Estimating a Kernel Fisher Discriminant in the Presence of Label Noise

Neil D. Lawrence NEIL@THELAWRENCES.NET
Bernhard Schölkopf BS@CONCLU.DE
Microsoft Research, St George House, 1 Guildhall Street, Cambridge, CB2 3NH, U.K.

Abstract

Data noise is present in many machine learning problems domains, some of these are well studied but others have received less attention. In this paper we propose an algorithm for constructing a kernel Fisher discriminant (KFD) from training examples with *noisy labels*. The approach allows to associate with each example a probability of the label being flipped. We utilise an expectation maximization (EM) algorithm for updating the probabilities. The E-step uses class conditional probabilities estimated as a by-product of the KFD algorithm. The M-step updates the flip probabilities and determines the parameters of the discriminant. We demonstrate the feasibility of the approach on two real-world data-sets.

1. Introduction

The presence of noise in data is a common problem. In the context of a supervised learning task this noise may either be on input data, $\mathbf{x}$, or on the observed 'target data', y, which may take the form of a regression target or a class label. The most widely studied noise model is perhaps independent Gaussian noise on a regression target. In this paper we wish to concentrate on noisy class labels, noise of this type may be present through data mis-labellings. To our knowledge, this form of model has not been widely studied, although some work may be found in (Norton & Hirsh, 1993; Guyon et al., 1996; Graepel & Herbrich, 2001).

In the next section we describe the general form of a probabilistic model which allows learning in the presence of label noise. For the purposes of illustration we consider specific, fairly restrictive, functional forms for the component distributions of the model. Section 3 describes how those distributions' parameters may be optimised through an expectation-maximisation (EM) algorithm, which is demonstrated on some toy data in Section 4. We extend the representational power of our model in Section 5 by fitting the probabilistic models in a high dimensional feature space using 'the kernel trick'. In the following section (Section 6) the kernelised approach is evaluated using OCR data as well as a photographic image data-set. Finally in Section 7 we discuss some limitations of the approach and how we hope to overcome them in future works.

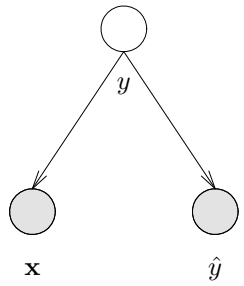

Figure 1. Graphical representation of the underlying probabilistic model. Such a representation signifies that the joint distribution of the variables may be factorised as follows $p(\mathbf{x}, y, \hat{y}) = P(\hat{y}|y)p(\mathbf{x}|y)P(y)$.

Notationally we choose to represent a discrete probability with $P(\cdot)$ and a probability density function by $p(\cdot)$.

2. The Probabilistic Model

The general approach we propose is to model the noise process probabilistically, in particular we approach the entire classification problem with a *generative model* and model the noise process as one component part of that model. Whilst it is thought that in a traditional classification task a purely discriminative approach (such as a neural network or a support vector machine) should be usually more effective, casting the problem as a generative model allows us to incorporate our model for the label noise in a straightforward manner. Specifically we assume that the joint distribution of the variables factorises as in Figure 1.

The model may now be fully specified by describing the functional form of the probabilistic relationships. Firstly, the class conditional densities, $p(\mathbf{x}|y)$, could simply be modelled by a Gaussian with mean $\mathbf{m}_y$ and covariance $\mathbf{\Sigma}_y$,

$$p(\mathbf{x}|y) = \mathcal{N}(\mathbf{x}|\mathbf{m}_y, \mathbf{\Sigma}_y). \quad (1)$$

Secondly, the probabilistic relationship between the noisy observed class label, $\hat{y}$, and the actual class label (the conditional distribution $P(\hat{y}|y)$) could then be specified by a probability table. For a two-class classification such a table would be parameterised as follows

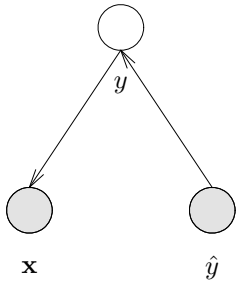

Figure 2. Graphical representation of the underlying probabilistic model. Such a representation signifies that the joint distribution of the variables may be factorised as follows $p(\mathbf{x}, y, \hat{y}) = P(y|\hat{y})p(\mathbf{x}|y)P(\hat{y})$. Note the difference between this plot and Figure 1 is that the direction of the arrow between y and $\hat{y}$ is reversed.

		$\hat{y}$	
		0	1
y	0	$(1-\gamma_0)$	γ_0
	1	γ_1	$(1-\gamma_1)$.

