Mach Learn (2009) 74: 191–234
DOI 10.1007/s10994-008-5092-4

Incremental data-driven learning of a novelty detection model for one-class classification with application to high-dimensional noisy data

Randa Kassab · Frédéric Alexandre

Received: 24 February 2007 / Revised: 4 November 2008 / Accepted: 7 November 2008 /
Published online: 2 December 2008

Abstract Most conventional learning algorithms require both positive and negative training data for achieving accurate classification results. However, the problem of learning classifiers from only positive data arises in many applications where negative data are too costly, difficult to obtain, or not available at all. This paper describes a new machine learning approach, called ILoNDF (Incremental data-driven Learning of Novelty Detector Filter). The approach is inspired by novelty detection theory and its learning method, which typically requires only examples from one class to learn a model. One advantage of ILoNDF is the ability of its generative learning to capture the intrinsic characteristics of the training data by continuously integrating the information relating to the relative frequencies of the features of training data and their co-occurrence dependencies. This makes ILoNDF rather stable and less sensitive to noisy features which may be present in the representation of the positive data. In addition, ILoNDF does not require extensive computational resources since it operates on-line without repeated training, and no parameters need to be tuned. In this study we mainly focus on the robustness of ILoNDF in dealing with high-dimensional noisy data and we investigate the variation of its performance depending on the amount of data available for training. To make our study comparable to previous studies, we investigate four common methods: PCA residuals, Hotelling's T^2 test, an auto-associative neural network, and a one-class version of the SVM classifier (lately a favored method for one-class classification). Experiments are conducted on two real-world text corpora: Reuters and WebKB. Results show that ILoNDF tends to be more robust, is less affected by initial settings, and consistently outperforms the other methods.

Keywords ILoNDF · Novelty detection · One-class classification · Neural networks · Text categorization

Editor: Tom Fawcett.

R. Kassab (✉) · F. Alexandre
LORIA, INRIA Lorraine, Campus Scientifique, BP. 239, 54506 Vandœuvre-lès-Nancy Cedex, France
e-mail: randa.kassab@loria.fr

F. Alexandre
e-mail: frederic.alexandre@loria.fr

1 Introduction

In standard two-class classification, data from two classes—positive and negative—are considered when developing classification models. However, the need to train classification models from positive data alone arises in many applications where negative data are too costly, difficult to obtain, or not available at all. This leads to the problem of one-class classification, which can also be described as a two-class classification problem where each of the two classes has a special meaning, but only training data from the so-called "target" or "positive" class are available. The other class, for which no examples are available, is considered the "outlier" or "negative" class and represents all other possible data not belonging to the target class. A prototypical application of one-class classification is novelty or anomaly detection where examples of the novel or abnormal class are very difficult or expensive to acquire. Furthermore, in applications with highly unbalanced data, studies have shown that ignoring all of the negative data and applying a one-class classifier yields better performance, especially for the minority classes (Raskutti and Kowalczyk 2004; Kassab and Lamirel 2007).

One-class classification is important in areas such as information filtering, web navigation assistance and novel event detection. Such domains may lack clearly labeled negative examples for a variety of reasons (Žižka et al. 2006; Yu et al. 2004). For example, some applications attempt to learn about users' interest by recording visited documents or links, which are interpreted as positive examples of interest while non-visited ones are interpreted as negative examples. But as Schwab et al. (2000) point out, the non-selection of a document does not necessarily mean it is irrelevant or uninteresting; the non-visited documents may perhaps be visited later, and classifying them as negative examples may be wrong. Moreover, users are often very busy and cannot give enough examples of their needs, so a system may need to be trained on the basis of relevant documents only, such as a user's bookmarks or some early documents collected by the user over a long time which are often the interesting results (Denis et al. 2003; Žižka et al. 2006). Finally, for marketing reasons, users may actually be prevented from giving negative ratings to products, advertisements or web pages.

The absence of negative data makes the one-class classification task more challenging and very different from that of normal two-class classification. It is possible to apply adapted versions of traditional two-class learning algorithms to solve a one-class problem, but this often results in substantial degradation of performance. This degradation seems to be more extreme for discriminative approaches than for generative ones (Japkowicz 2001). An approach that works well for a two-class or multi-class problem is not necessarily the best for a single-class problem. Apart from performance considerations, there is also a problem of setting an appropriate threshold (boundary) between two classes using data from just one. Indeed, most of the adapted algorithms provide only a relevance ranking approach for sorting new data with respect to the positive class, like the one-class KNN approach proposed in (Žižka et al. 2006). In contrast, few others provide a parametric threshold like the one-class SVM approach (Schölkopf et al. 2001). In both cases the decision threshold must be provided by the user, and this may be neither desirable nor easy.

This paper deals with a new machine learning approach inspired by novelty detection theory. The basic idea of novelty detection is to learn a model of the normal data of the domain being monitored and then to automatically detect novel data that deviate from the learnt model. The principle of novelty detection is mainly interesting for solving the one-class classification problem since training typically involves examples from the normal class alone. Several methods for novelty detection are proposed in the literature (Markou and

Singh 2003a, 2003b). The specific novelty detector filter (ILoNDF) we use is an adaptation of a former novelty detection method (NDF) proposed by Kohonen (1989), which is based on orthogonal projection operators. In addition to its simplicity, this method has the desirable property that it operates on-line, which is essential in applications where memory space is limited and real-time response is crucial. Although Kohonen's novelty learning rule is used as the basis of this method, developing it has involved major modifications and the definition of appropriate strategies for exploiting the output of this new method to the classification goal. Our method is able to detect co-occurrence dependencies between features of the training data, making ILoNDF fairly stable and less sensitive to noisy features which may be present in the representation of the positive data.

The rest of this paper is organized as follows. We begin with a discussion of some work related to one-class classification. In Sect. 3 we present the basic aspects of the NDF method and illustrate how the ILoNDF method addresses some of NDF's problems. We also discuss the use of ILoNDF for one-class classification and the strategy of choosing the decision threshold. Sections 4, 5 and 6 briefly review the principle of four common methods for one class classification: PCA residuals, Hotelling's T^2 Test, auto-associative neural networks, and the one-class SVM. Section 7 describes the experimental settings, and Sect. 8 presents empirical results and analyzes the findings. Section 9 concludes with remarks and some directions for future work.

2 Related work

Although most existing studies of learning assume that examples of every class are available for training (Sebastiani 2002), some prior work exists on the one-class classification problem (Yu et al. 2004; Denis et al. 2003). Two principal approaches have been taken. The first approach tries to artificially infer negative examples from a set of unlabeled examples and then applies traditional two-class learning algorithms. The second approach learns directly from positive examples alone.

The approach of extracting negative examples from a set of unlabeled ones has been pursued by several researchers (Yu et al. 2003; Liu et al. 2003; Li and Liu 2003). They have argued that unlabeled examples are readily available, e.g. from the World Wide Web, so their main objective is to reduce manual labeling effort while maintaining performance as high as that of traditional learning approaches, which learn from labeled positive and negative examples. Most such work follows a two-stage approach:

1. An initial classifier is built using either positive examples alone (e.g. 1-SVM in Yu et al. 2003) or a combination of both positive and unlabeled examples (assuming the latter as negative examples, Li and Liu 2003; Liu et al. 2003). The resulting classifier is then used to identify strong negative examples from the unlabeled set;
2. A two-class learning method (e.g. SVM and EM, Yu et al. 2003; Li and Liu 2003; Liu et al. 2003) is run iteratively to find additional negative examples from the remaining unlabeled set. The best or final classifier is then selected.

Classification performance varies depending on the different techniques used in each stage. Details are available in (Liu et al. 2003; Fung et al. 2006). A recent paper (Fung et al. 2006) presents an algorithm called PNLH which follows a somewhat different strategy. At the second stage it tries to enlarge the positive example set by extracting strong positive examples from the unlabeled set as well. Results show that PNLH generally improves the quality of classification with respect to the former strategy, especially when the number of positive examples is extremely small.

There are three drawbacks to the above general approach.

1. Extracting negative examples from unlabeled ones is unreliable and requires caution. Indeed, this approach is based on the assumption that unlabeled examples are noisy negatives, i.e. the proportion of positive examples in the unlabeled set is very small (cf. Yu et al. 2003), so that the classifier used in the first stage would identify strong negatives and exclude positive ones. Even if this were not the case, many positive examples would wrongly be extracted as negative examples, and the resulting performance would be poor and would degrade with iteration steps. For this reason we advocate the use of one-class classifier for the first stage (e.g. 1-SVM in Yu et al. 2003) in order to address these concerns;
2. Most existing methods perform poorly when the number of positive examples is small. Moreover, the unlabeled set must be sufficiently large to find the best classifier;
3. Finally, the iterative approach to find negative examples greatly increases the time complexity of the learning algorithm.

Although these drawbacks may not be important in all applications, they present serious limitations in some, and for these the strategy of learning from positive examples alone becomes an attractive alternative. Relatively little attention has been paid to the problem, however. Some early work on this issue was explored in the fields of pattern recognition and novelty detection (Japkowicz et al. 1995; Japkowicz 2001). We briefly review some of this work.

Early work on the one-class learning problem was based on neural networks. A novelty detection approach to classification based on the use of an auto-associative neural network (AANN) is reported in (Japkowicz 2001). The auto-associator is a feedforward three-layer network, with the same number of input and output neurons and significantly fewer hidden neurons. The AANN is trained to reconstruct the input data (positive examples) as output. After training, the reconstruction error of each novel input example is calculated and classification is performed, relying on the assumption that positive data will be accurately reconstructed while negative data will not. However, this technique involves few negative examples to adequately set the decision threshold in the case of noisy data where the expected quality of separation between positive and negative reconstruction errors is low.

More recently, one-class versions of support vector machines have been developed (Schölkopf et al. 2001; Tax and Duin 2001; Manevitz and Yousef 2001). The idea of the 1-SVM approach proposed by Schölkopf et al. (2001) is to map the positive data into a feature space corresponding to a kernel and to separate them from the origin, the only negative data, with maximum margin. In addition to requiring a user to provide the basic parameters of the SVM, the 1-SVM approach requires fixing a priori the percentage of positive data allowed to fall outside the description of the positive class. This makes 1-SVM more tolerant to outliers in the positive data. However, setting this parameter is not intuitive and its value strongly influences the performance of 1-SVM.

Another approach, the SVDD method proposed by Tax and Duin (2001), tries to find a hypersphere with minimum volume that encloses all or most data in the positive class. The constraint of minimal volume is applied to minimize the chance of accepting negative data. SVDD automatically optimizes the 1-SVM parameter by using artificially generated unlabeled data uniformly distributed in a hypersphere around the positive class. Even so, the optimization method is not applicable in high-dimensional spaces (e.g. more than 30 dimensions).

A somewhat different version of 1-SVM (outlier-SVM) is proposed by Manevitz and Yousef (2001). Their approach is to assume not only the origin as in the negative class but

also all data that are "close enough" to the origin are to be considered as negatives or outliers. The identification of outliers is accomplished by counting the number of non-zero features of a datum vector; if this number is less than a threshold then the datum is considered as a negative example, otherwise it is positive. The assumption behind this approach is that a datum which shares very few features with the feature set selected to represent the positive data is not a good representative of the positive class and can be treated as an outlier. Experimental results obtained on a text benchmark dataset show this approach does generally worse than the 1-SVM approach.

In the same paper, Manevitz and Yousef compare 1-SVM with one-class versions of the algorithms: Rocchio, Nearest Neighbor, Naive Bayes, feed-forward neural network (AANN). They conclude that 1-SVM and AANN are better than other methods tested. However, it is not clear how the decision threshold was set for the evaluated methods apart from 1-SVM. Lee and Cho (2006) also compared the performance of 1-SVM and AANN for the purpose of novelty detection. Empirical results from six benchmark datasets show that 1-SVM performs consistently better than AANN.

3 The ILoNDF model

Having discussed two general approaches to this problem, we introduce ILoNDF, a method of the second kind which learns from positive examples alone.

3.1 Background

Kohonen and Oja (1976) introduced the first novelty filter. It is an orthogonalizing algorithm that passes through only the "novelty" component of an input vector with respect to all of the earlier input data. Basically, the novelty filter relies on the properties of orthogonal projection operators. Let there be m distinct Euclidean vectors, denoted $x_1, x_2, \ldots, x_m \in \Re^n$, which span a subspace $\zeta \subset \Re^n$. The complement space of ζ, denoted $\zeta^\perp$, is spanned by all vectors in $\Re^n$ which are orthogonal to ζ. Then, any arbitrary vector $x \in \Re^n$ can be uniquely decomposed into the sum of two vectors $\hat{x} \in \zeta$ and $\tilde{x} \in \zeta^\perp$. One particular property of the orthogonal projections is that of all possible decompositions of $x = \dot{x} + \ddot{x}$ where $\dot{x} \in \zeta$, the one stated above is subject to $\|\tilde{x}\| = \min_{\|\ddot{x}\|} \|\ddot{x}\|$; in other words, the norm of $\tilde{x}$ is equivalent to the distance of x from ζ. The component $\hat{x}$ of the vector x is the orthogonal projection of x on ζ while the component $\tilde{x}$ is the orthogonal projection of x on $\zeta^\perp$. The matrix operator representation of the projections provides a guideline which can help to calculate the orthogonal projections $\hat{x}$ and $\tilde{x}$ as outlined hereafter. Let $X \in \Re^{n \times m}$ be a matrix with the x_i vectors as its columns, and X^+ be the pseudo-inverse of X.[1] Then XX^+ is a matrix operator which projects a vector x on ζ:

$$\hat{x} = XX^+x. \tag{1}$$

Similarly, $I - XX^+$ is the operator which projects x on $\zeta^\perp$:

$$\tilde{x} = (I - XX^+)\, x \tag{2}$$

with I being an identity matrix.

[1] Penrose showed that there exists a unique generalized pseudo-inverse for any rectangular matrix (Penrose 1955).

With respect to the above findings, the component $\tilde{x}$ can be regarded as the residual contribution in x that is left when a particular information processing operation is applied to x. So if a system is established with the matrix of Eq. 2 as its transfer function, $\tilde{x}$ may be interpreted as the amount that is maximally new or independent in an input vector x according to previous vectors x_i. In (Kohonen and Oja 1976) an artificial neural network for implementing the principle of novelty detection is developed and shown to be equivalent to the orthogonal projection operators under certain conditions. The next section describes the architecture of the neural system, viz. the NDF model.

3.2 NDF implementation

Kohonen and Oja (1976) implemented the NDF model with a recurrent network of neuron-like adaptive elements with internal connections. It consists of n fully connected feedback neurons, where n is the dimensionality of the input vectors, and so that all neurons are both input and output neurons. Figure 1 shows the typical architecture of such a network. The output of each neuron $\tilde{\eta}_i$ is determined by a linear combination of the external input η_i to that neuron and the feedback it receives from the output:

$$\tilde{\eta}_i = \eta_i + \sum_j m_{ij}\tilde{\eta}_j. \tag{3}$$

The weights m_{ij} associated with the feedback connections provide the variable internal state of the network. They are initially set to zero and are updated after each presentation of an input vector using an anti-Hebbian learning rule:

$$\frac{dm_{ij}}{dt} = -\alpha\tilde{\eta}_i\tilde{\eta}_j \tag{4}$$

where α is a small positive parameter that can be adaptively modified during learning.

Thus, strongly correlated output neurons will have strong inhibitory connections, which will reduce their correlation. This decorrelation can lead to suppressing the redundant activity of input features which code the overlap between similar data. Aside from removing correlations between the features, self-feedback connections tend to decrease the activity of individual features. The above module can now be expressed as a matrix operator $\Phi \in \Re^{n\times n}$ with the following equations:

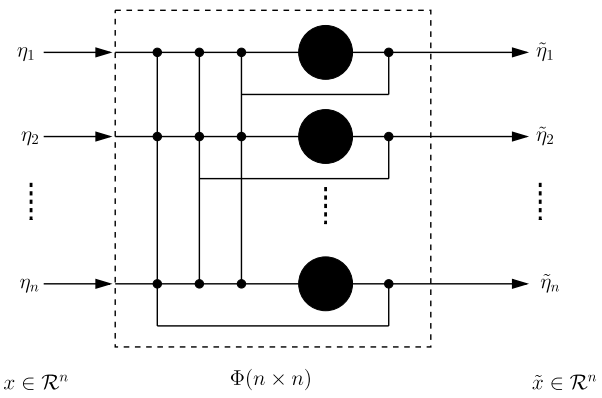

Fig. 1 The basic architecture of the NDF model

$$\begin{aligned} \tilde{x} &= x + M\tilde{x} = (I - M)^{-1}x = \Phi x, \\ \frac{dM}{dt} &= -\alpha \tilde{x}\tilde{x}^T. \end{aligned} \tag{5}$$

The differential equation for Φ is given by:

$$\frac{d\Phi}{dt} = -\alpha \Phi^2 x x^T \Phi^T \Phi. \tag{6}$$

This is a Bernoulli matrix differential equation of degree 4. Although its solution seems difficult, Kohonen and Oja (1976) show that it has stable asymptotic solutions if $\alpha \geq 0$. The approximate solutions are obtained under certain simplifying assumptions on the input data and the initial conditions of the matrix Φ. The assumptions are: (1) x is applied at the input and held constant for a sufficiently long period of time; (2) Φ_0 is initially a projection matrix, i.e. symmetrical and idempotent. Thereby, the state matrix Φ is allowed to approximately converge towards another projection matrix on the space $\mathcal{R}(\Phi_0) \cap \zeta^{\perp}$ where $\mathcal{R}(\Phi_0)$ is the range space of Φ_0 and $\zeta^{\perp}$ is the orthogonal complement of the space spanned by the x_i vectors (cf. Sect. 3.1). In particular, a special case of projection matrices which is consistent with the definition of the novelty filter is the identity matrix. Starting from a projection matrix $\Phi_0 = I$, the filter will initially act as an identity operator for any data applied to its input and the state matrix will always attain a stable state given by:

$$\Phi_c = I - XX^+ \tag{7}$$

which is the projection operator on the space $\mathcal{R}(I) \cap \zeta^{\perp} = \zeta^{\perp}$, which is orthogonal to the space spanned by the x_i vectors.[2]

However, for high-dimensional data, the neural model of NDF is computationally costly since the network is fully-connected and the number of neurons equals the dimensionality of the input space. To prevent this complexity, a computationally easier method is to directly compute the stable state Φ_c using any pseudo-inverse algorithm (Ben-Israel and Greville 2003). In this study we are interested in Greville's theorem for the purpose of an on-line solution (Greville 1960). Moreover, Greville's method has been found to give the fastest computation time (Noda et al. 1997). The theorem provides a recursive computation of the Moore-Penrose inverse of a matrix yielding a recursive expression of the transfer function of the NDF model (Eq. 7) as described below.

