Robust subspace mixture models using t-distributions

Dick de Ridder* and Vojtěch Franc†
* Pattern Recognition Group, Imaging Science & Technology Dept.
Delft University of Technology
Lorentzweg 1, 2628 CJ Delft, The Netherlands
dick@ph.tn.tudelft.nl
http://www.ph.tn.tudelft.nl/~dick
† Center for Machine Perception, Dept. of Cybernetics,
Faculty of Electrical Engineering, Czech Technical University
121 35 Praha 2, Karlovo nám. 13, Czech Republic
xfrancv@cmp.felk.cvut.cz
http://cmp.felk.cvut.cz/~xfrancv

Abstract

Probabilistic subspace mixture models, as proposed over the last few years, are interesting methods for learning image manifolds, i.e. nonlinear subspaces of spaces in which images are represented as vectors by their grey-values. However, for many practical applications, where outliers are common, these methods still lack robustness. Here, the idea of robust mixture modelling by t-distributions is combined with probabilistic subspace mixture models. The resulting robust subspace mixture model is shown experimentally to give advantages in density estimation and classification of image data sets.

1 Introduction

The last few years have seen an increasing interest in subspace methods, specifically when applied to image data. Many approaches approximate image data, represented by grey-values as vectors in high-dimensional spaces, using linear subspaces. These can be found using a number of well-known subspace techniques, such as principal component analysis (PCA) or factor analysis (FA). The subspaces themselves can be used as models for the projected image data (so-called appearance-based methods), or they can be used as a form of feature extraction to ease further processing.

Work has also proceeded on modelling nonlinear subspaces, i.e. curves or manifolds. Methods to do so broadly fall into three main categories: fitting principal curves and surfaces to the data; embedding the data; and modelling the data by (constrained) mixture models. This latter group of methods has received much attention. Most mixture models approximate a manifold using a relatively small number of localised subspaces, which are modelled by Gaussians with restricted covariance matrices [12; 3], possibly with a constraint on the relations between the models [1; 9]. Their popularity is due to the fact

BMVC 2003 doi:10.5244/C.17.35

that they are usually trained by the well-understood expectation-maximisation (EM) algorithm, for which a large body of literature exists. Methods of optimising the number of models to use, be it by using greedy algorithms or Bayesian learning, are readily available.

However, mixture models still have their drawbacks. First, the EM algorithm can only be guaranteed to converge to a *local* optimum. Second, as the models are density-based, dense data sets are necessary to train them. Finally, use of Gaussian densities means these methods are not robust to outliers. Especially when mixture models are used for subsequent classification this can lead to performance degradation. Although supervised mixture model techniques exist (e.g. [5; 10]), these algorithms are complex, computationally intensive and/or limited to specific data models.

In this paper, the problem of robustness to outliers will be addressed. First, an overview of probabilistic subspace models will be given in Section 2. Inspired by recent developments in statistics [8], a mixture of subspace t-distributions will be proposed in Section 3. This method can be shown to be much more robust against outliers than methods based on Gaussian densities. Its usefulness will be demonstrated on a toy example and a real-world problem in Section 4. The paper ends with conclusions and an outlook to further applications.

2 Subspace mixture models

2.1 Probabilistic subspace models

The two main models for mixtures of subspaces are mixtures of probabilistic principal component analysers (PPCA, [12]) and mixtures of factor analysers (FA, [3]). Both are based on a subspace model in which an observed variable $\vec{x} \in \mathbb{R}^d$ is generated by a some low-dimensional variable $\vec{s} \in \mathbb{R}^m$, where typically $m \ll d$:

$$\vec{x} = \vec{A}\vec{s} + \vec{\mu} + \vec{\varepsilon} \tag{1}$$

The low-dimensional variable is thus shifted w.r.t. to the center $\vec{\mu}$ of the subspace, mapped into the high-dimensional space by a projection operator $\vec{A}$, and i.i.d. Gaussian noise $\vec{\varepsilon}$ with $p(\vec{\varepsilon}) = N(\vec{\varepsilon}; \vec{0}, \vec{\Psi})$ is added. The distributions of the variables are:

$$p(\vec{s}) = N(\vec{s}; \vec{0}, \vec{I}) \tag{2}$$

$$p(\vec{x}|\vec{s}) = N(\vec{x}; \vec{A}\vec{s} + \vec{\mu}, \vec{\Psi}) \tag{3}$$

