Shared Kernel Information Embedding for Discriminative Inference

Roland Memisevic, Leonid Sigal, *Member*, *IEEE*, and David J. Fleet, *Senior Member*, *IEEE*

Abstract—Latent variable models, such as the GPLVM and related methods, help mitigate overfitting when learning from small or moderately sized training sets. Nevertheless, existing methods suffer from several problems: 1) complexity, 2) the lack of explicit mappings to and from the latent space, 3) an inability to cope with multimodality, and 4) the lack of a well-defined density over the latent space. We propose an LVM called the Kernel Information Embedding (KIE) that defines a coherent joint density over the input and a learned latent space. Learning is quadratic, and it works well on small data sets. We also introduce a generalization, the shared KIE (sKIE), that allows us to model multiple input spaces (e.g., image features and poses) using a single, shared latent representation. KIE and sKIE permit missing data during inference and partially labeled data during learning. We show that with data sets too large to learn a coherent global model, one can use the sKIE to learn local online models. We use sKIE for human pose inference.

Index Terms—Latent variable models, kernel information embedding, inference, nonparametric, mutual information.

1 Introduction

Many computer vision problems can be expressed as inference tasks on high-dimensional observations and/or high-dimensional unknowns, such as image features, pose representations, or object descriptions. Nonlinear latent variable models are increasingly used to resolve the well-known problems associated with such high-dimensional data. By utilizing the low-dimensional, nonlinear structure often inherent in such data sets, latent variable models are commonly used to improve performance in classification, recognition, and parameter estimation tasks. A common modeling assumption underlying many latent variable methods is that data are distributed along a lower dimensional, homogeneous manifold, or "sheet," in the high-dimensional data space. Learning amounts to estimating a parameterization of this manifold, and inference to finding a latent representative in this parameterization, given a high-dimensional query point not seen during training [1], [2], [3], [4].

The metaphor of a well-sampled, low-dimensional "sheet," unfortunately, is not always appropriate in real-world problems, where data can consist of multiple distinct clusters or "submanifolds," or where the intrinsic dimensionality can be different in different parts of the data space. Even when data are distributed along a homogeneous manifold, inference is often not unique, due to manifold curvature and noise. Consider, for example, a query point centered in a curved part of a one-dimensional manifold embedded in a higher dimensional space (e.g., see Fig. 4). When the query point has approximately the same euclidean distance to two points along the manifold, it is not clear how one should map the query point to the manifold, as it is inherently ambiguous. Instead of a simple mapping (or projection), a more appropriate representation than a single latent *point* would therefore be a—possibly multimodal—latent *density* conditioned on an observation. Unfortunately, treating latent variables fully probabilistically is straightforward only under linear-Gaussian assumptions (e.g., see [5]). The nonlinear case is typically addressed using geometric rather than probabilistic approaches (e.g., [3], [2]), or by *conditioning* on the latent variables, thereby treating them as deterministic parameters rather than as random variables [6], [7].

We describe a latent variable model, called Kernel Information Embedding (KIE) [8], that parameterizes a latent data representation in terms of a low-dimensional *density*. We show how, with kernel density estimates, it is possible to elegantly combine the *geometric* reasoning behind manifold learning methods, such as LLE [3], ISOMAP [2], and many others, with the *probabilistic* approach behind factor analysis and probabilistic PCA. In this way, we retain the advantages of probabilistic models, including the possibility of properly dealing with ambiguities and uncertainties in inference, while allowing us to model highly nonlinear and non-Gaussian data distributions. We describe a training method whose time complexity is quadratic in the number of observations, and thus compares favorably with GPLVM and related methods, whose complexity is cubic in the number of observations [6].

The shared Kernel Information Embedding (sKIE) [9] is an extension of KIE in which latent densities are used to mediate between *multiple* high-dimensional spaces. The sKIE is motivated by the task to compute *mappings* between high-dimensional observations, such as from image observations to a 3D object model. Common examples of this task

- *R. Memisevic is with the Department of Computer Science, University of Frankfurt, Robert-Mayer-Str. 10, 60325 Frankfurt, Germany. E-mail: ro@cs.uni-frankfurt.de.*
- *L. Sigal is with Disney Research, Pittsburgh, 4615 Forbes Ave., Pittsburgh, PA 15213. E-mail: lsigal@disneyresearch.com.*
- *D.J. Fleet is with the Department of Computer Science, University of Toronto, 6 King's College Road, Toronto, ON M5S 3H5, Canada. E-mail: fleet@cs.toronto.edu.*

Manuscript received 29 Dec. 2010; revised 7 June 2011; accepted 3 July 2011; published online 28 July 2011.
Recommended for acceptance by C. Sminchisescu.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2010-12-0989.
Digital Object Identifier no. 10.1109/TPAMI.2011.154.

in computer vision problems include articulated human pose inference [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], articulated pose and shape estimation [20], and hand pose estimation [21]. Such mappings are typically approached with *discriminative methods*, where, given a set of training samples comprising image features, $\mathbf{x}$, and 3D poses, $\mathbf{y}$, i.e., $\{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\}_{i=1}^{N}$, the estimation of pose, $\mathbf{y}$, is viewed as a form of "regression," or, more generally, as the estimation of the conditional distribution, $p(\mathbf{y} \mid \mathbf{x})$.

1.1 Human Pose Inference

In this paper, we focus on human pose inference. This task requires a mapping for which both the input (features) and the output (pose) are high-dimensional vectors, i.e., $\mathbf{y} \in \mathbb{R}^{d_\mathbf{y}}$ and $\mathbf{x} \in \mathbb{R}^{d_\mathbf{x}}$, where usually $d_\mathbf{x} > 100$ and $d_\mathbf{y} > 30$. With high-dimensional problems, large data sets are usually necessary to learn a conditional distribution that will generalize well. Furthermore, since synchronized image and pose data are hard to obtain in practice, one is often forced to work with small or moderately sized labeled data sets, or with unlabeled data using semi-supervised learning [14], [15]. Finally, since pose inference is often ambiguous (i.e., one feature vector is consistent with multiple poses), the conditional distribution, $p(\mathbf{y} \mid \mathbf{x})$, is generally multimodal [17].

We show how the sKIE allows us to define a coherent, multimodal density over one or more *input* feature spaces, the *output* pose space, and a learned low-dimensional latent space. With this model, it is also easy to condition on a latent state, or some combination of input and output states. The model can be learned from small data sets, with complexity that is quadratic in the number of training points. The latent model helps to mitigate problems of overfitting that are common with high-dimensional data. With data sets too large to learn a coherent global model, one can also use sKIE to learn local models in an online fashion. sKIE can deal with missing data during inference, and partially labeled data during learning.

1.2 Related Work

Approaches to discriminative articulated pose estimation (and tracking) can be generally classified as either local or global. *Local* methods take the form of kernel regression [16], [19], where one first finds a subset of training exemplars that are similar to the input features (e.g., using K-nearest neighbors). These exemplars are then used to learn an online regressor, like linear locally weighted regression [16] or Gaussian Process (GP) regression [19]. The latter has the advantage of also producing a confidence measure over the regressed pose. While simple conceptually, the determination of the right local topology and a good distance measure within the neighborhood of the test features can be difficult. Typically, these methods require a large set of training exemplars to densely cover the pose space. Furthermore, these methods do not deal with multimodal conditional distributions (or one-to-many mappings), so one must first cluster the selected exemplars into local convex sets, for which regression is unimodal (one-to-one).

Global methods learn a coherent model across the entire training set. Early examples were formulated as Ridge Regression or Relevance Vector Regression [10], [11]. The problem with such approaches has been the multimodal nature of pose estimation from image features. Early work on multivalued regression that addressed this issue includes [22]. Recent work on multivalued regression for pose-estimation includes the multivariate RVM [23], or the conditional Mixture of Experts (MoE) model [14], [20], [17]. The MoE is fairly efficient since training and inference are both $O(N)$. On the other hand, the MoE model typically requires large training sets to properly fit the parameters of the gating and expert functions [15]. This makes it prone to overfitting with small data sets. The second issue with such approaches is the implicit independence of different output dimensions, and hence the failure to capture critical correlations between output dimensions (e.g., different joints of the body in pose estimation) [24].

Some recent latent variable models naturally capture correlations in the outputs, and are capable of learning models from small or moderately sized data sets. For example, models based on the Gaussian Process Latent Variable Model (GPLVM) [12], [15], [25] and the Spectral Latent Variable Model (SLVM) [26] have been proposed (see Fig. 2). These models exploit an intermediate low-dimensional latent space to effectively regularize the conditional pose distribution (i.e., pose conditioned on features). This helps avoid overfitting with small training sets. Nevertheless, as with other Gaussian Process (GP) methods, learning is expensive, $O(N^3)$ for N training exemplars, and therefore impractical for all but small data sets. Inference with the GP model is also expensive as it involves an $O(N^2)$ optimization (with multiple restarts) of the likelihood in the latent space [15]. Sparsification methods [27] reduce learning complexity from $O(N^3)$ to $O(Nd^2)$, where d is the number of *pseudo-inputs* (ideally $d \ll N$), but the use of such techniques is not always straightforward and effective.

The KIE is closely related to the GPLVM [6] and to existing embedding methods that utilize kernel densities, such as [7], [28]. In contrast to [6] and [7], inference and learning for the KIE do not scale with the dimensionality of the data space. In contrast with [28], the KIE optimizes positions of latent representatives rather than the bandwidths of a kernel density estimate on a spectral embedding. Optimizing latent positions was shown to give more accurate embeddings in [29]. Furthermore, while the KIE has many benefits in common with the GPLVM, it has lower complexity for both learning, $O(N^2)$, and inference, $O(N)$. Most importantly, the KIE provides an explicit density over the latent space and closed-form expressions for conditional distributions, allowing one to easily condition on either a data-space location or a latent-space location.

The sKIE is a generalization of KIE to handle multiple input and output spaces (e.g., see Fig. 2), allowing a dynamic regression from any subset of inputs to any subset of outputs at test time (without learning separate pair-wise models, as would be required with MoE, for example). Furthermore, the sKIE can also be used to learn local models in an online fashion (cf., [19]).