Let X be a matrix with $x_1, x_2, \ldots, x_k$ its columns and let it be partitioned as $[X_{k-1}, x_k]$ such that the matrix X_{k-1} comprises previously learnt data while x_k is a new input datum. The theorem of Greville states:

$$X_k^+ = \begin{bmatrix} X_{k-1}^+(I - x_k \mathcal{P}_k^T) \\ ---------- \\ \mathcal{P}_k^T \end{bmatrix}, \quad \text{where} \tag{8}$$

$$\mathcal{P}_k = \begin{cases} \frac{(I - X_{k-1}X_{k-1}^+)x_k}{\|(I - X_{k-1}X_{k-1}^+)x_k\|^2} & \text{if the numerator} \neq 0, \\ \frac{(X_{k-1}^+)^T X_{k-1}^+ x_k}{1 + \|X_{k-1}^+ x_k\|^2} & \text{otherwise.} \end{cases}$$

[2]The reader can refer to (Kohonen 1989) for more details and theoretical justifications of the model.

and the recursive procedure is initiated with

$$X_1^+ = \begin{cases} x_1^T (x_1^T x_1)^{-1} & \text{if } x_1 \neq 0, \\ 0^T & \text{if } x_1 = 0. \end{cases}$$

Based on the above, it then follows that:

$$X_k X_k^+ = X_{k-1} X_{k-1}^+ (I - x_k \mathcal{P}_k^T) + x_k \mathcal{P}_k^T. \tag{9}$$

If $(I - X_{k-1} X_{k-1}^+) x_k$ is a zero vector the above formula yields $X_k X_k^+ = X_{k-1} X_{k-1}^+$; for a non-zero vector the upper expression for $\mathcal{P}_k$ is applied. For both cases the recursive form of Φ after the presentation of $x_1, \ldots, x_k$ at the filter input can be written:

$$\begin{aligned} \Phi_k &= I - X_k X_k^+ \\ &= (I - X_{k-1} X_{k-1}^+) - \frac{(I - X_{k-1} X_{k-1}^+) x_k x_k^T (I - X_{k-1} X_{k-1}^+)}{\|(I - X_{k-1} X_{k-1}^+) x_k\|^2}. \end{aligned} \tag{10}$$

Note that $(I - X_{k-1} X_{k-1}^+)$ is the transfer function of NDF after the presentation of all the vectors $x_1, \ldots, x_{k-1}$. Therefore Eq. 10 can be put into the following form:

$$\Phi_k = \Phi_{k-1} - \frac{\tilde{x}_k \tilde{x}_k^T}{\|\tilde{x}_k\|^2} \tag{11}$$

where $\tilde{x}_k = \Phi_{k-1} x_k$ represents the orthogonal projection of the data x_k on a space that is orthogonal to the space spanned by the $k-1$ learnt data, and the recursion starts with $\Phi_0 = I$.

The amount of novelty in an input vector x_i with respect to the previous vectors x_k can be specified by

$$N_{x_i} = \frac{\|\tilde{x}_i\|}{\|x_i\|}. \tag{12}$$

The complementary amount $H_{x_i} = 1 - N_{x_i}$ can be used as indicator of the similarity of x_i (its habituation) with respect to the previously learnt data.

To recapitulate, the work of the NDF model with Eq. 11 as its transfer function, and its input and output related by $\tilde{x} = \Phi_k x$, can be summarized as follows. During the learning phase the NDF model adapts to the reference data presented as input. Once learning is complete, if one of the reference data or a combination of it is applied to the filter input, the novelty output will be zero. On the other hand, if a novel datum not belonging to the space spanned by the reference data is chosen as an input, the corresponding output will be nonzero; it will be seen as representative of the new features extracted from the input data with respect to the reference data.

3.3 On the strengths and weaknesses of NDF

The ability of NDF to operate in an on-line mode without repeated training is highly desirable for many on-line and real-time one-class classification applications, such as surveillance and event detection tasks. Moreover, NDF contains no parameters to be tuned before or during training, so there is no need to perform additional computations.

On the other hand, being biologically inspired by the orthogonalization ability of the hippocampus,[3] NDF tends to act as a long-term memory recognizing quickly a datum on which it was trained. Still, NDF is biologically implausible in the sense of being unassociated with time-dependent forgetting capacities. With knowledge of these characteristics, we can expect the behavior of NDF as follows:

- If an input datum appears only occasionally it will not be considered novel during its future appearances, and the NDF model will recognize it no matter the time interval between occurrences and whether the datum is more often seen or not. In the context of one-class classification, this behavior requires that training data be noise-free (which is often unrealistic); otherwise, NDF will fail to give a reliable description of the target class.
- The inability of NDF to forget makes it inadequate to deal with dynamic environments where the target data may change over time, for example in robotic exploration and inspection tasks (Marsland et al. 2000).

  However, note that an adaptation of the NDF model with forgetting is also discussed in (Kohonen 1989). This was motivated by the fact that memory traces are allowed to decay in physical systems. A forgetting factor ($\beta > 0$) is introduced that decreases all the feedback connection weights at a rate directly proportional to their values. The weight update formula, in matrix notation, then becomes:

$$\frac{dM}{dt} = -\alpha \tilde{x}\tilde{x}^T - \beta M. \tag{13}$$

  Instead of Eq. 6, the following more general differential equation is now obtained:

$$\frac{d\Phi}{dt} = -\alpha \Phi^2 x x^T \Phi^T \Phi + \beta(\Phi - \Phi^2). \tag{14}$$

  Unfortunately, the solution of this equation is not easy to derive and might not have the implicitly required convergence properties (Aeyels 1990). In addition, Kohonen (1989) stated that—under the same assumptions made for the NDF model—the asymptotic solution with forgetting has the same form as a solution of the original equation without forgetting, and thus, will tend to a matrix which is approximately a projection matrix.
- The characteristic time of habituation to a given feature—which denotes the number of reference data containing that feature and which have been presented to the NDF's input—is rather short (inversely propositional to the number of features present in the data). Allowing such fast habituation to the features during the learning phase has a disadvantage in technical applications where training data may include some noisy features, e.g. text or bioinformatics data. In fact, Kohonen's learning rule (cf. Eq. 11) is mainly designed to distinguish the novelty parts from the old (habituated) parts in the input data with respect to the previously seen reference data. Accordingly, only novel parts are passed through the filter and will be considered when modifying the internal state through the feedback connections. This will not only accelerate the habituation of the filter to the features that have not been seen before but also will prevent the filter from improving the acquired knowledge about previously habituated features. In this case, almost all of

[3]The hippocampus is a region of the brain which is believed to play an essential role in learning and memory processes, such as the representation and retrieval of long-term memories—any type of novelty detection requires the retrieval of previously seen stimuli—and the formation of new memories about experienced events (Sirois and Mareshal 2004). Another area of the brain which is thought to be involved in dealing with novelty is the perirhinal cortex also belonging to the hippocampic system (Brown and Xiang 1998).

the relevant features will be approximately habituated after some training data have been presented, and consequently their presence in the ensuing training data will nearly be ignored, allowing faster habituation to noisy features which might be present in the training data.[4] In this regard, relevant and noisy features will be of about equal importance in the description of training data, which can greatly diminish the classification or detection accuracy.

To illustrate some of the above points, consider the following two simple cases.

Case 1: Sample Data #1 and #2

Suppose we have the following three training data $d_1, d_2, d_3 \in \Re^3$:

Sample Data Set #1

	d_1	d_2	d_3
f_1	1	1	1
f_2	0	1	1
f_3	0	0	1

After training on the above data, the transfer function of the filter is $\Phi_3 = 0$, a zero matrix. Therefore the novelty space is the null space, meaning that all the training data and their linear combinations have been habituated totally ($\tilde{d}_1, \tilde{d}_2$ and $\tilde{d}_3$ are null vectors). In particular, we can calculate the novelty and habituation proportion of the features by projecting the unit vector $\overrightarrow{u_f}$ associated with each feature on the novelty space; using the following formulas:

$$N_f = \frac{\|\Phi_3 \overrightarrow{u_f}\|}{\|\overrightarrow{u_f}\|}, \qquad H_f = 1 - N_f.$$

Then, we get:

$$H_{f_1} = H_{f_2} = H_{f_3} = 1.$$

This means that all the features f_1, f_2 and f_3 have been totally habituated by the model and will have the same importance when representing the training data, despite the fact that f_1 is more correlated with the description of training data than the other features and that f_2 is more prevalent than f_3, which could be a noisy feature. This phenomenon is due to the fast learning of features that is characteristic of the NDF model. So once a feature is totally habituated it will no longer be considered in subsequent learning steps (i.e. the presence or absence of such a feature in the ensuing training data will not affect learning in subsequent steps). This is mainly because learning is guided by the novelty vector alone. For example, training the NDF model on the following training data:

Sample Data Set #2

	d_1	d_2	d_3
f_1	1	0	0
f_2	0	1	0
f_3	0	0	1

[4] This can explain the tendency of NDF to be more effective on low-dimensional data, as will be shown in the experimental results of Sect. 8.1.

will yield the same results as when training on the Sample Data Set #1. On this basis, we can conclude that the fact that the NDF's learning is only guided by the novel part of the input data (the novelty vector) increases the learning possibility of non-discriminating features. In other words, the NDF's learning allows noisy features (such as the feature f_3 in Sample Data Set #1) to become approximately as habituated as relevant features (such feature f_1 of the same data set). Obviously, this problem cannot be solved by further training since the filter matrix is null.

Case 2: Sample Data #3

Suppose now we have the following training data:

Sample Data Set #3

	d_1	d_2	d_3	d_4	d_5	d_6	$d_7 \ldots$
f_1	1	1	0	0	0	0	0
f_2	0	1	0	0	0	0	0
f_3	0	0	1	0	1	1	1
f_4	0	0	0	1	1	1	0
f_5	0	0	0	0	1	0	1

After presenting d_1 and d_2 at the input of NDF, the filter will be totally habituated to both data as well as to their features f_1 and f_2. If we continue training by presenting data with different characteristics, the filter will learn the new data and their features while always remembering the former ones (f_1, f_2). This is not suitable for classification since although the first two data are not frequently presented to the filter, their features are totally habituated and will play an important role in describing the training data. Moreover, for some applications, it would be desirable to gradually discard the knowledge acquired about the old data and to consider them as novel if they are not frequently seen.

3.4 Incremental data-driven learning of NDF

The above concerns motivate an adaptation of the NDF's learning rule. Two critical issues should be considered in designing the new learning rule. First, all of the features of the input data should be considered throughout the course of training, regardless of whether they are already habituated. Second, knowledge about how much a datum or feature is novel should be taken into account continuously in order to maintain the principle of novelty detection.

With respect to these requirements, several adaptations of the NDF's learning rule have been investigated. Two among these were found rational to guarantee the fulfillment of the requirements necessary to address the NDF's defects. The first adaptation was guided by the assumption that the main defect of NDF comes from the fast habituation to the features of the training data, which often leads to a full saturation of the model (this corresponds to the case where $\Phi = 0$; that is, the novelty space is equivalent to the null space). This is mainly because learning begins with an identity matrix $\Phi_0 = I$, so if a feature f_i appears alone in a training datum (or several times with other features) it will become totally habituated by the model ($\Phi_{ik} = 0$, $\Phi_{ki} = 0$, $\forall k$). After this it will not be taken into account in further learning. Therefore, the simplest solution is to use the original learning rule while starting the recursion with a scalar matrix in which all of the diagonal elements are set to the number of training data. The results obtained by this solution are slightly inferior to those of the second solution we will present (Eq. 15). In addition, the number of training data must be known in advance, which is unsuitable for on-line classification applications.

The second solution originated from the fact that NDF's learning update is only guided by the novel part of the input data (the novelty vector $\tilde{x}$). This implies that once a feature becomes totally habituated by the filter it will no longer be considered in the updating process. To prevent this, the best strategy we found is to introduce the identity matrix at each step of training and to project each input datum on both identity and filter matrices. The resulting learning rule can be put in the form:

$$\Phi_k = I + \Phi_{k-1} - \frac{\tilde{x}_k \tilde{x}_k^T}{\|\tilde{x}_k\|^2} \tag{15}$$

where $\tilde{x}_k = (I + \Phi_{k-1})x_k$ and Φ_0 is a zero, or null, matrix. In this way, we can satisfy the first requirement by projecting each of the training data x_k on the identity matrix. This implies that each of the features present in the data x_k will take part in the updating process, regardless of the amount of novelty or redundancy in the data under consideration. This is especially important in dealing with situations in which a datum or feature has already been seen and its projection on the space spanned by the filter is zero. The second requirement is satisfied by simultaneously projecting the same data x_k on the NDF transfer function corresponding to the older training data. As learning progresses, features which frequently appear in the training data become more and more habituated as compared with less frequent ones. This typically helps discriminate more accurately the positive and negative data. The learning is driven not only by the proper novelty part in the data ($\Phi_{k-1}x_k$) but also by the data itself ($Ix_k \equiv x_k$). At each presentation of datum x our model can extract new knowledge about that datum and its relationships with respect to the old learnt data. Accordingly, we call this model Incremental data-driven Learning of NDF (ILoNDF).

A better description of the model may be obtained by investigating the behavior of the ILoNDF learning rule in the eigenspace corresponding to its state matrix. It is well known that if $\{\lambda_i\}$ are the eigenvalues of a square matrix X then $\{\lambda_i + \lambda\}$ are the eigenvalues of the matrix $X + \lambda I$; in addition, matrices X and $X + \lambda I$ have a common orthonormal basis of eigenvectors $\{u_i\}$. Therefore, matrices $\{\Phi_{k-1}\}$ and $\{I + \Phi_{k-1}\}$ of Eq. 15 have the same eigenvectors and their corresponding eigenvalues differ by 1. In other words, the introduction of the identity matrix does not change the direction of the eigenvectors of the matrices $\{\Phi_{k-1}\}$ but transforms them to a sequence of positive-definite matrices $\{I + \Phi_{k-1}\}$, with eigenvalues strictly greater than zero, while preserving the relative ordering of the eigenvalues. As a result, eigenvectors that are not orthogonal to the space spanned by the reference data will reside in the space spanned by the model but will stay less significant than other orthogonal eigenvectors. The orthogonality conditions are no longer satisfied between the space spanned by the filter and that of the reference data. Instead, the dimensionality of the space spanned by the model will still be of the same order as the representation space.

