The generative self-organizing map: a probabilistic generalization of Kohonen's SOM

Jakob Verbeek, Nikos Vlassis, Ben Krose

Universiteit
van
Amsterdam

Submitted to: European Symposium on Artificial Neural Networks 2003

IAS technical report IAS-UVA-02-03

The Generative Self-Organizing Map

A Probabilistic Generalization of Kohonen's SOM

J.J. Verbeek, N. Vlassis, and B.J.A. Kröse
Intelligent Autonomous Systems,
University of Amsterdam
The Netherlands

We present a variational Expectation-Maximization algorithm to learn probabilistic mixture models. The algorithm is similar to Kohonen's Self-Organizing Map algorithm and can be applied on any mixture model for which we can find a standard Expectation Maximization algorithm. We maximize the variational free-energy which sums data log-likelihood and Kullback-Leibler divergence between the neighborhood function and the posterior distribution on the components, given data. We illustrate the algorithm with an application on word clustering.
Keywords: self-organizing map, Gaussian mixture, free-energy minimization, Expectation-Maximization algorithm, vector quantization.

IAS
intelligent autonomous systems

Contents

Intelligent Autonomous Systems
Informatics Institute, Faculty of Science
University of Amsterdam
Kruislaan 403, 1098 SJ Amsterdam
The Netherlands
Tel (fax): +31 20 525 7461 (7490)
http://www.science.uva.nl/research/ias/

Corresponding author:
J.J. Verbeek
tel: +31 20 525 7550
jverbeek@science.uva.nl
http://www.science.uva.nl/~jverbeek/

1 Introduction

Kohonen's Self-Organizing Map (SOM) [5] is a data analysis method that combines vector quantization with topology preservation. With each quantizer we associate a *fixed* location in a 'latent' space. The latent space is of much lower dimension (typically just two) than the data space. The SOM algorithm finds locations for the quantizers in the data space such that the summed squared distance from data to closest node is small and simultaneously topology is preserved. Topology preservation means that nearby components in latent space are also nearby in the data space. As a consequence, data points associated with nearby components in latent space are coming from similar locations in the data space. Hence, to check if data points are 'close' in the data space, we can check whether their associated components are close in the latent space. Topology preservation makes SOMs useful for data visualization and dimensionality reduction.

We present a constrained or variational Expectation-Maximization (EM) learning algorithm to learn probabilistic mixture models similar to SOM. The algorithm applies to a wide class of mixture models, in principle to any mixture model for whcih we can find a standard EM algorithm.

In the next section we discuss the Gaussian mixture model that will serve as the example mixture model throughout and also briefly discuss the EM algorithm. In section 3 we present our Self-Organizing Mixtures algorithm. We compare our algorithm to several closely related algorithms in Section 4. A brief example application is discussed in Section 5 and we present our conclusions in Section 6.

2 A simple generative model and EM

As generative model consider the Mixture of Gaussians (MoG), given by :

$$p(\mathbf{x}) = \frac{1}{k} \sum_{s=1}^{k} p(\mathbf{x} \mid s),$$

for D-dimensional data, where $p(\mathbf{x}|s)$ is an isotropic Gaussian with inverse variance β and mean $\boldsymbol{\mu}_s$. Given a data $\mathbf{x} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, and an initial parameter vector $\boldsymbol{\theta} = \{\beta, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k\}$ the EM algorithm [6] finds a local maximizer $\boldsymbol{\theta}$ of the log-likelihood $\mathcal{L}(\boldsymbol{\theta})$. Assuming data are independent and identically distributed:

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_n \log p(\mathbf{x}_n \mid \boldsymbol{\theta}) = \sum_n \log \frac{1}{k} \sum_s p(\mathbf{x}_n \mid s).$$