The above table implies that a data-point, $(\mathbf{x}_n, \hat{y}_n)$, is mislabelled with probability γ_0 when the true class is 0 and γ_1 when the true class is 1. The final distribution we need to consider is $P(y)$, the prior probability of class y.

$$P(y=1) = \pi, \quad (2)$$
$$P(y=0) = (1-\pi). \quad (3)$$

The above definitions and equations provide us with all the details we require to perform inference and learning for a classification problem with noisy labels. However it is convenient to consider a slightly different factorisation of the joint distribution to implement the algorithm.

Whilst it may make intuitive sense to model the addition of label noise as we have described above, in a practical setting the prior distribution parameter, π, must be estimated. Typically such priors are estimated by considering the proportion of the data-set in each class, i.e., if out of a data-set of N points N_1 belong to class 1, π is estimated as $\frac{N_1}{N}$. When the labels are noisy however, such an estimate does not always makes sense. It is easier to directly estimate the parameters of $P(\hat{y})$. Making use of the above definitions we can obtain the prior associated with the observed labels:

$$P(\hat{y}=1) = (1-\pi)\gamma_0 + \pi(1-\gamma_1) \stackrel{\text{def}}{=} \hat{\pi}, \quad (4)$$
$$P(\hat{y}=0) = (1-\hat{\pi}). \quad (5)$$

In fact we can rewrite our factorisation of the joint distribution, utilising $P(\hat{y})$ (Figure 2) without affecting in any way our model's representative power.

The conditional distribution $P(y|\hat{y})$ can then be parameterised

		y	
		0	1
$\hat{y}$	0	$(1-\hat{\gamma}_0)$	$\hat{\gamma}_0$
	1	$\hat{\gamma}_1$	$(1-\hat{\gamma}_1)$.

where it can be shown that

$$\hat{\gamma}_1 = \frac{(1-\pi)\gamma_0}{(1-\pi)\gamma_0 + \pi(1-\gamma_1)}, \quad (6)$$
$$\hat{\gamma}_0 = \frac{\pi\gamma_1}{(1-\pi)(1-\gamma_0) + \pi\gamma_1}. \quad (7)$$

In summary, our model is a simple probabilistic model of the class conditional densities and the noise generating process. Next, we must consider how learning can occur in such a model.

3. Optimising the Model Parameters

The process of optimising a generative model for classification normally involves the fitting of the class conditional densities, $p(\mathbf{x}|y)$, through maximisation of a likelihood or log-likelihood function. Typically, for a data-set containing N points, $\{\mathbf{x}_n, y_n\}_{n=1}^N$, the log-likelihood objective function, $L(\boldsymbol{\theta})$ will take the form[1]

$$L(\boldsymbol{\theta}) = \sum_{n=1}^{N} \ln p(\mathbf{x}_n|y_n, \boldsymbol{\theta}), \quad (8)$$

where $\boldsymbol{\theta}$ are the parameters of the class conditional density. In the noisy label case, therefore, it seems reasonable to fit the perceived class conditional density, $p(\mathbf{x}|\hat{y})$, in a similar manner.

$$L(\boldsymbol{\theta}) = \sum_{n=1}^{N} \ln p(\mathbf{x}_n|\hat{y}_n, \boldsymbol{\theta}). \quad (9)$$

To determine each $p(\mathbf{x}_n|\hat{y}_n, \boldsymbol{\theta})$ we must marginalise the associated latent variable, y_n. This marginalisation is often performed in combination with the optimisation through an EM algorithm (Dempster et al., 1977). Here, rather than performing the marginalisation explicitly, we optimise a modified form of the log-likelihood

$$L(\boldsymbol{\theta}) \geq \sum_{y} Q(y|\mathbf{x}, \hat{y}) \ln p(\mathbf{x}, y|\hat{y}, \boldsymbol{\theta}) + \mathcal{H}(Q(y|\mathbf{x}, \hat{y})) \stackrel{\text{def}}{=} \mathcal{L}(\boldsymbol{\theta}), \quad (10)$$

where $\mathcal{H}(p(\cdot))$ is the entropy of $p(\cdot)$ and we have dropped the index n for clarity of notation. The bound becomes an equality if $Q(y|\mathbf{x}, \hat{y}) = P(y|\mathbf{x}, \hat{y}, \boldsymbol{\theta})$. Maximisation of

[1] Note that this equation implies that the label noise is independent across different data-points. More complex relationships are conceivable, and in many situations they are warranted, but they are beyond the scope of this paper.