2 Kernel Information Embedding

Given samples $\{\mathbf{x}^{(j)}\}_{j=1}^{N}$ drawn from a data distribution $p(\mathbf{x})$, we aim to find a low-dimensional latent distribution, $p(\mathbf{z})$,

that captures the structure of the data distribution. For probabilistic models, a natural measure of the goodness of the latent distribution is the *mutual information* (MI) between the latent distribution and $p(\mathbf{x})$, i.e.,

$$I(\mathbf{x},\mathbf{z}) = \int p(\mathbf{x},\mathbf{z}) \log \frac{p(\mathbf{x},\mathbf{z})}{p(\mathbf{x})p(\mathbf{z})}\, d\mathbf{x}\, d\mathbf{z} \tag{1}$$

$$= H(\mathbf{x}) + H(\mathbf{z}) - H(\mathbf{x},\mathbf{z}), \tag{2}$$

where $H(\cdot)$ is the usual (differential) Shannon entropy (see, e.g., [30]).

In practice, MI and other entropy-based quantities are not often used for nonlinear embedding because they are hard to evaluate for all but very specific distributions, e.g., Gaussians. Most nonlinear embedding methods rely instead on geometric arguments and try to match local geometric properties of the set of data points and a corresponding set of latent representatives (e.g., [3], [2]). Here, we suggest *combining* geometric with probabilistic reasoning using kernel density estimation (KDE).

A KDE models a probability density as a superposition of local kernel functions and is thus based on the assumption of local smoothness of the underlying density [31]. With a KDE, we can capture geometric structure through the *locality* of the kernel, providing probabilistic nonlinear embeddings. To this end, the KIE formulation begins with a set of latent representatives $\{\mathbf{z}^{(j)}\}_{j=1}^N$, one for each of the data samples $\{\mathbf{x}^{(j)}\}_{j=1}^N$. The latent representatives support a KDE in the latent space:

$$\hat{p}(\mathbf{z}) \;=\; \frac{1}{N}\sum_{j=1}^{N} k_{\mathbf{z}}\big(\mathbf{z},\mathbf{z}^{(j)}\big). \tag{3}$$

With this kernel density estimate, we can approximate the latent entropy as

$$H(\mathbf{z}) = -\int p(\mathbf{z}) \log p(\mathbf{z})\, d\mathbf{z} \tag{4}$$

$$\approx -\frac{1}{N}\sum_{j=1}^{N} \log p\big(\mathbf{z}^{(j)}\big) \tag{5}$$

$$\approx -\frac{1}{N}\sum_{i=1}^{N} \log \sum_{j=1}^{N} k_{\mathbf{z}}\big(\mathbf{z}^{(i)},\mathbf{z}^{(j)}\big) + \log(N) \tag{6}$$

$$=: \hat{H}(\mathbf{z}). \tag{7}$$

In doing so, we are making two approximations. First, in (5) we use a Monte Carlo approximation to the expected log probability of the latent distribution. Second, in (6) we approximate the latent distribution using the kernel density estimate $\hat{p}(\mathbf{z})$.

To learn a KIE model, we approximate all of the high-dimensional integrals in (2) with kernel density estimates for $p(\mathbf{x})$, $p(\mathbf{z})$, and $p(\mathbf{x},\mathbf{z})$. That is, if we let $k_{\mathbf{x}}(\cdot,\cdot)$ and $k_{\mathbf{z}}(\cdot,\cdot)$ denote kernels for the data and latent spaces, we can approximate the entropies and the mutual information as

$$\begin{aligned}\hat{I}(\mathbf{x},\mathbf{z}) &= \hat{H}(\mathbf{x}) + \hat{H}(\mathbf{z}) - \hat{H}(\mathbf{x},\mathbf{z}) \\ &= -\frac{1}{N}\sum_{i=1}^{N} \log \sum_{j=1}^{N} k_{\mathbf{x}}\big(\mathbf{x}^{(i)},\mathbf{x}^{(j)}\big) \\ &\quad -\frac{1}{N}\sum_{i=1}^{N} \log \sum_{j=1}^{N} k_{\mathbf{z}}\big(\mathbf{z}^{(i)},\mathbf{z}^{(j)}\big) \\ &\quad +\frac{1}{N}\sum_{i=1}^{N} \log \sum_{j=1}^{N} k_{\mathbf{z}}\big(\mathbf{z}^{(i)},\mathbf{z}^{(j)}\big)\; k_{\mathbf{x}}\big(\mathbf{x}^{(i)},\mathbf{x}^{(j)}\big).\end{aligned} \tag{8}$$

In what follows, we use isotropic, Gaussian kernels for convenience

$$k_{\mathbf{x}}(\mathbf{x}^{(i)},\mathbf{x}^{(j)}) = \frac{1}{\big(2\pi\sigma_{\mathbf{x}}^2\big)^{d_{\mathbf{x}}/2}} \exp\left(-\frac{\|\mathbf{x}^{(i)}-\mathbf{x}^{(j)}\|^2}{2\sigma_{\mathbf{x}}^2}\right), \tag{9a}$$

$$k_{\mathbf{z}}(\mathbf{z}^{(i)},\mathbf{z}^{(j)}) = \frac{1}{\big(2\pi\sigma_{\mathbf{z}}^2\big)^{d_{\mathbf{z}}/2}} \exp\left(-\frac{\|\mathbf{z}^{(i)}-\mathbf{z}^{(j)}\|^2}{2\sigma_{\mathbf{z}}^2}\right), \tag{9b}$$

but one can also use anisotropic kernels.

Because the data distribution is fixed, maximizing the mutual information (2) reduces to maximizing $H(\mathbf{z}) - H(\mathbf{x},\mathbf{z})$. This is equivalent to minimizing the conditional entropy $H(\mathbf{x} \mid \mathbf{z})$, i.e., the expected negative log likelihood of the data under the joint density $p(\mathbf{x},\mathbf{z})$. It is interesting to note the similarity to the GPLVM [6], which directly minimizes the negative log likelihood of the data. Unlike the GPLVM, the KIE provides an explicit density over the latent space.

In general, kernel density estimates are known to perform poorly in high-dimensional spaces, even in the case where data are distributed along a lower dimensional manifold. The reason is that the choice of kernel bandwidth implies a tradeoff between capturing structure in the data on the one hand and reducing waste of probability mass on the other: A small bandwidth leads to density estimates where single points form isolated clusters; a large bandwidth necessarily wastes mass in areas where there are no data, that is, in directions orthogonal to the manifold. Since high-dimensional spaces are usually sampled sparsely, the waste of probability mass is particularly common in such cases.

It is important to note that KIE does not rely on a particularly "good" density estimate in the high-dimensional space because KIE projects data into a low-dimensional space, where it can avoid the waste of probability mass. Like other latent variable models, training a KIE model can be viewed as cutting a low-dimensional, nonlinear "slice" through the "thickened" manifold defined by the high-dimensional kernel density estimate. Since KIE normalizes that slice such that it defines an optimal density estimate in the *low-dimensional* space, unlike for the high-dimensional KDE its bandwidth can be chosen to properly reflect the structure *within the manifold*.

When data are sampled densely enough, one can also use anisotropic kernels to deal with the problem of high dimensions (e.g., see [32]). In the experiments that follow, we did explore the use of anisotropic kernels in the data space of the KIE, but we did not see any demonstrable advantage over isotropic kernels.

2.1 Regularization

Interestingly, the KIE learning problem as stated above does admit degenerate solutions. A trivial way to maximize $\hat{I}(\mathbf{x}, \mathbf{z})$ is to drive all latent positions infinitely far from one another. This can be seen by rewriting objective (8) as a linear combination of data-space kernel function evaluations, where the weights in the linear combination are given by a *softmax* which depends on the latent-space elements:

$$\hat{I}(\mathbf{x}, \mathbf{z}) = \hat{H}(\mathbf{x}) + \frac{1}{N}\sum_{i=1}^{N} \log \sum_{j=1}^{N} \frac{k_{\mathbf{z}}(\mathbf{z}^{(i)}, \mathbf{z}^{(j)})}{\sum_{l=1}^{N} k_{\mathbf{z}}(\mathbf{z}^{(i)}, \mathbf{z}^{(l)})} k_{\mathbf{x}}\big(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\big). \tag{10}$$

The easiest way to optimize the objective is then by converting the *softmax* to the *max*-function that picks, instead of averaging, the best kernel to evaluate.

To prevent the latent representatives from moving arbitrarily far apart, and thereby encouraging more meaningful solutions, one can adopt a prior (or regularizer). Here, we consider generalized Gaussian priors over the latent positions. This simply adds $\frac{\lambda}{N}\sum_j \|\mathbf{z}^{(j)}\|^{\beta}$ to (8) to create the regularized objective function. Here, λ controls the influence of the regularizer and β its shape, where $\beta = 2$ results in a Gaussian, i.e., radially symmetric, prior.

Similar priors are used to avoid degenerate solutions in the GPLVM and in other embedding methods, including spectral methods (such as [1], [2], [3], [26], [28]). While spectral methods constrain latent embeddings to be mean-centered and uncorrelated, KIE allows for a more general way to avoid degenerate solutions, as we demonstrate in Section 2.4.

The influence of the regularization constant λ on the embedding is illustrated in Fig. 3. A large value for λ leads to an oversmoothed latent density (leftmost plot on the top). A small value leads to overfitting (rightmost plot). A simple way to find an optimal value for λ is with cross validation. This can be achieved by using a sweep over different values, starting with a large value that is slowly decreased, and tracking a validation cost, such as reconstruction error[1] on a validation set, during the optimization. Slowly reducing the regularization can be viewed as a kind of "annealing" that helps find a good local optimum of the nonconvex objective function. We used this approach in most of our experiments.

2.2 Inference

Since KIE defines a joint kernel density estimate over latent representatives and observations, inference takes a particularly simple form. It is straightforward to show that the conditional distributions $p(\mathbf{x} \mid \mathbf{z})$ and $p(\mathbf{z} \mid \mathbf{x})$ are given by weighted kernel density estimates:

$$p(\mathbf{x} \mid \mathbf{z}) = \sum_{i=1}^{N} \frac{k_{\mathbf{z}}(\mathbf{z}, \mathbf{z}^{(i)})}{\sum_{j=1}^{N} k_{\mathbf{z}}(\mathbf{z}, \mathbf{z}^{(j)})} k_{\mathbf{x}}(\mathbf{x}, \mathbf{x}^{(i)}), \tag{11a}$$

$$p(\mathbf{z} \mid \mathbf{x}) = \sum_{i=1}^{N} \frac{k_{\mathbf{x}}(\mathbf{x}, \mathbf{x}^{(i)})}{\sum_{j=1}^{N} k_{\mathbf{x}}(\mathbf{x}, \mathbf{x}^{(j)})} k_{\mathbf{z}}\big(\mathbf{z}, \mathbf{z}^{(i)}\big). \tag{11b}$$

These conditional distributions are straightforward to compute and they may be multimodal. For Gaussian kernels, they become Gaussian mixtures.