As both $p(\vec{s})$ and $p(\vec{x}|\vec{s})$ are Gaussian, the marginal distribution of $\vec{x}$ will be so as well:

$$p(\vec{x}) = \int p(\vec{x}|\vec{s}) p(\vec{s}) d\vec{s} \tag{4}$$

whose mean and covariance matrix can easily be found from (1):

$$E(\vec{x}) = \vec{\mu} \quad \text{and} \quad E(\vec{x}\vec{x}^T) = \vec{A}E(\vec{s}\vec{s}^T)\vec{A}^T + \vec{\Psi} = \vec{A}\vec{A}^T + \vec{\Psi} \tag{5}$$

i.e., $\vec{x}$ is distributed as:

$$p(\vec{x}) = N(\vec{x}; \vec{\mu}, \vec{A}\vec{A}^T + \vec{\Psi}) \tag{6}$$

The difference between principal component analysis and factor analysis lies in the assumed noise model. FA assumes $\vec{\Psi}$ to be a diagonal matrix, whereas PPCA assumes $\vec{\Psi}$ to be a multiple of the identity matrix, $\sigma^2\vec{I}$. FA thus models the individual noise level in each of the dimensions (i.e. pixels, for image data), PPCA assumes all dimensions have an equal noise level. In this paper, only PPCA will be discussed; however, all observations are equally applicable to FA.

2.2 The EM algorithm for mixtures of PPCAs

Tipping and Bishop [12] derived an EM algorithm for maximum likelihood learning of a mixture of k PPCA's. It will be summarised here in enough detail to allow the reader to understand the effects of using a different density model later on; for a more detailed derivation, see the original paper. The probability of observing a sample $\vec{x}$ is given by:

$$p(\vec{x}) = \sum_{j=1}^{k} \pi_j p(\vec{x}|j) \tag{7}$$

where the π_j are mixing parameters, for which $\sum_{j=1}^{k} \pi_j = 1$, and $p(\vec{x}|j)$ is the probability of $\vec{x}$ under model j given by (6). The log-likelihood of observing a data set $\vec{X} = \{\vec{x}_1, \ldots, \vec{x}_n\}$ is:

$$\mathscr{L}(\vec{X}) = \log \prod_{i=1}^{n} p(\vec{x}_i) = \sum_{i=1}^{n} \log \sum_{j=1}^{k} \pi_j p(\vec{x}_i|j) \tag{8}$$

To maximise (8) w.r.t. the parameters $\{\pi_j, \vec{\mu}_j, \vec{A}_j, \sigma_j^2\}$, $j = 1, \ldots, k$, a hidden indicator variable z_{ij} is introduced, which is 1 when sample $\vec{x}_i$ is generated by model j, and 0 otherwise; the mixing parameters π_j are then equal to $p(z_{ij} = 1)$. The complete-data distribution can now be written as:

$$p(\vec{x}, \vec{s}, \vec{z}) = \prod_{i=1}^{n} p(\vec{x}_i|\vec{s}_i, \vec{z}_i) p(\vec{s}_i|\vec{z}_i) p(\vec{z}_i) = \prod_{i=1}^{n} \prod_{j=1}^{k} [\pi_j p(\vec{x}_i|\vec{s}_i, j) p(\vec{s}_i|j)]^{z_{ij}} \tag{9}$$

where n is the number of samples in the dataset, $p(\vec{x}_i|\vec{s}_i, j)$ and $p(\vec{s}_i|j)$ are given by (3) and (2), respectively, and $p(\cdot|j)$ is short for $p(\cdot|z_{ij} = 1)$.

The EM algorithm uses the fact that the maximum of the log of (9) can easily be found analytically, to maximise (8) w.r.t. the parameters [7]. The M step of the EM algorithm for mixtures of PPCAs [12] actually consists of two steps: in the first, only the mixing parameters π_j and the means $\vec{\mu}_j$ are updated; in the second, new values for the $\vec{A}_j$ and σ_j^2 are found. Combined, this gives:

- **E step**: for all i and j, calculate the posterior responsibility of each model for each sample, i.e. the expectation of z_{ij}:

$$r_{ij} = E(z_{ij}) = \frac{\pi_j p(\vec{x}_i|j)}{p(\vec{x}_i)} = \frac{\pi_j p(\vec{x}_i|j)}{\sum_{l=1}^{k} \pi_l p(\vec{x}_i|l)} \tag{10}$$