Learning is facilitated by introducing a set of N *hidden* variables. Each hidden variable indicates which of the k mixture components generated the corresponding data point. The EM algorithm maximizes the negative free-energy:

$$\begin{aligned} F(Q, \boldsymbol{\theta}) &= \mathrm{E}_Q \log p(\mathbf{x}, s; \boldsymbol{\theta}) + \mathcal{H}(Q) && (1) \\ &= \mathcal{L}(\boldsymbol{\theta}) - D_{KL}(Q \parallel p(s \mid \mathbf{x}; \boldsymbol{\theta})), && (2) \end{aligned}$$

where we used $\mathcal{H}$ to denote the entropy of a distribution and D_{KL} to denote the Kullback-Leibler (KL) divergence between two probability distributions. $Q = \prod_n q_n$ is a distribution over the hidden variables, q_n gives a distribution on the generating mixture component $s \in \{1, \ldots, k\}$ for data point n. Note that, due to the non-negativity of the KL-divergence, for *any* Q, $F(Q, \boldsymbol{\theta})$ is a lower bound on $\mathcal{L}(\boldsymbol{\theta})$.

The two forms in which we expanded F are associated with the M-step and the E-step of EM. In the M-step we change $\boldsymbol{\theta}$ as to maximize, or at least increase, $F(Q, \boldsymbol{\theta})$. The first decomposition includes $\mathcal{H}(Q)$ which is a constant w.r.t. $\boldsymbol{\theta}$. In the E-step we maximize F w.r.t. Q, the second decomposition includes $\mathcal{L}(\boldsymbol{\theta})$ which is constant w.r.t. Q. What remains is a KL divergence which is the effective objective function in the E-step of EM.

In standard applications of EM (e.g. mixture modeling) Q is unconstrained, which results in setting $q_n = p(s \mid \mathbf{x}_n; \boldsymbol{\theta})$ in the E-step since the non-negative KL divergence equals zero if and only if both arguments are equal. Therefore, after each E-step $F(Q, \boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta})$. Variational methods are used when optimization over the unconstrained Q is intractable, Q is then restricted to a certain class $\mathcal{Q}$ of distributions allowing for tractable computations. Variational EM maximizes F instead of $\mathcal{L}$, the objective sums log-likelihood and a penalty which is high if the true posterior is far from any member of $\mathcal{Q}$.

3 Using free-energy for self-organization

By constraining choosing an appropriate class of distributions $\mathcal{Q}$, we can also enforce a topological ordering between the mixture components, as we explain below. The approach is much like the one taken in [7, 8], where constraints on Q are used to align the local coordinate systems of the components of a probabilistic mixture of factor analyzers.

We associate with each mixture component s a latent coordinate $\mathbf{g}_s$. It is convenient to take the components as located on a regular grid in the latent space. We set $\mathcal{Q}$ to be class of discretized isotropic Gaussians in the latent space, centered on either one of the k component locations $\mathbf{g}_s$ and with a particular *fixed* variance. The distributions q_n play a similar role as the *neighborhood function* in Kohonen's SOM [5]. The q_n are thus restricted to be contained in the finite set of distributions $\mathcal{Q} = \{p_1, \ldots, p_k\}$, where:

$$p_r(s) = \frac{\exp\left(-\lambda \parallel \mathbf{g}_s - \mathbf{g}_r \parallel^2\right)}{\sum_t \exp\left(-\lambda \parallel \mathbf{g}_t - \mathbf{g}_r \parallel^2\right)}.$$

A small λ corresponds to a broad distribution (high entropy), and for large λ the distribution p_r becomes more peaked (low entropy). By performing EM steps, we can never decrease the objective. The M-step is as usual and in the E-step we select for each data point $\mathbf{x}_n$ the distribution $q_n \in \mathcal{Q}$ that increases the objective the most. Note that in the E-step we can always stick to the q_n of the previous EM round without changing (and hence definately not decreasing) the objective funciton.

