Pattern Recognition Letters 29 (2008) 1565–1570

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Growing a multi-class classifier with a reject option

D.M.J. Tax *, R.P.W. Duin

Information and Communication Theory Group, Mekelweg 4, 2628 CD Delft, The Netherlands

ARTICLE INFO

Article history:
Received 11 October 2007
Received in revised form 30 January 2008
Available online 30 March 2008

Communicated by F. Roli

Keywords:
Multi-class classification
Outlier detection
Rejection

ABSTRACT

In many classification problems objects should be rejected when the confidence in their classification is too low. An example is a face recognition problem where the faces of a selected group of people have to be classified, but where all other faces and non-faces should be rejected. These problems are typically solved by estimating the class densities and assigning an object to the class with the highest posterior probability. The total probability density is thresholded to detect the outliers. Unfortunately, this procedure does not easily allow for class-dependent thresholds, or for class models that are not based on probability densities but on distances. In this paper we propose a new heuristic to combine any type of one-class models for solving the multi-class classification problem with outlier rejection. It normalizes the average model output per class, instead of the more common non-linear transformation of the distances. It creates the possibility to adjust the rejection threshold per class, and also to combine class models that are not (all) based on probability densities and to add class models without affecting the boundaries of existing models. Experiments show that for several classification problems using class-specific models significantly improves the performance.

1. Introduction

In standard problems one has to classify an object and assign it to one of a set of known classes. In practice one also has to reject the objects that do not fit to any of the classes (Dubuisson et al., 1985). In these applications some classes may be known, but novel classes can appear and these are unknown. In a face recognition problem a model for each person in the training set has to be trained. The system should recognize a novel person and it should not assign this outlier person to one of the known persons. Furthermore, an extra practical demand is that the system should be easily extendible to include new persons, and it should be simple to remove known persons. Moreover, such an extension should not affect the decision boundaries between existing models. These types of demands are not only typical for face recognition (Kang and Choi, 2006), but also for the classification of crops, industrial products, disease detection in medical imaging etc.

The standard approach to rejection in pattern recognition is to estimate the class conditional probabilities, and to reject the most unreliable objects, that is, the objects that have the lowest class posterior probabilities. This is called the *ambiguity reject* (Chows rule, Chow, 1970). This reject rule is optimal when the posterior probabilities are estimated without error. In the case of estimation errors, it was recognized in Fumera et al., 2000 that a per-class threshold may be required.

Furthermore, using the class posterior probability for rejection ignores the possibility of having objects from unknown classes. These objects do not typically appear in areas with a low posterior probability (i.e. in areas between the known classes), but they are often distributed around the known classes, where the total data probability density is low, but the posteriors are high. In Dubuisson and Masson, 1993 the ambiguity reject was extended to the *distance reject* in which objects are rejected for which the full data density is below a threshold. This can be seen as outlier detection, or novelty detection, and numerous other outlier detection algorithms in a wide range of scientific fields have been proposed (Davies and Gather, 1993; Japkowicz et al., 1995; Tarassenko et al., 1995; Cerioli and Riani, 1999; Baker et al., 1999; Pan et al., 2000; Ramaswamy et al., 2000; Tax and Duin, 2001; Marsland, 2001).

It appears that some of these outlier detection methods do not rely on a probability density estimate. To estimate a probability density requires a large amount of training data, and when the feature space is large in comparison to the training set size, density estimators suffer from the curse of dimensionality (Duda et al., 2001). It is therefore often better to avoid an explicit density estimation and to use an approximate model. Unfortunately, this makes the combination of the models to a multi-class classifier more complex, in particular when one wants to do more than simple voting. Confusion often occurs in situations where objects are accepted by more than one model. Unfortunately, in many cases it is just two models, and in this situation voting is not applicable.

* Corresponding author. Tel.: +31 (0) 15 27 88434; fax: +31 (0) 15 27 81843.
E-mail addresses: D.M.J.Tax@tudelft.nl (D.M.J. Tax), R.Duin@ieee.org (R.P.W. Duin).

0167-8655/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2008.03.010

For these situations the soft outputs of the models have to be compared. But because each model may have a different way to measure the similarity of an object to its class, the output values of the different class models have to be normalized.

In this paper we investigate and compare two rescaling heuristics for one-class models. Both heuristics are constructed such that the decision boundaries between the classes and the outliers are not affected. The first scaling makes use of the assumption that all class models give a fixed output for outlier objects, and for that it requires a non-linear scaling of distances. The second scaling assumes that the average output for a class is constant, and assumes that classes are relatively well sampled. In Section 2 we start with discussing the standard approach of rejecting objects. Next, the combination of models and the required normalization are presented in Section 3. In Section 4 the experimental evaluation is done and the paper finishes with conclusions in Section 5.