the functional consists of two steps. The 'expectation' or E-step consists simply of the computation of the posterior distribution of the latent variable y,

$$P(y|\mathbf{x}, \hat{y}, \boldsymbol{\theta}) = \frac{p(\mathbf{x}, y|\hat{y}, \boldsymbol{\theta})}{p(\mathbf{x}|\hat{y}, \boldsymbol{\theta})}, \tag{11}$$

and then by setting $Q(y|\mathbf{x}, \hat{y}) = P(y|\mathbf{x}, \hat{y}, \boldsymbol{\theta})$ the bound is made into an equality. The 'maximisation step' or M-step is then the optimisation of the function $\mathcal{L}(\boldsymbol{\theta})$ with respect to the parameters, $\boldsymbol{\theta}$. For our model, we achieve this through differentiation and rearranging to form the following fixed point update equations:

$$\mathbf{m}_y = \frac{1}{\nu_y} \sum_{n=1}^{N} P(y|\mathbf{x}_n, \hat{y}_n)\mathbf{x}_n, \tag{12}$$

$$\boldsymbol{\Sigma}_y = \frac{1}{\nu_y} \sum_{n=1}^{N} P(y|\mathbf{x}_n, \hat{y}_n)(\mathbf{x}_n - \mathbf{m}_y)(\mathbf{x}_n - \mathbf{m}_y)^{\mathrm{T}}, \tag{13}$$

$$\hat{\gamma}_0 = \frac{1}{\nu_y} \sum_{n=1}^{N} P(y|\mathbf{x}_n, \hat{y}_n)(1 - y)\hat{y}_n, \tag{14}$$

$$\hat{\gamma}_1 = \frac{1}{\nu_y} \sum_{n=1}^{N} P(y|\mathbf{x}_n, \hat{y}_n)(1 - \hat{y}_n)y, \tag{15}$$

where $\nu_y = \sum_{n=1}^{N} P(y|\mathbf{x}_n, \hat{y}_n)$ is the expected number of points in class y. The update equations for our EM algorithm are very similar to those for EM in mixtures of Gaussians.

To implement the algorithm we must first initialise the parameters, perhaps by assuming that the labels are noiseless and then computing the covariances and means of the Gaussian distributions. The posterior probability of each data-point may then be computed (the E-step) and used in the above update equations to obtain new parameter values through optimisation of the fixed point equations (the M-step). E and M-steps are then alternated until convergence is achieved.

In the next section we demonstrate our approach on a simple toy data-set.

4. Toy Problem

To create our toy problem, we sampled from two Gaussian distributions, one of which contains strong correlations ($y = 0$) and the other of which contains weaker correlations ($y = 1$). Each data-point was four times more likely to be sampled from the weakly correlated Gaussian than the strongly correlated Gaussian, i.e., $\pi = 0.8$. In total two hundred data-points were sampled. The sampled data is shown in Figure 3 (b), where the dots are from weakly correlated Gaussian and the crosses are from the strongly correlated one. The labels of the data-set were then flipped with a probability of 0.2 ($\gamma_0 = \gamma_1 = 0.2$) to form the observed labels, $\hat{y}$. The observed labels are shown in Figure 3 (a).

Two models were trained on the data. The first involved fitting Gaussian class conditional distributions to the data without modelling the label noise. The Gaussian distributions learnt by this model are represented in Figure 4(a) as solid lines. Also shown are the true underlying Gaussians which generated the data (dotted lines). The second model made use of our model for label noise (Figure 4(b)).

Initialisation For training the model label noise was initialised with $\hat{\gamma}_0 = \hat{\gamma}_1 = 0.3$. The parameters of the Gaussians were initialised as if there was no label noise present.

Training The EM algorithm was considered to have converged when both the log-likelihood and the parameters changed by less than 10^{-2}. Convergence occurred after 37 iterations of the EM algorithm.

Results The statistics learnt by the algorithm are summarised in Table 1 along with statistics for data-sets with more data-points. Note that as N is increased, the statistics improve.

Table 1. Examples of the different statistic estimates against varying numbers of data-points, N, along with the true values of the statistics.

