Hierarchical Spectral Latent Variable Models (HSLVM)

Cristian Sminchisescu and Catalin Ionescu
University of Bonn, Faculty of Mathematics and Natural Sciences, INS,
HUMOREV, Computer Vision and Machine Learning Group
{Cristian.Sminchisescu|Catalin.Ionescu}@ins.uni-bonn.de; http://sminchisescu.ins.uni-bonn.de

We present algorithms for learning hierarchical latent variable representations that combine the computational properties of factorized, tree dependency structures, with latent variable models offering the geometrical and topological features of spectral, non-linear embeddings. We build on our recent work on Spectral Latent Variable Models (SLVM) [1] and extend it to learning hierarchies that can represent multiple levels of correlation in the data. Our stagewise, hierarchical algorithm (HLSVM) boils down to learning a partially observed, non-linear, directed graphical model with tree-dependency. We use probabilistic formulations and ML training, see [2] for an alternative model and cost. For motivation, consider the case of a running person. The intrinsic dimensionality of a runner is the one of a 1d harmonic oscillator. But this minimalist model does not account for stylistic differences among people or for the lack of synchronization produced by external influences (obstacles, carry-on bags) present in many real-world situations. What seems appropriate is a hierarchy, with the strongest, lowest-dimensional model of correlation at the top (say), the weakest high-dimensional model of limbs moving unrestrictedly at the bottom, and various degrees of flexibility in-between (*e.g.* regularities among subsets of variables for each leg or arm, but no global constraints among them). The hierarchy can be used for the visual inference of 3D human body pose from images, either by estimating several representation levels simultaneously, or by automatically adapting the level of complexity to match the statistical regularity of the observation.

Spectral Latent Variable Models (SLVM): Assume vector-valued points in ambient space $\mathcal{Y} = \{\mathbf{y}_i\}_{i=1\ldots N}$ captured from a high-dimensional process, and corresponding latent space points $\mathcal{X} = \{\mathbf{x}_i\}_{i=1\ldots N}$, initially obtained using a spectral, non-linear embedding method like ISOMAP, LLE, Hessian or Laplacian Eigenmaps, *etc*. We model the joint distribution over latent and ambient variables as: $p(\mathbf{x},\mathbf{y}) = p(\mathbf{x})p(\mathbf{y}|\mathbf{x})$. The latent space prior $p(\mathbf{x})$ is modeled as a non-parametric kernel density estimate, with covariance $\boldsymbol{\theta}$: $p(\mathbf{x}) = \frac{1}{K}\sum_{i=1}^{K} K_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{x}_i)$. In the model, we assume that ambient vectors are related to the latent ones using a nonlinear vector-valued function with parameters $\mathbf{W}$ and noise covariance $\boldsymbol{\sigma}$: $p(\mathbf{y}|\mathbf{x},\mathbf{W},\boldsymbol{\sigma}) \sim \mathcal{N}(\mathbf{y}|\mathbf{F}(\mathbf{x},\mathbf{W}),\boldsymbol{\sigma})$, where $\mathcal{N}$ is a Gaussian distribution with mean $\mathbf{F}$ and covariance $\boldsymbol{\sigma}$. $\mathbf{F}$ is a generalized regression model: $\mathbf{F}(\mathbf{x},\mathbf{W}) = \mathbf{W}\boldsymbol{\phi}(\mathbf{x})$ with $\boldsymbol{\phi}(\mathbf{x}) = [K_{\boldsymbol{\delta}}(\mathbf{x},\mathbf{x}_1),\ldots,K_{\boldsymbol{\delta}}(\mathbf{x},\mathbf{x}_M)]^\top$, and kernels with covariance $\boldsymbol{\delta}$ placed at an M-size subset of $\mathbf{x}_i$. $\mathbf{W}$ is a weight matrix of size DxM. We use hierarchical priors on the parameters $\mathbf{W}$ in order to select a sparse subset for prediction [3, 1].