2. Multi-class classifiers with reject and class models

Assume we are given objects $\boldsymbol{x}$ from c classes $\omega_1, \ldots, \omega_c$, with prior probabilities $p(\omega_i)$. All objects are represented by p-dimensional feature vectors in a bounded area in the feature space: $\boldsymbol{x} \in \mathscr{D} \subset \mathbb{R}^p$. A training set $\mathscr{X}_i^{tr} = \{\boldsymbol{x}_j, j = 1, .., n_i\}$ is available for each of the classes ω_i. The standard pattern recognition approach to classification is to estimate the class conditional probabilities $p(\boldsymbol{x}|\omega_i)$, $i = 1,\ldots,c$. By applying Bayes rule the posterior probabilities $p(\omega_i|\boldsymbol{x})$ can be computed using the class conditional probabilities and the class priors:

$$p(\omega_i|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\omega_i)p(\omega_i)}{\sum_{j=1}^{c} p(\boldsymbol{x}|\omega_j)p(\omega_j)} = \frac{p(\boldsymbol{x}|\omega_i)p(\omega_i)}{p(\boldsymbol{x})}. \quad (1)$$

In the standard rejection approach, the ambiguity reject (Chow, 1970), the objects $\boldsymbol{x}$ are rejected for which the maximum posterior probability $\max_i p(\omega_i|\boldsymbol{x})$ is below a threshold.

In real applications objects from other, novel classes may appear. This situation can be modeled by an extra reject (or outlier) class ω_0 that has a uniform distribution in the area $\mathscr{D}$. To distinguish this outlier class from the c known classes, one can put a threshold on the total data density of the known classes (Bishop, 2006). The total classifier with reject therefore becomes:

$$\hat{y} = \begin{cases} \omega_0 & p(\boldsymbol{x}) \leqslant \theta, \\ \omega_i & p(\omega_i|\boldsymbol{x}) > p(\omega_j|\boldsymbol{x}), \quad i \neq j \quad \text{and} \quad p(\boldsymbol{x}) > \theta. \end{cases} \quad (2)$$

This approach is suitable when a sufficiently large training sample is available for all of the classes and when the training sample is not contaminated by outliers. In this case $p(\boldsymbol{x}|\omega_i)$ can be estimated reliably by some model $\hat{p}(\boldsymbol{x}|\omega_i)$. A first problem may be that the different classes in the training data may be contaminated by different amounts of outliers. In that case a different rejection threshold θ_i per class (Fumera et al., 2000) has to be used. In this case a *density-based* one-class model for class ω_i is obtained:

$$\hat{y} = \begin{cases} \omega_0 & \hat{p}(\boldsymbol{x}|\omega_i) < \theta_i \\ \omega_i & \text{otherwise}. \end{cases} \quad (3)$$

Even when we know the class priors and we are using proper density models (as in (3)), we cannot just use the standard Bayes rule (Eq. (1)) for finding the most probably class. The Bayes rule does not incorporate the model thresholds θ_i, and these thresholds may vary significantly in value, especially when classes have a large spread. For classes with a large spread, the probability density values tend to be low, because the probability densities are normalized to integrate to one. On the other hand, classes that are very compact will have a much higher probability densities. When a single rejection threshold is chosen in the standard Bayes rule, most of the rejected objects will therefore come from the class with the highest spread, while (almost) none of the objects from a very compact class is rejected.

A second problem is that to apply Bayes rule (1) a good estimate of the class priors $p(\omega_i)$ should be available. When the training set reflects well what can be expected in the practical application, these priors can be simply obtained. For situations that new classes may appear, for instance because new types of diseases or new types of defects may appear in the objects that should be classified, it may be hard to find these priors.

A third significant problem for formulation (2) is that density estimation is a hard problem. For high dimensional feature spaces many training objects are required to avoid the curse of dimensionality (Duda et al., 2001). When a limited training set is available, approximations to the class densities have to be made, like k-means clustering centers, self-organizing maps, subspace models using PCA or hypersphere models inspired by the support vector machines (Tax, 2001). These methods often use a distance to prototypes or subspaces, and are therefore called *distance-based* class models. In contrast to the density models, the distance-based class models give a high output to the outliers:

$$\hat{y} = \begin{cases} \omega_0 & d_i(\boldsymbol{x}) > \theta_i \\ \omega_i & \text{otherwise}, \end{cases} \quad (4)$$

where $d_i(\boldsymbol{x})$ is the distance of object $\boldsymbol{x}$ to the model of class ω_i.

Although these distance-based class models (4) may describe a class better, they lack a common output scaling and it is not clear how they can be compared and combined. A similar problem appears when density-based models and distance-based models are combined, because one cannot directly compare (3) with (4); the first one increases while the second one decreases when one approaches a class.

To generalize formulation (2) to both density and distance-based class models, a normalization has to be defined. We define this normalization with two demands. The first demand is that the normalized output for a class is high for objects that come from that class. The second demand is that the decision boundaries of the different models between the outlier objects and their corresponding class objects are not changed.

Because each model characterizes the same outlier class with their threshold θ_i, these thresholds should coincide. On the other hand, each model characterizes a different 'target' class, and therefore these class outputs have to be compared to find the most probable output class. The exact construction of the normalization is explained in the next section.

3. Combination of class models

We propose to use the following transformation for the normalization the outputs of models (3) and (4). For the density-based models we chose to use a simple linear rescaling, for the distance-based models we have a possibly nonlinear transformation g:

$$\tilde{p}_i(\boldsymbol{x}) = \begin{cases} \frac{1}{Z_p}\hat{p}(\boldsymbol{x}|\omega_i) + p_0 & \text{density-based models}, \\ \frac{1}{Z_d}g(d_i(\boldsymbol{x})) + d_0 & \text{distance-based models}, \end{cases} \quad (5)$$

with the two free parameters Z_p, p_0 for the density models, and g, Z_d, d_0 for the distance models.