2.3 Learning

Learning the KIE entails the maximization of $\hat{I}(\mathbf{x}, \mathbf{z})$ (8) plus the regularization term on the latent representatives $\mathbf{z}^{(i)}$. This can be done using any gradient-based optimization method (e.g., conjugate gradient). Toward this end, it follows from (8) that the gradient of $\hat{I}(\mathbf{x}, \mathbf{z})$ with respect to latent position $\mathbf{z}^{(i)}$ is the sum of two terms (since $\hat{H}(\mathbf{x})$ does not depend on $\mathbf{z}^{(i)}$):

$$\frac{\partial \hat{H}(\mathbf{z})}{\partial \mathbf{z}^{(i)}} = -\frac{1}{N}\sum_{j=1}^{N} \big(\kappa_{\mathbf{z}}^{i} + \kappa_{\mathbf{z}}^{j}\big) \frac{\partial k_{\mathbf{z}}(\mathbf{z}^{(i)}, \mathbf{z}^{(j)})}{\partial \mathbf{z}^{(i)}}, \tag{12}$$

$$\frac{\partial \hat{H}(\mathbf{x}, \mathbf{z})}{\partial \mathbf{z}^{(i)}} = -\frac{1}{N}\sum_{j=1}^{N} \big(\kappa_{\mathbf{xz}}^{i} + \kappa_{\mathbf{xz}}^{j}\big) k_{\mathbf{x}}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \frac{\partial k_{\mathbf{z}}(\mathbf{z}^{(i)}, \mathbf{z}^{(j)})}{\partial \mathbf{z}^{(i)}}, \tag{13}$$

where

$$\kappa_{\mathbf{z}}^{i} \equiv \frac{1}{\sum_{l=1}^{N} k_{\mathbf{z}}(\mathbf{z}^{(i)}, \mathbf{z}^{(l)})} \tag{14}$$

and

$$\kappa_{\mathbf{xz}}^{i} \equiv \frac{1}{\sum_{l=1}^{N} k_{\mathbf{x}}(\mathbf{x}^{(i)}, \mathbf{x}^{(l)}) k_{\mathbf{z}}(\mathbf{z}^{(i)}, \mathbf{z}^{(l)})}. \tag{15}$$

For the isotropic Gaussian kernel,

$$\frac{\partial k_{\mathbf{z}}(\mathbf{z}^{(i)}, \mathbf{z}^{(j)})}{\partial \mathbf{z}^{(i)}} = -k_{\mathbf{z}}(\mathbf{z}^{(i)}, \mathbf{z}^{(j)}) \frac{\mathbf{z}^{(i)} - \mathbf{z}^{(j)}}{\sigma_{\mathbf{z}}^{2}}.$$

Since we are optimizing the positions of latent representatives (and assuming isotropic kernels), any change in the bandwidth of the latent kernel, $k_{\mathbf{z}}$, is equivalent to rescaling the entire latent space. The choice of the latent-space bandwidth $\sigma_{\mathbf{z}}$ is therefore arbitrary, and for the remainder we assume a fixed bandwidth of $\sigma_{\mathbf{z}} = 1$.

The same argument does not hold for the bandwidth of the data-space kernel, $\sigma_{\mathbf{x}}$. A common heuristic, which we use below, is to set the bandwidth based on the average distance of nearest neighbors:

$$\sigma_{\mathbf{x}} = \frac{1}{N}\sum_{i=1}^{N} \big\| \mathbf{x}^{(i)} - \mathbf{x}^{(j(i))} \big\|, \tag{16}$$

where $j(i)$ is the index of the nearest neighbor of $\mathbf{x}^{(i)}$. One could also learn the bandwidth using cross validation.

Since the objective function is not convex, one should expect to find local optima. One strategy to obtain a good local optimum is to slowly decrease the influence of the regularization, as we discuss in Section 2.1.

2.4 Example: 2D S-Curve

Fig. 1 shows a learned KIE model of a two-dimensional "S-curve" manifold embedded in three dimensions. The

1. Fig. 3 shows average marker error in pose space, but one can use other task-specific measures of merit. See Section 4 for a formal definition of average marker error.

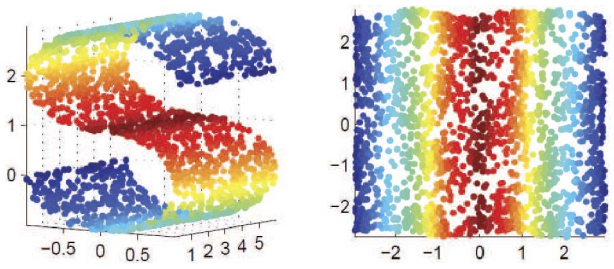

Fig. 1. Three-dimensional "S-curve" data set (left) and its two-dimensional embedding (right). The alignment of the embedding with the coordinate axes is encouraged with a penalty on the latent variables.

data set consists of 2,000 points sampled uniformly from the "S"-shaped sheet. The right plot shows a two-dimensional embedding with a color-coding scheme that reveals the underlying mapping.

We used a Gaussian kernel with $\sigma_{\mathbf{x}} = \sqrt{10.0}$. We used the regularization described in Section 2.1 with $\beta = 4$, resulting in an L4-penalty on the latent space. The contours of the L4-norm in two dimensions look like a "rounded box" that is aligned with the coordinate axes. Using this regularizer therefore encourages the embedding to align with the coordinate axes as well, as can be seen in the figure. One can also use an L2-penalty, but one would lose the axis alignment in this case (and, arguably, some of the interpretability of the embedding). Note that this kind of alignment would be impossible to achieve using, for example, a spectral method.

We initialized the latent representative $\mathbf{z}^{(i)}$ to small random values. We set λ to 0.1 initially and gradually reduced it (by 80 percent of its previous value) for 20 steps during the optimization. We have repeated the experiment 10 times with different random initializations and with no noticeable difference in the result (except that, in some cases, the embedding is rotated by 90 degrees in the latent space).

We also experimented with a wide range of kernel bandwidths, again without significant differences in the results. Note that, regardless of the bandwidth, the kernel density estimate necessarily underestimates the true density in this data set. The data set is drawn from an infinitely thin, two-dimensional sheet in the three-dimensional space, so the density is infinite within this sheet. Even though the estimate is necessarily suboptimal, KIE finds the underlying two-dimensional structure.

3 Shared Kernel Information Embedding

The Shared KIE (sKIE) is an extension of KIE to multiple (high-dimensional) data sets, with a single hidden cause that is responsible for the variability across all data sets. In what follows, we consider the case of two data sets, comprising image feature vectors, $\mathbf{x}$, and poses, $\mathbf{y}$. As illustrated in Fig. 2, the sKIE model is undirected, and hence, it has several natural factorizations and different ways of generating samples or performing inference. For example, with two data sets one could obtain samples $(\mathbf{x}, \mathbf{y})$ by first sampling from the latent distribution $\mathbf{z}^* \sim p(\mathbf{z})$, and then sampling $\mathbf{x}$ and $\mathbf{y}$ (independently) from their conditionals $p(\mathbf{y} \mid \mathbf{z}^*)$ and $p(\mathbf{x} \mid \mathbf{z}^*)$. Alternatively, given an observed feature vector $\mathbf{x}^*$, one could draw a sample from

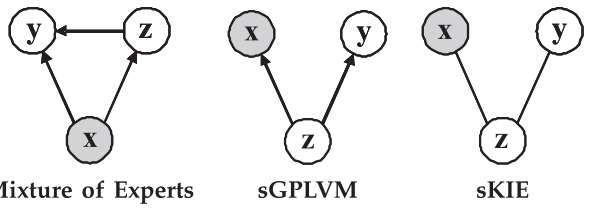

Fig. 2. Graphical models for regression problems. Gray and white nodes depict observed and hidden variables. MoE and sGPLVM are directed models. The Shared KIE is undirected, and can be factored easily in multiple ways.

the conditional latent distribution $\mathbf{z}^* \sim p(\mathbf{z} \mid \mathbf{x}^*)$, and then sample the conditional pose distribution, $p(\mathbf{y} \mid \mathbf{z}^*)$. While we continue to focus on two data sets below, the generalization to more than two is straightforward.

The sKIE joint embedding for two data sets is obtained by maximizing the mutual information $I((\mathbf{x}, \mathbf{y}), \mathbf{z})$. Assuming conditional independence of $\mathbf{x}$ and $\mathbf{y}$ given $\mathbf{z}$, the MI can be expressed as a sum of two MI terms, i.e.,

$$I((\mathbf{x}, \mathbf{y}), \mathbf{z}) = I(\mathbf{x}, \mathbf{z}) + I(\mathbf{y}, \mathbf{z}). \tag{17}$$

Like the KIE objective in (8), we maintain a fully nonparametric model, using kernel density estimates for the integrals in (17):

$$\hat{I}((\mathbf{x}, \mathbf{y}), \mathbf{z}) = \hat{I}(\mathbf{x}, \mathbf{z}) + \hat{I}(\mathbf{y}, \mathbf{z}). \tag{18}$$

In contrast to KIE, here the labeled training data $\{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{N}$ share a single hidden cause. Thus, each pair $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})$ is represented by a single, shared latent element $\mathbf{z}^{(i)}$. The use of shared hidden variables is reminiscent of canonical correlation analysis (CCA) and similar methods (e.g., [33]), which also represent two observable variables using a single shared cause. Like the KIE, which can be viewed as a nonlinear extension of probabilistic PCA, one could think of the sKIE as the corresponding analog of (a probabilistic formulation of) CCA. Because of the conditional independence assumption (cf., (17)), the sKIE, in contrast to an embedding of the concatenation $(\mathbf{x}, \mathbf{y})$, learns a representation of the *relationship* between $\mathbf{x}$ and $\mathbf{y}$. Only the similarities *within* an individual space determine the value of the kernel density estimate in each space, not the similarities across the spaces. It is the *embedding* that then represents structure across the individual density estimates and that thereby encodes their relations. A repercussion of this is that scaling and other properties of these spaces do not need to be comparable or to be measured on the same scale since no joint kernel has to simultaneously accommodate all of these. It would also be straightforward to extend sKIE to the case of factorized orthogonal latent models [34].