As an illustration, Fig. 2 shows the behavior of ILoNDF in the eigenspace corresponding to the state matrix of the model. At the first step ILoNDF acts like NDF. After training on x_1 the state matrix has two eigenvectors: one is orthogonal to the training data x_1 and the other is in the direction of x_1 with a zero eigenvalue. As stated earlier, adding the identity matrix to Φ_1 results in an increase of the eigenvalues by 1, i.e. the amount of information explained by the eigenvector u_1 becomes only two times greater than the amount of information explained by the eigenvector u_2. After training on x_2, the eigenvectors are rotated in different directions with different eigenvalues in order to enhance the overall representation of the various features in the training data. Specifically, u_1 represents 72.4% of the information with the remaining 27.6% being represented by u_2. Now if x_1 (or x_2 or any linear combination of them) is applied again to the model, it will not be seen as a redundant

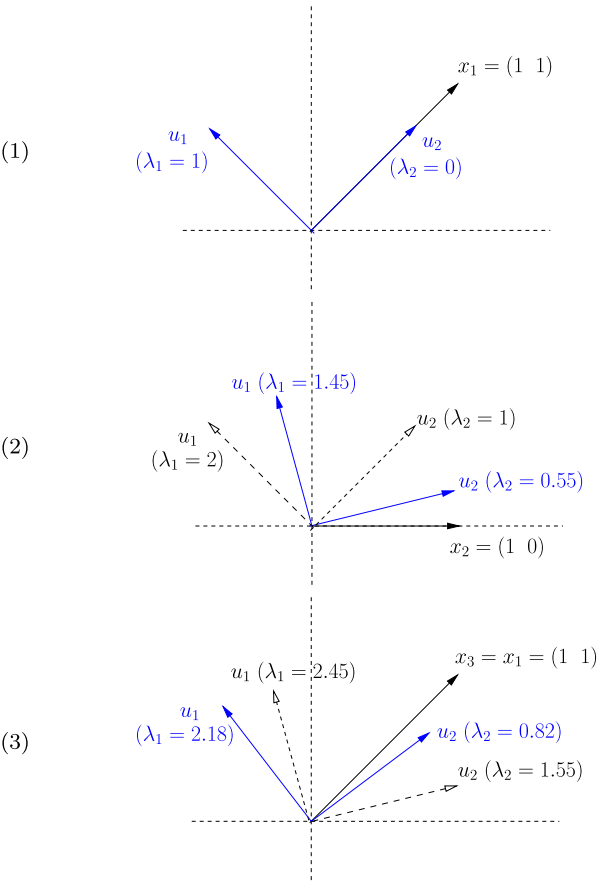

Fig. 2 Example of the behavior of the ILoNDF model in the eigenspace corresponding to the state matrix of the model. Training data are $x_1 = (1\ 1)$, $x_2 = (1\ 0)$ and $x_3 = (1\ 1)$. The eigenvectors are indicated by u_1, u_2 and their corresponding eigenvalues by λ_1, λ_2

information and the model will continuously change its knowledge about the training data. In contrast, with NDF the novelty space spanned by the model will be the null space $\Phi_2 = 0$ after the presentation of x_1 and x_2. Afterwards the presentation of x_1 or any data which can be explained as a linear combination of x_1 and x_2 will not change the acquired knowledge of the model.

Our modification of the original learning rule results in another interesting aspect of ILoNDF: it has the ability to capture co-occurrence relationships between features in the training data. An analysis of the learning process of ILoNDF reveals that, during learning, if a feature i appears with a feature j within a training datum, the value of the ij^{th} element of the filter matrix will decrease. Subsequently, if the feature j appears with a feature k without i within a training datum, the value of the jk^{th} element of the filter matrix will decrease while the values of the ij^{th} and ik^{th} elements of the filter matrix will increase. When learning terminates, the negative value of a matrix element, say Φ_{ij}, will thus reveal the co-occurrence dependency between features i and j. This means that they appear at least one time together, whereas a positive value indicates independence between the two features. Thus we can interpret the resulting matrix Φ_k of ILoNDF as a feature-by-feature co-occurrence dependency matrix. This ability helps ILoNDF identify features that correlate

within the training data, making it more robust with respect to noisy features.[5] Still, it is worth noting that such a relationship between features is captured by the NDF learning rule as well. However, once a feature becomes completely habituated all information about its relation with other features will be lost. This can be verified by observing the internal structure of the state matrices Φ^{NDF} and Φ^{ILoNDF} of the models in Sect. 3.6.

To clarify how the modified rule (Eq. 15) can address the main defects of the original rule, consider the simple cases of Sect. 3.3.

Case 1: By training ILoNDF on Data Set #1 we obtain the transfer function of the model:

$$\Phi_3 = 3 \times \begin{pmatrix} 0.556 & -0.192 & -0.099 \\ -0.192 & 0.657 & -0.127 \\ -0.099 & -0.127 & 0.789 \end{pmatrix}.$$

If we now calculate the habituation proportions of the features and divide them by the number of training data (cf. Sect. 3.5, Eq. 17), we get:

$$H_{f_1} = 0.405 > H_{f_2} = 0.304 > H_{f_3} = 0.195.$$

As can be seen, the presence of f_1 within all the training data are taken into account and accordingly, it is now more habituated than the other features and will be considered more important when representing the training data.

Training ILoNDF on Data Set #2 will not yield the same results. The features will be equally learnt as they have the same frequency in the training data.

Case 2: This case serves to highlight how the ILoNDF model tends to implicitly incorporate some forgetting effects on old or rare data. Figure 3 shows the habituation proportions of the f_1 and f_2 features over training time steps. Unlike with the NDF model, the habituation of the ILoNDF model to both f_1 and f_2 tends to decrease over time.

3.5 Using ILoNDF for classification

Training ILoNDF on a set of positive data leads to the development of the transfer function of the filter under a matrix representation Φ. If a new example is then presented as input, a vector will appear at the output which represents the new features extracted from the input data after taking into account the occurrence frequency of each of the features and their relationships in the training data. Applying the ILoNDF model to one-class classification is straightforward and can be done in either of the following ways.

- *The Direct-Projection Method (DPM)*: The straightforward way is to project each new datum, say x_i, on the filter matrix Φ to generate a novelty vector $\tilde{x}_i = \Phi \cdot x_i$. Two proportions can thus be computed:
  - The "*novelty proportion*" which quantifies the novelty in the data under consideration with respect to the data that were seen during training.

$$N_{x_i} = \frac{\|\tilde{x}_i\|}{n \times \|x_i\|} \tag{16}$$

  where n is the number of positive training data.

[5] This ability of ILoNDF to capture co-occurrence relationships between features can be of benefit in many analysis tasks. We have undertaken a first application in the context of information filtering for analyzing different types of user's needs (Kassab and Lamirel 2007).

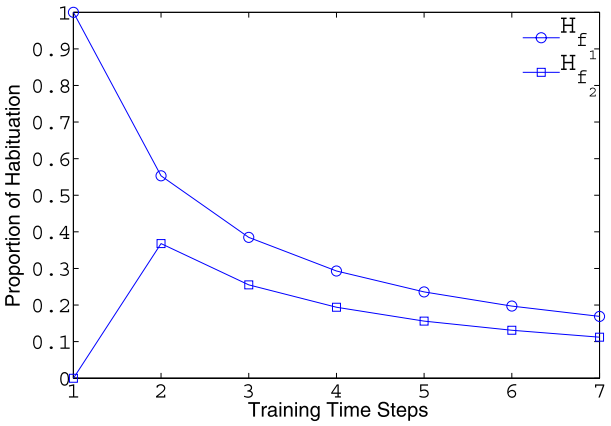

Fig. 3 Forgetting effect of the ILoNDF model. ILoNDF is trained on the sample data #3. Owing to the fact that features f_1 and f_2 do not appear any more in the description of training data after the second time step, their proportions of habituation decrease monotonically over time

– The "*habituation proportion*" which quantifies the redundancy (or similarity) of the data with the previously learnt ones.

$$H_{x_i} = 1 - \frac{\|\tilde{x}_i\|}{n \times \|x_i\|}. \tag{17}$$

This second proportion could be considered a "*classification score*" of the data x_i indicating the likelihood that the example belongs to the positive class: the higher the habituation proportion, the higher the similarity of the data to the positive class.

- *The Vector-Based Projection Method (V-PM)*: It is also possible to create from the matrix representation of the filter (Φ) a representative vector (P_v) of the positive training data. The *classification score* of each new example is then computed by comparison to the representative vector as explained below.

The projection of the unit vector $\overrightarrow{u_f}$—associated with the direction of a feature f in the representation space—on the filter matrix Φ can be used to compute the habituation proportion of this feature as follows:

$$H_f = 1 - \frac{\|\Phi \overrightarrow{u_f}\|}{n \times \|\overrightarrow{u_f}\|}. \tag{18}$$

The representative vector of positive data is represented as a linear combination of the unit vectors associated with the features of the representation space F in a unique way, that is:

$$P_v = \sum_{f \in F} H_f \overrightarrow{u_f}. \tag{19}$$

In other words, the representative vector is defined as a vector of weights corresponding to the habituation proportion of the features in the data representation space F. Finally, the classification score of an example x_i with unknown class is computed using the cosine similarity between the representative vector and the example (Salton 1971):

$$\mathrm{Cos}(x_i, P_v) = \frac{x_i \centerdot P_v}{\|x_i\| \, \|P_v\|} \tag{20}$$

where $x_i \centerdot P_v$ denotes the dot product of these two vectors.

The two methods above behave differently and can have a significant impact on classification performance. In general, DPM is a specific data-matching procedure which determines the novelty of a given example with respect to the training data (with the NDF model, the matching is performed between the example under consideration and the closest training data. In contrast, the ILoNDF model considers the co-occurrence of features and their relationships within all the training data when performing the matching process. See the example below for clarification). In contrast, V-PM is a feature-matching procedure which takes advantage of the diversity of the habituation values of the learnt features to distinguish between positive and negative data. It will favor data containing a greater number of features with high habituation values. In our previous work we found V-PM to be more accurate than DPM, especially when only a few positive data are available (Kassab and Lamirel 2006, 2007). Nevertheless, if all the features have approximately the same proportion of habituation (often owing to the lack of a correct representation of the training data, e.g. using insufficient or inappropriate features) the quality of the representative vector of the training data can be very poor. In this case, the values of the vector components would have low variance. We can view the variance of the vector components as indicative of its usefulness for classification.

To avoid such a situation, a weighted combination of the classification scores according to both DPM and V-PM into a global score *CS* may be calculated as follows:

$$CS(x_i) = (1 - \lambda) H_{x_i} + \lambda \operatorname{Cos}(x_i, P_v) \tag{21}$$

with

$$\lambda = \frac{standard\ deviation\ of\ P_v}{max\ value - min\ value\ of\ the\ P_v\ components}.$$

The pseudocode for the ILoNDF learning method is shown in Algorithm 1.

3.6 An illustration

For clarity, we present an example showing the main defect of the original learning rule of the NDF method (Eq. 11) when applied to the classification task, which motivates our modification of this rule. The example also illustrates the impact of both DPM and V-PM on the classification solution of both NDF and ILoNDF models.

Let us consider the data presented in Table 1 and suppose that the first three data items d_1, d_2 and d_3 are selected for training the NDF and ILoNDF models.

After training, we obtain the transfer function of the NDF model:

$$\Phi^{\mathrm{NDF}} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0.6 & -0.2 & -0.4 & -0.2 \\ 0 & -0.2 & 0.4 & -0.2 & 0.4 \\ 0 & -0.4 & -0.2 & 0.6 & -0.2 \\ 0 & -0.2 & 0.4 & -0.2 & 0.4 \end{pmatrix}$$

and the transfer function of the ILoNDF model:

$$\Phi^{\mathrm{ILoNDF}} = 3 \times \begin{pmatrix} 0.792 & -0.089 & 0.018 & -0.089 & -0.107 \\ -0.089 & 0.769 & -0.152 & -0.231 & -0.079 \\ 0.018 & -0.152 & 0.798 & -0.152 & 0.050 \\ -0.089 & -0.231 & -0.152 & 0.769 & -0.079 \\ -0.107 & -0.079 & 0.050 & -0.079 & 0.871 \end{pmatrix}.$$

Algorithm 1 LearnILoNDF(X, F, Φ_0)

Inputs:

$X = \{x_1, \ldots, x_n\}$ a set of positive training data;
$F = \{f_1, \ldots, f_m\}$ a set of features used for representing the training data;
Φ_0 the initial matrix of the ILoNDF model (default: a zero matrix).

Outputs:

Φ_n the final matrix of the ILoNDF model;
P_v the representative vector of target class.

Initialization:

$P_v^0 = 0$ {a zero, or null, vector}.

begin
for each $x_k \in X$ **do**

$$\tilde{x}_k = (I + \Phi_{k-1})x_k$$

$$\Phi_k = I + \Phi_{k-1} - \frac{\tilde{x}_k \tilde{x}_k^T}{\|\tilde{x}_k\|^2}$$

end for
for each $f_i \in F$ **do**

$$H_{f_i} = 1 - \frac{\|\Phi_n \overrightarrow{u_{f_i}}\|}{n \times \|\overrightarrow{u_{f_i}}\|}$$

$$P_v^{(i)} = P_v^{(i-1)} + H_f \overrightarrow{u_f}$$

end for
end

Table 1 Sample feature-by-data matrix

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8
f_1	1	0	1	0	1	0	0	1
f_2	1	1	1	0	1	1	0	1
f_3	1	1	0	1	0	0	1	1
f_4	1	1	1	0	0	0	0	1
f_5	0	0	1	0	0	1	1	1

On the basis of these matrices, the representative vectors of training data with respect to the NDF and ILoNDF models can be constructed using Eq. 19. They are respectively:

$$P_v^{\mathrm{NDF}} = \begin{pmatrix} 1 & 0.225 & 0.368 & 0.225 & 0.368 \end{pmatrix}^T,$$

$$P_v^{\mathrm{ILoNDF}} = \begin{pmatrix} 0.191 & 0.174 & 0.172 & 0.174 & 0.114 \end{pmatrix}^T.$$

Recall that the value of each component in the representative vector corresponds to the habituation proportion of one feature in the representation space. As can be seen from P_v^{NDF}, f_1 is much more habituated than the others, and accordingly it will be considered more relevant to the description of training data. Moreover, f_5, which is less frequent in the training data, is more habituated than other more frequent features such as f_4. In contrast, applying

Table 2 Ranking and Classification scores according to the DPM and V-PM methods (Comparison between NDF and ILoNDF)

	DPM		V-PM		CS	
	Doc.	Score	Doc.	Score	Doc.	Score
NDF	d_1, d_2, d_3	1	d_8	0.835	d_1, d_3	0.907
	d_8	0.717	d_1, d_3	0.776	d_8	0.766
	d_5, d_6	0.452	d_5	0.740	d_2	0.752
	d_4	0.368	d_7	0.444	d_5	0.572
	d_7	0.106	d_2	0.403	d_6	0.413
			d_6	0.358	d_4	0.345
			d_4	0.314	d_7	0.246
ILoNDF	d_2	0.560	d_8	0.987	d_8	0.704
	d_1	0.530	d_1	0.952	d_1	0.692
	d_8	0.527	d_3	0.874	d_2	0.653
	d_3	0.511	d_2	0.804	d_3	0.650
	d_5	0.255	d_5	0.691	d_5	0.422
	d_6	0.209	d_6	0.544	d_6	0.338
	d_4	0.172	d_7	0.541	d_4	0.282
	d_7	0.083	d_4	0.460	d_7	0.258

the modified learning rule of Eq. 15 reduces the gap between the habituation of f_1 and the other features (f_2, f_3 and f_4) while reducing the habituation of f_5.

Now we can calculate the classification scores of the data in Table 1 (using DPM, V-PM or CS) and rank them in terms of their similarity to the training data. Table 2 shows the ranking of the data by different methods in decreasing order of similarity to the training data.

From Table 2 we may see that only NDF with the direct projection strategy was perfectly habituated to the training data (d_1, d_2, d_3). This behavior is consistent with the design of the model and the bit-wise matching of the DPM strategy. Note also that the NDF model is unable to distinguish well between features that are correlated with the description of training data (the relevant features) and those that are not (irrelevant features). This situation is especially obvious in the case of d_5 and d_6, which are assigned the same score, whereas d_5 is more similar to the training data. In contrast, when looking at the ranking performed by the ILoNDF model with the direct projection strategy, we find that the model is not totally habituated to the training data and that the matching is performed against all the training data and not just the closest training data. This is beneficial for distinguishing between relevant and irrelevant features, and accordingly, for distinguishing between relevant and irrelevant data. As can be seen from Table 2, d_6 was assigned a lower similarity score than d_5 by the ILoNDF model.

If we compare DPM with V-PM when using the NDF model, we find that V-PM could solve some discrimination problems of NDF with the DPM strategy (when d_5 is classified as more similar to the training data than d_6) but other discrimination problems arise; for example, d_2 has a lower similarity score than d_5 or d_7. Again, this is because the quality of the representative vector created from the trained NDF model is fairly poor. The weighted combination of the two methods is achieved with $\lambda = 0.416$ which slightly favors DPM over V-PM. The combination method (CS) performs better but the results are still far from

satisfactory. With the ILoNDF model, the application of V-PM changes the ranking order of the first four data (d_1, d_2, d_3 and d_8) such that the greater the number of relevant features, the higher the classification score; but the significant change in the ranking concerns d_4 and d_7. Note that if f_5 were a noisy feature it would be better to use the DPM strategy, otherwise V-PM will be preferable. As the variance of the representation vector of the ILoNDF model is rather low, the combination method will favor DPM with $\lambda = 0.383$ in order to prevent likely problems.

3.7 Setting the threshold

Up to this point, classification scores (cf. Eq. 17, Eq. 20 and Eq. 21) were only used to rank data by similarity to the positive class. To make a decision boundary discriminating between the positive and negative data, a threshold should be set on the classification scores: if the classification score of a datum is less than a specified threshold then the datum is classified as negative, otherwise it is classified as positive.