To remove the first free parameter, we use the assumption that all one-class models are assumed to model the *same* outlier distribution with their threshold θ_i. The rejection thresholds θ_i in (3) and (4) should therefore coincide for all classes. This removes one of the free parameters.

To fix the second free parameter two alternatives are possible; the first is based on the expected output for the outlier class data, the second is based on the expected output for the target class:

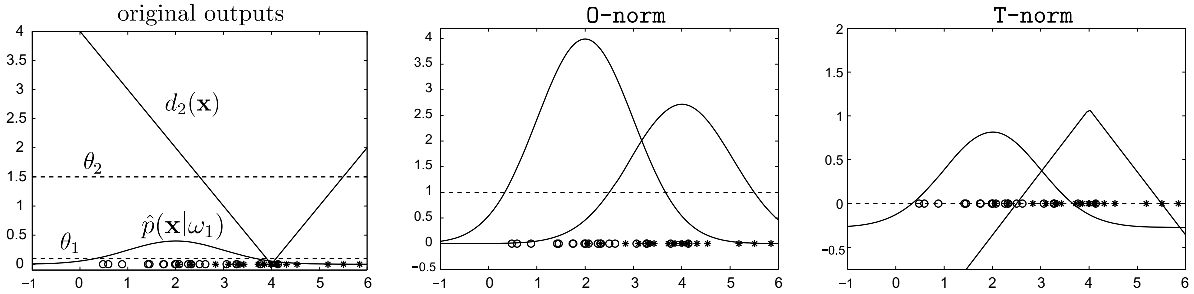

Fig. 1. Class model normalization. Left plot: the original outputs of two class models (a density $\hat{p}(\boldsymbol{x}|\omega_1)$ and a distance $d_2(\boldsymbol{x})$ model) with their thresholds θ_1 and θ_2, indicated by the dashed lines. Center plot: the outputs rescaled using `O-norm`, Right plot: the outputs rescaled using `T-norm`. Note that in the center and right plot just a single threshold is defined.

- Outlier normalization, `O-norm`

  Here the outputs of the models for objects that are (infinitely) far away from the given training data (i.e. the outliers) are standardized. This fixed value is chosen to be 0, such that the density-based models automatically fulfill this. The normalized output for objects on the rejection boundary (i.e. objects $\boldsymbol{x}^*$ for which $\hat{p}(\boldsymbol{x}^*|\omega_i) = \theta$) can be standardized to any positive value, and here it is chosen to be 1: $\tilde{p}_i(\boldsymbol{x}^*) = 1$. Using these constraints the free parameters for the density models in (5) can be solved: $p_0 = 0$, $Z_p = \theta$.

  For the distance-based models the classifier output has to be transformed such that an infinite distance is mapped to a 0. A linear scaling of the distance is not sufficiently flexible to satisfy the constraints that outliers obtain a 0 output. Inspired by Kang and Choi, 2006, we chose a nonlinear exponential function, $\tilde{p}_i(\boldsymbol{x}) = \frac{1}{Z_d}\exp(-d_i^2(\boldsymbol{x})) + d_0$. Solving the free parameters in this model, we obtain $Z_d = \exp(-\theta^2)$ and $d_0 = 0$. Combined, the following `O-norm` normalization is obtained:

$$\tilde{p}_i(\boldsymbol{x}) = \begin{cases} \frac{p(\boldsymbol{x}|\omega_i)}{\theta_i} & \text{density-based models}, \\ \exp(-d_i^2(\boldsymbol{x}) + \theta_i^2) & \text{distance-based models}. \end{cases} \tag{6}$$

- Target normalization, `T-norm`

  In this normalization the outputs for the target class are standardized. We choose to fix the integral of the output over ω_i to π_i, the empirical class priors: $\int_{\mathscr{D}} \tilde{p}_i(\boldsymbol{x})\mathrm{d}\boldsymbol{x} = \pi_i$.[1] In general, this integration cannot be performed exactly, but it can be approximated by summing the classifier output $\tilde{p}_i$ over all training objects from class ω_i:

$$\frac{1}{n_i}\sum_{j=1}^{n_i} \tilde{p}_i(\boldsymbol{x}_j) = \pi_i, \tag{7}$$

where n_i is the number of training objects in class ω_i. The output for objects on the rejection threshold can be fixed to any value lower than $\min_i \pi_i$, and here we choose to fix it to zero: $\tilde{p}_i(\boldsymbol{x}^*) = 0$. For this situation the identity function can be used for g. Solving the free parameters p_0, d_0, Z_p and Z_d, results in the following `T-norm` normalization:

$$\tilde{p}_i(\boldsymbol{x}) = \begin{cases} \pi_i \frac{p(\boldsymbol{x}|\omega_i) - \theta_i}{\sum_j^{n_i} p(\boldsymbol{x}_j|\omega_i)/n_i - \theta_i} & \text{density-based models}, \\ \pi_i \frac{\theta_i - d_i(\boldsymbol{x})}{\theta_i - \sum_j^{n_i} d_i(\boldsymbol{x}_j)/n_i} & \text{distance-based models}. \end{cases} \tag{8}$$