The ambient marginal is obtained by integrating the latent space. The evidence, as well as derivatives w.r.t. model parameters, are computed using a Monte Carlo (MC) estimate using, say S, samples from the prior. This gives the MC estimate of the ambient marginal: $p(\mathbf{y}|\mathbf{W},\boldsymbol{\sigma}) = \int p(\mathbf{y}|\mathbf{x},\mathbf{W},\boldsymbol{\sigma})p(\mathbf{x})\mathbf{dx} \approx \frac{1}{S}\sum_{s=1}^{S} p(\mathbf{y}|\mathbf{x}^{(s)},\mathbf{W},\boldsymbol{\sigma})$. The latent space conditional is obtained using Bayes' rule: $p(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{y})} = \frac{S}{K}\frac{p(\mathbf{y}|\mathbf{x})\sum_{i=1}^{K} K_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{x}_i)}{\sum_{s=1}^{S} p(\mathbf{y}|\mathbf{x}^{(s)},\mathbf{W},\boldsymbol{\sigma})}$. For pairs of ambient data points j and MC latent samples i, we abbreviate $p_{(i,j)} = p(\mathbf{x}_i|\mathbf{y}_j)$. The choice of prior $p(\mathbf{x})$ influences the membership probabilities. We can compute either the conditional mean or the mode (better for multimodal distributions) in latent space, using the same MC integration method: $\mathrm{E}\{\mathbf{x}|\mathbf{y}_n,\mathbf{W},\boldsymbol{\sigma}\} = \int p(\mathbf{x}|\mathbf{y}_n,\mathbf{W},\boldsymbol{\sigma})\mathbf{x}\mathbf{dx} = \sum_{i=1}^{K} p_{(i,n)}\mathbf{x}_i$, where $i_{max} = \arg\max_i p_{(i,n)}$. The model contains the ingredients for efficient computation in both latent and ambient space: a prior in latent space, an ambient marginal, the conditional distribution from latent to ambient space, and vice-versa. Latent conditionals given partially observed $\mathbf{y}$ vectors are easy computable – the conditional distribution of $\mathbf{y}$ is Gaussian and unobserved components can be integrated analytically (see [1] for details). We train by maximizing the log-likelihood of the data: $\mathcal{L} = \log \prod_{i=1}^{N} p(\mathbf{y}_n|\mathbf{W}, \boldsymbol{\sigma}) = \sum_{n=1}^{N} \log\{\frac{1}{S} \sum_{s=1}^{S} p(\mathbf{y}_n|\mathbf{x}^{(s)}, \mathbf{W}, \boldsymbol{\sigma})\}$. The model is fitted using EM, and maximizing the likelihood gives estimates for $\mathbf{W}, \boldsymbol{\sigma}$. In the *E-step* we compute the membership probabilities of latent points generating datapoints, $p_{(i,j)}$. In the *M-step* we learn the sparse mapping and its noise model $(\mathbf{W}, \boldsymbol{\sigma})$, by solving a weighted regression problem [1]. The SLVM is the building block used for learning hierarchies, described next.

Learning a Hierarchical SLVM (HSLVM): Our learning algorithm follows three steps: the construction of intermediate latent variable sub-manifolds connected in a tree structure, the initialization of data and parameters at all pairs of nodes (or levels), and the EM-iteration. Learning boils down to computing marginals at each node using the current parameters, and solving a decoupled set of SLVM problems for each child-parent node, in breadth-first order.

Tree Construction: We build the tree by partitioning the original state space $\mathbf{y}$ (at the bottom level of the tree, at present not automatically) into subsets of variables and constructing latent variable model sub-manifolds for each. At the next, more compact level, subsets of latent variables are grouped again, and new latent variable models are learned for each group. This becomes the data modeled by the emerging level *above*, in order to obtain the next level of latent variables, and so on, all the way to the root of the tree. The directionality of the child-parent relations goes opposite to the way the tree is built – top-down from the root, corresponding to the most compact model, through intermediate levels that encode different degrees of correlation among variables, all the way to the leaves (groups of variables of the original state space). Each layer in the tree is as a possible representation of the data. Observed data $\mathbf{y}$ can be generated from a given level by following the dependency structure and local node mappings/conditionals, down to leaves.

Initialization: At each level of tree construction we compute non-linear embeddings using the datapoints representing subsets of variables in the level below, in order to obtain the data for the level above. The identity of the data is preserved, hence we always combine (complementary) latent representations corresponding to the same original datapoint and don't crossover between different ones. Once the data at each node is available, KDE approximations to marginals (latent 'priors' in SLVMs) are constructed and mappings to the level below are learnt (*i.e.* learn individual SLVMs for all parent-child node pairs).

EM iteration: *E-step:* Given parameters, compute marginals at each node. These are Gaussian mixtures obtained using MC estimates from the 'prior' of each SLVM, computed as the marginal at its parent node. *M-step:* Solve decoupled SLVM problems for each child-parent node. This requires samples from both marginals (the child's to compute the MC estimate and the parent's to generate the data used at current EM iteration). The latter is also used for the MC estimate of the next layer. Hence the SLVMs are optimally learned breadth first. Notice that following initialization, the only marginal that is not re-estimated is the one at the root (the initial KDE estimate).

Topic: visual processing and pattern recognition. Preference: poster

References

[1] A. Kanaujia, C. Sminchisescu, and D. Metaxas. Spectral Latent Variable Models for Perceptual Inference. In *CVPR*, 2007.

[2] N. Lawrence and A. Moore. Hierarchical Gaussian Process Latent Variable Models. In *ICML*, 2007.

[3] D. Mackay. Comparison of Approximate Methods for Handling Hyperparameters. *Neural Computation*, 11(5), 1998.