N	200	2000	20000	TRUE
$\hat{\pi}$	0.6550	0.6755	0.6805	0.6800
$\hat{\gamma}_0$	0.4106	0.4746	0.5084	0.5000
$\hat{\gamma}_1$	0.0902	0.0669	0.0504	0.0508

Discussion Whilst these results are encouraging, we should stress that real world problems are unlikely to be as easily modelled as the toy problem we have proposed. If the true class conditional densities are not well captured by the Gaussian model, the label noise will be over or under estimated. It is crucial to our approach that the complexity of our approximating distributions is well matched to the true underlying distributions. Gaussians are clearly not flexible enough to fulfill this criterion. For many real world problems, they will provide inaccurate estimates of the posterior $P(y|\mathbf{x}, \hat{y})$. If the approximating distributions are overly complex, these values will be over-estimated. If the converse is true the values will be underestimated. To resolve this issue we must utilise a more flexible model of the class conditional densities. In the next section we achieve this aim by seeking to model the class conditional densities of the input data mapped into a high dimensional feature space, rather than the input values directly.

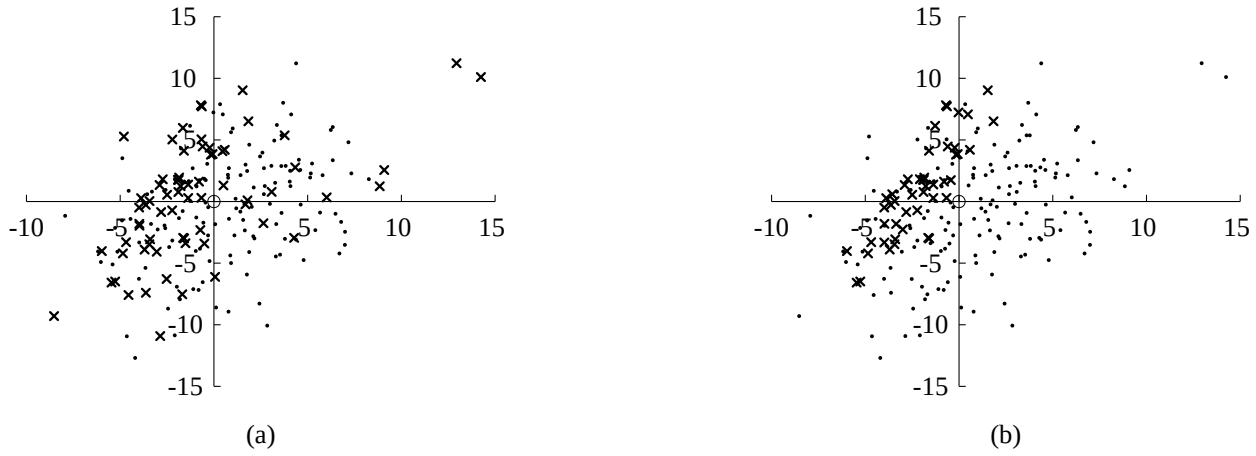

Figure 3. Two hundred data-points sampled the strongly and weakly correlated Gaussian. The observed labels associated with the data-points are depicted by showing each point as either a dot, or a cross. In (b) we show the true labellings of the data. In (a) the noisy labels are shown.

5. Data Models in Feature Space

In this section we are going to make use of 'the kernel trick' to model our data in a high dimensional feature space. The idea is that using a positive definite kernel k corresponds to mapping the data into the feature space by a map Φ, and taking a dot product in that space,

$$k(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x})^{\mathrm{T}}\Phi(\mathbf{x}'). \tag{16}$$

There is a large class of kernels that can be used, leading to a large class of available feature spaces. This allows us to consider a range of more complex models, but carries with it some further problems. We will seek to fit Gaussian distributions to the mapped data, however, in general, we will be unable to calculate the likelihood of the data under these distributions. The reason for this is that the kernel only allows us to work with the data implicitly, in terms of dot products. Only operations that can be reduced to dot products are feasible (Schölkopf et al., 1998). In the present case, we must exploit the fact that, when the Gaussians are subjected to certain constraints, incomputable terms in the likelihoods will cancel when we consider ratios between these likelihoods. These ratios are of the type we require to perform EM updates.

Consider first a further constraint imposed on the Gaussian class conditional densities. We constrain the covariance of the Gaussians to be equal, $\mathbf{\Sigma}_W = \mathbf{\Sigma}_1 = \mathbf{\Sigma}_0$, where we call $\mathbf{\Sigma}_W$ the within class covariance matrix. With this constraint the update equation for $\mathbf{\Sigma}_W$ becomes

$$\mathbf{\Sigma}_W = \frac{1}{N}\sum_y \sum_{n=1}^{N} P(y_n|\mathbf{x}_n, \hat{y}_n)(\mathbf{x}_n - \mathbf{m}_y)(\mathbf{x}_n - \mathbf{m}_y)^{\mathrm{T}}. \tag{17}$$