Since the objective function might have local optima and EM is only guaranteed to give locally optimal solutions, good initialization of the parameters of the mixture model is essential to finding a good solution. Analogous to the method of shrinking the extent of the neighborhood function with the SOM, we can start with a small λ (broad neighborhood function) and increase it iteratively until a desired value is reached. In implementations we started with λ such that the p_r are close to uniform over the components, then we run the EM algorithm until convergence. Note that if the p_r are almost uniform the initialization of $\boldsymbol{\theta}$ becomes irrelevant. After convergence we set $\lambda^{new} \leftarrow \eta\lambda^{old}$ with $\eta > 1$ (typically η is close to unity). In order to initialize the EM procedure with λ^{new}, we initialize $\boldsymbol{\theta}$ with the value found in running EM with λ^{old}.

Using $q_{ns} = q_n(s)$, for our MoG case F can be rewritten as:

$$F(Q, \boldsymbol{\theta}) = \frac{1}{2} N D \log \beta - \sum_{ns} q_{ns} \Big[\beta \parallel \mathbf{x}_n - \boldsymbol{\mu}_s \parallel^2 / 2 + \log q_{ns} \Big].$$

For fixed λ an EM algorithm can be derived by differentiation:

- **E:** Determine (by means of exhaustive or sparse search in $\mathcal{Q}$, see below) for each $\mathbf{x}_n$ the distribution $p_{r^*} \in \mathcal{Q}$ that maximizes F, set $q_n = p_{r^*}$.
- **M:** Set: $\boldsymbol{\mu}_s = \sum_n q_{ns}\mathbf{x}_n / \sum_n q_{ns}$ and $\beta = ND / \sum_{ns} q_{ns} \parallel \mathbf{x}_n - \boldsymbol{\mu}_s \parallel^2$.

The component r^* on which p_{r^*} is centered for data point n, is referred to as the 'winner' for $\mathbf{x}_n$. The computational cost of the E-step is $O(Nk^2)$, a factor k slower than Kohonen's SOM and possibly prohibitively slow in large-scale applications. However, by restricting the search for a winner in the E-step to a limited number of candidate winners we can obtain an $O(Nk)$ algorithm. A straightforward choice is to use the l components with the largest joint likelihood $p(\mathbf{x}, s)$ as candidates, corresponding for our MoG to smallest Euclidean distance to the data point. If none of the candidates yields a higher value of $F(Q, \boldsymbol{\theta})$ we keep the winner of the previous step, in this way we are guaranteed never to decrease the objective in every step. We found $l = 1$ to work well and fast in practice, in this case we only check whether the winner from the previous round should be replaced with the closest node.

Let us consider why our algorithm yields topology preservation. The q_n are by construction localized in the latent space. This implies that although the winner gets the most mass from q_n, the neighbors of the winner also get some mass and far-away components get practically no mass. As a consequence nearby components in latent space are forced to model similar data.

Also, consider a topology preserving configuration of the mixture model. We can destroy the topology preservation if we permute the mixture components in the data space but keeping their locations in the latent space fixed. Note that $\mathcal{L}(\boldsymbol{\theta})$ is the same for both configurations. Clearly, for the topology preserving configuration the mass of the posterior distribution is more or less localized in the latent space, where this will not be the case for the permuted model. Therefore, the KL divergence is expected to much smaller for the topology preserving configuration.

4 Discussion

Discussion. Our algorithm is very similar to Kohonen's SOM when applied on the example MoG. If we use only $l = 1$ candidate winner in the E-step the difference with Kohonen's winner selection is that we we only accept the closest node as a winner when it increases the energy and keep the previous winner otherwise. Our M-step coincides exactly with the update rule of the batch SOM: it puts component s at the weighted sum of the data, each data item weighted proportionally to the value of the neighborhood function (centered on the winner of the data point) at s. For an increasingly peaked neighborhood function both SOM and our algorithm aplied to the simple MoG reduce to the k-means vector quantization algorithm.

In [3, 4] another SOM-like algorithm is proposed with objective function:

$$-\sum_{ns} q_{ns}[\beta \sum_r h_{sr} \parallel \mathbf{x}_n - \boldsymbol{\mu}_r \parallel /2 + \log q_{ns}].$$

There, the neighborhood function, implemented by the h_{sr}, is fixed, but the winner assignment is soft. Instead of selecting one 'winner' an unconstrained distribution over the components is used: the q_{ns}. The β is used for annealing: for very small β the entropy term, with only one *global* optimum, becomes dominant, whereas for large β the quantization error, with many local optima, becomes dominant. By gradually increasing β more and more structure is added to the objective function.