When the outputs of the class models are normalized, we assign a new object $\boldsymbol{z}$ to the class with the highest output:

$$\hat{y} = \begin{cases} \omega_0 & \tilde{p}_i(\boldsymbol{x}) \leqslant \theta_i, \quad \forall i \\ \omega_i & \tilde{p}_i(\boldsymbol{x}) > \tilde{p}_j(\boldsymbol{x}), \quad j \neq i \quad \text{and} \quad \tilde{p}_i(\boldsymbol{x}) > \theta_i. \end{cases} \tag{9}$$

This classification rule is very similar to (2), with the important difference that it reproduces the decision boundaries that were obtained using the individual models (3) and (4) (except where class models overlap).

Both normalizations are shown in Fig. 1. The left subplot shows a one-dimensional dataset containing two equi-probable classes. The two classes are described by a density and a distance-based model, respectively. Their model outputs and their thresholds are drawn using solid and dashed lines. In the center subplot both class model outputs are normalized using `O-norm`: the thresholds are set to 1 and remote outliers will have output 0. In the right subplot, the `T-norm` is applied. Here the rejection threshold becomes 0, while the average training object output becomes $\pi_1 = \pi_2 = \frac{1}{2}$.

Although the philosophy behind the heuristics are different, in practice the results are often very similar. Both normalizations are constructed such that the decision boundaries between the classes and the outliers are not changed (around $x = 0.3$ and $x = 3.7$ for class ω_1, and around $x = 2.5$ and $x = 5.5$ for class ω_2). Objects that are classified as outliers by the individual class models, are also rejected in both normalization schemes. Only in the areas where the non-reject classes overlap (that is around $x = 3$ in the example shown in Fig. 1) a difference between the normalization schemes can be observed.

In the `O-norm` the distances are transformed in a non-linear manner. When identical models for all classes are used, this non-linear transformation changes the output for the classes in an identical manner, and the non-linear transformation has no effect on the final outcome. Changes only occur when density and distance-based models are combined. The `O-norm` is inspired by the transformation that transforms the Mahalanobis distance to the center of a Gaussian distribution to a true density. When the model distance can be interpreted reasonably well as a Mahalanobis distance, and the resulting output is similar enough to a density, the combination of a density output with a true density output can be very fruitful.

A drawback of the `O-norm` is that the output $\tilde{p}$ for a distance model is nearly flat around $d_i = 0$, with a constant value of exp(1) (compare the right graphs in the center and right subplot of Fig. 1). This is important when two classes heavily overlap and the class means are close, or when a class with large variance covers a class with a much smaller variance. The outputs for the two classes $\tilde{p}_1$ and $\tilde{p}_2$ both become exp(1) in the `O-norm`, and it becomes hard to distinguish the two classes. The `T-norm` is therefore better than the `O-norm` for combining distance models with heavily overlapping classes. This is shown in Fig. 2. Here the two classes severely overlap. The means of the classes are located at $\mu_1 = 2$ and

[1] Another idea may be to fix the *maximum* value of the classifier output for the training set. In practice, it appears that this normalization is very noise sensitive. Small variations in the position of the training data change the smallest distance to the model center, which appear to have a significant influence on the final scaling factor. Therefore in this paper `T-norm` normalization is defined based on the average training set output.

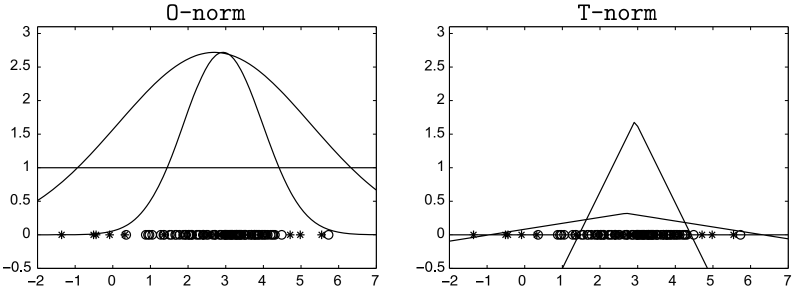

Fig. 2. The difference between `O-norm` (left plot) and `T-norm` (right plot) for overlapping and imbalanced classes. Because of the constant output of the `O-norm` around $d_i = 0$, the distinction between the classes vanishes and the `O-norm` classifies almost all data to the 'star'-class. The `T-norm` can distinguish the classes, and classifies objects around $x = 3$ to the 'circle'-class.

$\mu_2 = 3$, while the variances are $\sigma_1^2 = 6$ and $\sigma_2^2 = 1$. On both classes a support vector data description is fitted, and the outputs using the `O-norm` and `T-norm` are shown in the left and right subplot respectively. Clearly the `O-norm` suppresses the difference between the class outputs around $x = 3$, while the `T-norm` retains it very clearly. This difference is more pronounced in high dimensional feature spaces where differences in class variances tend to be larger.