At first sight this appears to be a foolish thing to do. We have further constrained the complexity of our model. The advantage is that the model may now be easily be 'kernelised'. In the absence of label noise ($\gamma_0 = \gamma_1 = 0$) the model is recognised as a Fisher discriminant (FD) (Fisher, 1936; Fukunaga, 1990). Several authors (Mika et al., 1999; Roth & Steinhage, 2000; Baudat & Anouar, 2000) have shown how the Fisher discriminant may be kernelised. The main trick centres around using the *Rayleigh coefficient*,

$$J(\mathbf{w}) = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{\Sigma}_B\mathbf{w}}{\mathbf{w}^{\mathrm{T}}\mathbf{\Sigma}_W\mathbf{w}}, \tag{18}$$

where $\mathbf{\Sigma}_B$ is known as the between class covariance matrix, and is given by

$$\mathbf{\Sigma}_B = (\mathbf{m}_1 - \mathbf{m}_0)(\mathbf{m}_1 - \mathbf{m}_0)^{\mathrm{T}}. \tag{19}$$

When performed in the feature space, it has been found advantageous to add a small regularisation term to the denominator of Eqn (18), either equal to a multiple of the identity matrix, or of the kernel Gram matrix (Mika et al., 1999).

Fisher's discriminant may be computed through maximisation of the Rayleigh coefficient with respect to $\mathbf{w}$. Naturally in the standard version of the Fisher discriminant the means ($\mathbf{m}_0$ and $\mathbf{m}_1$) and the covariances ($\mathbf{\Sigma}_W$ and $\mathbf{\Sigma}_B$) are not the posterior weighted versions, however it can be shown that the M-step in our EM version of the Fisher discriminant merely involves substituting the posterior weighted versions of these parameters.

The optimum value of $\mathbf{w}$ can be shown to be proportional to $\mathbf{\Sigma}_W^{-1}(\mathbf{m}_0 - \mathbf{m}_1)$. This vector defines a discriminating hyper-plane. Kernelisation of the Fisher discriminant relies

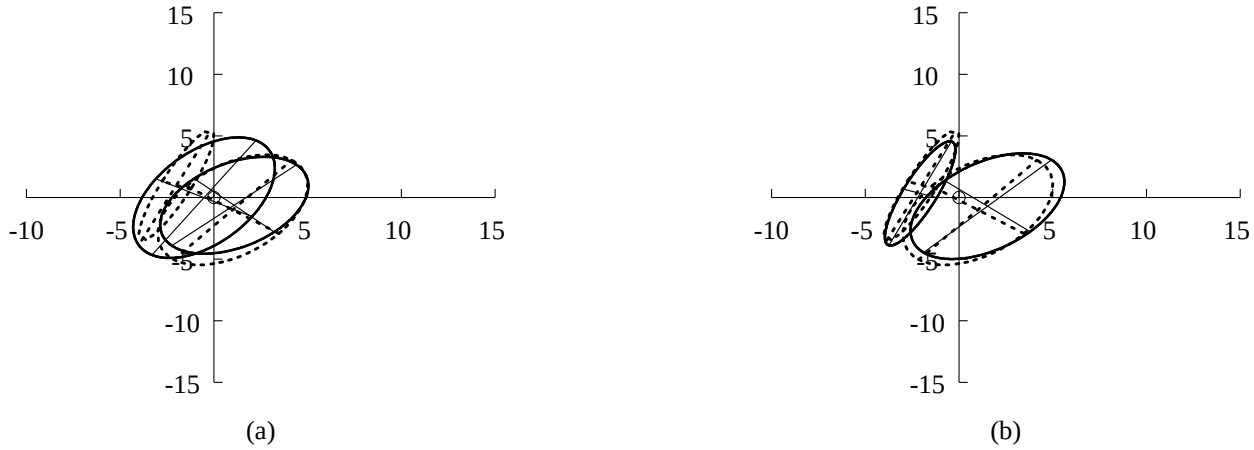

Figure 4. In (a) Gaussians have simply been fitted to the data from Figure 3(a) assuming that there is no label noise. In (b) our approach has been utilised to learn the label noise. The covariance of the Gaussians which generated the data are marked as dotted ellipses. The lines in the ellipses represent the eigenvalues and eigenvectors of the covariance matrices. These lines cross at the mean of the generating Gaussian. The learnt covariances are represented by solid ellipses. Note how the learnt covariances correspond to the true covariances much better in (b) than in (a).

on expanding the direction of discrimination as

$$\mathbf{w} = \sum_{n=1}^{N} \alpha_n \mathbf{x}_n \tag{20}$$

and substituting this into the Rayleigh coefficient. The resulting matrices of dot products can be replaced with the so-called Gram matrix $(k(\mathbf{x}_i, \mathbf{x}_j))_{ij}$ in order to perform the algorithm in the feature space.