For the sKIE, semi-supervised learning is also possible with multiple data sets. For example, one might have image feature vectors $\mathbf{x}^{(j)}$ that correspond to images for which pose was unavailable. Alternatively, one might have pose data without the corresponding images. To handle incomplete data like this, some latent representatives will only be constrained directly by elements in one of the two data sets.

More formally, in the general case, let $\mathcal{I}_{\mathcal{X}}$ and $\mathcal{I}_{\mathcal{Y}}$ denote two sets of indices for the available training data points,

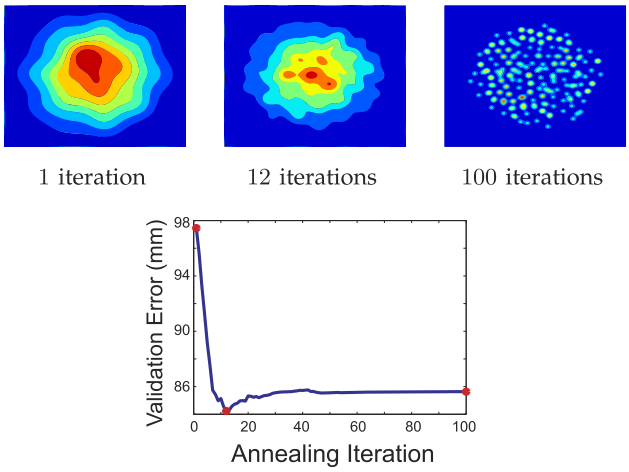

Fig. 3. Annealing with cross validation. The plot on the bottom shows cross-validation error as a function of the number of annealing iterations. The plots on the top show latent distributions $p(\mathbf{z})$ at annealing levels before, at, and after the minimum of the cross-validation curve. The latent space is initially concentrated, but then spreads out as the annealing progresses. In the limit, the regularizer has no influence and the latent points drift far apart.

with cardinalities $N_{\mathbf{x}}$ and $N_{\mathbf{y}}$. Indices for labeled training samples are included in both sets. Indices for training samples of $\mathbf{x}$ (respectively, $\mathbf{y}$) for which there is no corresponding sample from $\mathbf{y}$ ($\mathbf{x}$) exist only in $\mathcal{I}_{\mathcal{X}}$ ($\mathcal{I}_{\mathcal{Y}}$). If we then ignore the terms of the approximate mutual information (18) that are independent of the latent positions, $\mathbf{Z} \equiv \{\mathbf{z}^{(j)}\}_{j=1}^{N}$, where N is the cardinality of $\mathcal{I} = \mathcal{I}_{\mathcal{X}} \cup \mathcal{I}_{\mathcal{Y}}$, the sKIE objective function becomes

$$
\begin{aligned}
L(\mathbf{Z}) = \frac{1}{N_{\mathbf{x}}} \Bigg[& \sum_{i \in \mathcal{I}_{\mathcal{X}}} \log \sum_{j \in \mathcal{I}_{\mathcal{X}}} k_{\mathbf{z}}\big(\mathbf{z}^{(i)}, \mathbf{z}^{(j)}\big)\; k_{\mathbf{x}}\big(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\big) \\
& - \sum_{i \in \mathcal{I}_{\mathcal{X}}} \log \sum_{j \in \mathcal{I}_{\mathcal{X}}} k_{\mathbf{z}}\big(\mathbf{z}^{(i)}, \mathbf{z}^{(j)}\big) \Bigg] \\
& + \frac{1}{N_{\mathbf{y}}} \Bigg[\sum_{i \in \mathcal{I}_{\mathcal{Y}}} \log \sum_{j \in \mathcal{I}_{\mathcal{Y}}} k_{\mathbf{z}}\big(\mathbf{z}^{(i)}, \mathbf{z}^{(j)}\big) k_{\mathbf{y}}\big(\mathbf{y}^{(i)}, \mathbf{y}^{(j)}\big) \\
& - \sum_{i \in \mathcal{I}_{\mathcal{Y}}} \log \sum_{j \in \mathcal{I}_{\mathcal{Y}}} k_{\mathbf{z}}\big(\mathbf{z}^{(i)}, \mathbf{z}^{(j)}\big) \Bigg].
\end{aligned} \tag{19}
$$

3.1 Learning

Learning the sKIE entails the maximization of $L(\mathbf{Z})$ with respect to the unknown latent positions $\mathbf{Z}$. In the experiments below, we use a gradient-based approach. The gradients for sKIE optimization have the same basic form as those for KIE in (12) and (13).

Like KIE, regularization is necessary to constrain the model, for example, by using a mean-zero Gaussian prior over latent positions. We can utilize this prior by performing "annealing" in conjunction with a validation set in order to obtain a good local optimum of the learning objective. As mentioned above, we use cross validation to determine when to stop the annealing procedure. In particular, we use MSE in discriminative pose inference on a validation set as a measure of cross-validation error. Fig. 3 shows the effect of typical annealing on the latent sKIE space and illustrates the cross-validation error as a function of annealing iterations; the details of the data and model being learned here are given in Fig. 10.

3.2 Inference

For discriminative pose inference, we want to find likely poses $\mathbf{y}$ conditioned on input image features $\mathbf{x}^*$. We are therefore interested in the conditional pose distribution:

$$
p(\mathbf{y} \mid \mathbf{x}^*) = \int_{\mathbf{z}} p(\mathbf{y} \mid \mathbf{z})\, p(\mathbf{z} \mid \mathbf{x}^*)\, d\mathbf{z}. \tag{20}
$$

We have explicit closed-form expressions for the two conditional factors in the integrand in (20), but $p(\mathbf{y} \mid \mathbf{x}^*)$ is not straightforward to express or compute in closed form. While the second factor in the integrand is a mixture of Gaussian functions of $\mathbf{z}$, the first factor is a conditional mixture whose weights given by *normalized* Gaussian functions of $\mathbf{z}$ (cf., (11)).

The integral can be approximated with Monte Carlo integration, by drawing latent samples and then pose samples as described above, but this can be computationally expensive, especially when $p(\mathbf{y} \mid \mathbf{x}^*)$ is multimodal. Alternatively, here we focus on identifying the principal modes of $p(\mathbf{y} \mid \mathbf{x}^*)$. To this end, we assume that the principal modes of $p(\mathbf{y} \mid \mathbf{x}^*)$ coincide with the principal modes of the conditional latent distribution $p(\mathbf{z} \mid \mathbf{x}^*)$. That is, we first search for local maxima (MAP estimates) of $p(\mathbf{z} \mid \mathbf{x}^*)$, denoted $\{\mathbf{z}_k^*\}_{k=1}^{K}$ for K modes. From these latent points, it is straightforward to perform either MAP inference or take expectations over the conditional pose distributions $p(\mathbf{y} \mid \mathbf{z}_k^*)$.

To understand this form of approximate inference, it is useful to note that pose is often not fully constrained by image features. That is, it is often the case that the conditional distribution over pose, i.e., $p(\mathbf{y} \mid \mathbf{x}^*)$, is multimodal. Under the sKIE formulation, the latent distribution models the joint distribution, and therefore, when we condition the latent space on input features, we expect the distribution $p(\mathbf{z} \mid \mathbf{x}^*)$ to be similarly multimodal. The conditional distribution over pose, given a single latent position, say $\mathbf{z}_k^*$, is typically unimodal.

3.2.1 Local Modes of $p(\mathbf{z} \mid \mathbf{x}^*)$

As with KIE, conditional distributions for sKIE take the form of weighted kernel density estimates (cf., (11)). To find modes of $p(\mathbf{z} \mid \mathbf{x}^*)$, one can choose one or more starting points, and then some form of gradient ascent (e.g., with mean shift [35]). Starting points could be found by evaluating (11b) for each latent representative, and then selecting those with the highest conditional density. One could also draw random samples from $p(\mathbf{z} \mid \mathbf{x}^*)$, or, among the training exemplars, one could find nearest neighbors to $\mathbf{x}^*$ in feature space, and then begin gradient ascent from their latent representatives (cf., [15]). The most probable latent representatives found in this way are often sufficiently probable in the latent space that subsequent optimization is unnecessary.

3.2.2 Pose Inference

Because the conditional pose distribution is typically unimodal, it is reasonable to use conditional expectation

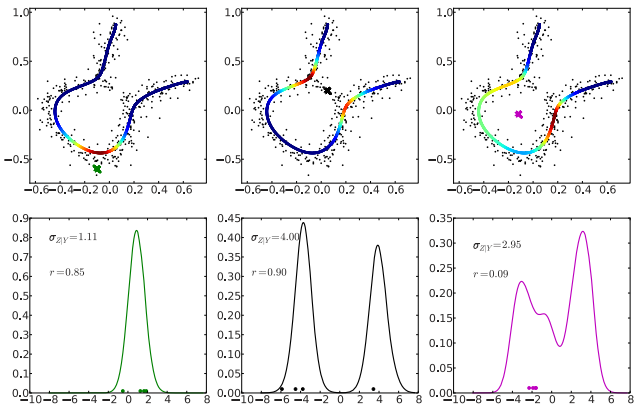

Fig. 4. Multimodal inference. The plots in the top row show a two-dimensional data set (the same in each plot) and the learned one-dimensional KIE manifold. Conditioning on an observable point $\mathbf{x}$ (a different one in each plot) can lead to an inferred latent density that is unimodal (left plot) or multimodal (center plot, right plot). The density is shown using color coding, where red corresponds to high-density regions and dark blue to low-density regions. For better visibility, the row below depicts the same latent densities in the one-dimensional latent space.

to find the mean and covariance in the pose space. From the form of the conditional distributions (11a), the mean of $p(\mathbf{y} \mid \mathbf{z}^*)$ is a convex combination of training exemplars. That is,

$$E[\mathbf{y} \mid \mathbf{z}^*] = \sum_{i=1}^{N} \frac{k_{\mathbf{z}}(\mathbf{z}^*, \mathbf{z}^{(i)})}{\sum_{j=1}^{N} k_{\mathbf{z}}(\mathbf{z}^*, \mathbf{z}^{(j)})} \mathbf{y}^{(i)} \,. \tag{21}$$

With some algebraic manipulation, one can also derive the form of the conditional covariance, that is,

$$\begin{aligned} C_{\mathbf{y} \mid \mathbf{z}^*} = C_{k_{\mathbf{y}}} + \sum_i \frac{k_{\mathbf{z}}(\mathbf{z}^*, \mathbf{z}^{(i)})}{\sum_l k_{\mathbf{z}}(\mathbf{z}^*, \mathbf{z}^{(l)})} \mathbf{y}^{(i)} \mathbf{y}^{(i)\mathrm{T}} \\ - \sum_{i,j} \frac{k_{\mathbf{z}}(\mathbf{z}^*, \mathbf{z}^{(i)}) k_{\mathbf{z}}(\mathbf{z}^*, \mathbf{z}^{(j)})}{\left(\sum_l k_{\mathbf{z}}(\mathbf{z}^*, \mathbf{z}^{(l)})\right)^2} \mathbf{y}^{(i)} \mathbf{y}^{(j)\mathrm{T}}, \end{aligned} \tag{22}$$

where $C_{k_{\mathbf{y}}}$ is the kernel covariance matrix; here, $C_{k_{\mathbf{y}}} = \sigma_{\mathbf{y}}^2 \mathbf{I}$.