- **M step**: for all j, update the estimates of the parameters (a prime indicating new parameters):

$$\pi_j' = \frac{1}{n} \sum_{i=1}^{n} r_{ij} \quad \text{and} \quad \vec{\mu}_j' = \frac{\sum_{i=1}^{n} r_{ij} \vec{x}_i}{\sum_{i=1}^{n} r_{ij}} \tag{11}$$

The per-model weighted sample covariance matrix $\vec{S}'_j$,

$$\vec{S}'_j = \frac{\sum_{i=1}^{n} r_{ij}(\vec{x}_i - \vec{\mu}'_j)(\vec{x}_i - \vec{\mu}'_j)^T}{\sum_{i=1}^{n} r_{ij}} \tag{12}$$

can then be used to find $\vec{A}'_j$ and $\sigma_j^{2'}$, using normal eigendecomposition:

$$\vec{A}'_j = \vec{E}_j(\vec{\Lambda}_j - \sigma_j^{2'}\vec{I})^{\frac{1}{2}} \quad \text{and} \quad \sigma_j^{2'} = \frac{1}{d-m}\sum_{l=m+1}^{d} \lambda_l \tag{13}$$

where $\vec{\Lambda}_j$ contains the m leading eigenvalues of $\vec{S}'_j$ on its diagonal, $\vec{E}_j$ the corresponding eigenvectors, and the λ_l^j are the trailing eigenvalues.

Tipping and Bishop actually describe a faster version of the EM algorithm as well, updating $\vec{A}_j$ and σ_j^2 iteratively. However, this still uses just $\vec{S}'_j$.

2.3 Application to manifold learning

The mixture model described above is clearly a mixture of Gaussians, in which the covariance matrices are restricted to a specific form, controlled by the parameter m. Although the Gaussian distribution is mathematically elegant and allows one to derive the EM algorithm above, it is not necessarily optimal for manifold learning. The problem is that it may assign high probability to large empty volumes in space, in the presence of a few outliers. This is illustrated in Figure 2(a)-(c) for a simple 2D problem, where 200 samples are distributed along 2 1D curves, with some added Gaussian noise. When just a few uniformly distributed outliers are present, one or more of the Gaussians are used to model these. The resulting model no longer describes the manifolds well.

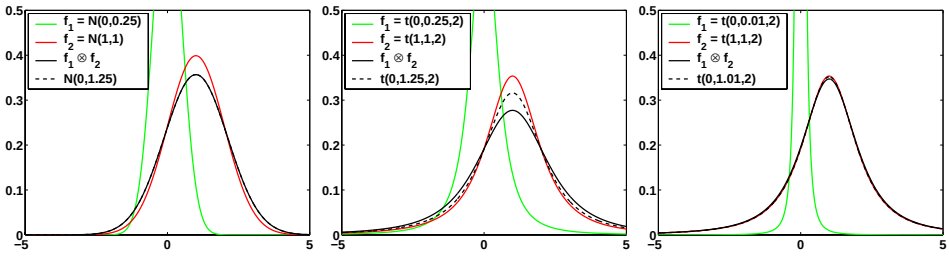
What is needed for cases such as these is a more robust mixture model. There has recently been interest in the vision and statistics communities in modelling single subspaces more robustly (e.g. [2; 6]), as manifold learning is applied to increasingly more problems[1]. However, many of the proposed techniques are cumbersome, iterative, take a large amount of computation or are not applicable in a mixture model.

We propose that for manifold learning, finding an exact local PCA solution is not necessary, *as long as the main axes of the densities are aligned with the manifold*. Then, more robust densities can be used in the mixture, resulting in models that will mainly assign high probability to samples on the manifold. From the statistics literature, there is a wide range of possible robust approaches (see e.g. [4]). A simple solution in mixture modelling consists of adding a uniform density into the mixture to capture the outliers. The problem with this approach is that the range over which this uniform distribution is defined will have to be set, which is not trivial in high-dimensional spaces. Another interesting approach is to use two Gaussian components per model in the mixture:

$$p(\vec{x}) = (1-c)N(\vec{x};\vec{\mu},\vec{C}) + cN(\vec{x};\vec{\mu},\alpha\vec{C}) \tag{14}$$

Here, the second Gaussian is used to model outliers. Obviously, this leads to the question how to set or learn α. Below, a generalised and more elegant implementation of this idea, using the t-distribution, will be discussed.