As with many machine learning approaches, the problem of setting an appropriate threshold to distinguish between two classes using data from just one is nontrivial. Indeed, most of the adapted algorithms provide a user-defined parametric threshold like the one-class SVM approach (Schölkopf et al. 2001). Thus, the decision threshold must be provided by the user which may be neither desirable nor easy. Here we propose a simple thresholding strategy which can be applied to any on-line learning approach. In developing our thresholding strategy, we consider the following two aspects:

1. The classification scores corresponding to the data after being used in the training process can be used as a potentially good indicator of classification scores of data which tend to be positive and which are easy to be detected owing to the fact that they are strongly similar to the data used for training to model the positive class. Consequently, the mean of these scores can be accepted as an upper bound for the decision threshold of the classification.
2. The classification scores corresponding to the data before being used in the training process can be used as a potentially good indicator of classification scores of data which tend to be positive but which are not easy to be detected owing to the fact that they are not strongly similar to the data used for training to model the positive class. Consequently, the mean of these scores can be accepted as an lower bound for the decision threshold of the classification.

So, at each learning step, we calculate these two mean values over training data and a combination value is obtained as a weighted mean of the two values, one being weighted by the number of trained data, other being weighted by the number of untrained data. Using this weighting is important to quantify reliability of each of the mean values computed across learning steps. The final threshold is a percentage ratio τ of the weighted mean obtained over the learning steps (see Algorithm 2).

4 Multivariate statistical-based approaches

Principal Component Analysis (PCA) is one of the most widely used multivariate techniques and it has found application in a wide variety of areas (Jolliffe 1986). The PCA model is a transformation in which the data are represented by orthogonal features, called the principal components, which are linear combinations of the original features. The principal components (PCs) are extracted so that the first PC accounts for the maximum variation of the

Algorithm 2 ThresholdSetting(X, T)

Inputs:

$X = \{x_1, \ldots, x_n\}$ a set of positive training data;
$F = \{f_1, \ldots, f_m\}$ a set of features used for representing the data.

Output:

s the threshold value.

Initialization:

$\Phi_0 = 0$ a zero matrix; $subLX_0 = \{\}$; $subNLX_0 = X$.
$s = 0$; $nb\text{Step} = 0$;

begin
  for $i = 0$ to n **do**
    if $((i \ \% \ \frac{n}{10}) \ != 0)$ **then**
      $subLX_i = subX_{i-1} \cup x_i$
      $subNLX_i = subNLX_{i-1} \setminus x_i$
    else
      $(\Phi_i, P_i) = LearnILoNDF(subLX_i, T, \Phi_{i-1})$
      $s_1 = MeanClassificationScores(subLX_i, P_i)$
      $s_2 = MeanClassificationScores(subNLX_i, P_i)$
      $s = s + (|subLX_i| \times s_1 + |subNLX_i| \times s_2)/n$
      $nb\text{Step} = nb\text{Step} + 1$
    end if
  end for
  $s = \tau \frac{s}{nbStep}$
end

Note:

MeanClassificationScores(X, P) returns the mean of classification scores of data in X using one of the formulas Eq. 17, Eq. 20 or Eq. 21.

data and subsequent orthogonal PCs account for progressively smaller amounts of residual variation. Typically, PCs are found by extracting the eigenvectors and eigenvalues of the covariance matrix of the data. The eigenvectors denote the direction of the principal components and the eigenvalues indicate the variance accounted for by the corresponding principal component. It is often the case that a small number of the first extracted principal components capture the majority of the total variation and are sufficient to replace the original features without major loss of information. Accordingly, only k components with the largest variances (that is, the eigenvectors with the largest associated eigenvalues) are preserved when constructing the target PCA model (there are a number of different ways to establish the dimensionality of the PCA model subspace (k), such as the average eigenvalue approach which is adopted in the present investigation, Valle et al. 1999). In the following we present a brief review of two multivariate statistical indexes relying on the use of PCA and their application to one-class classification.

4.1 PCA residuals

PCA may be viewed as a subspace decomposition in which the feature space of the positive data is divided into two orthonormal subspaces, the PCA model and residual subspaces. Let $X \in \Re^{n \times p}$ be a sample of positive training data with n being the number of examples

and p the number of features. After PCA transformation the PCA model subspace, which is spanned by the first k principal components (P_k), is associated with systematic data variations in agreement with the feature correlations. In contrast, the residual subspace, which is spanned by the remaining n-k principal components, is associated with random variations due to errors or noise in the positive data. Therefore, a new datum x_i can be decomposed as

$$x_i = \hat{x}_i + \tilde{x}_i$$

with $\hat{x} = P_k P_k^T x$ and $\tilde{x} = (I - P_k P_k^T)x$ being the projections of the datum x_i onto the model and residual subspaces, respectively. The "goodness of fit" of the data to the positive class can then be assessed by Squared Prediction Error (SPE) of the residual. This statistic is also called the Q-statistic index and is defined as:

$$\text{SPE} = \|\tilde{x}_i\|^2 = x_i^T \, (I - P_k P_k^T) \, x_i .$$

A negative datum is predicted when SPE exceeds a Q-statistic threshold (Jackson and Mudholkar 1979).

4.2 Hotelling's T^2 test

While the analysis of PCA residuals offers a way to test if a datum shifts outside the model space, Hotelling's T^2 statistic provides an indication of novel variability within the model space. Having established a PCA model of the positive training data, Hotelling's T^2 statistic can be computed based on the first k principal components of the model (Kim and Beale 2002; Detroja et al. 2007). Let x_i be a $1 \times n$ data vector auto-centered by the mean of the positive training data. The T^2 statistic for the data is:

$$T_i^2 = t_i^T \Lambda^{-1} t_i = x_i^T P_k \Lambda^{-1} P_k^T x_i$$

where $t_i = P_k^T x_i$ is the orthogonal projection of the data x_i into the model subspace defined by the k first principal components, and Λ is a diagonal matrix containing the first k eigenvalues of the covariance matrix of the positive training data.

A large value of T^2 indicates a large deviation of the data under consideration from the positive class. A threshold (T_α^2) can be obtained using the F-distribution (Detroja et al. 2007).

5 Auto-associative neural networks

Auto-associative neural networks (AANNs), also known as auto-encoders or "bottleneck" neural networks, are a special kind of feed-forward multilayer perceptron network. Originally introduced by Rumelhart et al. (1986), AANNs are trained to produce an approximation of the identity mapping between the input and the output of the network. In its conventional form, an auto-associative neural network consists of an input layer of neuron units, followed by one or more hidden layers, and an output layer of the same dimension as the input layer (see Fig. 4). One of the hidden layers, referred to as the "bottleneck" layer, should be of smaller dimension than either input or output (or other hidden layers if any). This restriction is required to inhibit the network from memorizing the data while

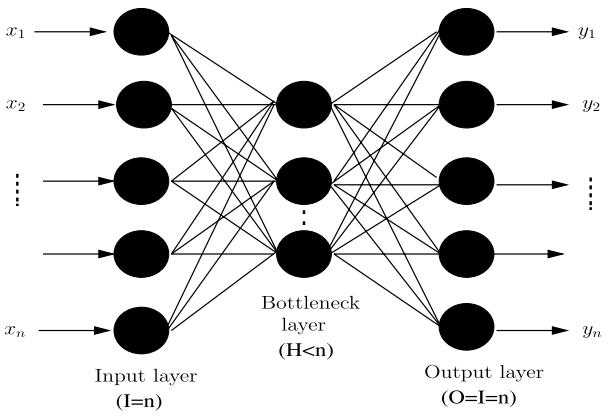

Fig. 4 A schematic presentation of a three-layer auto-associative neural network

capturing the most significant features, similar to principal components, of the input data by compressing their redundancies.[6,7]

The principle of applying AANNs for solving one class classification problems involves two processes: training the network and identifying a suitable decision threshold for it. The goal of training is to properly adjust the connection weights of the network under a supervised learning scheme in such a way that the network is able to reconstruct in its output layer the positive data presented as input. Back propagation (Baldi et al. 1995) is the most widely employed learning algorithm to minimize the mean-squared error (MSE) between the input X and the output Y vectors:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N} \|X - Y\|^2$$

where N is the number of positive training data.

After training, classification may be performed with the AANN under the assumption that positive data will accurately be reconstructed while negative data will not. However, finding an accurate boundary (decision threshold) between the positive and negative data is problematic. In our preliminary experiments, we adopted the practice of Japkowicz et al. (1995), whose method considers the upper-bound of the reconstruction error of the positive training data at each training epoch and relaxes it by a fixed percentage (e.g. 25%). New data

[6]Typically, AANNs are most known as compression networks with a bottleneck layer of smaller dimension than the input layer. Even though, the design of AANN with no bottleneck, that is, with hidden layer of higher dimension than the input layer, albeit feasible (Japkowicz et al. 2000), can be applied in the case of one-class classification. Experimentally, we have investigated this possibility but found that non-bottleneck AANNs are not inherently better than bottleneck AANNs. Often, the non-bottleneck AANNs yielded slightly worse results. Furthermore, they would not be an efficient design because of the increased computational requirements posed by larger numbers of hidden neurons.

[7]The mapping of the input space can be either linear or non-linear depending on the architecture of the network and the type of activation functions of the output layer. When used with linear activation functions, an AANN will perform a type of principal component analysis. If non-linear functions are introduced, the AANN network will be able to solve problems where the principal component analysis is degenerate (Japkowicz et al. 2000; Cottrell and Munro 1988). Nonlinearity can also be achieved by adding additional hidden layers with non-linear functions between the bottleneck layer and the input and output layers (Oja 1991). However, the training of such networks is more difficult and requires more computational time (Kambhaltla and Leen 1994).

are then classified by comparing their reconstruction error to that of the relaxed boundary. If the error exceeds the boundary in at least a certain proportion of the epochs considered, a datum is classified as negative, else it is classified positive. However, we found the precision of this technique to be very low, so we decided to use a somewhat different strategy consisting of computing the mean of the reconstruction error of the training data at each of the k first epochs (in our experiments, k is set to two times the number of the best epoch corresponding to the lowest validation error). The average of the mean values over the k epochs is then used as the decision threshold.

For the experiments reported in this paper, we developed and trained a three-layer auto-associative neural network using a standard back-propagation algorithm. The number of neurons in the input and output layers ($I = O$) was determined by the dimensionality of the input data, while we experimented with various values for the number of units in the hidden layer (H) . We considered the proportion of hidden units (H) to input units (I) and tested $\frac{H}{I}$ at 10%, 25%, 50% and 75%. The learning rate and the momentum factor of the back-propagation learning algorithm were fixed at 0.1 and 0.9, respectively.

6 One-class support vector machines

Support Vector Machines (SVM) were introduced by Vapnik (1995) as an effective learning algorithm that operates by finding an optimal hyperplane to separate the two classes of training data. The basic idea behind SVM is to "non-linearly" map the input space of the training data into a higher dimensional feature space corresponding to an appropriate kernel function. SVM then finds a linear separating hyperplane in the feature space by maximizing the distance, or margin, to the separating hyperplane of the closest training data from the two classes (cf. Fig. 5(a)).

An extension of SVM (1-SVM) was subsequently proposed by Schölkopf et al. (2001) to handle one-class classification. The strategy is to separate the training data (positive examples) from the origin (considered as negative examples) with maximum margin in the feature space. Under this strategy, a bounding hypersphere is computed around the training data in a way that captures most of these data while minimizing its volume (see Fig. 5(b)). The algorithm can be briefly described as follows. Given a training data set of l positive examples $\{x_1, \ldots, x_l\}$, $x_i \in \Re^n$ and $\Phi : X \rightarrow F$ a feature map which maps the given data X

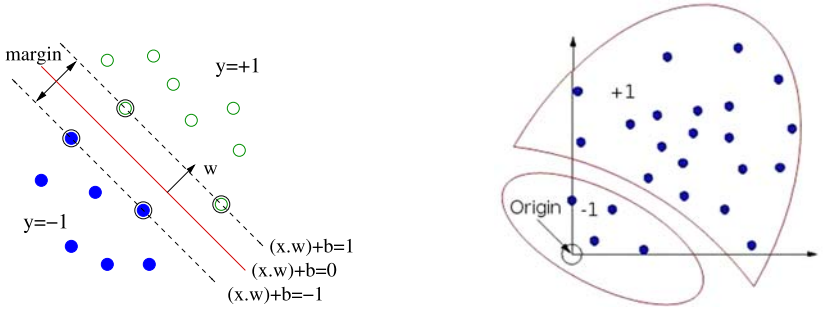

Fig. 5 Geometry interpretation of SVM-based classifiers: (**a**) Two-class SVM classifier. (**b**) One-class SVM classifier

into the feature space F, a one-class SVM requires solving the quadratic problem:

$$\begin{aligned} &\min_{w,\xi,\rho} \frac{1}{2} w.w + \frac{1}{\nu l} \sum_{i=1}^{l} \xi_i - \rho \\ &\text{subject to } (w.\Phi(x_i)) \geq \rho - \xi_i, \quad \xi_i \geq 0,\ i = 1, \ldots, l, \end{aligned} \tag{22}$$

where ξ_i are slack variables that allow some data to be misclassified and $\nu \in [0, 1]$ is a free parameter that controls the impact of the slack variables, i.e. the fraction of training data which are allowed to fall outside the hypersphere (when $\nu = 0$, it becomes the hard margin case).

Introducing Lagrange multipliers α_i and applying the Karush-Kuhn-Tucker conditions (Fletcher 1987) for the primal problem above, we obtain:

$$w = \sum_i \alpha_i \Phi(x_i). \tag{23}$$

We can now solve the primal quadratic problem by solving its dual problem, given by:

$$\begin{aligned} &\min_{\alpha} \frac{1}{2} \sum_{ij} \alpha_i \alpha_j K(x_i, x_j) \\ &\text{subject to } 0 \leq \alpha_i \leq \frac{1}{\nu l}, \quad \sum_i \alpha_i = 1 \end{aligned} \tag{24}$$

where $K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$ is the kernel function corresponding to the dot product of the two data in the feature space. The data x_i with non-zero α_i are the so-called support vectors.

Once the optimal values of the parameters are found, one can classify the new data according to the following decision function: $f(x) = \text{sgn}(\sum_i \alpha_i K(x_i, x) - \rho)$ such that the data corresponding to $f(x) \geq 0$ lie within the hypersphere and are classified as positive data. Otherwise, they lie outside the hypersphere and they are classified as outliers or negative data.

One-class SVM has the same advantages as two-class SVM classification, such as efficient handling of high-dimensional and sparse feature spaces, and handling of non-linear classification using kernel functions. However, one-class SVM requires many more positive training data to give an accurate decision boundary because its support vectors come only from the positive data. In our study, we used a free SVM software package called mySVM (Rüping 2000) for 1-SVM training and classification experiments.

7 Experimental settings

7.1 Data collections

The empirical evaluations are carried out over two test corpora: Reuters-21578[8] and WebKB.[9] These corpora are publicly available and have been widely used for text categorization tasks.

[8] http://www.daviddlewis.com/resources/testcollections/reuters21578/.

[9] http://www.cs.cmu.edu/webkb/.

Table 3 The number of training and testing documents of the Reuters and WebKB corpora

Reuters			WebKB		
Category	Train	Test	Category	Train	Test
corn	181	56	staff	116	21
ship	197	89	depart	181	1
wheat	212	71	project	484	20
trade	369	117	course	886	44
interest	347	131	faculty	1089	34
crude	389	189	student	1513	128
grain	433	149			
money	538	179			
acq	1650	719			
earn	2877	1087			

• *Reuters-21578.* The Reuters-21578 newswire collection (distribution 1.0) is a multi-label dataset widely used in text categorization research (Debole and Sebastiani 2005; Dumais et al. 1998). We followed the ModApte split which consists of 9603 training documents and 3299 test documents classified into 135 topic categories. The distribution of the documents among the categories is highly skewed; the top 10 most common categories contain 75% of the training documents and the remaining documents are distributed among the other categories. Our experimental results are reported for the set of the 10 most common categories (out of the total of 135). For these 10 categories, the number of training documents per category also varies widely, from 181 to 2877 with an average of 719.3 documents per category.

• *WebKB.* The WebKB collection contains HTML pages gathered from computer science departments of various universities. There are about 8280 web pages which are divided into seven categories: student, faculty, staff, department, course, project, and other. In our experiments, we ignore the category "other" due to its unclear definition. The number of training documents per category varies from 116 to 1090 with an average of 711.7 documents per category. The WebKB collection is a single-label dataset, however, it is a relatively difficult dataset since its words have been found to be very scattered over categories making these later very closely related to each other and thus hard to discriminate. Table 3 gives the number of training and testing documents in each of the categories of the Reuters and WebKB corpora.