3.3 Latent Densities

Unlike other nonlinear embedding models, such as the GPLVM and spectral methods, the latent representation of an out-of-sample observation $\mathbf{x}$ for both the KIE and the sKIE is a *density*, not a point-estimate (cf., (11b)).

Fig. 4 illustrates how the KIE latent densities make ambiguities in the mapping from data space to latent space explicit, using a one-dimensional synthetic example. The plots in the top row show a two-dimensional data set and a learned low-dimensional manifold, defined as the conditional mean of the latent space mapped back into the data space. The bottom row of the figure shows the same densities in the one-dimensional latent space.

In each of the three plots, a different element $\mathbf{x}$ is chosen as a conditioning data point (depicted using a cross symbol). The color of the manifold encodes the value of the conditional latent density at the corresponding latent position and shows how ambiguities in the mapping lead to multimodal latent densities. Multiple factors can influence these densities: In Fig. 4 (center plot), the curvature of the manifold leaves unclear which "branch" of the manifold generated the observation, leading to a projection ambiguity and thus to a bimodal latent density.

Latent densities also reflect nonuniformities in the *data density*, as shown in Fig. 4 (right plot), where a dense region in the data density provides extra support for one of the latent modes (the "rightmost" mode in the bottom plot) which, as a result, is larger than the other two.

Also shown in the plots at the bottom are some random samples drawn from the latent densities. Sampling latent-space elements is a common operation in many applications of both probabilistic and nonprobabilistic latent variable models (see also Section 3.2). The KIE densities make it possible to know how *representative* a sample set is of the density.

A simple way of computing how well a sample set represents the density is to compare sample statistics with expectations under the distribution. For example, let r be the latent standard deviation divided by the empirical standard deviation of the sample set. The closer r is to 1, the more representative the sample set is. In particular, "missed" modes in the latent-space density will typically correspond to a small value for r. In Fig. 4, latent-space variances and the ratios r are shown next to the plots of the latent densities (bottom row). The rightmost plot illustrates the case where the sample set is not representative of the latent density.

One might also compute the "significance" of a mode by fitting a Gaussian centered at the mode. To this end, note that the Hessian $\boldsymbol{H}(\mathbf{z} \mid \mathbf{x})$ of the negative log (conditional) density $p(\mathbf{z} \mid \mathbf{x})$, using Gaussian kernels (9), takes the form

$$\boldsymbol{H}(\mathbf{z} \mid \mathbf{x}) = \frac{\partial^2 L}{\partial \mathbf{z} \partial \mathbf{z}^{\mathrm{T}}} = 2\boldsymbol{I} + 4 \sum_{i,l} (\gamma_i \gamma_l - \delta_{il} \gamma_i)(\mathbf{z} - \mathbf{z}^i)(\mathbf{z} - \mathbf{z}^l)^{\mathrm{T}}, \tag{23}$$

where

$$\gamma_i = \frac{\omega_i k(\mathbf{z}, \mathbf{z}^i)}{\sum_j \omega_j k(\mathbf{z}, \mathbf{z}^j)}, \quad \omega_i = \frac{k(\mathbf{x}, \mathbf{x}^i)}{\sum_j k(\mathbf{x}, \mathbf{x}^j)} \,, \tag{24}$$

$\delta_{il} = 1$, iff $i = l$, and $\boldsymbol{I}$ is the $q \times q$ identity matrix. One can obtain a local approximation to the latent-space density at mode $\mathbf{z}_0$ using a Gaussian with inverse covariance matrix $\boldsymbol{H}(\mathbf{z}_0 \mid \mathbf{x})$ whose normalizing constant is given by $p(\mathbf{z}_0 \mid \mathbf{x})(2\pi)^{-\frac{d}{2}} \mid \boldsymbol{H}(\mathbf{z}_0 \mid \mathbf{x}) \mid^{-\frac{1}{2}}$. For unimodal densities, this approximation is known as the Laplace approximation (e.g., see [36]).

3.4 Complexity

Learning KIE and sKIE has complexity $O(N^2)$, while inference (cf., (11)) is $O(N)$. Since learning and the search for latent modes during inference (where required) do involve iterative optimization,; however, the number of iterations is an additional multiplicative constant. Because of this and since KIE is a nonparametric method, these operations can be time consuming on large data sets, and it can be useful to make use of parallelization, as we discuss also in Section 3.6.

In general, the complexity of learning and inference compares favorably with the GPLVM, for which learning is $O(N^3)$ due to the inversion of the $N \times N$ kernel matrix, and inference is $O(N^2)$. Indeed, in quadratic time with the KIE

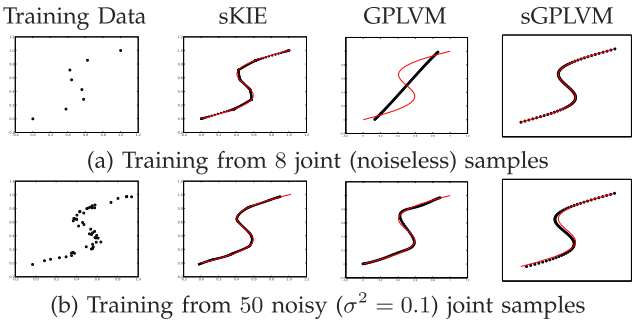

Fig. 5. Comparison of sKIE, GPLVM, and sGPLVM on S-curve data. The red curves depict the true mean of the joint density. For the sKIE and GPLVM, the blue points are produced by uniformly sampling latent positions $\mathbf{z}$, and then taking the mean of $p(\mathbf{x}, \mathbf{y} \mid \mathbf{z})$. For sGPLVM, the latent positions are optimized with NN initialization before taking the mean of $p(\mathbf{y} \mid \mathbf{z})$.

one could, in principle, use gradient ascent on the conditional distributions (11) from each training point to find the majority of the modes.

For both the GPLVM and KIE, one can also achieve greater efficiencies with sparsification methods (e.g., [27]), and numerical approximations such as fast multipole methods (e.g., [37]). For KIE, one can also employ fast mean-shift algorithms for efficient mode finding on the conditional distributions (e.g., [38]).

3.5 S-Curve Data

To demonstrate sKIE on a simple problem, following [15], [17], we first consider data from a one-dimensional synthetic "S-curve." Points were sampled from a 2D density $p(\mathbf{x}, \mathbf{y})$ defined by

$$\mathbf{x} = t + sin(2\pi t) + \eta_{\mathbf{x}}, \quad \mathbf{y} = t + \eta_{\mathbf{y}},$$

where $\eta_{\mathbf{x}}$ and $\eta_{\mathbf{y}}$ are IID mean-zero Gaussian noise with variance σ^2 and $t = \mathcal{U}(0, 1)$ is uniform. The conditional density $p(\mathbf{y} \mid \mathbf{x})$ has up to three modes.

Fig. 5 shows sKIE, GPLVM, and the shared GPLVM (sGPLVM) models learned from noisy and noiseless data (the GPLVM was learned from the joint data samples). Our sGPLVM model was based on the formulation and implementation in [12]. Bandwidths were optimized based on the standard GPLVM hyperprior. All models learn a 1D latent space, encoding the shared structure of the 2D joint distribution. The GPLVM does not capture the S-curve with only eight samples, consistent with [15]. sKIE and sGPLVM do a better job with eight points. Interestingly, our sGPLVM results are inconsistent with the results in [15], which claimed that eight samples are insufficient to capture the structure of the S-curve. We found that initialization using shared PCA and proper setting of kernel bandwidth for back-constraints in sGPLVM are critical in achieving the illustrated sGPLVM performance. In the presence of the noise, sKIE recovers the structure of the curve well, while the GPLVM and sGPLVM exhibit a small bias in the upper lobe. Finally, we note that the Mixture of Experts (MoE) model cannot be trained with eight points. With 50 points, MoE can be trained but typically overfit and overestimate the variance (see [15]).

Fig. 6 shows sKIE learned from partially labeled data. While sKIE tolerates significant amounts of unlabeled data, the model often generalizes well without it, so the unlabeled data mainly help to reduce the variance of the output estimates. With only eight fully labeled samples, the reconstruction of the joint distribution is smooth (see Fig. 5), but the variance is high because the data are sparse.

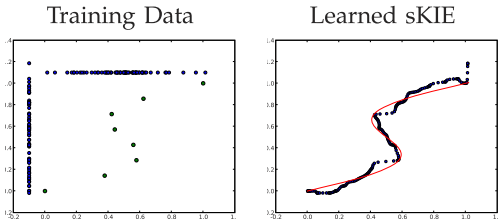

Fig. 6. Semi-supervised learning. The sKIE is learned from 8 joint and 142 marginal noisy samples. The curve is not as smooth as in the fully supervised case, but the model does tolerate nearly 20 times as many unlabeled as labeled samples.

3.6 Implementation on Graphics Hardware

For both the KIE and the sKIE, inference (cf., Sections 2.2 and 3.2) as well as gradient-based learning (cf., Sections 2.3 and 3.1) comprises element-wise evaluation of simple expressions, such as "exp," "power 2," and "multiply," over large arrays. This allows for straightforward parallelization using graphics processing units (GPU). To that end, note that because KIE is a nonparametric model, it is defined as a function of the training data. Therefore, once training data and other storage have been initialized on a GPU, learning does not require any further transfer of data between GPU and main memory. However, when using a general-purpose, gradient-based optimizer for learning, the gradients and model parameters (latent representatives) do need to be transferred between parameter updates. Since the main computational bottleneck in learning is the calculation of the gradient and objective function, not memory transfer, we found this to be a reasonable approach nevertheless. Conjugate gradients, in particular, can be very efficient at optimizing the model objective. For inference, one can use batches of test data to minimize the effect of latency when transferring data to and from the GPU. We provide GPU-based implementations[2] of KIE and sKIE in the Python language at http://learning.cs.toronto.edu/~rfm/kie.

