal value of the regularization parameter was a 2-base logarithm space search from 2^{-5} to 2^{5}, i.e. all the integer powers of 2 in these interval were tried. The chosen parameter was the one with the highest score which was calculated as the percent of properly predicted classes, i.e. with the highest value of Strict Accuracy measure. Of course, in this case the factors FP and TN vanish. To improve results a stratified 3-fold cross validation was used instead of a simple self-validation in the training set.

The best estimated value for C was 8 and it was used in all further computations.

The final scores of the Strict Accuracy measure that this model achieves with the chosen parameters are 99.07% at training set, 90.20% at testing set and 96.41% at both sets joined.

The accuracy matrix for the complete set of native classes is shown in Table 1. Rows represent real classes and columns the predicted ones. All the values are percentages. For instance, the row labeled 1 informs that 99.03% of 1's is correctly classified, 0.26% of 1's is classified as 2's, 0.09% of 1's is classified as 3's etc. Taking the column labeled 1 we have 99.03% of 1's correctly classified, 0.10% of 3's classified as 1's and 0.10% of 7's classified as 1's. Percentages in rows sum up to 100% ($\pm$0.01% due to rounding error). Percentages at the diagonal define the accuracy of the classifier for each class separately.

We also compared our architecture to a popular multi-class output architecture for SVMs — the one-vs-one architecture as implemented in LIBSVM. The results are comparable (96.96% at both sets joined). However, our architecture is computationally more efficient since it builds less SVMs (in this case: 9 in our architecture vs 45 in the one-vs-one architecture).

4.2 The Local Rejection Model with One-class SVMs

Employing one-class SVM for each native class is the simplest method for rejecting foreign elements, compare left part of Figure 5. The question arises for an optimal value of the parameter ν. This is quite in-

Table 2: Results for SVM-based classifiers with rejection.

rejection by	one-class SVM				two-classes SVM			
details	$\nu = 0.01$		$\nu = 0.22$		train set A		train set B	
results on set	A	B	A	B	A	B	A	B
Strict Accuracy	90.17	68.30	85.69	74.06	97.97	67.03	86.78	97.64
Accuracy	91.78	69.84	86.97	75.28	99.65	68.64	88.39	99.18
Native Precision	89.24	62.04	95.31	72.50	99.99	60.46	82.04	99.97
Foreign Precision	94.67	91.13	81.22	78.21	99.31	98.47	97.89	98.48
Native Sensitivity	95.02	95.02	77.76	77.76	99.31	99.31	98.31	98.31
Foreign Sensitivity	88.54	46.81	96.17	73.02	99.99	40.59	78.48	99.97
Native F-measure	92.04	75.06	85.64	75.04	99.65	75.16	89.44	99.13
Foreign F-measure	91.50	61.85	88.06	75.53	99.65	57.49	87.12	99.22

tuitive that the higher its value, the better acceptance of native elements and the worse rejection of foreign elements. Since real foreign elements are not known at the stage of classifier's construction, semi-synthetic ones, as described in Section 3.1, were used for both training and testing purposes.

The results were first computed for $\nu = 0.01$ and are high for set A. Rotated digits differ enough from non-rotated ones to be easily rejected. In the case of set B, there is a substantial drop in both native precision and foreign sensitivity (as well as a small drop in foreign precision). This means that in the space of our 106 features these prepared symbols are quite similar to digits and one-class SVMs accept them more often than it is in the case with the rotated digits.

The question arises if results can be improved by modifying parameter ν. A simple linear search with step equal to 0.01 reveals $\nu = 0.22$ as the best value for set B in terms of accuracy. However, better results for set A could not be achieved with this step.

With $\nu = 0.22$ foreign sensitivity got improved in both cases but native one dropped considerably. It means that the classifier was able to reject more foreign elements but also rejected more native elements (which is not what we want). Furthermore, it resulted in a drop in foreign precision since more native elements were incorrectly rejected. Overall, better accuracy with set B but worse with set A were achieved.

4.3 The Local Rejection Model with Two-classes SVMs

In this model one-class SVMs are replaced by two-classes SVMs, cf. Figure 5. In case of one-class SVMs it was straightforward how to train them — use data for particular class only. In case of two-classes SVMs, however, there needs to be an additional anti-class which acts as an explicit foreign elements class. This model was built against two separate anti-classes:

1. Train set A: The class of rotated digits.
2. Train set B: The class of rotated digits printed over normal digits.

The sensitivity of foreign elements is high in set A when it is trained against itself. This is the expected result since that is the goal of training a two-class SVM. However, it is considerably lower in set B, lower even than the lowest one obtained with one-class SVMs.