Although the `O-norm` avoids the loss of class distinction for overlapping classes, it assumes that the training data for the classes is sampled well. That means that the class distributions in training and testing should be relatively similar. When the training data for a certain class is sampled such that it only *covers* the class area in the feature space (as may be sufficient for models that do not perform a density estimation, like the SVDD (Tax and Duin, 1999)), then the average outcome of the objects in the test set is very different from that in the training set. This may result in classes that are totally overwhelmed by other classes. Fortunately, in most applications the sampling of the trainingset is sufficiently reliable, as can be observed in the next section.

4. Experiments

In this section we report experiments on some multi-class datasets, for which it is beneficial to adapt different models per class. First the datasets and some class models are discussed in Section 4.1. In Section 4.2 the results are shown and discussed.

4.1. The datasets and classifiers

For the comparison of the normalizations some standard multi-class UCI dataset(Newman et al., 1998) (see Table 1) are used. To simulate a realistic real world setting, the datasets have to contain some outliers in the test set. Without this outlier data in the test set, classifiers without reject option will obviously always perform best. For datasets that only contain two or three classes (or many tiny classes, like Glass), some artificial outlier data is added to the test set. These outliers are generated from a Gaussian distribution that has a covariance matrix that is four times larger than that of the dataset itself. The number of generated outliers is 50% of the size of the genuine data. That means that when a classifier rejects all test data, a classification performance of 33.3% is obtained. For datasets that contain many larger classes, a few of the classes are relabeled to outlier. The outlier data is ignored during training, only examples from the labeled classes are used. More details on the datasets is shown in Table 1. In the experiments the following class models are fitted:

Table 1
Datasets used and their characteristics

Dataset	Train classes	# Train objects	Outlier classes	# Outlier objects	Dim.
Datasets with generated outliers					
Thyroid	1–3	727		360	21
Hepatitis	1–2	155		77	19
Ionosphere	1–2	351		175	34
Glass	1–7	323		161	9
Datasets with classes are labeled as outliers					
Vowel	1–6	540	7–11	450	10
Face	1–10	100	11–40	300	25[a]
Digits	1–5	2000	6–10	2000	256
Pump	1–3	1350	4	450	64

[a] The Face data is originally 10305 dimensional, but the feature size was reduced by applying PCA first.

Gaussian model. In the experiments, the Gaussian class model is implemented as a distance model, by computing the Mahalanobis distance to the class mean:

$$d_i(\boldsymbol{x}) = (\boldsymbol{x} - \mu_i)^T \sum_i^{-1} (\boldsymbol{x} - \mu_i) \tag{10}$$

When a single Gaussian distribution is estimated on each of the classes the standard quadratic discriminant (Duda et al., 2001) is obtained. This classifier is called 'QD' in the experiments when a single rejection threshold is used. When the rejection threshold is adapted for each individual Gaussian model, it is called 'Gaussian'.
Parzen density model. This density-based model estimates the class conditional density as:

$$\hat{p}(\boldsymbol{x}|\omega_i) = \frac{1}{n_i} \sum_{i=1}^{n_i} \mathcal{N}(\boldsymbol{x}; \boldsymbol{x}_i, h^2) \tag{11}$$

where $\mathcal{N}(\boldsymbol{x}; \boldsymbol{x}_i, h^2)$ is the Gaussian kernel function centered on $\boldsymbol{x}_i$ and with parameter h, evaluated at $\boldsymbol{x}$. The width parameter in the density estimator is estimated using a leave-one-out procedure (Duin, 1976).
Naive Parzen density model. This is the Naive Bayes approach for estimating the class density per feature and combining it to a full density estimate (Hastie et al., 2001). In this case the class model per feature is a Parzen density. The full class density is computed by multiplying the per-feature densities:

$$\hat{p}(\boldsymbol{x}|\omega_i) = \prod_{l=1}^{p} \frac{1}{n_i} \sum_{i=1}^{n_i} \mathcal{N}(x_l; x_{il}, h_l^2) \tag{12}$$

where x_l is the lth element of feature vector $\boldsymbol{x}$. Again, the width parameters in the density estimator are estimated using a leave-one-out procedure.
hypersphere class model. Here a single hypersphere, with center $\vec{a}_i$, is fitted on a class ω_i. The distance to the sphere center is computed like:

$$d_i(\boldsymbol{x}) = \|\boldsymbol{x} - \vec{a}_i\|^2 \tag{13}$$

A hypersphere boundary may be a poor and inflexible model for that class, but for small sample sizes it may be sufficient. Furthermore, the description can be made more flexible by applying the kernel trick, resulting in the Support Vector Data description (Tax and Duin, 2001).

k-means class model. Instead of using a single hypersphere for a class, a set of spheres may be more suitable. The locations of the spheres for class ω_i, $\{\vec{a}_{ij}, j = 1, \ldots, k\}$, can be found by a clustering procedure, like k-means clustering (Tax, 2001). The distance to the nearest center is used as the output of the classifier:

$$d_i(\boldsymbol{x}) = \min_j \|\boldsymbol{x} - \vec{a}_{ij}\|^2 \qquad (14)$$

In the experiments in this paper, per default $k = 5$.

Nearest-neighbor model. When each single training object is considered as the center of a hypersphere, a nearest neighbor approach is obtained. It can be generalized to consider not only the distance to the closest object, but to consider the average distance to the k nearest objects. Without the rejection option, this results in the k-nearest neighbor classifier.