4 Human Pose Estimation

We next consider human pose inference. We use two data sets: one synthetic, called POSER [11], and the HUMANEVA data set [39] with synchronized video and mocap data.

4.1 Poser Data Set

POSER contains 1,927 training and 418 test images, synthetically generated from mocap data[3] (54 joint angles per frame). The image features and error metric are provided with the data set [11]. The 100D feature vectors encode the image silhouette using vector-quantized shape contexts.

2. Training an embedding of an S-curve with 2,000 points using a full 15-step annealing schedule on an NVIDIA GeForce GTX 580 takes well less than 1 minute. For comparison, a naive Python implementation on a standard PC takes about 20 minutes. Similarly, inference on 2,000 query-points using mode-finding takes about 40 seconds on the GPU versus several minutes with the native Python implementation on a CPU.

3. The data set is available at http://lear.inrialpes.fr/pubs/2004/AT04/Data/index.html.

To measure errors in estimated poses, we use the mean RMS error, averaged over all joints. This is given by

$$E_{ang}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{M} \sum_{m=1}^{M} | \, (\hat{\mathbf{y}}_i - \mathbf{y}_i) \mod 360^\circ \, | \, . \qquad (25)$$

Here, $M = 54$, and $\mathbf{y}_i$ and $\hat{\mathbf{y}}_i$ correspond to the ith joint angles for the true and estimated poses.

4.2 HumanEva Data Set

HUMANEVA-I [39] contains synchronized multiview video and mocap data. It comprises three subjects performing multiple activities. Here, we use walking and jogging sequences[4] with observations from three color cameras.[5] This includes 5,985 training samples (image-pose pairs) and 6,291 test samples (for global models, we only use subject S1, for which there are 2,190 training and 2,625 test samples). Where smaller training sets are used, data are randomly sampled from the entire training set.

4.2.1 Features

Following [40], our features are based on shape context descriptors. From each image, we extract a silhouette using background subtraction to which we fit a bounding box. The shape context representation is constructed by randomly sampling 400 points on internal edges and outer contours of the silhouette, from which histograms are constructed.[6] We then cluster 40,000 randomly sampled histograms to learn a codebook of size 300. Shape context histograms are subsequently vector-quantized using that codebook. The final 300D feature vectors are normalized to unit length. This choice of feature vector was motivated by simplicity and ease of implementation; better features have been shown to perform favorably on HUMANEVA-I (e.g., hierarchical features [14], HMAX, Spatial Pyramid, Hyperfeatures, and Vocabulary Trees have all been explored).

4.2.2 Errors

Pose is encoded by 15 3D joint centers defined relative to the pelvis in camera-centric coordinates, so $\mathbf{y} \in \mathbb{R}^{45}$. Estimation errors (in mm) are measured as an average euclidean distance to the $M = 15$ markers [39]:

$$E_{pos}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{M} \sum_{m=1}^{M} \|(\hat{\mathbf{y}}_i - \mathbf{y}_i)\|. \qquad (26)$$

4.3 Monocular Pose Inference

Fig. 7 shows how the performance of sKIE depends on the number of training examples. For each of POSER and HUMANEVA, we perform 10 runs using a different random initialization and a different random subset of the training data in each run. We fixed the number of latent dimensions to 10 as we found it to yield good performance on these types of data. One could also set this number using cross validation; this would take longer to learn, but would likely improve performance.

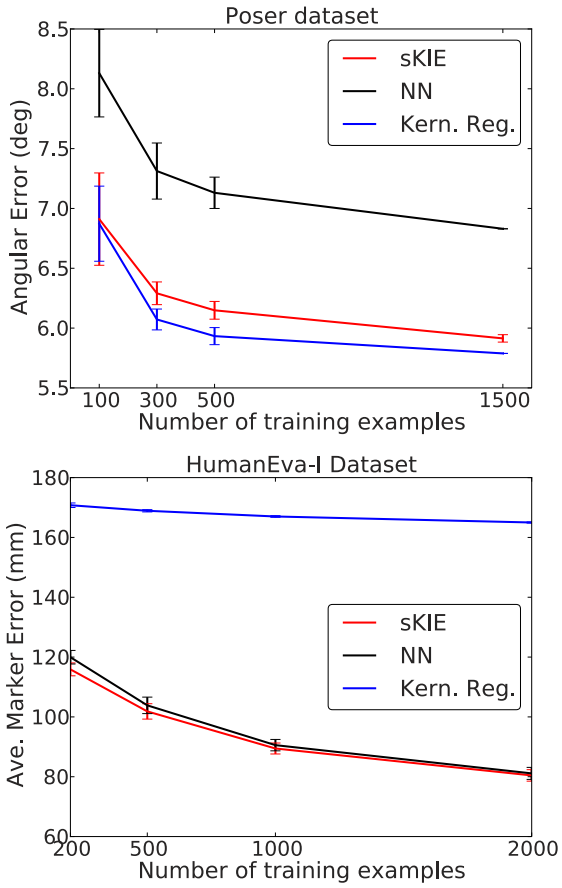

Fig. 7. Single hypothesis inference. The plots show the average error with standard error bars for sKIE, NN Regression, and Kernel Regression on POSER (top) and HUMANEVA-I (bottom) as a function of the training set size.

For this experiment and others below, unless stated otherwise sKIE inference proceeds as follows: Given a feature vector $\mathbf{x}^*$, we find the most probable training exemplar according to the conditional latent distribution $p(\mathbf{z} \mid \mathbf{x}^*)$, from which we use mean shift to find a local maximum. Conditioned on this point, $\mathbf{z}^*$, we compute the mean pose from $p(\mathbf{y} \mid \mathbf{z}^*)$. We also implemented Nearest Neighbor (NN) regression and kernel regression (with Gaussian kernels). Performance for all estimators is simply the average error (i.e., (25) and (26)) over the respective test sets. Standard error bars are also shown in all cases.

sKIE consistently outperforms NN Regression. Kernel regression performs favorably compared to sKIE on POSER data when the number of training examples is greater than 100. We postulate that this is due in part to the relatively clean data that does not exhibit much multimodality. Kernel regression is sensitive to the kernel bandwidth and performance can quickly degrade as the kernel bandwidth increases.[7] sKIE seems to be relatively insensitive to kernel bandwidths in both the input and output spaces.

4. We ignore walking from subject 3 due to corrupt motion capture data.
5. Like [40], [19], we do not use the four grayscale views.
6. Histograms are computed at 12 angular and 5 radial bin log-polar resolution, with minimal and maximal radial extent set to 1/8 and 3 of the mean distance between all 400 sampled points. The histograms are thereby invariant to the overall scale of the silhouette.

7. Performance of kernel regression on HUMANEVA-I in Fig. 7 can be improved by manually tuning kernel bandwidths, but for consistency with sKIE we used the same heuristic, (16), for all models in all experiments.

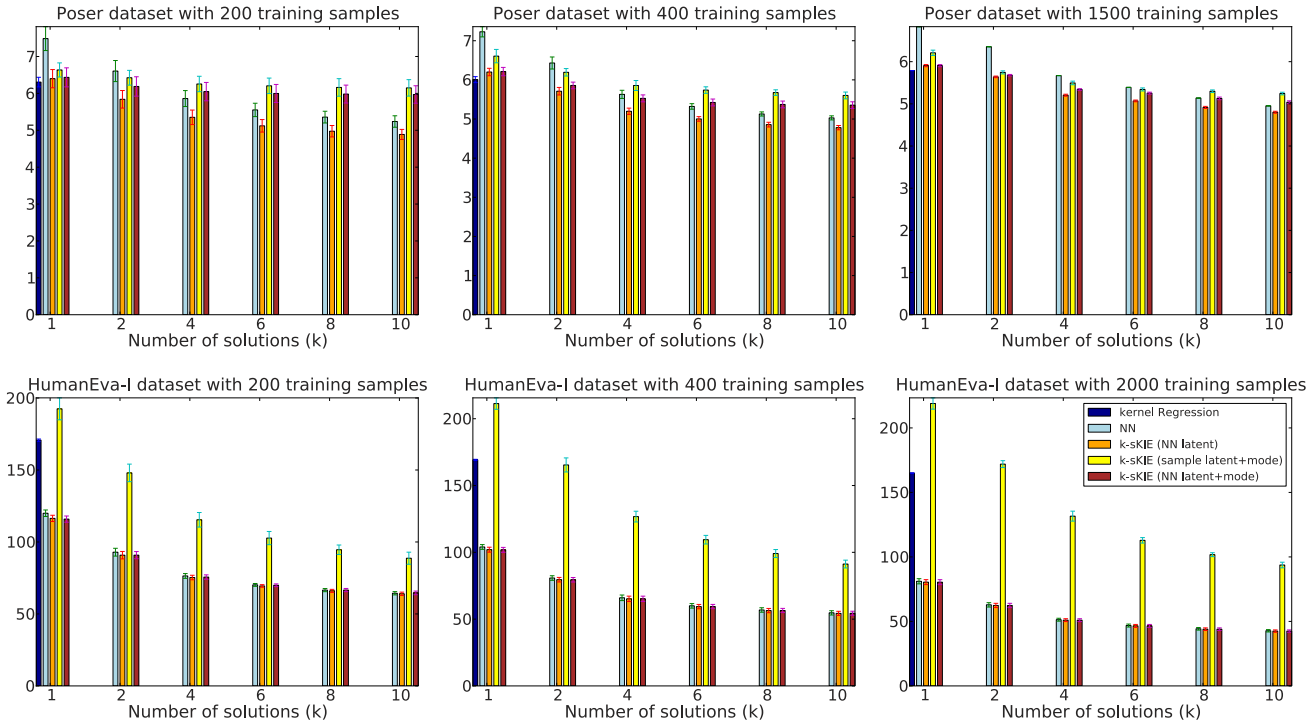

Fig. 8. Multihypothesis inference. Graphs show the performance of sKIE as a function of the number of hypotheses, k (for 200, 400, and 1,500 training examples for POSER data and 200, 400, and 2,000 training examples for HUMANEVA-I data). We compare performance of sKIE with k-Nearest Neighbor Regression and Kernel Regression; different inference methods are explored as described in the text. The plots show angular error (in degrees) for POSER and average marker distance (in mm) for HUMANEVA-I. The plots show 1-standard deviation error bars.