Our work differs in several ways. We use *one localized distribution* in the latent space to constrain q_n, as opposed to a *mixture of localized distributions* in the latent space. As a consequence the easy speed-up of our algorithm does not apply to the algorithms in [3, 4]. We use the neighborhood function width (controlled by λ) for annealing as opposed to using β. Note

that although both λ and β can be used for annealing, only β can be optimized efficiently as a free parameter. Both our objective function and the one of [3, 4] can be interpreted as a log-likelihood plus a penalty term for non-topology preserving configurations. The interpretation provided in this work is simpler and applies trivially to any mixture model where for the interpretation of [3, 4] it is not clear whether it applies to any mixture model.

Another similar model is presented in [1]. There, in the E-step we set:

$$q_n = \arg\max_r \sum_s p_r(s) p(\mathbf{x}_n|s).$$

The M-step finds a new parameter vector that maximizes:

$$\sum_n \log \sum_s q_n(s) p(\mathbf{x}_n|s).$$

This algorithm does not optimize a single likelihood function, even if we keep the neighborhood function width fixed, since the mixing weights vary for each data item and change throughout the iterations of the algorithm. The algorithm has run-time $O(nk^2)$, but can benefit from the same speed-up used here.

Another model, often presented as the probabilistic version of the self-organizing map, is the Generative Topographic Map (GTM) [2]. However, the manner in which GTM achieves the topographic organization is quite different from those used in the SOM models. In GTM mixture components are parameterized by a linear combination of nonlinear functions of the locations of the components in the latent space. The parameters of the linear map are learned from data. The number of nonlinear basis functions and their smoothness are to be found by model complexity selection procedures.

5 Illustration

We illustrate our algortihm with an example where the mixture does not contain Gaussians as components but products of Bernoulli distributions, to underline that we are not limited to using Gaussian mixtures.

Figure 1 shows results of modeling word occurrence data on a part of the 20 newsgroup data set, word $\mathbf{x}_n$ is shown at location $\mathbf{g}_n = \sum_s p(s|\mathbf{x}_n)\mathbf{g}_s$. Each of the 100 words is a data item with 16242 binary features indicating its occurrence in each of 16242 documents.

We learned a $k = 25$ component mixture model with our algorithm, where each mixture component is a product of Bernoulli distributions:

$$p(\mathbf{x}_n|s) = \prod_{i=1}^{16242} p(x_n^{(i)}|s) = \prod_{i=1}^{16242} {p_{s,i}}^{x_n^{(i)}} (1 - p_{s,i})^{(1-x_n^{(i)})},$$

where $x_n^{(i)}$ denotes the i-th element of the vector $\mathbf{x}_n$.

Due to the huge dimensionality of this data, the posteriors are rather peaked (low entropy), resulting in $\mathbf{g}_n$ almost equal to one of the $\mathbf{g}_s$. The result is that the words locations $\mathbf{g}_n$ are almost equal to one of the component locations $\mathbf{g}_s$. Smoothing the posteriors changes the plot from one in which the $\mathbf{g}_n$ form 25 heaps of points on the $\mathbf{g}_s$, into one where the $\mathbf{g}_n$ are drawn toward other similar heaps and the points are more spread. To 'smoothen' the representation we used $p'(s|\mathbf{x}_n) \propto p(s|\mathbf{x}_n)^\alpha$ for $\alpha < 1$ such that the entropies of the $p'(s|\mathbf{x}_n)$ were close to two bits.[1]

[1] It is not difficult to show that $p' \propto p^\alpha$ is the best approximation in Kullback-Leibler sense to p such that $\mathcal{H}(p') = c$ where α depends on c.

The different occurance patterns of the words through the documents can be identified in the latent space. For example, the lower right corner is devoted to words that occur in computer related documents.