The combined class models are compared to standard classifiers that have a fixed rejection threshold (as defined in (2)). In all cases, the threshold θ is set such that 10% of the training data is rejected. Finally, a combination of *different* class models is fitted, where each class obtains its own optimized model. The optimization of the class models is done manually, by fitting the combinations of class models that create the smallest training error. Although this may introduce a bit more overfitting (and in some cases may be suboptimal), results on the test set show still satisfactory performances, as can be observed in the following section.

4.2. Results

In Table 2 the results for the datasets with artificial outliers are given, and in Table 3 the results for datasets with 'real' outliers (i.e. the outliers are from unseen classes). The best results (and the ones that are not significantly worse) are indicated in bold. The significance is computed using a paired-differences t-test on 10-fold cross validation (Dietterich, 1998). The first five lines give the results for the standard classifiers with a fixed threshold (using (2)). The threshold θ is fixed to obtain 10% rejection. Next, the results are shown for the two normalization methods `O-norm` and `T-norm`, (6) and (8). The thresholds θ_i are fixed such that on each class a 10% rejection rate is obtained. The last two lines in the table show the results for the manually optimized models per class. In the last cases both the class models $p(\boldsymbol{x}|\omega_i)$, $d_i(\boldsymbol{x})$ as the thresholds θ_i are optimized by hand.

Table 2
Classification performance (%) on four datasets for which artificial outlier data is added. The best results, and the ones that are not significantly worse, over the automated procedures are indicated in bold. All classifiers are trained to reject 10% of the data, except for the hand-optimized classifiers in the last two lines

classifier	Thyroid	Hepatitis	Ionosphere	Glass
Standard multi-class classifiers using Bayes rule				
QD with reject	42.7 (0.6)	67.7 (3.1)	*80.7 (1.9)*	62.2 (1.8)
Parzen with reject	60.6 (2.1)	56.7 (4.3)	74.9 (0.9)	*62.4 (1.9)*
Naive Parzen with reject	41.3 (0.3)	56.5 (5.0)	42.5 (0.1)	*65.6 (2.3)*
1-NN with reject	59.2 (0.2)	53.3 (1.7)	56.0 (1.3)	43.9 (1.1)
SVM 2nd degree polyn.	62.6 (0.1)	49.4 (3.0)	57.5 (1.9)	32.8 (2.9)
Model normalization using `O-norm`				
`O-norm` Gaussian	89.8 (0.5)	67.7 (3.1)	63.5 (3.0)	61.5 (1.9)
`O-norm` Parzen	56.0 (2.1)	41.4 (2.0)	39.8 (1.5)	59.5 (2.1)
`O-norm` NaiveParzen	*92.0 (0.4)*	*78.4 (3.3)*	81.2 (2.2)	58.3 (2.5)
`O-norm` SVDD	85.0 (0.4)	*78.0 (3.4)*	46.6 (1.4)	38.0 (3.4)
`O-norm` *k*-Means	82.0 (0.5)	73.1 (3.2)	52.2 (2.6)	*61.7 (2.4)*
`O-norm` *k*-NN	84.1 (0.6)	*78.0 (3.6)*	58.6 (2.8)	38.7 (2.8)
Model normalization using `T-norm`				
`T-norm` Gaussian	89.5 (0.6)	67.7 (3.1)	52.4 (2.0)	*66.6 (2.8)*
`T-norm` Parzen	56.0 (2.1)	41.4 (2.0)	39.8 (1.5)	63.3 (3.0)
`T-norm` NaiveParzen	*92.0 (0.4)*	75.0 (4.1)	*82.1 (2.6)*	*61.5 (2.5)*
`T-norm` SVDD	84.9 (0.4)	*76.3 (3.8)*	56.2 (2.3)	30.8 (3.0)
`T-norm` *k*-Means	85.6 (0.6)	70.3 (4.0)	51.3 (2.3)	*62.8 (2.5)*
`T-norm` *k*-NN	89.9 (0.4)	*78.0 (3.6)*	58.6 (2.8)	38.7 (2.8)
Hand-optimized model normalization using variable rejection rate				
Optimized `O-norm`	94.7 (0.3)	84.8 (2.5)	78.6 (2.8)	62.2 (2.5)
Optimized `T-norm`	95.8 (0.3)	85.1 (1.7)	86.2 (1.7)	68.1 (3.2)

Table 3
Classification performances on four datasets for which some classes are used as outlier classes. The best results, and the ones that are not significantly worse, over the automated procedures are indicated in bold. All classifiers are trained to reject 10% of the data, except for the hand-optimized classifiers in the last two lines