We also conducted a set of experiments in which sKIE produces multiple predictions (called k-sKIE, for k predictions). In doing so, we explore several alternate forms of inference: 1) finding the latent representatives associated with the k nearest neighbors of $\mathbf{x}^*$ in the input feature space and computing the k mean poses conditioned on the latent points; 2) sampling k samples from $p(\mathbf{z} \mid \mathbf{x}^*)$, followed by mode finding, and then estimation of the mean pose from the conditional pose distribution; and 3) like (1) but with intermediate latent mode finding. In all cases, the annealing schedule was $\lambda_{i+1} = 0.9\lambda_i$ starting with $\lambda_0 = 0.5$ and was run for a maximum of 20 annealing iterations.

Fig. 8 shows the results. For comparison, we also include the performance of k-nearest neighbors and kernel regression. At $k = 1$, the performance of conditional mode inference is similar to that of conditional neighbor inference for POSER data. For HUMANEVA-I data, mode inference performs slightly better. It is interesting to note that for POSER data the performance of conditional mode inference (initialized with both sampling or nearest neighbors) tends to *level out* (at $k = 2$ for 200 training points, $k = 4$ for 400 training points, and $k = 8$ for 1,500 points). This is a strong indication that the number of modes is very small so that mode-finding reduces the diversity in the set of neighbors (or samples) used as initialization and suggests restricting the number of solutions to the respective number k at which the leveling-off occurs.

In any case, it is clear that the latent model carries significant value in regularizing the inference, i.e., with sKIE we always perform better than k-NN for any k both in the POSER and HUMANEVA-I experiments.

4.4 Local Monocular Pose Estimation

For very large training data sets, learning a coherent global model can be prohibitively expensive. In such cases, we can learn local sKIE models online for inference.

We demonstrate the use of local sKIE models using the entire HUMANEVA-I data set. For each test case, we first find the 25 training samples whose features vectors are closest to the test feature vector. From those 25 samples, we learn an sKIE with a 2D latent space. To speed up the online learning, sKIE was trained by running a fixed (5) number of annealing iterations starting at $\lambda = 0.5$ and taking 100 gradient steps for every annealing iteration. Once trained, the mode of the conditional latent distribution was found, from which the conditional mean was used as the estimated pose. The full data sets were used for both testing and training. As shown in Fig. 9, the online sKIE performs favorably with respect to all methods considered, including the GPLVM, whose performance was reported with POSER data in [12].

Algorithm / Dataset	HUMANEVA-I	POSER
Linear Regression [12]	-	7.70 (deg)
Nearest Neighbor	81.12 (mm)	6.83 (deg)
GPLVM [12]	-	6.50 (deg)
Kernel Regression	164.99 (mm)	5.79 (deg)
Gaussian RVM [11]	-	6.00 (deg)
Global sKIE	80.44 (mm)	5.91 (deg)
Local sKIE (25 neigh)	64.63 (mm)	5.77 (deg)

Fig. 9. Performance of the local sKIE model.

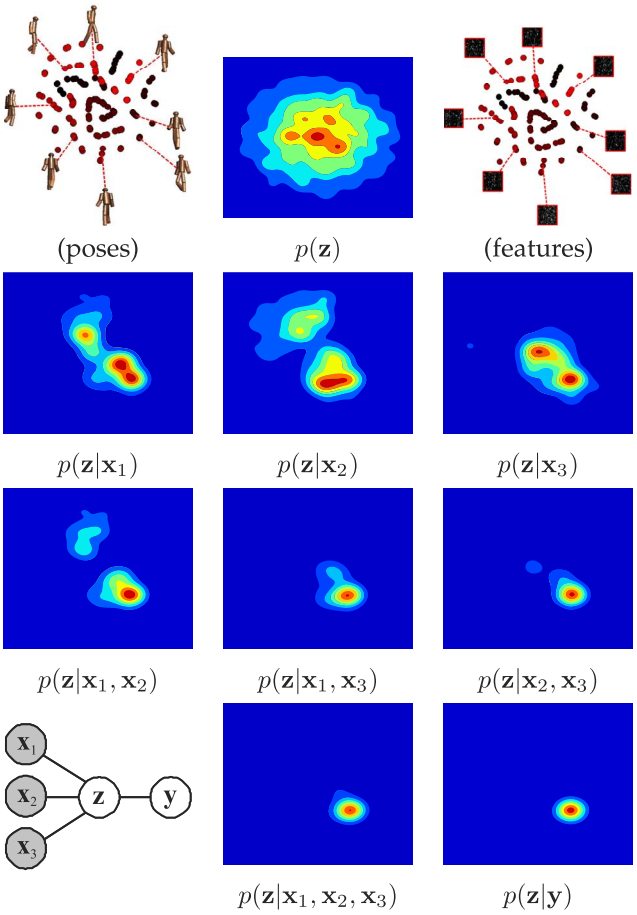

Fig. 10. Shared KIE for multiview pose inference. The sKIE is learned from 200 random samples of walking from Subject 1 in HUMANEVA-I. Inputs are 300D shape context features from each of three synchronized cameras, $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$, plus a 45D pose vector (i.e., a joint model over 947-dimensional data). The top row depicts the learned 2D shared latent space. The remaining rows depict conditional latent-space distributions obtained by conditioning on subsets of the views and on the true pose (*bottom-right*). The top-left figure illustrates individual training sample positions in the latent space colored by their proximity within the sequence. Nearby poses do end up close to one another in the latent space. Also note that the conditional distributions obtained by conditioning on observations and on the pose give consistent densities in the latent space.

4.5 Multivariable Shared-KIE Models

An advantage of sKIE over direct regression models (e.g., MoE) is its ability to learn joint models over many variables, allowing conditioning on any subset of them at test time. With other regression models, a separate regression function (or conditional distribution) would have to be learned between all combinations of test inputs. To illustrate these benefits of sKIE, we trained two models with more than one input and one output, one for multiview pose inference and one for tracking.

Multiview pose inference. Using the HUMANEVA-I data set, we learned an sKIE with input features from three synchronized cameras $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$ and pose $\mathbf{y}$. The model can be conditioned on any subset of inputs. The results, explained in Fig. 10, clearly show the ambiguities that arise in pose inference from only one or two views.

Monocular tracking. For Bayesian pose tracking, we want to condition on both the current image features, $\mathbf{x}_t$, and the pose at the previous time, $\mathbf{y}_p$. Fig. 11 shows such a model. It can be used to initialize a tracking discriminatively, or generatively as a motion prior. Other methods (e.g., [10], [17]) would need several models to perform the same task. Again, in Fig. 11 it is interesting to see that the conditional latent distributions (conditioned on image features) are sometimes multimodal (frame 430) and sometimes unimodal (frame 586).

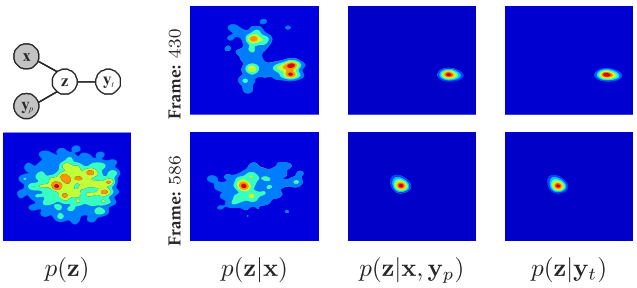

Fig. 11. Shared KIE for tracking. Illustrated are the latent space, the latent space conditioned on one current observation, and latent space conditioned on the current observation and previous pose. We utilize 200 samples from HUMANEVA-I data set (camera 1) used to train the model in Fig. 10.

5 Conclusions

We introduced a probabilistic latent variable model called the KIE and its extension, the sKIE. We applied the model to human pose inference. The model has several appealing properties, namely that 1) it has favorable complexity of $O(N^2)$ for training and $O(N)$ for inference, 2) it utilizes a latent space as an intermediary in the inference process and hence allows learning from small data sets (i.e., can generalize well), and 3) it provides closed-form multimodal conditional distributions, conditioning on the input space (or spaces) of features, the shared latent space, and the output pose space. Furthermore, the model can deal with missing data during inference, and partially labeled data during learning. For training, one can use gradient-based optimization. Computation of gradients and cost functions allow for the straightforward use of graphics hardware.

An interesting direction for future research is to use sampling of latent and observable densities during learning, which could further improve the accuracy of the learned embeddings. Furthermore, additional parametric mappings (e.g., [41], [42]) could be incorporated to obtain fast feedforward inference where required, while retaining a fully probabilistic model that can help detect and resolve ambiguities and assess uncertainties in the predictions.

Another direction for research is the use of sparse priors on the latent variables. This would allow the model to learn sparse, overcomplete representations in a fully nonparametric way, which can be useful, for example, in nonlinear demixing tasks.

Acknowledgments

This work was supported in part by NSERC, Canada, by the Canadian Institute for Advanced Research (CIFAR), and by the German Federal Ministry of Education and Research (BMBF) in the project 01GQ0841 (BFNT Frankfurt). The authors would also like to thank the reviewers for several useful suggestions.

REFERENCES

[1] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear Component Analysis as a Kernel Eigenvalue Problem," *Neural Computation,* vol. 10, pp. 1299-1319, July 1998.

[2] J.B. Tenenbaum, V. Silva, and J.C. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction," *Science,* vol. 290, no. 5500, pp. 2319-2323, Dec. 2000.

[3] S.T. Roweis and L.K. Saul, "Nonlinear Dimensionality Reduction by Locally Linear Embedding," *Science,* vol. 290, pp. 2323-2326, 2000.

[4] Y. Bengio., J.-F. Paiement, P. Vincent, O. Delalleau, N.L. Roux, and M. Ouimet, "Out-of-Sample Extensions for LLE, Isomap, MDS, Eigenmaps, and Spectral Clustering," *Proc. Advances in Neural Information Processing Systems,* pp. 177-184, 2004.

[5] M.E. Tipping and C.M. Bishop, "Probabilistic Principal Component Analysis," *J. Royal Statistical Society, Series B,* vol. 61, pp. 611-622, 1999.

[6] N.D. Lawrence, "Probabilistic Non-Linear Principal Component Analysis with Gaussian Process Latent Variable Models," *J. Machine Learning Research,* vol. 6, pp. 1783-1816, Nov. 2005.

[7] P. Meinicke, S. Klanke, R. Memisevic, and H. Ritter, "Principal Surfaces from Unsupervised Kernel Regression," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 27, no. 9, pp. 1379-1391, Sept. 2005.