6 Conclusions

We presented a penalized log-likelihood probabilistic mixture modeling method, similar to Kohonen's SOM. The probabilistic formulation offers several benefits: (i) The method is directly applicable to any mixture model for i.i.d. data for which we can find a regular EM algorithm. (ii) The generative model allows seamless embedding in larger probabilistic systems. It is for example straightforward to learn mixtures of Self-Organizing maps. (iii) There is a direct and clear link to the mixture model data log-likelihood.

In current research we are investigating a similar method and in a sense opposite method. In the work presented here, we started with fixed locations in the latent space and fitted a mixture by maximizing the free energy. The opposite approach uses a fixed mixture and finds a corresponding latent representation that maximizes a similar free energy. The latter free-energy has single unique maximum as a function of the latent space representation which can be found by finding few eigenvectors of a matrix with edge size k.

References

[1] A. Anouar, F. Bedran, and S. Thiria. Probabilistic self organized map: application to classification. In M. Verleysen, editor, *Proceedings of European Symposium on Artificial Neural Networks*, Belgium, 1997. D-Facto.

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

[3] T. Graepel, M. Burger, and K. Obermayer. Self-organ. maps: generalizations and new optimiz. techniques. *Neurocomputing*, 21:173–190, 1998.

[4] T. Heskes. Self-organizing maps, vector quantization, and mixture modeling. *IEEE Transactions on Neural Networks*, 12:1299–1305, 2001.

[5] T. Kohonen. *Self-Organizing Maps*. Springer-Verlag, 2001.

[6] R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M.I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. Kluwer, 1998.

[7] S.T. Roweis, L.K. Saul, and G.E. Hinton. Global coordination of local linear models. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, USA, 2002. MIT Press.

[8] J.J. Verbeek, N. Vlassis, and B. Kröse. Coordinating Principal Component Analyzers. In J.R. Dorronsoro, editor, *Proceedings of Int. Conf. on Artificial Neural Networks*, pages 914–919, Madrid, Spain, 2002. Springer.