When training with set B the sensitivity of foreign elements in that set improved but got worse in set A due to different anti-class. There is also a small drop in native sensitivity. That might be due to higher similarity between elements in set B and native elements.

In both cases, however, the native sensitivity is very high, higher than with any of the presented one-class SVMs-based models. This is the main result of abandoning one-class SVMs — native elements are more likely to be accepted.

4.4 Training with Other Anti-classes

There was an idea to train these rejecting two-classes SVMs using foreign samples with a random number of randomly distributed black pixels, using both uniform and normal probability distributions. However, in performed experiment, it turned out that such samples are not a good anti-class. It occurred that such samples are so much different comparing to native samples (handwritten digits), that they were perfectly identified as foreigners, with no mistakes! Therefore, this idea was abandoned and the results are not included.

5 CONCLUSIONS

In this paper we presented two ways in which standard classifiers can be enhanced to reject foreign elements at local level. Namely, first, all elements, native and foreign, subjected to recognition were classified to native classes. Then, for every native class a SVM dedicated to the identification of foreign elements was applied. While one-class SVMs are the more straightforward way to do the rejection, two-classes SVMs gave similarly good results keeping Native Sensitivity high.

There are several interesting research directions related to recognition with rejection. An interesting topic is a possibility to generate such random samples that would improve quality measures in so called reinforced learning of rejecting elements. Another interesting topic is related to the architecture of classifiers. Architectures with rejecting at different levels of classification would be more effective than ones with rejection at local level only. Another topic is related to re-classification of rejected elements. This topic is especially important for classification with low Foreign Sensitivity. Alike, other measures would be considered in construction of classifiers and the re-classification process.

ACKNOWLEDGEMENTS

The research is partially supported by the Foundation for Polish Science under International PhD Projects in Intelligent Computing. Project financed from The European Union within the Innovative Economy Operational Programme (2007- 2013) and European Regional Development Fund.

The research is supported by the National Science Center, grant No 2012/07/B/ST6/01501, decision no UMO-2012/07/B/ST6/01501.

REFERENCES

Chang, C. C. and Lin, C. J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):1–27.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

Homenda, W., Luckner, M., and Pedrycz, W. (2013). Classification with rejection: concepts and formal evaluations. In *Proc. 8th Int. Conf. on KICSS*, pages 161–172.

Homenda, W., Luckner, M., and Pedrycz, W. (2014). Classification with rejection based on various SVM techniques. In *Proceedings of the WCCI 2014 IEEE World Congress on Computational Intelligence*, pages 3480–3487.

LeCun, Y., Cortes, C., and Burges, C. (1996). The mnist database of handwritten diggits. In *http://yann.lecun.com/exdb/mnist/*.

Ng, A. Y., Jordan, M. I., and Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, volume 14, pages 849–856.

Pedregosa, F. et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pillai, I., Fumera, G., and Roli, F. (2011). A classification approach with a reject option for multi-label problems. In *Proc. 16th Int. Conf. on Image Analysis and Processing*, volume 6978, pages 98–107.

Schölkopf, B., Smola, A. J., Williamson, R. C., and Bartlett, P. L. (2000). New support vector algorithms. *Neural Computation*, 12(5):1207–1245.

Schölkopf, B., Williamson, R. C., Smola, A. J., Shawe-Taylor, J., and Platt, J. C. (1999). Support vector method for novelty detection. In *Proc. Advances in Neural Information Processing Systems*, volume 12, pages 582–588.

Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905.

Stefano, C., Sansone, C., and Vento, M. (2000). To reject or not to reject: that is the question — an answer in case of neural classifiers. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 30(1):84–94.

von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416.

Weber, D.M.; Du Preez, J. (1993). A comparison between hidden markov models and vector quantization for speech independent speaker recognition. In *Proc. South African Symposium on Comm. and Signal Processing*, pages 139–144.

Yu, S. X. and Shi, J. (2003). Multiclass spectral clustering. In *Proc. 9th Int. Conf. on Comp. Vision*, volume 1, pages 313–319.

Zhang, B. (2011). Breast cancer diagnosis from biopsy images by serial fusion of random subspace ensembles. In *Proc. 4th Int. Conf. on BMEI*, pages 180–186.