7.2 Document representation

We performed standard text preprocessing steps: document parsing, tokenization, stop word removal and stemming. Only single words were used for content representation. The informative word terms were selected according to document frequency (DF), such that only terms that appeared in the highest number of documents are retained. In order to explore the influence of the dimensionality of representation space (i.e. feature set size) on classification performance, we experimented with four document frequency thresholds: (T10) where only the 10 more frequent terms are retained, (T20) where only the 20 more frequent terms are retained, ($>10\%$) where only the terms that appear in at least 10% of the training documents are retained, and ($>5\%$) where only the terms that appear in at least 5% of the training documents are retained and considered for use in document representation. The retained terms

are used for representing both training and testing documents by vectors of term-weights. Several versions of term weighting schemes were implemented:

- Normalized Term Frequency (NTF). Term Frequency (TF) is the number of times a term appears in a given document. Cosine normalization, where each term weight is divided by the Euclidean document length, is afterwards used to remove the advantage of long documents over the short ones (Salton and Buckley 1988).
- Normalized $\log(\mathrm{TF}+1)$. A slight variation of NTF is to take the logarithm of term frequency thereby reducing the gap between large differences in term frequencies.
- Augmented $\mathrm{NTF} = (0.5 + 0.5 * \frac{\mathrm{TF}}{\max \mathrm{TF}})$ where $\max \mathrm{TF}$ is the maximum TF for all terms in the given document.
- Binary. This weighting scheme assigns the weight 1 to a given term if it appears at least once in the given document, else 0.

It is worth noting that both term selection and term weighting should be category-specific processes because only information of the category under study is supposed to be available in the context of one-class classification (e.g. Manevitz and Yousef 2001). Therefore, different sets of terms were separately selected for each category using only the training documents belonging to that category. Training documents of each category and the test documents were then represented using the specific set of terms.

7.3 Evaluation metrics

Performance was evaluated mainly with the standard *Precision* and *Recall* metrics. *Precision* quantifies the percentage of data predicted to be positive by the classifier that are actually positive; *Recall* quantifies the percentage of positive data predicted to be positive by that classifier. Firstly, we evaluated the effectiveness of classifiers for ranking data from high to low expectations of being positive. This is motivated by the need to assess the absolute learning performance of the classifiers independent of any specific threshold. Furthermore, some applications require the data to be ranked instead of (or in addition to) hard classification labels (in medical applications this is common). To this end, the *Recall-Precision (R-P)* curve is suitable for evaluating classifiers by integrating their performance over a range of decision thresholds. It depicts the relation between recall (x-axis) and precision (y-axis) varying the range of thresholds corresponding to the 11 standard levels of recall ($0\%, 10\%, \ldots, 100\%$) (Salton 1971). The lower the misclassification error of a classifier, the closer the corresponding point is to the upper right-hand corner of the R-P curve. The Mean Average Precision *MAP* measure, which is roughly the area under the R-P curve, is also used in this investigation.

Though the effectiveness of classifiers is closely related to their ranking effectiveness, this relationship is not strong enough to use only ranking measures to evaluate classification models. Measures that incorporate the choice of decision thresholds are necessary to assess the thresholding strategies and then to evaluate classification performance in realistic conditions. To this end, we used the F_1 measure, which reflects the balanced harmonic mean of *Precision* and *Recall* for certain decision thresholds (van Rijsbergen 1979). The overall performance results are averaged across categories by means of macro and micro averaging scores (Salton 1971; Lewis 1991). Macro averaging is the standard average of the individual values of a given measure which are obtained for each of the categories, so each category has the same weight regardless of how many documents belong to it. In contrast, micro averaging gives equal weight to every document by merging the decisions of individual classes and globally computing the value of a given measure. In general, micro

averaging scores tend to be dominated by the classifier's performance on common categories, while macro averaging tend to be biased towards the classifier's performance on rare categories.

8 Results and discussion

In this section we present experimental results of classification from both corpora under study. First, we are interested in analyzing the methods for robustness with regard to different initial settings. For this purpose, four different dimensions for the term representation space and four variants of weighting schemes were implemented for each of the corpora (Reuters and WebKB, cf. Sect. 7.2). The robustness and flexibility of the studied methods were also investigated under various parameter settings. Next, we focus on comparing the performance of the ILoNDF model with that of the best methods identified. Finally, we give a short summary of the main findings emanating from the results.

8.1 Results for different settings

8.1.1 NDF vs. ILoNDF

The first series of experiments studied the behavior of the NDF and ILoNDF methods. Tables 4 and 5 summarize the MAP scores under different weighting schemes and various dimensions of the representation space. Figures 6 and 8 depict the 11-point Recall-Precision curves of the methods according to various dimensions of the representation space on the Reuters and WebKB corpora. Figures 7 and 9 depict the 11-point Recall-Precision curves of the methods according to various weighting schemes. The results are reported when the classification scores are computed with respect to the direct-projection method (DPM), the

Table 4 Summary of the MAP scores of the NDF model on the Reuters and WebKB corpora under different weighting schemes and various dimensions of the representation space. Bold entries denote the statistically highest scores among the different settings while underlined entries indicate the performance scores at the reference settings supported by the overall results

		Reuters				WebKB			
		T10	T20	>10%	>5%	T10	T20	>10%	>5%
DPM	NTF	0.1634	0.1861	0.2105	0.2891	0.2382	0.23	0.1937	0.2650
	logNTF	0.2008	0.2088	0.2023	0.3312	0.2258	0.2154	0.2066	0.3681
	ANTF	0.2561	0.2383	0.2110	0.3277	0.2073	0.2168	0.1941	0.2792
	binary	0.1850	0.2112	0.2015	0.3117	0.2391	0.2165	0.1798	0.2646
V-PM	NTF	0.4798	0.4833	0.3646	0.3222	0.3238	0.3135	0.2809	0.4038
	logNTF	0.5064	0.4781	0.3233	0.2710	0.4010	0.4802	0.32	0.3264
	ANTF	0.5012	0.4636	0.3109	0.2556	**0.5261**	0.4859	0.3031	0.2831
	binary	0.4791	0.4490	0.3101	0.2525	0.5244	0.3972	0.3291	0.2770
CS	NTF	0.4811	0.4838	0.3645	0.3556	0.3234	0.3136	0.2802	0.2891
	logNTF	**0.5068**	0.4779	0.3233	0.3529	0.4009	0.4804	0.3516	0.4055
	ANTF	0.5006	0.4635	0.3110	0.3512	0.5257	0.4863	0.2897	0.4075
	binary	0.4770	0.4532	0.3099	0.3302	0.5208	0.4091	0.2516	0.4115

Table 5 Summary of the MAP scores of the ILoNDF model on the Reuters and WebKB corpora under different weighting schemes and various dimensions of the representation space. Bold entries denote the statistically highest scores among the different settings while underlined entries indicate the performance scores at the reference settings supported by the overall results

		Reuters				WebKB			
		T10	T20	>10%	>5%	T10	T20	>10%	>5%
DPM	NTF	0.4390	0.4448	0.4770	0.5254	0.2268	0.2428	0.3378	0.3443
	logNTF	0.5130	0.5226	0.6001	0.6699	0.2882	0.3645	0.4814	0.5038
	ANTF	0.5584	0.5560	0.6236	0.6857	0.3592	0.4796	0.5094	0.5102
	binary	0.5755	0.5494	0.6044	0.6496	0.5221	0.5077	0.5117	0.5014
V-PM	NTF	0.5322	0.5294	0.5308	0.5492	0.2495	0.3262	0.3702	0.3901
	logNTF	0.6161	0.6267	0.6540	0.6748	0.3720	0.4414	0.3886	0.4903
	ANTF	0.6125	0.6434	0.6773	0.6937	0.4964	0.3836	0.4865	0.4969
	binary	0.6133	0.6189	0.6451	0.6556	0.5167	0.4040	0.4929	0.4901
CS	NTF	0.4903	0.4883	0.5032	0.5376	0.2466	0.2667	0.3528	0.3568
	logNTF	0.5765	0.5820	0.6318	0.6762	0.3168	0.4512	0.4813	0.5067
	ANTF	0.6081	0.6117	0.6544	**0.6962**	0.4963	0.4767	0.5051	0.5118
	binary	0.6107	0.6107	0.6380	0.6599	**0.5285**	0.4170	0.4970	0.4982

vector-based projection method (V-PM) and the weighted combination of the DPM and V-PM scores (CS).

From the results, we observe that the DPM method is not suitable for the NDF model in the context of classification because of the bit-wise strategy which is based on comparing the current input data with just its closest match in the training data. Often the model works substantially better when the V-PM method is applied. A further improvement is achieved by applying the combination method (CS), particularly for the >5% dimensionality of the documents. With ILoNDF, the difference between DPM and V-PM is within a relatively narrow range. On the Reuters corpus, V-PM yields significantly better results than DPM but performs worse on the WebKB corpus. The explanation has to do again with the fact that the V-PM method favors data containing high proportions of features appearing in the training data. If the representation space is not well formed (i.e., there are many noisy features and few relevant features), V-PM should have poor performance. As can be seen from Table 6, the terms in the WebKB corpus are quite scattered. Statistically, on the WebKB corpus the dispersion of terms over categories is roughly twice as high as that found on the Reuters corpus. Consequently, there are many fewer specific terms for each category and many non discriminating (noisy) terms in the WebKB corpus. This is especially clear when comparing the T10 and T20 dimensionality on the WebKB corpus. The effect of noisy terms may be partially reduced by certain weighting schemes which consider the term frequencies within the documents, but the performance is still far from satisfactory. The CS method presents a reasonable solution in such a situation.

On the basis of the above findings, the focus of the following discussion will only cover the behavior of the NDF/ILoNDF models with the CS method.

When comparing the impact of the different initial settings, we observe that weighting schemes have almost the same impact on both models (cf. Figs. 7 and 9). In contrast with two-class classification, term frequency (NTF) produces the worst results with one-class classification. Better performance is obtained by logNTF but it is still lower than that of other

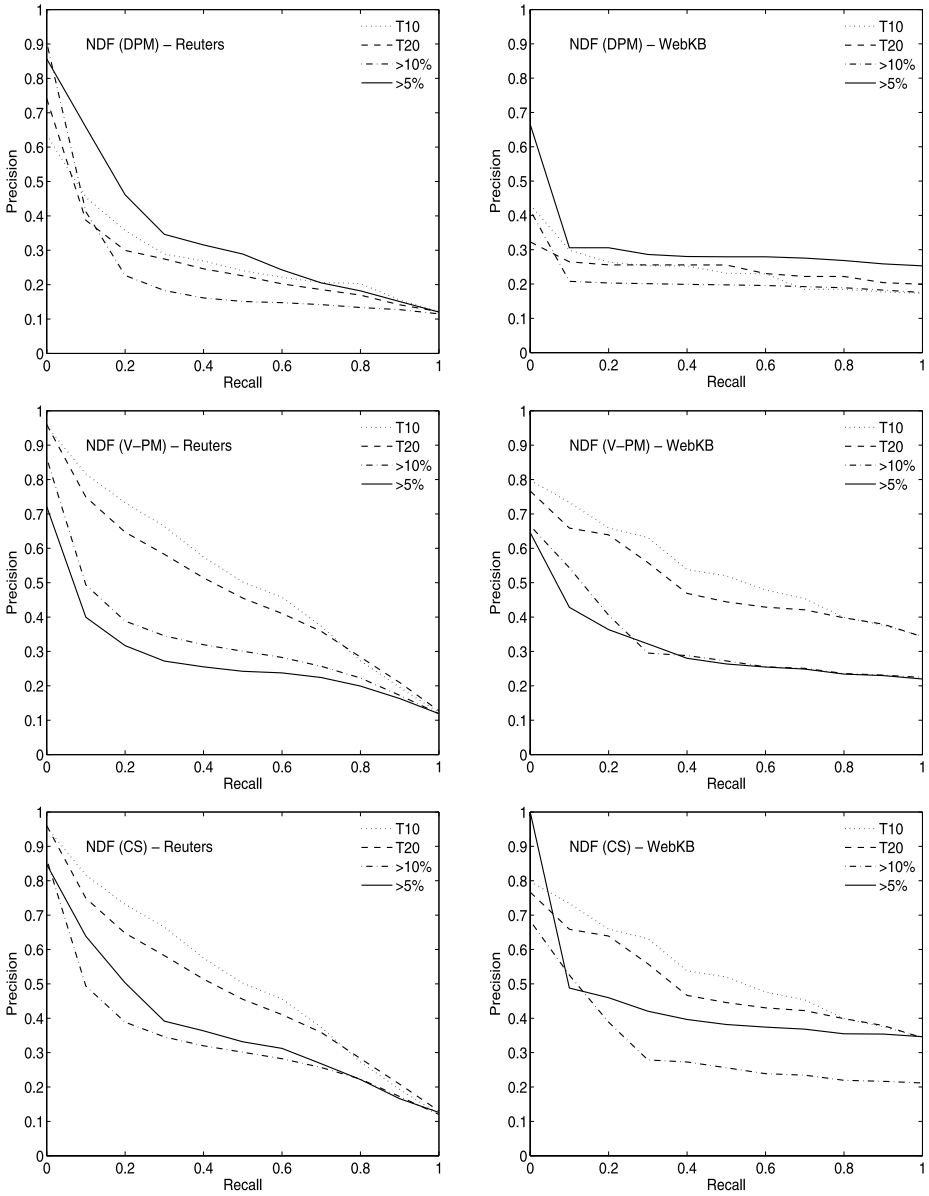

Fig. 6 The 11-point Recall-Precision curves of the NDF model according to various dimensions of the representation space on the Reuters and WebKB corpora. Results are reported according to the ANTF weighting scheme

weighting schemes. The ANTF weighting scheme has not yet been evaluated for one-class classification. In this study, ANTF turns out to perform better than the binary representation in almost all situations; accordingly, the ANTF representation was chosen to be the reference weighting scheme for the NDF and ILoNDF models. Figures 6 and 8 show that the models behave differently with respect to the dimensionality of the representation space. ILoNDF

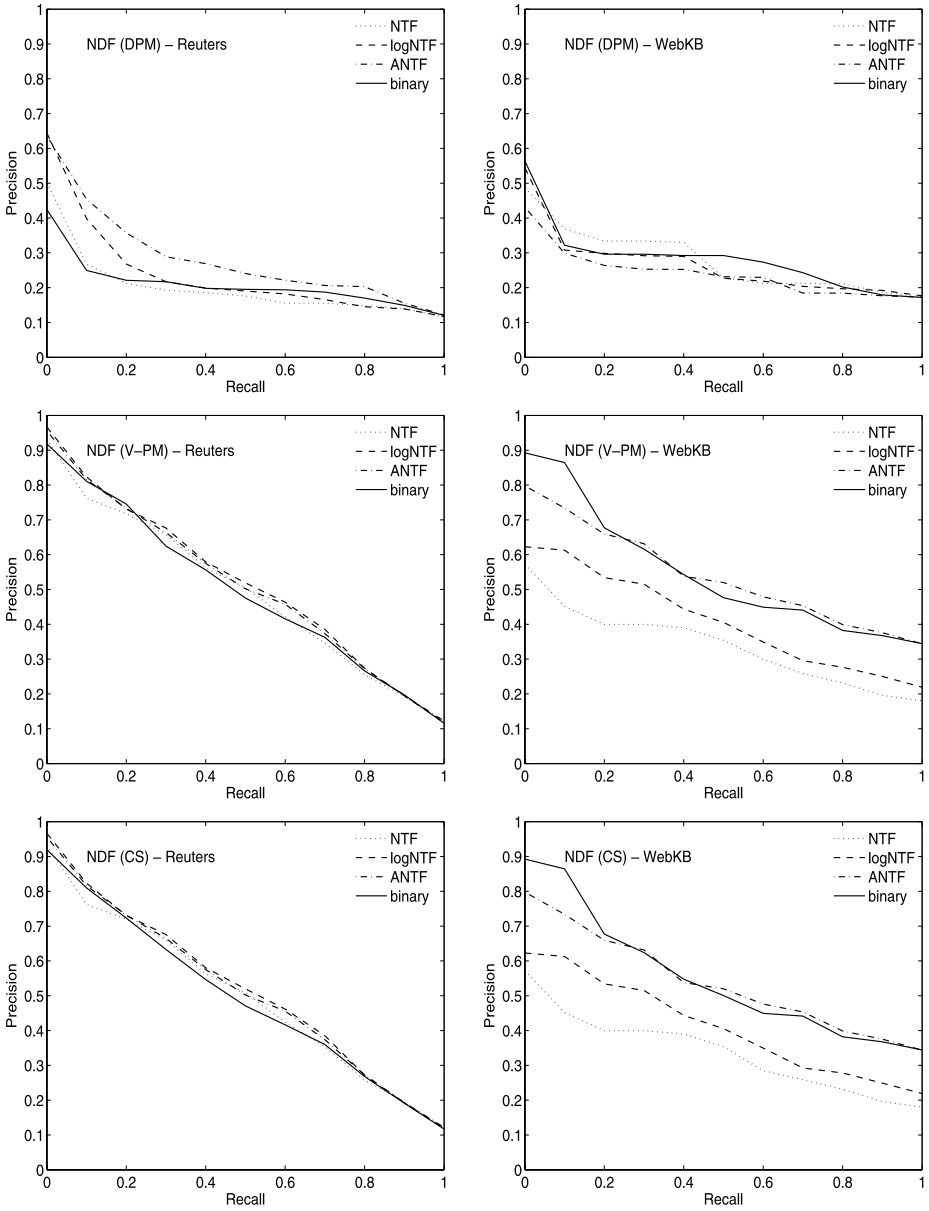

Fig. 7 The 11-point Recall-Precision curves of the NDF model according to various weighting schemes on the Reuters and WebKB corpora. Results are reported according to the $T10$ dimensionality of the representation space

shows quite stable performance little affected by the size of term set, and tends to achieve the best results for relatively high-dimensional spaces ($>5\%$ dimensionality). In contrast, the size of the term set can seriously affect the performance of NDF, which shows degradation in performance as the number of terms increases.