[8] R. Memisevic, "Kernel Information Embeddings," *Proc. Int'l Conf. Machine Learing,* pp. 633-640, 2006.

[9] L. Sigal, R. Memisevic, and D. Fleet, "Shared Kernel Information Embedding for Discriminative Inference," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* pp. 2852-2859, 2009.

[10] A. Agarwal and B. Triggs, "Learning to Track 3D Human Motion from Silhouettes," *Proc. Int'l Conf. Machine Learning,* pp. 9-16, 2004.

[11] A. Agarwal and B. Triggs, "3D Human Pose from Silhouettes by Relevance Vector Regression," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* vol. 2, pp. 882-888, 2004.

[12] C. Ek, P. Torr, and N. Lawrence, "Gaussian Process Latent Variable Models for Human Pose Estimation," *Proc. Int'l Conf. Machine Learning for Multimodal Interaction,* pp. 132-143, 2007.

[13] T. Jaeggli, E. Koller-Meier, and L.V. Gool, "Monocular Tracking with a Mixture of View-Dependent Learned Models," *Proc. Conf. Articulated Motion and Deformable Objects,* pp. 494-503, 2006.

[14] A. Kanaujia, C. Sminchisescu, and D. Metaxas, "Semi-Supervised Hierarchical Models for 3D Human Pose Reconstruction," *Proc. IEEE Conf. Computer Vision Pattern Recognition,* 2007.

[15] R. Navaratnam, A. Fitzgibbon, and R. Cipolla, "The Joint Manifold Model for Semi-Supervised Multi-Valued Regression," *Proc. 11th IEEE Int'l Conf. Computer Vision,* 2007.

[16] G. Shakhnarovich, P. Viola, and T. Darrell, "Fast Pose Estimation with Parameter-Sensitive Hashing," *Proc. Ninth IEEE Int'l Conf. Computer Vision,* vol. 2, pp. 750-759, 2003.

[17] C. Sminchisescu, A. Kanaujia, Z. Li, and D. Metaxas, "Discriminative Density Propagation for 3D Human Motion Estimation," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* vol. 1, pp. 390-397, 2005.

[18] Z. Lu, M. Carreira-Perpinan, and C. Sminchisescu, "People Tracking with the Laplacian Eigenmaps Latent Variable Model," *Advances in Neural Information Processing Systems 20,* J. Platt, D. Koller, Y. Singer, and S. Roweis, eds. MIT Press, pp. 1705-1712, 2008.

[19] R. Urtasun and T. Darrell, "Local Probabilistic Regression for Activity-Independent Human Pose Inference," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2008.

[20] L. Sigal, A. Balan, and M.J. Black, "Combined Discriminative and Generative Articulated Pose and Non-Rigid Shape Estimation," *Proc. Neural Information Processing Systems,* 2007.

[21] T. de Campos and D. Murray, "Regression-Based Hand Pose Estimation from Multiple Cameras," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* vol. 1, pp. 782-789, 2006.

[22] M.Á. Carreira-Perpiñán, "Reconstruction of Sequential Data with Probabilistic Models and Continuity Constraints," *Proc. Neural Information Processing Systems,* pp. 414-420, 1999.

[23] A. Thayananthan, R. Navaratnam, B. Stenger, P. Torr, and R. Cipolla, "Multivariate Relevance Vector Machines for Tracking," *Proc. European Conf. Computer Vision,* pp. 124-138, 2006.

[24] L. Bo and C. Sminchisescu, "Twin Gaussian Processes for Structured Prediction," *Int'l J. Computer Vision,* 2010.

[25] A. Shon, K. Grochow, A. Hertzmann, and R. Rao, "Learning Latent Structure for Image Synthesis and Robotic Imitation," *Proc. Neural Information Processing Systems,* pp. 1233-1240, 2006.

[26] A. Kanaujia, C. Sminchisescu, and D. Metaxas, "Spectral Latent Variable Models for Perceptual Inference," *Proc. IEEE Int'l Conf. Computer Vision,* 2007.

[27] J. Quiñonero-Candela and C. Rasmussen, "A Unifying View of Sparse Approximate Gaussian Process Regression," *J. Machine Learning Research,* vol. 6, pp. 1939-1959, 2006.

[28] M.A. Carreira-Perpiñan and Z. Lu, "The Laplacian Eigenmaps Latent Variable Model," *J. Machine Learning Research W&P,* vol. 2, pp. 59-66, 2007.

[29] R. Memisevic, "Non-Linear Latent Factor Models for Revealing Structure in High-Dimensional Data," PhD dissertation, Univ. of Toronto, 2008.

[30] T.M. Cover and J.A. Thomas, *Elements of Information Theory.* John Wiley & Sons, 1991.

[31] D.W. Scott, *Multivariate Density Estimation: Theory, Practice, and Visualization* (Wiley series in probability and statistics). Wiley, Sept. 1992.

[32] T. Brox, B. Rosenhahn, D. Cremers, and H.-P. Seidel, "Nonparametric Density Estimation with Adaptive Anisotropic Kernels for Human Motion Tracking," *Proc. Second Int'l Workshop Human Motion,* 2007.

[33] M. Kuss and T. Graepel, "The Geometry of Kernel Canonical Correlation Analysis," Technical Report 108, Max Planck Inst. for Biological Cybernetics, Tübingen, Germany, May 2003.

[34] M. Salzmann, C.H. Ek, R. Urtasun, and T. Darrell, "Factorized Orthogonal Latent Spaces," *Proc. 13th Int'l Conf. Artificial Intelligence and Statistics,* 2010.

[35] D. Comaniciu and P. Meer, "Mean Shift: A Robust Approach Toward Feature Space Analysis," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 5, pp. 603-619, May 2002.

[36] D.J.C. MacKay, *Information Theory, Inference, and Learning Algorithms.* Cambridge Univ. Press, 2003.

[37] V. Raykar and R. Duraiswami, *The Improved Fast Gauss Transform with Applications to Machine Learning.* MIT Press, 2006.

[38] B. Han, D. Comaniciu, Y. Zhu, and L. Davis, "Sequential Kernel Density Approximation: Application to Real-Time Visual Tracking," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 30, no. 7, pp. 1186-1197, July 2008.

[39] L. Sigal and M.J. Black, "HumanEva: Synchronized Video and Motion Capture Data Set for Evaluation of Articulated Human Motion," Technical Report CS-06-08, Brown Univ., 2006.

[40] L. Bo, C. Sminchisescu, A. Kanaujia, and D. Metaxas, "Fast Algorithms for Large Scale Conditional 3D Prediction," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2008.

[41] M. Ranzato, C. Poultney, S. Chopra, and Y. Lecun, "Efficient Learning of Sparse Representations with an Energy-Based Model," *Proc. Advances in Neural Information Processing Systems,* 2006.

[42] M.Á. Carreira-Perpiñán and Z. Lu, "Parametric Dimensionality Reduction by Unsupervised Regression," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1895-1902, 2010.

Roland Memisevic received the PhD degree in computer science from the University of Toronto, Canada, in 2008. During and after his graduate studies, he held positions as a research intern at Microsoft Research, Redmond, Washington, as a research scientist at PNYLab LLC in Princeton, New Jersey, and as a postdoctoral fellow at the University of Toronto and at ETH Zurich. In 2011, he joined the Department of Computer Science, University of Frankfurt, Germany, as an assistant professor of computer science. His research interests include machine learning and computer vision, in particular in unsupervised learning and feature learning. His scientific contributions include higher order and relational sparse coding models, approaches to learning motion and transformation patterns from images and videos, and approaches to invariant recognition. He presented this work at conferences such as NIPS, CVPR, ICML, ICCV, AAAI, and in journals such as the *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *Neural Networks*, and *Neural Computation*. He has served as a program committee member or reviewer for most of these and other conferences and journals in machine learning and computer vision.

Leonid Sigal received the BSc degrees in computer science and mathematics from Boston University in 1999, the MA degree from Boston University in 1999, the MS degree from Brown University in 2003, and the PhD degree from Brown University in 2008. He is a research scientist at Disney Research, Pittsburgh, Pennsylvania, colocated with Carnegie Mellon University. From 1999 to 2001, he worked as a senior vision engineer at Cognex Corporation. He was a postdoctoral fellow in the Department of Computer Science at the University of Toronto from 2007 to 2009. His research interests include computer vision, visual perception, machine learning, and character animation. He has published numerous research articles and book chapters in top venues within these fields, including the *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *International Journal of Computer Vision*, CVPR, ICCV, ECCV, NIPS, and ACM Siggraph. The topics of published articles include human pose estimation and tracking, human motion perception, latent variable models, physics-based models, fast approximate cloth simulation, and motion-capture. His work received the Best Paper Award at the Articulate Motion and Deformable Objects Conference in 2006 (with Prof. Michael J. Black). He maintains an active professional service within the community. As part of that service, he regularly serves on program committees and reviews papers for major computer vision, machine learning, and computer graphics conferences, and has organized three workshops and two tutorials (in conjunction with NIPS, CVPR, ECCV, and ICCV). He has also coedited the book *Guide to Visual Analytics of Humans: Looking at People* (Springer, 2011). He is a member of the IEEE.

David J. Fleet received the PhD degree in computer science from the University of Toronto in 1991. He was on the faculty at Queen's University in Kingston from 1991 to 1998, and then area manager and research scientist at the Palo Alto Research Center (PARC) from 1999 to 2003. In 2004, he joined the University of Toronto as a professor of computer science. His research interests include computer vision, image processing, visual perception, and visual neuroscience. He has published research articles, book chapters, and one book on various topics including the estimation of optical flow and stereoscopic disparity, probabilistic methods in motion analysis, modeling appearance in image sequences, motion perception and human stereopsis, hand tracking, human pose tracking, latent variable models, and physics-based models for human motion analysis. In 1996, he was awarded an Alfred P. Sloan Research Fellowship for his work on computational models of perception. He has won paper awards at ICCV 1999, CVPR 2001, UIST 2003, and BMVC 2009. In 2010, he was awarded the Koenderink Prize with Michael Black and Hedvig Sidenbladh for his work on human pose tracking. He has served as area chair for numerous computer vision and machine learning conferences. He was program cochair for the 2003 IEEE Conference on Computer Vision and Pattern Recognition. He will be program cochair for the 2014 European Conference on Computer Vision. He has been an associate editor and associate editor-in-chief for the *IEEE Transactions on Pattern Analysis and Machine Intelligence*, and currently serves on the TPAMI Advisory Board. He is a senior member of the IEEE.