[1] Note that not all robust approaches are only concerned with ignoring outliers; some also aim at being robust against outlying pixel values. Robustness in this sense is not addressed in this work.



(a) (b) (c)

Figure 1: (a) The convolution of two Gaussians $N(x;\mu,\sigma^2)$ is itself a Gaussian; (b) in general, the convolution of two t-pdfs $t(x;\mu,\sigma^2,\nu)$ is not a t-pdf, however (c) for small σ^2 it approximates it well.

3 Mixtures of t-distributed subspaces

A recent development in the statistics community is that of a mixture of t-distributions [8]. The motivation is as follows: a generalisation of the two-component normal mixture model (14) is the normal scale model, where

$$p(\vec{x}) = \int N(\vec{x};\vec{\mu},u^{-1}\vec{C})dH(u) \tag{15}$$

This is identical to (14) when H is a pdf with $H(1) = 1-c$, $H(\frac{1}{c}) = c$ and 0 elsewhere. If, however, H is replaced by a χ^2 pdf with ν degrees of freedom, with a gamma prior on u of $p(u) = \gamma(u;\nu/2,2/\nu)$, the resulting distribution is a t-distribution [7; 8]. Here,

$$\gamma(u;\alpha,\theta) = u^{\alpha-1}\exp(-u\theta^{-1})\Gamma(\alpha)^{-1}\theta^{-\alpha} \tag{16}$$

The t-distribution is a heavier-tailed alternative to the Gaussian, with an additional parameter ν, the degrees of freedom. The pdf of a random variable $\vec{x}$ with a multivariate t-distribution is given by:

$$t(\vec{x};\vec{\mu},\vec{C},\nu) = (\pi\nu)^{-\frac{d}{2}}\Gamma\left(\frac{\nu+d}{2}\right)\Gamma\left(\frac{\nu}{2}\right)^{-1}\left((\vec{x}-\vec{\mu})^T\vec{C}^{-1}(\vec{x}-\vec{\mu})+1\right)^{-\frac{\nu+d}{2}} \tag{17}$$

where Γ is the Gamma function, $\vec{\mu}$ is the mean and $\vec{C}$ the covariance matrix. For $\nu\to\infty$, the t-distribution becomes a Gaussian distribution.

The key element in deriving the EM algorithm for mixtures of PPCA was that the convolution of two Gaussians is itself a Gaussian; this was used to derive (6). For t-distributions, the convolution of

$$p(\vec{x}|\vec{s}) = t(\vec{x};\vec{A}\vec{s}+\vec{\mu},\sigma^2\vec{I}) \quad \text{and} \quad p(\vec{s}) = t(\vec{s};\vec{0},\vec{I}) \tag{18}$$

is not necessarily a t-distribution. However, it is to good approximation for large ν (as the t-distribution becomes like a Gaussian) *or for small* σ^2 in (18). The latter is a consequence of the fact that for small σ^2, convolution with a Gaussian will approximate convolution with a delta function, as is illustrated in Figure 1 for the univariate case.

In manifold learning, one will typically observe small values of σ^2 (provided the manifold dimensionality is chosen correctly). Approximately, then:

$$p(\vec{x}_i|j) \approx t(\vec{x}_i;\vec{\mu}_j,\vec{A}_j\vec{A}_j^T+\sigma_j^2\vec{I},\nu_j) \tag{19}$$

This allows us to use the EM algorithm derived for mixtures of constrained Gaussians above, and apply it to a mixture of constrained t-distributions. Again, a full derivation is not given; please see [8] for more information.

In using the EM algorithm to maximise (8) when a mixture of t-distributions is used, a new hidden variable u_{ij} is introduced. If $\vec{x}_i$ belongs to model j (i.e. $z_{ij} = 1$), then:

$$p(\vec{x}_i|\vec{s}_i, u_{ij}, j) = N(\vec{x}_i; \vec{A}\vec{s}_i + \vec{\mu}_j, u_{ij}^{-1}\vec{\Psi}_j) \tag{20}$$

Intuitively, u_{ij} is the weight assigned by model j to sample $\vec{x}_i$: outliers will be given low weight, and hence be described by a Gaussian with high covariance matrix elements. As above, given model j, the u_{ij} are assumed independently distributed according to a gamma distribution, $p(u_{ij}|j) = \gamma(u_{ij}; \nu_j/2, 2/\nu_j)$.