Kernelising the model introduces the flexibility required to model a large variety of class conditional densities. The complexity of the class conditional densities can be controlled by varying the parameters of the kernel matrix, e.g. in a squared exponential kernel[2] we may vary the width parameter. Kernelisation of the preceding update equations proceeds in a straightforward manner, similar to that of the standard kernel Fisher discriminant (KFD).

Whilst likelihoods under the class conditional distribution, $p(\mathbf{x}|\hat{y})$, may not be computable, ratios of the likelihoods may be computed because the determinant will cancel. We may thus obtain the necessary posterior probabilities in Eqn (11). It is for this reason that we constrain the class conditional variances to be equal.[3]

One problem with kernelisation of the algorithm is that it increases the complexity from $O(I^3)$, where I is the number of input features, to $O(N^3)$. As we shall see, in the applications we envisage, this may be a serious handicap. For the moment though we do not seek to fully resolve this issue. We simple sub-sample the matrix in a random manner to reduce the computational complexity.

6. Method Evaluation

Now that our entire algorithm is described we turn to two further evaluations of the approach. The first, an artificial OCR problem, is purely illustrative. The second however, an image understanding problem, shows how the algorithm may be utilised in practice.

6.1 Artificial Digits Problem

To understand better the characteristics of our approach we took data from the US Postal Service CEDAR CD-ROM of the digits 0 and 1. The data was pre-processed so that the digits were 16×16 gray-scale images. We then added artificial label noise to the data and trained a KFD using a squared exponential kernel[4] $k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-\|\mathbf{x}-\mathbf{x}'\|^2}{128}\right)$. To decrease computation time we constrained $\mathbf{w}$ to lie in the spaced spanned by only 200 of the possible feature vectors[5], i.e. if we define the set of 200 sub-sampled data points to be Ξ,

$$\mathbf{w} = \sum_{n \in \Xi} \alpha_n \mathbf{x}_n, \tag{21}$$

reducing the computational complexity to $O(200^3)$. We implemented our algorithm, initialising $\hat{\gamma}_0 = \hat{\gamma}_1 = 0.1$, as

[2] Sometimes referred to as a Gaussian kernel.

[3] Other workarounds are possible and being considered, see Tipping, 2001.

[4] This kernel (with the width 128) has proved effective in previous works (Schölkopf et al., 1997) on data of the same dimension.

[5] Another way of looking at this is that we *a priori* constrain all but two hundred of the α_ns to be zero.

well as the standard KFD (i.e., $\hat{\gamma}_0 = \hat{\gamma}_1 = 0$). Ten different models were trained using different sub-samples of the kernel matrix. The label noise modelling results utilised a maximum of ten iterations of the EM algorithm. Predictions were then made on the test set, to which no label noise had been added. The results are depicted in Figure 5 and the region of interest (between label noise of 0.4 and 0.6) is shown at a larger scale in Figure 6.

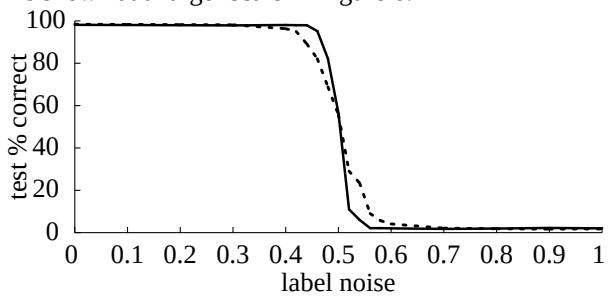

Figure 5. Plot of test set error vs. induced label noise for the digit classification problem. Dotted line is the standard KFD, solid line is KFD with label noise.

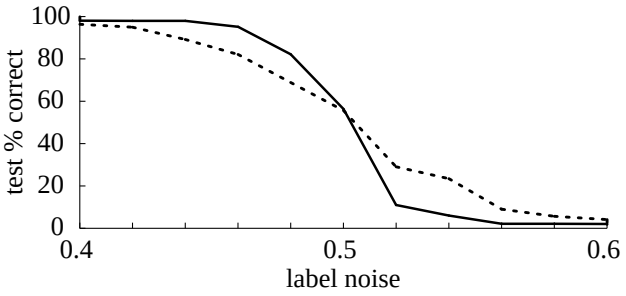

Figure 6. Detail of the test set error vs. induced label noise plot between noise of 0.4 and 0.6. Dotted line is the standard KFD, solid line is KFD with label noise.