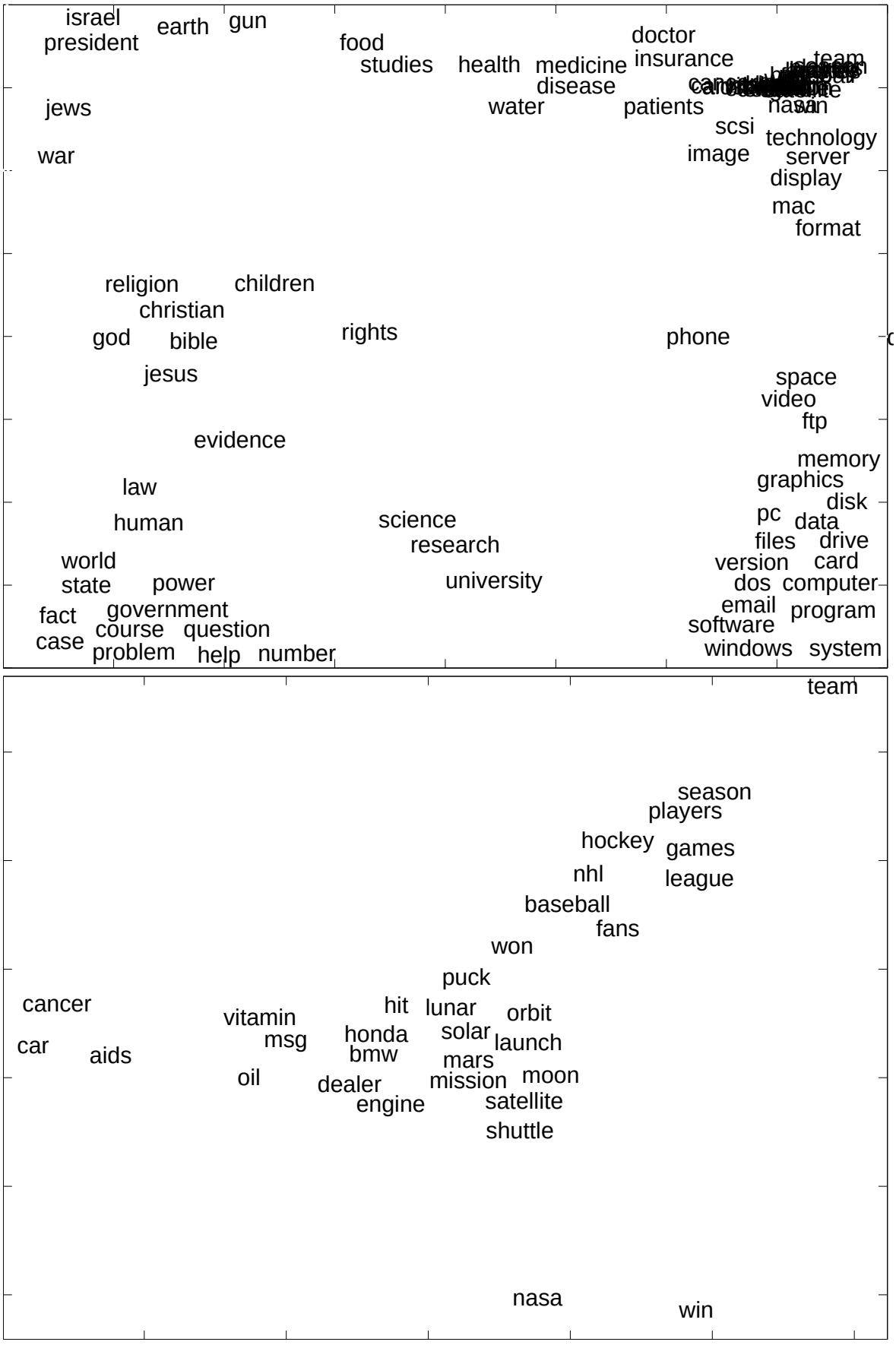

Figure 1: Self-organizing Bernoulli models, $k = 25$. Lower plot zooms dense area in upper plot.

Acknowledgements

This research is supported by the Technology Foundation STW, applied science division of NWO and the technology program of the Ministry of Economic Affairs.

IAS reports

This report is in the series of IAS technical reports. The series editor is Stephan ten Hagen (`stephanh@science.uva.nl`). Within this series the following titles appeared:

R. Bunschoten and B. Kröse. *Robust scene reconstruction from an omnidirectional vision system.* Technical Report IAS-UVA-02-02, Computer Science Institute, University of Amsterdam, The Netherlands, February 2002.

J.J. Verbeek, N. Vlassis, and B. Kröse. *Procrustes Analysis to Coordinate Mixtures of Probabilistic Principal Component Analyzers.* Technical Report IAS-UVA-02-01, Computer Science Institute, University of Amsterdam, The Netherlands, February 2002.

J.J. Verbeek, N. Vlassis, and B.J.A. Kröse. *Efficient Greedy Learning of Gaussian Mixtures.* Technical Report IAS-UVA-01-10, Computer Science Institute, University of Amsterdam, The Netherlands, May 2001.

M.B. van Leeuwen and F.C.A. Groen. *Vehicle Detection with a Mobile Camera.* Technical Report IAS-UVA-01-09, Computer Science Institute, University of Amsterdam, The Netherlands, October 2001.

L.A. Cornelissen and F.C.A. Groen. *Automatic color landmark detection and retrieval for robot navigation.* Technical Report IAS-UVA-01-08, Computer Science Institute, University of Amsterdam, The Netherlands, October 2001.

J. Kok, R. de Boer, and N. Vlassis. *Towards an optimal scoring policy for simulated soccer agents.* Technical Report IAS-UVA-01-06, Computer Science Institute, University of Amsterdam, The Netherlands, October 2001.

All IAS technical reports are available for download at the IAS website, `http://www.science.uva.nl/research/ias/publications/reports/`.