The complete-data distribution becomes:

$$\begin{aligned} p(\vec{x}, \vec{s}, \vec{z}, \vec{u}) &= \prod_{i=1}^{n} p(\vec{x}_i|\vec{s}_i, \vec{u}_i, \vec{z}_i) p(\vec{s}_i|\vec{u}_i, \vec{z}_i) p(\vec{u}_i|\vec{z}_i) p(\vec{z}_i) \\ &= \prod_{i=1}^{n} \prod_{j=1}^{k} [\pi_j p(\vec{x}_i|\vec{s}_i, j, u_{ij}, j) p(\vec{s}_i|u_{ij}, j) p(u_{ij}|j)]^{z_{ij}} \end{aligned} \tag{21}$$

with $p(\vec{x}_i|\vec{s}_i, u_{ij}, j)$ given by (20), $p(\vec{s}_i|u_{ij}, j) = N(0, u_{ij}^{-1}\vec{I})$ and $p(u_{ij}|j)$ given above.

The EM algorithm maximising (8) w.r.t. the parameters consist of:

- **E step**: for all i and j, calculate r_{ij} according to (10), and

$$u_{ij} = \frac{\nu_j + d}{\nu_j + (\vec{x}_i - \vec{\mu}_j)^T (\vec{A}_j \vec{A}_j^T + \sigma_j^2 \vec{I})^{-1} (\vec{x}_i - \vec{\mu}_j)} \tag{22}$$

- **M step**: for all j, re-estimate the π_j's as in (11) and

$$\vec{\mu}_j' = \frac{\sum_{i=1}^{n} r_{ij} u_{ij} \vec{x}_i}{\sum_{i=1}^{n} r_{ij} u_{ij}} \quad \text{and} \quad \vec{S}_j' = \frac{\sum_{i=1}^{n} r_{ij} u_{ij} (\vec{x}_i - \vec{\mu}_j')(\vec{x}_i - \vec{\mu}_j')^T}{\sum_{i=1}^{n} r_{ij}} \tag{23}$$

  $\vec{S}_j'$ can then be used to find $\vec{A}_j'$ and $\sigma_j^{2'}$ using normal eigendecomposition as in (13), or it can be used in Tipping and Bishop's faster version of the algorithm.

The only parameters not re-estimated yet in the M step are the ν_j's. Following [8], they can be updated by equating the derivative of all terms involving ν_j in the log-likelihood to zero. A solution can then be found using a nonlinear solver. However, in all experiments in this paper ν_j was fixed at 2, $\forall j$, as we are specifically interested in robust manifold learning (for more discussion on this, see Sections 4.1 and 5).

It is easy to see that the changes this EM algorithm introduces w.r.t. that outlined in Section 2.2 are minimal. An additional weight u_{ij} is calculated for each sample $\vec{x}_i$ and model j, which is used to re-estimate the means and sample covariance matrices robustly. Besides the actual model, a useful extra output of this algorithm are exactly these weights, which for outliers will be low.

4 Experiments

4.1 Density estimation

Figure 2 demonstrates the robustness of the proposed mixture of t-subspaces on a toy data set. The data was sampled from 2 semicircles, with some added Gaussian noise. To these 200 samples q uniformly distributed noise samples were added. Mixtures of PPCAs (MPPCA) with $k = 6$, $m = 1$ and mixtures of t-distributed subspaces (MTS) with identical settings were trained on this set. For the MTS, ν_j was fixed at 2, $\forall j$.

For $q = 0$, i.e. when no noise is added, both methods give more or less similar results. The likelihood of the MPPCA model is slightly higher than that of the MTS model. Note that MTS tends to assign low probability to samples at the "edges" of the manifolds; this is due to the fact that the within-subspace probability model is t-distributed as well. Samples lying along the main axes of a subspace, but far away from its mean, will therefore be assigned low weight u_{ij}. When outliers are added, MPPCA loses the manifold structure quite quickly. Already when $q = 10$, i.e. a fraction of outliers of 5%, 1 of the 6 PPCA models is used to model these. For 50% outliers, the manifold structure is lost completely. MTS is more robust: although the quality of the density estimate along the manifold deteriorates, it manages to downweight the outliers. Furthermore, the likelihood of the MTS becomes higher than that of the MPPCA.