Whilst the label noise modelling KFD did perform better in some regions than the standard approach, the standard KFD did slightly better up to about 30 % label noise. However, the performance of the standard KFD did drop off fairly rapidly after this point whilst the label noise modelling approach was almost unaffected up to a label noise of around 44 %.

6.2 Real-world Image Problem

Whilst in the digit problem above some improvement was seen in the performance of the model for large levels of label noise, the performance of the standard KFD was very good for low levels of label noise. If the label noise is the product of occasional mis-labellings by a human expert, therefore, we are unlikely to be operating in the central regions of Figure 5 so where is the merit in the approach?

Let us formulate our classification task with the merits of our new algorithm in mind. Consider an image problem where the task is to label pixels as being 'sky' or 'not sky'. The standard machine learning approach to the problem is to create a training set containing labelled examples of sky pixels and not-sky pixels. However the creation of such a training set by a human expert is a wearisome task. The task is much simplified if the expert merely has to say whether the picture contains sky or not. If the data is labelled in this manner on an image by image basis, yet we apply the labels to the data on a pixel by pixel basis we have constructed a problem with a noisy labelling. The noise in the not-sky class, $\hat{y} = 0$, is, $\hat{\gamma}_0 = 0$. However the noise in the sky class, $\hat{y} = 1$, is large. If the proportion of sky in those pictures containing sky is typically 30% then we have $\hat{\gamma}_1 = 0.7$. Whilst this noise level is greater than 50%, learning should still be possible because the other class is noiseless.

We tested our approach on a data-set of construction images taken from The Corel Gallery 1,000,000 Collection (Corel Corporation Ltd, 1998). In the training set there were twenty images containing sky and twenty images without sky. The input data was taken from a 9×9 block of pixels. The aim was to predict the central pixel's label, i.e., the true label was considered to be sky if the central pixel was thought to be from a sky portion of the picture. Not all the pixels in the block were utilised, they were sub-sampled as in the Figure 7. The seventeen sub-samples across three channels (red, green and blue) led to 51 input values for each data-point. The full training data-set was

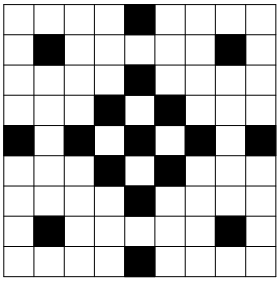

Figure 7. Arrangement of the pixels from each 9×9 block. Black pixels are those that were utilised.

uniformly sub-sampled to obtain the actual training data. The images were 256×384 pixels in size which would have led to $98,304$ data-points in each image. We used $1,176$ uniformly spaced samples from each image leading to a total training set of $47,040$ points. This sub-sampling also leads to greater independence between the data-points. Once again we sub-sampled kernels for computational reasons. Once again, for computational reasons, we assumed *a priori* that the solution for $\mathbf{w}$ lay in the space spanned by only 200 of the possible $47,040$ data-points. A squared exponential kernel was used, the width of which was chosen

as 15 through using a *labelled validation set* of 20 images, sampled in the same manner to the training set. The performance of the algorithms was evaluated on a *labelled test set* of a further 20 images, again pre-processed in a similar manner to the validation and training sets.

We compared both a FD and KFD with label noise to the standard KFD algorithm. We also trained a standard FD and KFD on a labelled version of the data-set. The results are summarised in Table 2.

Table 2. Summary of results for the image understanding problem. In the table, 'labelled data' refers to the case where the training set was labelled pixel-wise by a human observer.

Method	Test accuracy
Standard KFD	84.5%
FD with label noise	91.1%
KFD with label noise	94.2%
FD with labelled data	91.9%
KFD with labelled data	95.9%
(Fraction of sky	34.2%)

In Figure 8 and Figure 9 we show two of the original test images alongside the corresponding images with the sky removed.

Note that our kernelised and linear approaches obtained 98.2 % and 99.1% respectively of the accuracy of the corresponding fully labelled data approaches with much less effort from the human expert.

7. Discussion

To summarise, we have proposed an algorithm for performing classification in the presence of noisy data labels. Initial results are promising. Whilst there appears to be little benefit in utilising the approach when the label noise is small, we have shown how modelling the noise allows us to perform 'sloppy labellings' like in the sky problem. For an alternative approach to this type of problem see (Keeler et al., 1991; Maron & Lozano-Prez, 1998).