classifier	Vowel	Face	Digits	Pump
Standard multi-class classifiers using Bayes rule				
QD with reject	*58.9 (1.6)*	75.0 (0.0)	63.8 (0.5)	51.9 (1.8)
Parzen with reject	47.3 (0.5)	*91.0 (1.9)*	57.9 (0.4)	25.1 (0.1)
Naive Parzen with reject	*59.0 (2.1)*	89.0 (1.4)	10.0 (0.0)	49.0 (1.7)
1-NN with reject	42.2 (1.8)	75.0 (0.0)	45.9 (0.4)	56.3 (1.3)
SVM 2nd degree polyn.	39.5 (1.2)	24.2 (0.5)	48.0 (0.2)	48.7 (1.8)
Model normalization using `O-norm`				
`O-norm` Gaussian	56.3 (1.7)	75.0 (0.0)	60.6 (0.4)	47.3 (1.5)
`O-norm` Parzen	45.5 (0.0)	75.0 (0.0)	50.0 (0.0)	25.0 (0.0)
`O-norm` NaiveParzen	50.7 (2.1)	80.8 (1.8)	50.0 (0.0)	35.7 (2.0)
`O-norm` SVDD	32.2 (3.1)	*88.0 (2.0)*	59.1 (2.1)	30.4 (3.5)
`O-norm` *k*-Means	*61.1 (1.5)*	80.2 (1.6)	67.9 (3.1)	40.2 (2.3)
`O-norm` *k*-NN	55.7 (1.1)	*93.2 (1.5)*	*70.7 (3.1)*	*79.7 (1.5)*
Model normalization using `T-norm`				
`T-norm` Gaussian	56.6 (1.8)	75.0 (0.0)	60.5 (0.5)	49.3 (1.5)
`T-norm` Parzen	45.5 (0.0)	75.0 (0.0)	42.0 (5.3)	25.0 (0.0)
`T-norm` NaiveParzen	53.3 (2.4)	80.8 (1.8)	50.0 (0.0)	45.0 (1.9)
`T-norm` SVDD	32.0 (2.7)	*88.0 (2.0)*	59.4 (2.1)	30.7 (3.7)
`T-norm` *k*-Means	*62.7 (1.5)*	78.5 (1.1)	66.0 (3.1)	42.1 (3.5)
`T-norm` *k*-NN	55.7 (1.1)	*93.2 (1.5)*	*70.7 (3.1)*	*79.7 (1.5)*
Hand-optimized model normalization using variable rejection rate				
Optimized `O-norm`	68.0 (2.1)	94.0 (1.3)	75.3 (2.8)	82.4 (2.1)
Optimized `T-norm`	65.3 (1.6)	94.0 (1.3)	75.3 (2.8)	82.4 (2.1)

Having a single threshold on all the class models performs reasonably well, in particular when the classes are of the same size and shape and (some of) the classes are all relatively small. This holds for the datasets Ionosphere, Vowel and Face. Here the classes are almost balanced, and the results of the classifiers using a global threshold θ match the performance of the classifiers using a per-class threshold θ_i.

For other classification problems it appears to be important that the thresholds per class model are adjusted. This happens in the Thyroid, Hepatitis, Glass, Digits and Pump. The classifiers with a threshold per class significantly outperform the classifiers with a fixed global threshold.

Generally, the difference between the `O-norm` and `T-norm` is not very large. This is because a large contribution to the error is the misclassification of outlier data and the rejection is identical under the two normalizations. This is visible for the Thyroid, Face and Digits datasets. Only in cases where classes significantly overlap, differences between the normalizations can be observed. The results show a slightly better performance for the `T-norm` than for the `O-norm`. A clear example is the Glass dataset. In the Glass dataset two classes overlap heavily, and one of the classes has a much smaller variance than the other. This is the situation that is shown in Fig. 2: almost all objects of one of these classes are assigned to the other class by the `O-norm`, resulting in a high classification error.

Furthermore, the performance can be improved even further my manual optimization of the rejection threshold. For four datasets (Thyroid, Hepatitis, Digits, Pump) the class models are identical for all classes (Naive Parzen for Thyroid and nearest neighbor for Hepatitis, Digits and Pump), but the thresholds are changed to (10%, 10%, 3%), (10%, 5%), (5%, 10%, 10%, 20%, 20%) and (2%, 10%, 2%) rejection rate per class, respectively. This can result in a performance increase of 5%.

Adjusting the class models per class can also be done by changing the complexity of the class model or by using different models. For the Vowel and Face datasets the class models are fixed to k-means and k-NN, but the parameter k is optimized for the classes: $k = (10, 5, 5, 10, 5, 10)$ for Vowel, $k = (1, 1, 1, 1, 1, 3, 1, 1, 1, 3)$ for Face. Although the combiner now combines only distance models, the distance distribution for the models with different k appear to be very different. The normalization is therefore still essential to combine the class models.

Finally, in the datasets Ionosphere and Glass density and distance models are combined. For Ionosphere a Gaussian model (in this implementation a distance model, using the Mahalanobis distance) and a Naive Parzen model is combined. For the Glass dataset a wide variety is required, due to the large variance in class complexity and class size: three Gaussian models, a Parzen density and two Naive Parzen density models.

5. Conclusions

In real-world classification problems, not only several classes have to be distinguished, but also outliers have to be rejected, and possibly new classes may have to be added. The standard assumption in pattern recognition that only objects from a known set of classes will appear, is not always realistic. For these situations it might be worth to develop a model for each class. When different models for different classes are defined, a normalization heuristic should be used to combine the individual class models into a multi-class classifier. When an object does not fit any of the class models, the object has to be rejected. This heuristic should be such that the addition of new models does not change the acceptance of objects by old models.