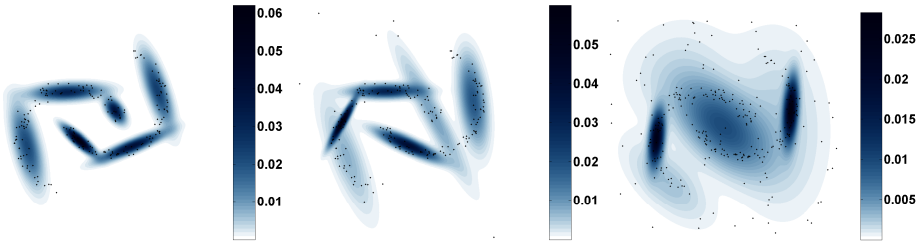
The results obtained in Figures 2(d)-(f) were obtained when ν_j was fixed to 2, for all models. If ν_j is set higher, robustness is gradually lost; Figure 2(g) illustrates this for higher settings (here ν_j was fixed at 5). When ν_j is learned, after initialisation at 5, the result (Figure 2(h)) is nearly as poor as that of MPPCA.

4.2 Handwritten digit recognition

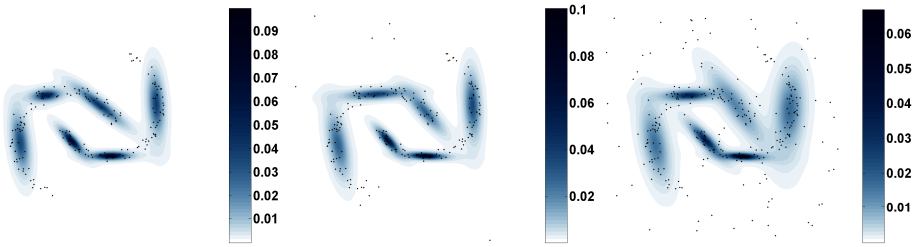
The MPPCA and MTS models were also compared on a classification problem, handwritten digit recognition. The database used was a pre-processed version of the NIST database. The original NIST database digits [13] were resized to 16×16 pixel images (preserving the aspect ratio), put upright, normalised for pencil-width and rescaled in grey value to $[-1, 1]$. A data set of 1000 training samples per class was thus created, as well as a test set containing 1000 samples per class.

Global principal component analysis on the training set left 51 dimensions. In this 51D space, mixture models were trained on each individual digit, for various settings of the number of subspaces, k, and the number of dimensions per subspace, m. All experiments were repeated for 10 different initialisations, by the k-means algorithm. The covariance matrices $\vec{C}_j = \vec{A}_j \vec{A}_j^T + \sigma_j^2 \vec{I}$ found in each iteration of the EM algorithm were regularised by adding $10^{-3}\vec{I}$, to prevent the likelihood from becoming infinite. For the MTS models, the ν_j's were fixed at 2; no attempt has been made to optimise this setting. Table 1 presents the classification results obtained using MPPCA and MTS models, obtained by Bayesian classification using the densities found for each digit.

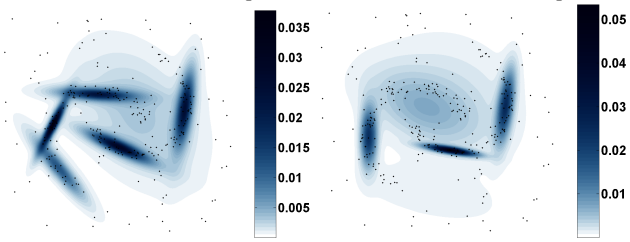
The table demonstrates that the proposed MTS models perform slightly better, for individual settings of k and m, than MPPCA models. The minimum error reached is also lower, at 1.89% vs. 2.27%. Another interesting observation is that, where MPPCA has a clear optimal model for $k = 4$ and $m = 12 - 16$, MTS performance decreases more gracefully with different parameter settings. Even for very large models (e.g. $k = 16$,



(a) $q = 0$, $\mathscr{L} = -882$ (b) $q = 10$, $\mathscr{L} = -998$ (c) $q = 100$, $\mathscr{L} = -1683$