Table 6 The dispersion percentage of terms over categories for both Reuters and WebKB corpora

	T10	T20	>10%	>5%
Reuters	21.7%	20.6%	27.4%	32.3%
WebKB	37%	42.6%	45.5%	49.1%

Table 7 Summary of the MAP scores of PCA residuals, Hotelling's T^2 test, AANN ($H/\mathrm{I} = 10\%$), and 1-SVM (linear kernel function) on the Reuters and WebKB corpora under different weighting schemes and various dimensions of the representation spaces. Bold entries denote the statistically highest scores among the different settings while underlined entries indicate the performance scores at the reference settings supported by the overall results

		Reuters				WebKB			
		NTF	logNTF	ANTF	binary	NTF	logNTF	ANTF	binary
PCA residuals	T10	0.3167	0.4202	0.4680	0.3255	0.2529	0.2654	0.2342	0.2650
	T20	0.3308	0.4307	0.4772	0.2095	0.2198	0.2443	0.2303	0.2112
	>10%	0.2967	0.3915	0.4314	0.1530	0.2568	0.2641	0.2472	0.1799
	>5%	0.3569	0.4307	**0.4898**	0.1694	0.2880	**0.3304**	0.2889	0.1834
Hotelling T^2 test	T10	0.1740	0.1715	0.1801	0.1492	0.2099	0.2321	0.3251	**0.3713**
	T20	0.1562	0.1553	0.1583	0.1028	0.1988	0.2985	0.2940	0.2441
	>10%	0.1736	**0.2148**	0.1828	0.0713	0.2039	0.2707	0.3596	0.1674
	>5%	0.1432	0.1952	0.1892	0.0802	0.2043	0.2420	0.2619	0.1726
AANN	T10	0.4261	0.5024	0.3452	0.2989	0.2460	0.2776	0.3217	0.4099
	T20	0.4209	0.4852	0.2363	0.1466	0.2492	0.3013	0.2324	0.2448
	>10%	0.4234	0.5903	0.1294	0.1380	0.3451	0.4055	0.1733	0.1703
	>5%	0.4495	**0.6272**	0.1370	0.1148	0.3361	**0.43**	0.1824	0.18
1-SVM	T10	0.4846	0.5439	0.5756	0.5823	0.2360	0.2893	0.3387	**0.5377**
	T20	0.4703	0.5333	0.5589	0.5510	0.2479	0.4512	0.4819	0.4901
	>10%	0.4838	0.5711	0.5832	0.4788	0.3271	0.3954	0.5030	0.3725
	>5%	0.5152	0.6204	**0.6356**	0.4277	0.3384	0.5035	0.5038	0.3277

Overall, if we contrast the performance of NDF with its improved version ILoNDF, we find that ILoNDF substantially outperforms NDF, particularly with high-dimensional data spaces.

8.1.2 Other candidate one-class classification methods

The goal of the second series of experiments was to examine the effect of different settings on the behavior of other candidate methods, viz. PCA residuals, Hotelling's T^2 test, AANN and 1-SVM. Table 7 summarizes the MAP scores achieved by the methods under different weighting schemes and various dimensions of the representation space. Figure 10 depicts the 11-point Recall-Precision curves of the methods according to the best weighting scheme while varying dimensions of the representation space on the Reuters and WebKB corpora. Figures 11 depicts the 11-point Recall-Precision curves of the methods according to the best dimensionality while varying the weighting schemes.

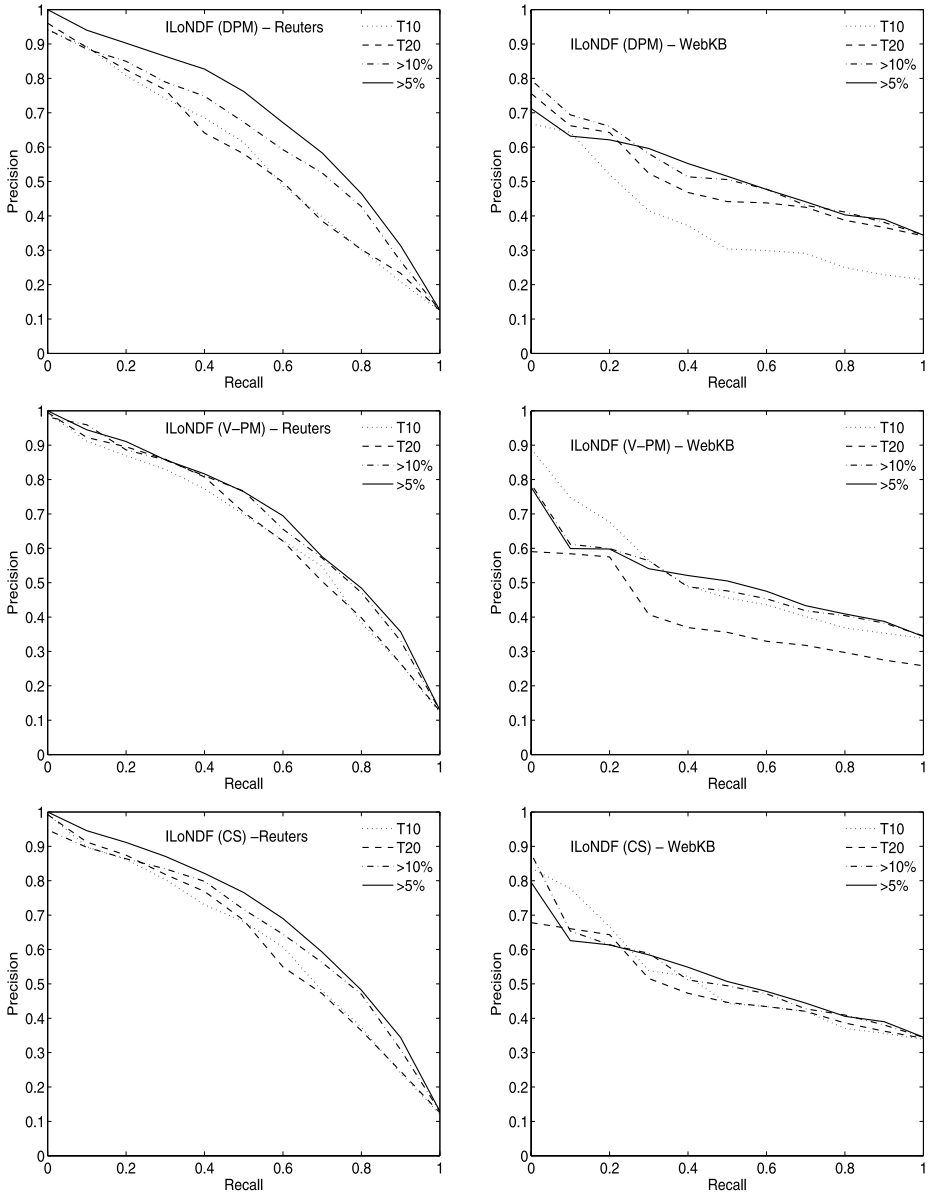

Fig. 8 The 11-point Recall-Precision curves of the ILoNDF model according to various dimensions of the representation space on the Reuters and WebKB corpora. Results are reported according to the ANTF weighting scheme

Overall, the PCA-based methods (PCA residuals and Hotelling's T^2 test) show very poor performance on both the Reuters and WebKB corpora. The performance of PCA residuals is somewhat better than that of T^2 on Reuters but rather worse on WebKB. It is somewhat misleading to identify the best initial settings for these methods. Roughly speaking, both the logNTF and ANTF weighting schemes appear to be good candidates for PCA residu-

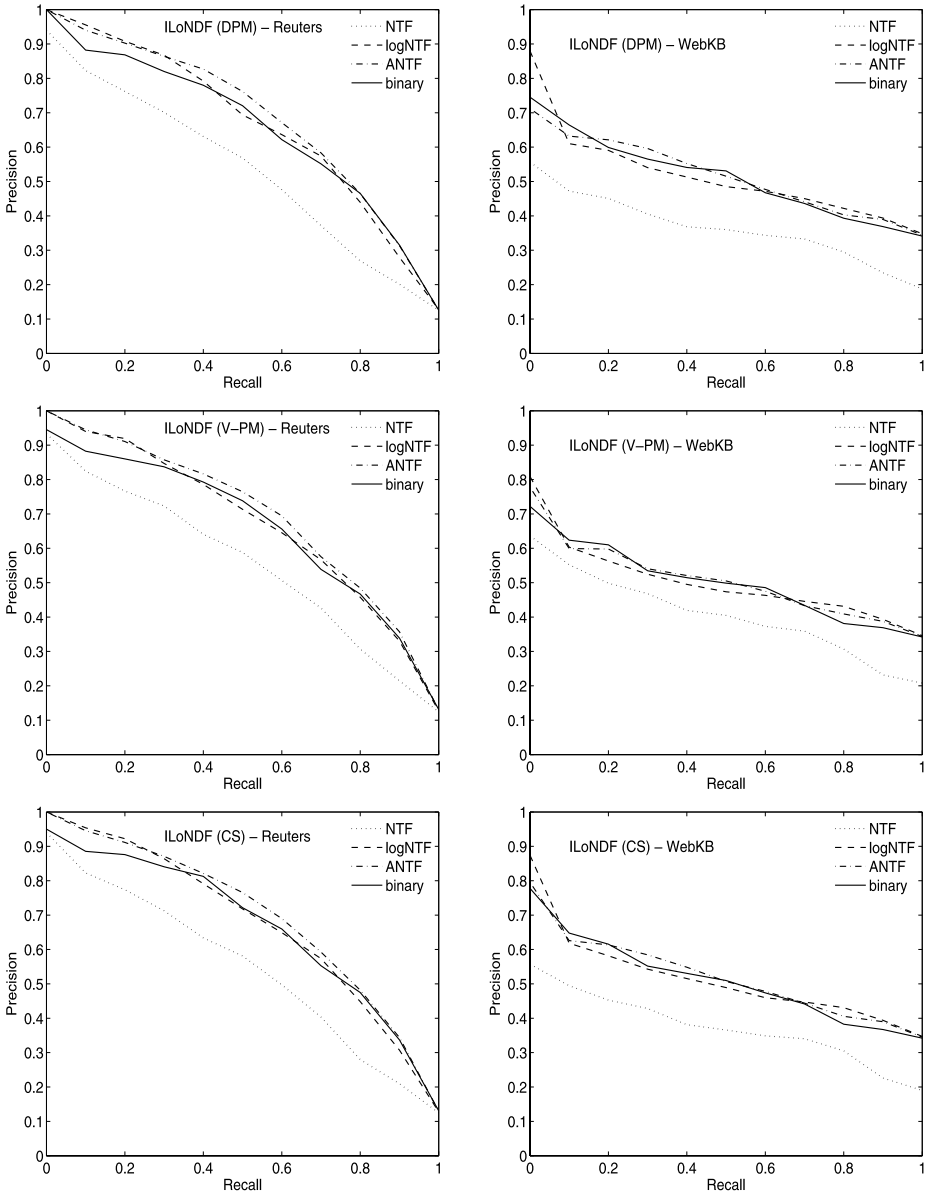

Fig. 9 The 11-point Recall-Precision curves of the ILoNDF model according to various weighting schemes on the Reuters and WebKB corpora. Results are reported according to the >5% dimensionality of the representation space

als, particularly when applied on high-dimensional data. In contrast, Hotelling's T^2 test may produce better results on low-dimensional data but the best weighting scheme varies depending on the collection. The method seems to work better with the ANTF weighting scheme on the Reuters corpus, while best results are obtained with binary weights on the WebKB corpus. In some respects, the results reveal a degree of similarity between the performance

Table 8 The impact of the network size (the number of neuron units in the hidden layer (H) with respect to that of the input/output layers (I)) of AANN on the classification performance according to the MAP scores (weighting scheme is logNTF). Bold entries denote the statistically highest scores among the different settings while underlined entries indicate the performance scores at the reference settings supported by the overall results

(H/I)	Reuters				WebKB			
	10%	25%	50%	75%	10%	25%	50%	75%
T10	0.5024	0.5042	0.4580	0.4877	0.2776	0.3030	0.2509	0.2962
T20	0.4852	0.4801	0.5072	0.4762	0.3013	0.2868	0.2790	0.2981
>10%	0.5903	0.5829	0.6257	0.5918	0.4055	0.3836	0.4035	0.4084
>5%	0.6272	0.6110	0.5947	**0.6335**	**0.43**	0.4183	0.3785	0.3510

of the PCA-based methods and that of NDF associated with DPM (cf. Table 4). This is probably because all of these methods achieve a high degree of decorrelation amongst data. The decorrelation ensures the detection of data that significantly deviate from the training data. However, if there is some correlated noise in the representation of the training data, the decorrelating mechanism may significantly affect the classification results.

The AANN method shows a better performance than the PCA-based methods. However, it shows great sensitivity with regard to the initial settings such as weighting schemes and dimensionality of the representation space, but it is less affected by the size of its hidden layer (cf. Table 8). Its best performance is obtained at the highest dimension (>5%) using the logNTF weighting scheme and a small number of hidden units ($H/I = 10\%$).

Among the four candidate methods tested, 1-SVM achieves the best performance. When using the binary weighting scheme 1-SVM shows constant degradation in classification performance as the dimensionality of the representation space increases, which is in agreement with previous studies (Manevitz and Yousef 2001). However, 1-SVM shows an opposite tendency when using the other weighting schemes and attains its best performance at the highest dimensionality (>5%) through the use of the ANTF weighting scheme. In addition to the linear kernel function, the 1-SVM classifier was tested with three non-linear kernel functions (Polynomial, Radial basis, Sigmoid). Results shown that non-linear kernels are not particularly useful. This is because text classification problems are generally linearly separable so no mapping to a higher dimensional space is required (Joachims 2001). On the other hand, the choice of an appropriate value of the parameter ν is an important setting which can make 1-SVM more tolerant to noise that might be present in the training data. Figure 12 displays the curves of the MAP scores on the Reuters and WebKB corpora when varying the value of ν. By increasing ν, the description of the positive class becomes more specific to the training data, and consequently recognizing a new positive data will be more difficult. The optimal value of ν was chosen at the best value of the MAP scores and was found to be very low on both Reuters and WebKB corpora ($\nu \simeq 0.001$). The reason is that experimental benchmarks are usually carefully labeled and so are unlikely to contain many noisy data. However, we believe that setting such parameters, without any prior knowledge about the nature of the training data, is ultimately a matter of exploration.