This paper presents a new normalization heuristic. The heuristic does not change the rejection boundary between the classes and outlier objects as they are defined by the individual class models, but it defines how objects that are accepted by several class models will be classified. This heuristic fixes the average class output, in contrast to the more common heuristic that fixes the output for the outliers (using a nonlinear transformation of distances). It appears that it performs better in situations where classes severely overlap. It has a higher sensitivity to differences between overlapping classes, and avoids that a larger class (both in terms of spread and number of objects) overwhelms a smaller class. The price to be paid is that the training set distribution should be similar to what is observed in testing. The experiments show that in most practical situations this demand is satisfied.

Furthermore, because the *outputs* of the class models are normalized, not only the class models themselves may differ in terms of the functional form of $p(\boldsymbol{x}|\omega_i)$ or $d_i(\boldsymbol{x})$, but they may also contain different feature preprocessings (resulting in a different $\boldsymbol{x}$ per model), or they may even work in different feature spaces. The important assumption is that the models output a distance or density, and supply a threshold for the rejection of outliers. This therefore allows for the integration of completely independently constructed class models into one classification system.

One note of warning should be added. In the proposed combination of class models it is assumed that the class overlap is very severe. When large class overlap exist, good density models have to be constructed in order to distinguish the classes. In this paper we considered class models that are based on distances. In order to combine these with other models (both density-based and distance-based) ad-hoc heuristics have to be used and no proofs of optimal performance can be given.

Finally, optimizing the models per class appears to be very fruitful. A fixed model per class may easily introduce overtraining for some of the classes and undertraining for others, in particular when classes differ in size, complexity, and the amount of outliers differs in the training set of each class. By optimizing the complexity of the model to each of the classes, the final classification performance may be improved significantly.

References

Baker, D., Hofmann, T., McCallum, A., Yang, Y., 1999. A hierarchical probabilistic model for novelty detection in text. Technical Report, Just Research.

Bishop, C.M., 2006. Pattern Recognition and Machine Learning. Springer.

Cerioli, A., Riani, M., 1999. The ordering of spatial data and the detection of multiple outliers. J. Comput. Graph. Statist. 8 (2), 239–277.

Chow, C.K., 1970. On optimum recognition error and reject tradeoff. IEEE Trans. Inf. Theory IT-16 (1), 41–46.

Davies, L., Gather, U., 1993. The identification of multiple outliers. J. Amer. Statist. Assoc. 88, 782–792.

Dietterich, T.G., 1998. Approximate statistical test for comparing supervised classification learning algorithms. Neural Comput. 10, 1895–1923.

Dubuisson, B., Masson, M., 1993. A statistical decision rule with incomplete knowledge about classes. Pattern Recognition 26 (1), 155–165.

Dubuisson, B., Usai, M., Malvache, P., 1985. Computer aided diagnostic with an incomplete learning set. Progr. Nucl. Energy 15, 875–880.

Duda, R.O., Hart, P.E., Stork, D.G., 2001. Pattern Classification, second ed. John Wiley & Sons.

Duin, R.P.W., 1976. On the choice of the smoothing parameters for Parzen estimators of probability density functions. IEEE Trans. Comput. C-25 (11), 1175–1179.

Fumera, G., Roli, F., Giacinto, G., 2000. Reject option with multiple thresholds. Pattern Recognition, 2099–2101.

Hastie, T., Tibshirani, R., Friedman, J., 2001. The Elements of Statistical Learning: Data Mining, Inference and Prediction. Springer Series in Statistics. Springer, New York, N.Y.

Japkowicz, N., Myers, C., Gluck, M., 1995. A novelty detection approach to classification. In: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, pp. 518–523.

Kang, W.-S., Choi, J.Y., 2006. SVDD-based method for face recognition system. In: Proceedings of the SCIS & ISIS, 2006.

Marsland, S., 2001. On-line novelty detection through self-organisation, with application to inspection robots. Ph.D. Thesis, University of Manchester.

Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J., 1998. UCI repository of machine learning databases. URL:<http://www.ics.uci.edu/~mlearn/MLRepository.html>.

Pan, J.-X., Fung, W.-K., Fang, K.-T., 2000. Multiple outlier detection in multivariate data using projection pursuit techniques. J. Statist. Planning Inference 83 (1), 153–167.

Ramaswamy, S., Rastogi, R., Shim, K., 2000. Efficient algorithms for mining outliers from large data sets. Technical Report, Bell Laboratories.

Tarassenko, L., Hayton, P., Brady, M., 1995. Novelty detection for the identification of masses in mammograms. In: Proceedings of the Fourth International IEE Conference on Artificial Neural Networks, vol. 409, pp. 442–447.

Tax, D.M.J., 2001. One-class classification. Ph.D. Thesis, Delft University of Technology, June. <http://ict.ewi.tudelft.nl/~davidt/thesis.pdf>.

Tax, D.M.J., Duin, R.P.W., 1999. Data domain description using support vectors. In: Verleysen, M. (Ed.), Proceedings of the European Symposium on Artificial Neural Networks 1999, D. Facto, Brussel, pp. 251–256.

Tax, D.M.J., Duin, R.P.W., 2001. Uniform object generation for optimizing one-class classifiers. J. Mach. Learn. Res., 155–173.