(d) $q = 0$, $\mathscr{L} = -910$ (e) $q = 10$, $\mathscr{L} = -1000$ (f) $q = 100$, $\mathscr{L} = -1642$



(g) $q = 100$, $\mathscr{L} = -1634$ (h) $q = 100$, $\mathscr{L} = -1664$

Figure 2: Density estimates on a simple 2D toy dataset of 200 samples distributed along 2 1D curves, with some added Gaussian noise and q added uniformly distributed noise samples: (a)-(c) MPPCA, (d)-(f) MTS (both $k = 6$, $m = 1$). (g) MTS with ν_j fixed at 5, $\forall j$. (h) Same, but with ν_j initialised at 5, $\forall j$, and optimised by the EM algorithm. For the latter, after training the ν_j's ranged between 18.8 and 20.6. All runs of the EM algorithm were randomly initialised.

$m = 20$) the error is only 2.58%, as compared to 5.07% for MPPCA. This clearly indicates that MTS is less likely to assign high probability to regions that do not contain data.

The MTS models not only give better performance, but also a slightly higher likelihood on both the training set and the test set, although with fixed ν_j's they have the same number of free parameters as MPPCA models. This is illustrated in Figure 3(a) for the test set. The difference is largest for digits with little natural variation in appearance, i.e. digits "0", "1", "4", "7" and "9". For digits with a larger amount of variation the likelihood difference is not as pronounced. This indicates that MPPCA tends to over-estimate variance due to the presence of outliers; in other words, by assigning low weight to outliers, MTS obtains a tighter fit around the manifold, while not overfitting.

Finally, Figures 3(b)-(c) show some histograms of $\sum_{j=1}^{k} u_{ij}$, the cumulative weight assigned to samples by an MTS mixture with $k = 4$,$m = 12$. Outliers have been correctly given low weight, and "prototypical" digits have been given high weight. Simple thresholding of the cumulative weight makes for an easy outlier removal procedure.

(a)

$m=$	4	8	12	16	20
$k=2$	4.23 (0.10)	3.42 (0.07)	2.94 (0.04)	2.79 (0.08)	2.64 (0.05)
4	3.36 (0.15)	2.54 (0.09)	2.27 (0.10)	2.31 (0.11)	2.34 (0.11)
8	3.14 (0.14)	2.50 (0.08)	2.57 (0.13)	2.81 (0.24)	4.00 (2.99)
12	4.09 (3.03)	3.66 (2.99)	4.08 (2.95)	3.50 (0.12)	3.87 (0.20)
16	5.93 (6.31)	3.94 (2.97)	4.48 (3.05)	5.16 (2.99)	5.07 (0.52)

(b)

$m=$	4	8	12	16	20
$k=2$	4.48 (0.09)	3.03 (0.09)	2.56 (0.06)	2.55 (0.02)	2.70 (0.05)
4	3.21 (0.08)	2.39 (0.08)	2.14 (0.05)	2.02 (0.10)	2.18 (0.06)
8	2.67 (0.13)	2.13 (0.08)	2.02 (0.13)	1.89 (0.12)	2.15 (0.12)
12	2.36 (0.12)	2.02 (0.08)	2.20 (0.59)	2.08 (0.18)	2.25 (0.20)
16	2.54 (0.62)	2.09 (0.11)	2.31 (0.40)	2.41 (0.29)	2.58 (0.27)

Table 1: Performance of (a) MPPCA and (b) MTS models with different settings for the number of subspaces k and number of dimensions per subspace, m. Numbers are % error on a test set, average and standard deviation over 10 different initialisations of the algorithm.

5 Conclusions

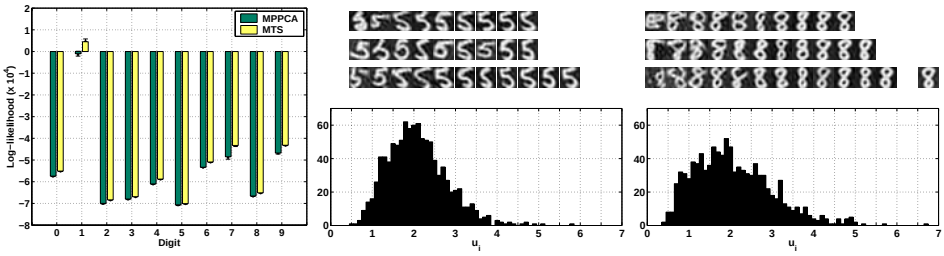
A method for robust manifold learning by EM has been presented, based on earlier work on mixtures of probabilistic PCA subspaces and mixtures of t-distributions. It is less suitable for density estimation in the absence of outliers, as it tends to ignore the "edges" of the data set. However, in the presence of outliers, it is much more robust than mixtures of PCA subspaces. As a consequence, it is useful for description and classification of high-dimensional data, such as images. The model was compared to mixtures of probabilistic PCAs on a handwritten digit recognition problem, and was shown to give good results.