8.1.3 Evaluation of thresholding strategies

Next we evaluate classification performance after applying the threshold strategies associated with the methods. Given the poor performance of NDF, PCA residuals and Hotelling's

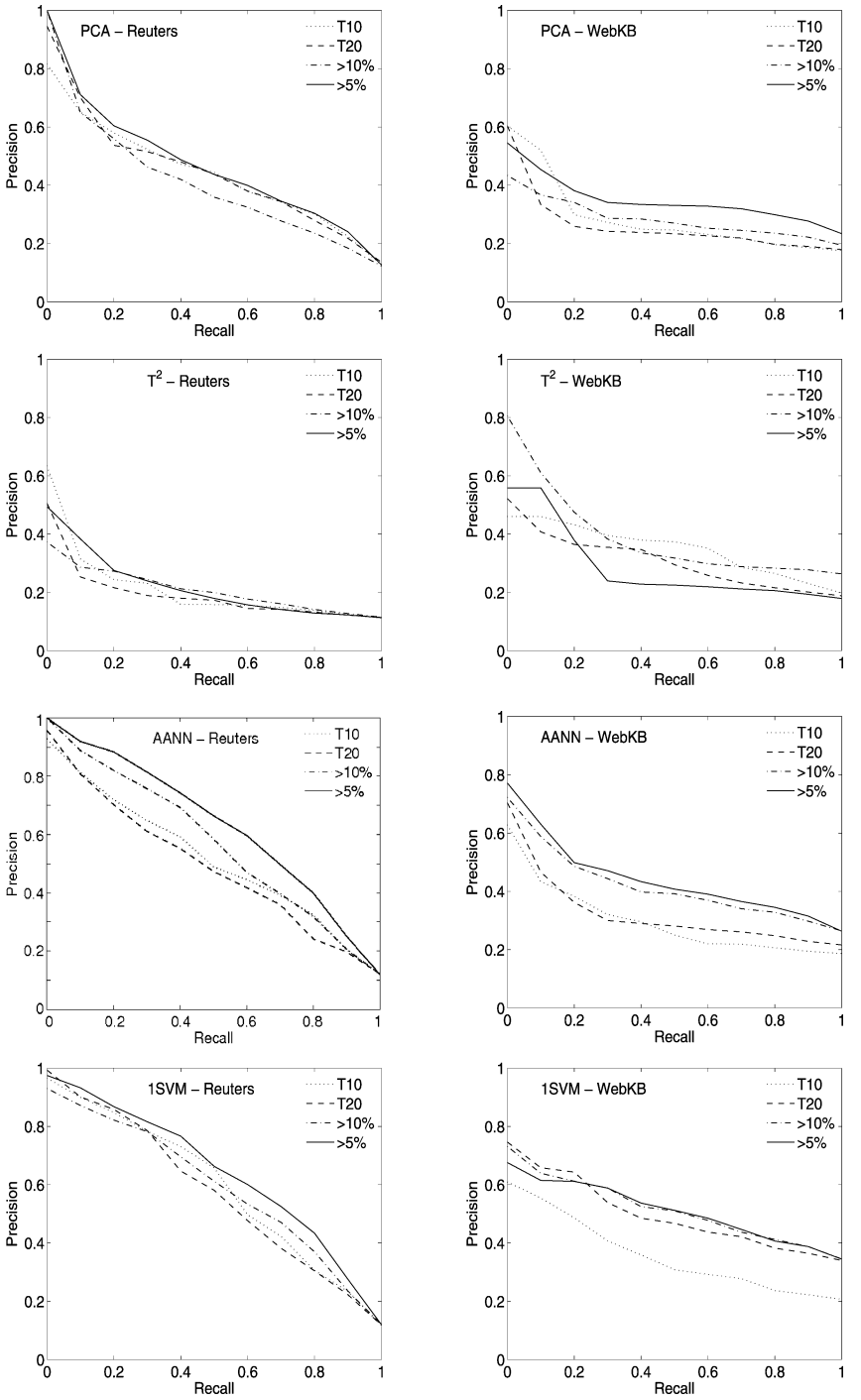

Fig. 10 The 11-point Recall-Precision curves of PCA residuals, Hotelling T^2, AANN and 1-SVM according to various dimensions of the representation space on the Reuters and WebKB corpora (Weighting schemes are respectively logNTF, ANTF, logNTF and binary)

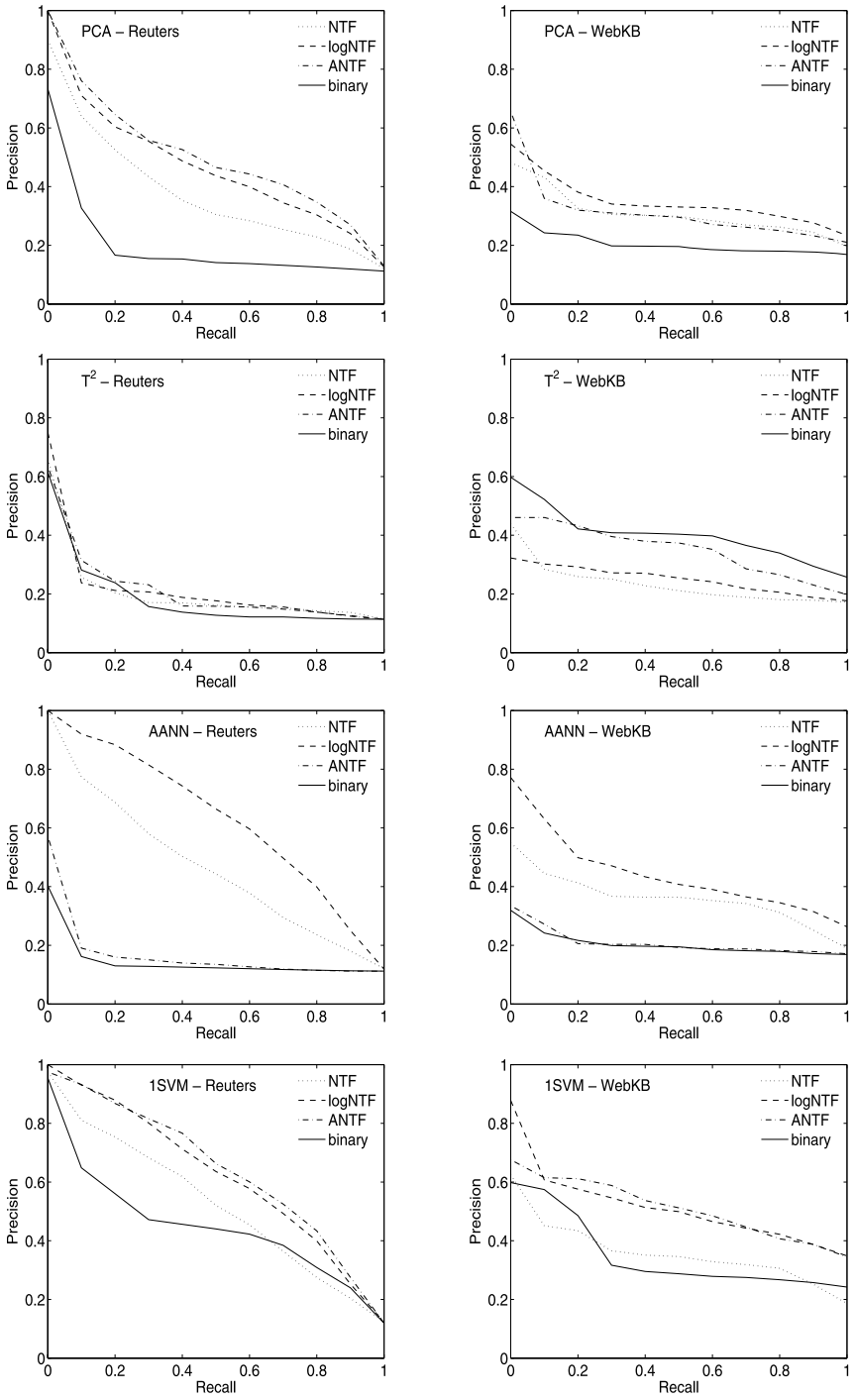

Fig. 11 The 11-point Recall-Precision curves of PCA residuals, Hotelling T^2, AANN and 1-SVM according to various weighting schemes on the Reuters and WebKB corpora. The results of Hotelling T^2 test are reported under the T10 dimensionality while the results of other methods are reported under the >5% dimensionality

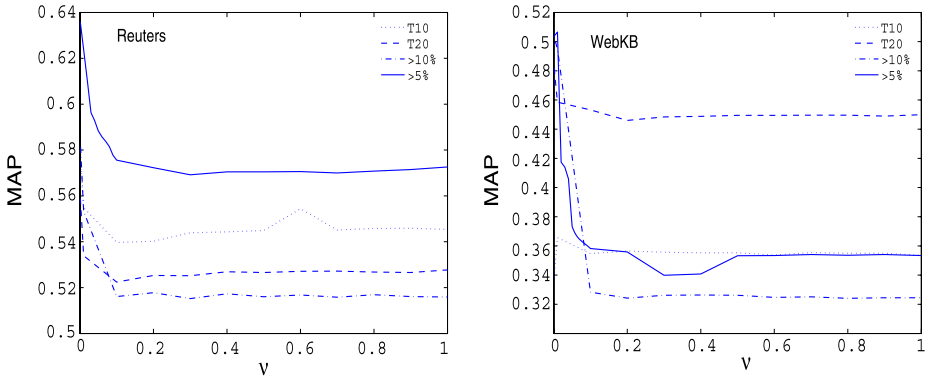

Fig. 12 The MAP scores of 1-SVM according to ν parameter value (**a**) on the Reuters corpus (**b**) on the WebKB corpus (Weighting scheme is ANTF)

T^2 test, we do not present details of their results, but instead concentrate on ILoNDF, AANN and 1-SVM. Table 9 summarizes the macroaveraged and microaveraged values of precision, recall and F_1 criteria on the Reuters and WebKB corpora. The F_1 values corresponding to the break-even points achieved by the methods are also given in Table 9. The results are reported according to the logNTF weighting scheme for AANN and the ANTF weighting scheme for both ILoNDF and 1-SVM methods.

From the results we notice that the strategy outlined in Sect. 5 for AANN is convincing. Training the network over several epochs produces a rather reliable estimate of the classification scores of positive data. However, the threshold turns out to be biased towards recall for low dimensionality versus precision for high dimensionality. In both cases there is no great difference between the F_1 values obtained according to this strategy and those according to the break-even points. The ability of 1-SVM to strike a compromise between precision and recall is better on Reuters than on WebKB. Often it assigns relatively more importance to recall than to precision even for the best value of the ν parameter.

The thresholding strategy of Sect. 3.7 applied to ILoNDF is interesting. The results are already good without the use of the relaxation parameter τ (i.e. τ is set to 1). In this case the resulting threshold of classification emphasizes the importance of precision over recall, especially for high-dimensional data. For the problem of one-class classification, achieving such precision while maintaining recall as high as possible is usually difficult but desirable, because target (positive) data are often much rarer than negative data so ensuring high recall will often be at the cost of precision, and vice versa. In other situations where the recall may be of higher importance, the relaxation parameter τ can be used to control the trade-off between precision and recall. An illustration is given in Table 9 when using the relaxation parameter τ with a value of 0.95.

From Table 9 we can also see that the microaveraged breakeven scores (dominated by the performance on the common categories) are often higher than the macroaveraged scores (dominated by the performance on the relatively small categories), meaning that all the studied classifiers tend to perform better with increasing amounts of training data. However, there is not much difference between the microaveraged and macroaveraged scores of ILoNDF. A larger gap can be observed with AANN and 1-SVM. Therefore, we expect that the performance of ILoNDF may be less sensitive to the amount of training data, even though classification accuracy will primarily depend on the characteristics of the training data.

Table 9 Comparison of the classification performance of the AANN, 1-SVM and ILoNDF models on the Reuters and WebKB corpora. Results are reported according to the logNTF weighting scheme for AANN and the ANTF scheme for 1-SVM and ILoNDF. Bold entries denote the highest performance scores achieved by the different models

		Macroaveraged				Microaveraged			
		P	R	F_1	BKP	P	R	F_1	BKP
Reuters AANN	T10	0.4695	0.5474	0.4218	0.4886	0.2887	0.4716	0.3582	0.5519
	T20	0.4896	0.5221	0.4484	0.4767	0.3845	0.4605	0.4191	0.5518
	>10%	0.6850	0.4544	0.5387	0.5561	0.7382	0.5159	0.6074	0.6078
	>5%	0.7432	0.4338	0.5421	0.5935	0.8329	0.5081	0.6312	0.7011
1-SVM	T10	0.5320	0.5880	0.5150	0.5650	0.4327	0.5644	0.4899	0.5780
	T20	0.5043	0.5815	0.5011	0.5512	0.4136	0.5619	0.4765	0.5503
	>10%	0.5807	0.5688	0.5591	0.5734	0.6278	0.6167	0.6222	0.6710
	>5%	0.6452	0.5598	0.5896	0.6086	0.7321	**0.6292**	0.6768	0.7192
ILoNDF $\tau = 1$	T10	0.6613	0.5175	0.5708	0.5942	0.6121	0.4672	0.5299	0.5706
	T20	0.6151	0.5217	0.5309	0.5851	0.5488	0.5020	0.5749	0.5749
	>10%	0.7254	0.5121	0.5871	0.6313	0.7524	0.5297	0.6217	0.7034
	>5%	**0.8173**	0.4708	0.59	**0.6552**	**0.8776**	0.5414	0.6696	**0.7514**
$\tau = 0.95$	T10	0.5171	**0.6275**	0.5327	–	0.4289	0.5952	0.4985	–
	T20	0.5392	0.6183	0.5374	–	0.4424	0.5943	0.5072	–
	>10%	0.6626	0.6056	0.6123	–	0.6828	0.6341	0.6575	–
	>5%	0.7514	0.5557	**0.6291**	–	0.8257	0.6249	**0.7114**	—
WebKB AANN	T10	0.2110	0.6242	0.2779	0.2643	0.2713	**0.6169**	0.3768	0.2853
	T20	0.2563	0.5663	0.31	0.2749	0.2473	0.4597	0.3216	0.3042
	>10%	0.3037	0.2088	0.2270	0.2914	0.3476	0.2621	0.2989	0.3257
	>5%	0.4587	0.4310	0.3746	0.3007	0.4490	0.3548	0.3964	0.3445
1-SVM	T10	0.2839	0.6257	0.3231	0.2946	0.3114	0.5524	0.3983	0.4262
	T20	0.3150	0.5554	0.3306	0.4584	0.3372	0.4637	0.3905	0.4637
	>10%	0.3747	0.5134	0.3652	0.4686	0.3732	0.4274	0.3985	0.5
	>5%	0.4297	0.4634	**0.4091**	**0.4810**	0.4322	0.4113	0.4215	**0.5323**
ILoNDF $\tau = 1$	T10	0.2906	0.5898	0.3268	0.4739	0.3351	0.5202	0.4076	0.4718
	T20	0.3083	0.5394	0.3136	0.4716	0.3182	0.4234	0.3633	0.4556
	>10%	0.3966	0.4509	0.3422	0.4889	0.3489	0.3306	0.3395	0.5081
	>5%	**0.4620**	0.4102	0.3816	0.4784	**0.4540**	0.3185	0.3744	0.5242
$\tau = 0.95$	T10	0.2439	**0.6480**	0.3063	–	0.2857	0.6048	0.3881	–
	T20	0.2589	0.6007	0.2970	–	0.2806	0.5081	0.3615	–
	>10%	0.32	0.5607	0.3359	–	0.3306	0.4839	0.3928	–
	>5%	0.3659	0.5012	0.3684	–	0.4094	0.4556	**0.4313**	–

On a final note, it is worth remarking that while ILoNDF performs generally better than other methods (cf. Tables 4, 5 and 7), its superiority, in particular with respect to 1-SVM, becomes less pronounced after thresholding. This suggests that the thresholding strategy should be further improved to take full advantage of the capacities of the ILoNDF model.

Table 10 Summary of the MAP scores and the F_1 values at the break-even points achieved by the methods according to the best weighting scheme and the highest dimensions of the representation space. Bold entries denote the best performance scores

	Reuters		WebKB	
	MAP	BKP	MAP	BKP
NDF	0.3512	0.3584	0.4075	0.4227
AANN	0.6272	0.5935	0.43	0.3007
1-SVM	0.6356	0.6086	0.5038	**0.4810**
ILoNDF	**0.6962**	**0.6552**	**0.5118**	0.4784

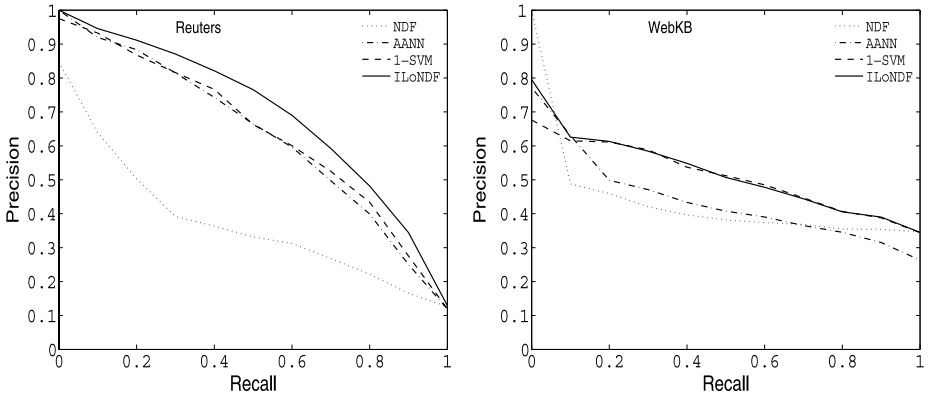

Fig. 13 The 11-point Recall-Precision curves of the methods NDF, AANN, 1-SVM and ILoNDF. The results are reported on the highest dimensionality of the representation space on the Reuters and WebKB corpora. The results of AANN are reported using the logNTF weighting scheme, while the ANTF scheme is used for NDF, 1-SVM and ILoNDF

8.2 A direct comparison of classification methods for high-dimensional data

This section presents a direct comparison of the performance of ILoNDF with that of the two best methods identified, AANN and 1-SVM. To better understand our modification of NDF's learning rule, we also present results of experiments with NDF. The focus of comparison will be on the highest dimensionality (>5%) of the data. Figure 13 depicts the 11-point Recall-Precision curves of the methods NDF, AANN, 1-SVM and ILoNDF. Figure 14 depicts the MAP scores and the F_1 values at the break-even points over the categories of the Reuters and WebKB corpora. Table 10 summarizes the MAP scores and the F_1 values at the breakeven points achieved by the methods.