Issues remain with the approach. Standard cross validation techniques do not really make sense for model selection with the algorithm. In our sky example we made use of a validation set to determine the correct model complexity. This rather basic approach could be refined by incorporating a proportion of the accurately labelled data with the sloppily labelled data.

The algorithm may also be utilised to perform active learning (Cohn et al., 1996; Freund et al., 1997). In active learning the aim is to select data for labelling such that it would provide the most information for the classifier. Data about which the algorithm is uncertain, i.e., the posterior of the true class label is close to 0.5, could be passed to a human expert for accurate labelling.

The approach could also use unlabelled data, in which case the label noise could be initialised in accordance with the class prior probabilities (see also Roth & Steinhage, 2000).

We mentioned earlier the computational problems associated with large data-sets and kernel methods. Future work will focus on resolving these issues (see for example Smola & Schölkopf, 2000).

Acknowledgements The authors would like to thank Mike Tipping and Ralf Herbrich for helpful discussions.

References

Baudat, G., & Anouar, F. (2000). Generalized discriminant analysis using a kernel approach. *Neural Computation, 12*, 2385–2404.

Cohn, D. A., Ghahramani, Z., & Jordan, M. I. (1996). Active learning with statistical models. *Journal of Artificial Intelligence Research, 4*, 129–145.

Corel Corporation Ltd (1998). Corel GALLERY Magic 1,000,000. `http://www.corel.com`. Superceded by Corel GALLERY 1,300,000.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, B, 39*, 1–38.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics, 7*, 179–188. Reprinted in *Contributions to Mathematical Statistics*, John Wiley: New York (1950).

Freund, Y., Seung, H. S., Shamir, E., & Tishby, N. (1997). Selective sampling using the query by committee algorithm. *Machine Learning, 20*, 133–168.

Fukunaga, K. (1990). *Introduction to statistical pattern recognition*. San Diego, CA: Academic Press. 2nd edition.

Graepel, T., & Herbrich, R. (2001). The kernel gibbs sampler. In (Leen et al., 2001). In press.

Guyon, I., Matic, N., & Vapnik, V. (1996). Discovering informative patterns and data cleaning. In U. Fayyad, G. Piatetsky-Shapiro, P. Smyth and R. Uthurusamy (Eds.), *Advances in knowledge discovery and data mining*, 181–203. Cambridge, MA: MIT Press.

Figure 8. A test image with the sky removed, number 85046 from the Corel Gallery 1,000,000 Collection.

Figure 9. A test image with the sky removed, number 667056 from the Corel Gallery 1,000,000 Collection.

Keeler, J. D., Rumelhart, D. E., & Leow, W.-K. (1991). Integrated segmentation and recognition of hand-printed numerals. *Advances in Neural Information Processing Systems* (pp. 557–563). San Mateo, CA: Morgan Kauffman.

Leen, T. K., Dietterich, T. G., & Tresp, V. (Eds.). (2001). *Advances in neural information processing systems*, vol. 13. Cambridge, MA: MIT Press. In press.

Maron, O., & Lozano-Prez, M. (1998). A framework for multiple-instance learning. *Advances in Neural Information Processing Systems* (pp. 570–576). Cambridge, MA: MIT Press.

Mika, S., Rätsch, G., Weston, J., Schölkopf, B., & Müller, K.-R. (1999). Fisher discriminant analysis with kernels. *Neural Networks for Signal Processing IX* (pp. 41–48). IEEE.

Norton, S. W., & Hirsh, H. (1993). Learning DNF via probabilistic evidence combination. *Proceedings of the International Conference in Machine Learning* (pp. 220–227). San Fransisco, CA: Morgan Kauffman.

Roth, V., & Steinhage, V. (2000). Nonlinear discriminant analysis using kernel functions. *Advances in Neural Information Processing Systems* (pp. 568–574). Cambridge, MA: MIT Press.

Schölkopf, B., Smola, A. J., & Müller, K.-R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, *10*, 1299–1319.

Schölkopf, B., Sung, K.-K., Burges, C. J. C., Girosi, F., Niyogi, P., Poggio, T., & Vapnik, V. N. (1997). Comparing support vector machines with Gaussian kernels to radial basis function classifiers. *IEEE Transactions on Signal Processing*, *45*, 2758–2765.

Smola, A. J., & Schölkopf, B. (2000). Sparse greedy matrix approximation for machine learning. *Proceedings of the International Conference in Machine Learning* (pp. 911–918). San Francisco, CA: Morgan Kauffman.

Tipping, M. E. (2001). Sparse kernel principal component analysis. In (Leen et al., 2001). In press.