In future work, we intend to investigate how to set ν in an optimal way. Experiments showed that optimising it by maximum likelihood makes the method as a whole less robust. However, the choice of $\nu = 2$ in the digit recognition experiments, while leading to good results, is rather arbitrary. Another interesting question is whether it will be feasible to use different in-subspace and out-of-subspace models, i.e. a Gaussian distribution for the latent variable $\vec{s}$ coupled with a t-distribution for the noise $\vec{\varepsilon}$. This might improve performance of the method as a robust density estimator.

An application of this model might be to learning global manifolds, i.e. for which a unique mapping $\mathscr{F} : \mathbb{R}^d \to \mathbb{R}^m$ exists. This can be achieved, for example, by post-coordinating the subspaces found [11]. Robustness is especially important for such a procedure, as a single poorly fitted subspace might result in a global misfit of the data.

Acknowledgements

The first author would like to thank Jiři Matas and the Center for Machine Perception, of the Czech Technical University of Prague. This paper was written while he was a



(a) (b) (c)

Figure 3: (a) Average log-likelihood of the test set over 10 experiments, per digit, for MPPCA and MTS models ($k = 4$, $m = 12$). Error bars indicate standard deviation. (b)-(c) Histograms of $\sum_{j=1}^{k} u_{ij}$ for MTS models ($k = 4$, $m = 12$) trained on digits "5" and "8". Images above the histogram indicate typical images having weights directly below.

visiting researcher supported by the Center of Excellence EU grant ICA 1-CT-2000-70002 MIRACLE. The work was partly sponsored by the the Czech Ministry of Education under Project MSM 212300013.

References

[1] C.M. Bishop, M. Svensén, and C.K.I. Williams. GTM: The generative topographic mapping. *Neural Computation*, 10(1):215–234, 1998.

[2] H. Chen, P. Meer, and D. Tyler. Robust regression for data with multiple structures. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2001)*, pages 1069–1075, Los Alamitos, CA, 2001. IEEE Computer Society Press.

[3] Z. Ghahramani and G.E. Hinton. The EM algorithm for mixtures of factor analyzers. Technical Report CRG-TR-96-1, University of Toronto, Toronto, Canada, 1996.

[4] P.J. Huber. *Robust statistics*. John Wiley & Sons, New York, NY, 1981.

[5] T. Jebara and A. Pentland. Maximum conditional likelihood via bound maximization and the CEM algorithm. In M.S. Kearns, S.A. Solla, and D.A. Cohn, editors, *Adv. in Neural Information Proc. Systems 11*, 1998.

[6] A. Leonardis and H. Bischof. Robust recognition using eigenimages. *Computer Vision and Image Understanding*, 78(1):99–118, 2000.

[7] G.J. McLachlan and T. Krishnan. *The EM algorithm and extensions*. John Wiley & Sons, New York, NY, 1997.

[8] D. Peel and G.J. McLachlan. Robust mixture modelling using the *t* distribution. *Statistics and Computing*, 10(4):339–348, 2000.

[9] S.T. Roweis and L.K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[10] L.K. Saul and D.D. Lee. Multiplicative updates for classification by mixture models. In *Adv. in Neural Information Proc. Systems 14*, Cambridge, MA, 2001. MIT Press.

[11] Y.W. Teh and S.T. Roweis. Automatic alignment of local representations. In S. Becker, S. Thrun, and K. Obermayer, editors, *Adv. in Neural Information Proc. Systems 15*, Cambridge, MA, 2002. MIT Press.

[12] M.E. Tipping and C.M. Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2):443–482, 1999.

[13] C.L. Wilson and M.D. Garris. Handprinted character database 3, february 1992. National Institute of Standards and Technology; Advanced Systems division.