From these results we observe the following. The performance of NDF is not particularly high on the Reuters corpus. ILoNDF substantially exceeds the performance of NDF by up to 35% according to the MAP scores and by up to 30% according to the F_1 scores at the break-even points on this same corpus. In addition, ILoNDF consistently outperforms other methods, yielding relative improvements of about 6–7% over 1-SVM and AANN, respectively. ILoNDF tends to be particularly useful for small ambiguous categories where the representation of data may be very noisy. This is essentially caused by the fact that these categories can have many noisy documents in the negative class. For example, the "ship" and "crude" categories are known to be highly overlapping categories with many common

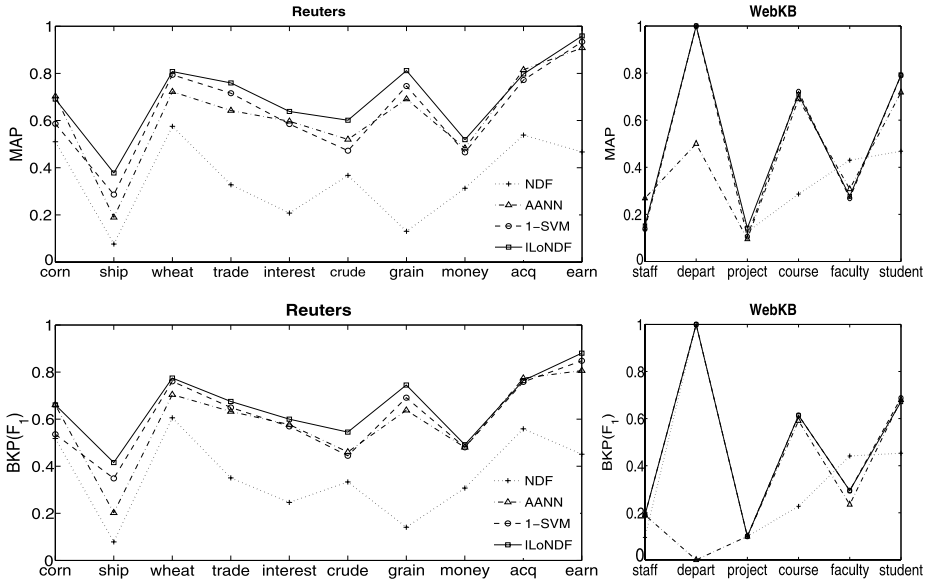

Fig. 14 Comparison between NDF, AANN, 1-SVM and ILoNDF on both Reuters and WebKB on the >5% dimensionality of the representation space. The results of AANN are reported using the logNTF weighting scheme, while the ANTF scheme is used for NDF, 1-SVM and ILoNDF

terms (the dispersion of terms over these two categories is 70.5%). Similarly, the "trade" category has a high degree of confusion with "earn" (the dispersion of terms is 55.5%) but "earn" is less sensitive to this phenomenon because of the greater number of test documents associated with "earn" as compared with "trade". On its own, 1-SVM seems to be slightly better than AANN with an overall improvement of about +1%.

We find that the performance of all the classifiers degrades from the Reuters corpus to the WebKB corpus. The gap in performance between NDF and its improved version ILoNDF decreases from 30–35% to 5–11%. Also, ILoNDF is still slightly better than others, though the improvement in performance is less significant than that obtained on the Reuters corpus (generally about +1–8%) according to the MAP scores. Our hypothesis is that the classification of web documents can be more difficult than the classification of normal documents. Indeed, the web documents—unlike many of the corpora typically used for experimental evaluation of text classification—lacks homogeneity and regularity. As outlined before in Table 6, the terms in the WebKB corpus are quite scattered. The dispersion of terms over categories is 49.1% on WebKB and 32.3% on Reuters. So, there are fewer specific terms for each category against too many non-discriminating (noisy) ones in the case of WebKB corpus, resulting in lower performance of the classifiers including ILoNDF.

In conclusion, it is important to note that ILoNDF has a comparable computational cost to 1-SVM, while being significantly less expensive than AANN. However, when tuning the different parameters of the 1-SVM classifier, a series of validation tests is generally conducted. Such tests involve a very expensive optimization, and thus will significantly increase the computational requirements of the 1-SVM method. On the other hand, ILoNDF does not need such optimization and it turns out to be less sensitive to experimental aspects such as representation and term weighting schemes.

8.3 Summary

The main points emerging from the results of the previous sections can be briefly summarized as follows.

First, we studied the behaviors of NDF and ILoNDF according to the various methods of calculating classification scores: DPM, V-PM and CS. We also tested their robustness with respect to various experimental aspects, viz. different weighting schemes and different dimensions for the representation space. Upon examining the results, the following points are noted:

- The behaviors of NDF and ILoNDF are very different with respect to DPM and V-PM. For both models, the method CS presents a reasonable solution to certain problems which may arise when using the previous methods;
- The best weighting scheme for both NDF and ILoNDF is ANTF;
- The NDF and ILoNDF methods behave differently relative to the dimensionality of the representation space. NDF yields better results in the case of relatively small-dimensional spaces, whereas ILoNDF yields better results in the case of relatively high-dimensional spaces;
- The superiority of ILoNDF over NDF is very apparent, particularly in the case of high-dimensional spaces.

Next, we examined the behavior of other methods of classification under the various experimental aspects. The studied methods are the PCA residuals, the Hotelling's T^2 test, auto-associative neural networks (AANN), and a one-class version of the SVM classifier (1-SVM). The experiments revealed the following results:

- The PCA-based methods (PCA residuals and Hotelling's T^2 test) have very poor performance of the same order as NDF. It was difficult to draw clear conclusions on the best weighting scheme and the best dimensionality for these methods;
- The auto-associative neural networks offer better performance than the PCA-based methods. They show a great sensitivity to the various experimental aspects. The best results are obtained using a restricted number of hidden neurons, the logNTF weighting scheme and the highest dimension of the representation space;
- 1-SVM, among the four studied methods, produced the best results. The best performance is reached when using the ANTF weighting scheme and the highest dimensional representation space. The use of non-linear kernel functions does not bring performance improvements over the use of linear function in the case of text data. The choice of appropriate value of the parameter ν is very important and affects directly the performance of 1-SVM.

Concerning the strategies of thresholding adopted for the various methods, we identified the following findings:

- The thresholding strategy applied to AANN tends to be biased towards recall for low dimensionality versus precision for high dimensionality;
- 1-SVM assigns relatively more importance to recall than to precision even for the best value of the ν parameter;
- The thresholding strategy applied to ILoNDF emphasizes the importance of precision over recall, especially for high-dimensional data, without the use of the relaxation parameter τ (i.e. $\tau = 1$). The relaxation parameter can be used to control the trade-off between precision and recall;

- The performance of ILoNDF seems be less sensitive to the amount of training data than the other studied methods.

Overall, the results of experiments indicated better performance of ILoNDF than the other studied methods, in particular in the case of high-dimensional spaces.

9 Conclusion and future work

In this study we have introduced ILoNDF, a new high-dimensional on-line learning method based on novelty detection theory with application to the one-class classification problem. An attractive aspect of this model is the ability of its generative learning to capture the intrinsic characteristics of the training data by continually incorporating information relating to the relative frequencies of the features of training data and their co-occurrence dependencies. This makes ILoNDF fairly stable and less sensitive to noisy features which may be present in the representation of the positive data. In addition, ILoNDF is less computationally expensive since it is an on-line method that does not need repeated training and it has no parameters that need to be tuned. The objective of our experiments was to demonstrate the potential of the proposed modification of the original NDF model, and to compare ILoNDF to other candidate methods, viz. PCA residuals, Hotelling's T^2 test, auto-associative neural network and a one-class version of SVM classifier in the context of high-dimensional noisy data. We have explored the effects of various dimensions of the representation space, weighting schemes and normalization techniques. From our experiments we conclude that ILoNDF is a robust model less affected by initial settings and its performance is often superior to that of other methods.

As future work, we intend to investigate more thoroughly two issues related to one-class classification. The first one concerns the exploitation of the rich information in a web document and the connectivity among documents to have more accurate classification of this kind of data. Hyperlinks, HTML tags, and metadata all provide rich information for web classification, which is not typically available in traditional text classification and which is often found useful for improving classification performance (Yang et al. 2002). The second issue concerns the setting of decision thresholds for which we plan to extend our proposed thresholding strategy by including the specific characteristic of the class under consideration, such as density and exhaustivity (Kassab and Lamirel 2007). Still, feature selection and weighting schemes are an open research issue. So, more intensive trials are required to suggest a better performance of one-class classification approaches.

In another aspect, while the main evidence about the behavior of ILoNDF are emphasized throughout the present work, it is still interesting to analyze its behavior when dealing with other types of data rather than textual data. An issue of particular interest is to study the ability of ILoNDF to solve more complex non-linear problems and to compare it to kernel methods.

References

Aeyels, D. (1990). On the dynamic behaviour of the novelty detector and the novelty filter. In B. Bonnard, B. Bride, J. P. Gauthier & I. Kupka (Eds.), *Analysis of controlled dynamical systems* (pp. 1–10).

Baldi, P., Chauvin, Y., & Hornik, K. (1995). Back-propagation and unsupervised learning in linear networks. In Y. Chauvin & D. E. Rumelhart (Eds.), *Back propagation: theory, architectures and applications* (pp. 389–432). Hillsdale: Lawrence Earlbaum Associates.

Ben-Israel, A., & Greville, T. N. E. (2003). *Generalized inverse: theory and applications* (2nd ed.). New York: Springer.

Brown, M. W., & Xiang, J. Z. (1998). Recognition memory: neuronal substrates of the judgement of prior occurrence. *Progress in Neurobiology*, *55*, 149–189.

Cottrell, G. W., & Munro, P. (1988). Principal components analysis of images via back propagation. In *Proceedings of the society of photo-optical instrumentation engineers* (pp. 1070–1077).

Debole, F., & Sebastiani, F. (2005). An analysis of the relative hardness of Reuters-21578 subsets. *Journal of the American Society for Information Science and Technology*, *56*(6), 584–596.

Denis, F., Gilleron, R., Laurent, A., & Tommasi, M. (2003). Text classification and co-training from positive and unlabeled examples. In *Proceedings of the ICML 2003 workshop: the continuum from labeled to unlabeled data* (pp. 80–87).

Detroja, K. P., Gudi, R. D., & Patwardhan, S. C. (2007). Plant-wide detection and diagnosis using correspondence analysis. *Control Engineering Practice*, *15*(12), 1468–1483.

Dumais, S., Platt, J., Heckerman, D., & Sahami, M. (1998). Inductive learning algorithms and representations for text categorization. In *Proceedings of the 7th international conference on information and knowledge management (CIKM 98)* (pp. 148–155).

Fletcher, R. (1987). *Practical methods of optimization*. New York: Wiley.

Fung, G. P. C., Yu, J. X., Lu, H., & Yu, P. S. (2006). Text classification without negative examples revisit. *IEEE Transactions on Knowledge and Data Engineering*, *18*(1), 6–20.

Greville, T. N. E. (1960). Some applications of the pseudoinverse of a matrix. *SIAM Review*.

Jackson, J. E., & Mudholkar, G. S. (1979). Control procedures for residuals associated with principal component analysis. *Technometrics*, *21*(3), 341–349.

Japkowicz, N. (2001). Supervised versus unsupervised binary-learning by feedforward neural networks. *Machine Learning*, *42*(1-2), 97–122.

Japkowicz, N., Myers, C., & Gluck, M. A. (1995). A novelty detection approach to classification. In *Proceedings of the fourteenth joint conference on artificial intelligence* (pp. 518–523).

Japkowicz, N., Hanson, S. J., & Gluck, M. A. (2000). Nonlinear autoassociation is not equivalent to PCA. *Neural Computing*, *12*(3), 531–545.

Joachims, T. (2001). A statistical learning model of text classification for support vector machines. In *Proceedings of the conference on research and development in information retrieval (SIGIR)* (pp. 128–136).

Jolliffe, I. T. (1986). *Principal component analysis*. New York: Springer.

Kambhaltla, N., & Leen, T. K. (1994). Fast non-linear dimension reduction. In J. D. Cowan, G. Tesauro & J. Alspector (Eds.), *Advances in neural information processing systems* (Vol. 6, pp. 152–159).

Kassab, R., & Lamirel, J. C. (2006). A new approach to intelligent text filtering based on novelty detection. In *The 17th Australian database conference* (pp. 149–156), Tasmania, Australia.

Kassab, R., & Lamirel, J. C. (2007). Towards a synthetic analysis of user's information need for more effective personalized filtering services. In *The 22nd ACM symposium on applied computing special track on information access and retrieval (SAC-IAR)* (pp. 852–859).

Kim, J. H., & Beale, G. O. (2002). Fault detection and classification in underwater vehicles using the T^2 statistic. *AUTOMATIKA—Journal for Control, Measurement, Electronics, Computing and Communications*, *43*(1-2), 29–37.

Kohonen, T. (1989). *Self organisation and associative memory* (3rd ed.). New York: Springer.

Kohonen, T., & Oja, E. (1976). Fast adaptive formation of orthogonalizing filters and associative memory in recurrent networks of neuron-like elements. *Biological Cybernetics*, *21*, 85–95.

Lee, H., & Cho, S. (2006). Application of LVQ to novelty detection using outlier training data. *Pattern Recognition Letters*, *27*(13), 1572–1579.

Lewis, D. D. (1991). Evaluating text categorization. In *Proceedings of speech and natural language workshop* (pp. 312–318). San Mateo: Morgan Kaufmann.

Li, X., & Liu, B. (2003). Learning to classify texts using positive and unlabeled data. In *Proceedings of the 18th international joint conference on artificial intelligence (IJCAI-03)* (pp. 587–594).

Liu, B., Dai, Y., Li, X., Lee, W. S., & Yu, P. S. (2003). Building text classifiers using positive and unlabeled examples. In *International conference on data mining* (pp. 179–186).

Manevitz, L. M., & Yousef, M. (2001). One-class SVMs for document classification. *Journal of Machine Learning Research*, *2*, 139–154.

Markou, M., & Singh, S. (2003a). Novelty detection: a review—part 1: statistical approaches. *Signal Processing*, *83*(12), 2481–2497.

Markou, M., & Singh, S. (2003b). Novelty detection: a review—part 2: neural network based approaches. *Signal Processing*, *83*(12), 2499–2521.

Marsland, S., Nehmzow, U., & Shapiro, J. (2000). A real-time novelty detector for a mobile robot. In *European advanced robotics systems masterclass and conference*. Salford: AAAI Press.

Noda, M. T., Makino, I., & Saito, T. (1997). Algebraic methods for computing a generalized inverse. *ACM SIGSAM Bulletin*, *31*(3), 51–52.

Oja, E. (1991). Data compression, feature extraction, and autoassociation in feedforward neural networks. In K. Kohonen, et al. (Eds.), *Artificial neural networks* (pp. 737–745).

Penrose, R. (1955). A generalized inverse for matrices. *Proceedings of the Cambridge Philosophical Society*, *52*, 406–413.

Raskutti, B., & Kowalczyk, A. (2004). Extreme re-balancing for SVMs: a case study. *SIGKDD Explorer Newsletter*, *6*(1), 60–69.

van Rijsbergen, C. J. (1979). *Information retrieval*. London: Butterworths.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representation by error propagation. In D. E. Rumelhart & J. L. McClelland (Eds.), *Parallel distributed processing: explorations in the microstructures of cognition* (pp. 318–362). Cambridge: MIT Press.

Rüping, S. (2000). *mySVM-Manual*. Universitat Dortmund, Lehrstuhl Informatik VIII, http://www-ai.cs.uni-dortmund.de/SOFTWARE/MYSVM/.

Salton, G. (1971). *The SMART retrieval system: experiments in automatic document processing*. Englewood Cliffs: Prentice Hall.

Salton, G., & Buckley, C. (1988). Term weighting approaches in automatic text retrieval. *Information Processing and Management*, *24*(5), 513–523.

Schölkopf, B., Platt, J., Shawe-Taylor, J., Smola, A. J., & Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. *Neural Computation*, *13*(7), 1443–1471.

Schwab, I., Pohl, W., & Koychev, I. (2000). Learning to recommend from positive evidence. In *Proceedings of intelligent user interfaces* (pp. 241–247). New York: ACM Press.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, *34*(1), 1–47.

Sirois, S., & Mareshal, D. (2004). An interacting systems model of infant habituation. *Journal of Cognitive Neuroscience*, *16*(8), 1352–1362.

Tax, D. M. J., & Duin, R. P. W. (2001). Uniform object generation for optimizing one-class classifiers. *Journal of Machine Learning Research*, *2*, 155–173.

Valle, S., Li, W., & Qin, S. J. (1999). Selection of the number of principal components: the variance of the reconstruction error criterion with a comparison to other methods. *Industrial & Engineering Chemistry Research*, *38*(11), 4389–4401.

Vapnik, V. N. (1995). *The nature of statistical learning theory*. New York: Springer.

Yang, Y., Slattery, S., & Ghani, R. (2002). A study of approaches to hypertext categorization. *Journal of Intelligent Information Systems*, *18*(2-3), 219–241.

Yu, H., Zhai, C., & Han, J. (2003). Text classification from positive and unlabeled documents. In *Proceedings of the twelfth international conference on information and knowledge management* (pp. 232–239). New York: ACM Press.

Yu, H., Han, J., & Chang, K. C. C. (2004). PEBL: Web page classification without negative examples. *IEEE Transactions on Knowledge and Data Engineering*, *16*(1), 70–81.

Žižka, J., Hroza, J., Pouliquen, B., Ignat, C., & Steinberger, R. (2006). The selection of electronic text documents supported by only positive examples. In *Proceedings of the 8th international conference on the statistical analysis of textual data (JADT)* (pp. 19–21).