Fast Algorithms for Large Scale *Conditional* 3D Prediction

Liefeng Bo[1], Cristian Sminchisescu[2,1]
[1]TTI-C, [2]University of Bonn

Atul Kanaujia[3], Dimitris Metaxas[3]
[3]Rutgers University

Abstract

*The potential success of discriminative learning approaches to 3D reconstruction relies on the ability to efficiently train predictive algorithms using sufficiently many examples that are representative of the typical configurations encountered in the application domain. Recent research indicates that sparse conditional Bayesian Mixture of Experts (cMoE) models (*e.g. *BME [21]) are adequate modeling tools that not only provide contextual 3D predictions for problems like human pose reconstruction, but can also represent multiple interpretations that result from depth ambiguities or occlusion. However, training conditional predictors requires sophisticated double-loop algorithms that scale unfavorably with the input dimension and the training set size, thus limiting their usage to 10,000 examples of less, so far. In this paper we present large-scale algorithms, referred to as f*BME*, that combine forward feature selection and bound optimization in order to* train probabilistic, BME models, with one order of magnitude more data (100,000 examples and up) and more than one order of magnitude faster. *We present several large scale experiments, including monocular evaluation on the HumanEva dataset [19], demonstrating how the proposed methods overcome the scaling limitations of existing ones.*

1. Introduction

This paper is motivated by our interest in making large scale conditional (also known as discriminative) 3D human pose prediction methods practical in terms of training time, number of examples and input / output dimensions. Although we demonstrate human pose prediction, the methods – generically known as conditional mixture of experts (cMoE) – are potentially relevant to a larger community, including researchers who study 3D reconstruction or object recognition.

The versatility of cMoE [21, 22] relies on a balanced combination of several attractive properties, some long sought by computer vision researchers: (*i*) *conditioning on input* eliminates the need for simplifying naive Bayes assumptions, common with generative models, and allows for diverse, potentially non-independent feature functions of the input (in this case, the image) to be encoded in its descriptor. This makes possible to model non-trivial image correlations and enhances the predictive power of the input representation. (*ii*) *multivaluedness of outputs* allows for multiple plausible hypotheses – as opposed to a single one – to be faithfully represented; (*iii*) *contextual predictions* offer versatility by means of ranking (gating) functions that are paired with the experts, and adaptively score their competence in providing solutions, for each input. This allows for nuanced, finely tuned responses; (*iv*) *probabilistic consistency* enforces data modeling according to its density via formal, conditional likelihood parameter training procedures; (*v*) *Bayesian formulations and automatic relevance determination mechanisms* favor sparse models with good generalization capabilities. All these features make the cMoE model suitable for fast, automatic feedforward 3D prediction, either as a stand alone, indexing system, or as an initialization method, in conjunction with complementary visual search and feedback mechanisms [21, 22, 3].

Nevertheless, a significant downside of existing cMoE algorithms [10, 5, 2, 11, 22] is their scalability. The algorithms require an expensive double loop algorithm (an iteration within another iteration) based on Newton optimization, to compute the gate functions, a factor that makes models with more than 10,000 datapoints and large input dimension impractical to train. In this paper we present new, computationally efficient cMoE algorithms that combine forward feature selection based on marginal likelihood and functional gradient boosting with techniques based on bound optimization, in order to train models that are one order of magnitude larger (100,000 examples and up), in time that is more than one order of magnitude faster than previous methods. We present several large scale experiments, including quantitative monocular evaluation on the HumanEva [19] dataset, demonstrating that the algorithms are accurate and overcome the scaling challenges of existing ones.

1.1. Related Work

This research connects to structured prediction, feature selection, and conditional mixture modeling, as well as vi-

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

sual human pose estimation. Discriminative methods for human pose reconstruction have recently seen a revival, as a result of advances in feature extraction and machine learning methods. The algorithms range from nearest-neighbor [18, 15], to regression, probabilistic mixture of predictors and their conditional counterparts [16, 21, 20]. See [11, 14] for multivalued extensions based on semi-supevised manifold methods.

Sparse probabilistic Bayesian formulations for regression and conditional mixture of experts have been presented in [1] and [21], respectively, but both use backward elimination to select features, which makes them less efficient – the sub-problems solved during the first steps of model training are high-dimensional and require large memory storage and substantial computational resources. Forward predictive regression methods exist [23, 24] but they have not been adapted to the conditional mixture of experts problem. Vincent and Bengio [24] proposed kernel matching pursuit and discussed back-fitting, and Friedman [7] shows how to perform feature selection in function space, for arbitrary differentiable loss functions. Efficient forward selection methods for Gaussian Process learning are given in [25, 17], for a tutorial see [8].

Besides the input dimension or the dataset size, models trained using Conditional-EM [10, 5, 2, 9, 21] face additional bottlenecks: fitting the gate functions requires iterative second-order methods. Bound optimization methods for general C-EM algorithms have been discussed in [9], but obtaining global variational upper bounds is expensive and requires the computation of the Hessian matrix w.r.t. the model parameters at each iteration.

Our fast, large-scale algorithm for conditional mixture of Bayesian expert models (f*BME*) employs techniques based on forward feature selection and bound optimization in order to sequentially (and greedily) optimize lower bounds on the conditional likelihood of the model, given training data. We use forward selection schemes based on decomposing the marginal likelihood with respect to one additional feature, in order to train the experts, and use feature selection based on functional gradient boosting for training the gates. Fitting of the gates is a convex, but non-quadratic problem that requires iterative methods. To make these fast, we exploit a remarkable, *input dependent, constant lower bound on the Hessian matrix* of the gate likelihood w.r.t. their parameters, and efficiently construct updates using an alternation scheme. To our knowledge, the algorithm we propose is novel for both computer vision and machine learning, and appears to be the first of this kind capable of training *conditional* Bayesian mixture of experts models (BME) with multivariate (high-dimensional) inputs and outputs, using datasets of 100,000 examples or more, in time that makes it reasonably practical.[1]

[1]Notice the difference between conditional models and clusterwise expert models, where data is partitioned and an expert is fit to each one. Clusterwise expert models do not face scaling problems, but lack expert ranking. To use multivalued models without a supplementary verification step, one needs *conditional parameterizations*. These do not only provide multiple predictions, but also their consistent contextual ranking.

2. Fast Conditional Algorithms (*f*BME)

This section describes efficient algorithms for training sparse conditional Bayesian mixtures of experts with high-dimensional inputs and for large training sets. To simplify notation, we review models with one state (output) dimension, being understood that the formulation generalizes to multivariate state spaces, either by training separate models for each output or by extending a single model to provide vector-valued (as opposed to scalar) prediction.

2.1. Conditional Mixture of Experts

We work with a probabilistic conditional model:

$$P(x|\mathbf{r}) = \sum_{j=1}^{M} g_j(\mathbf{r}) p_j(x) \tag{1}$$

with:

$$g_j(\mathbf{r}) \equiv g(\mathbf{r}|\boldsymbol{\lambda}_j) = \frac{e^{\boldsymbol{\lambda}_j^\top \mathbf{r}}}{\sum_k e^{\boldsymbol{\lambda}_k^\top \mathbf{r}}} \tag{2}$$

$$p_j(x) \equiv p_j(x|\mathbf{r}, \mathbf{w}_j, \sigma_j^2) \sim \mathcal{N}(x|\mathbf{w}_j^\top \mathbf{r}, \sigma_j^2 \mathbf{I}) \tag{3}$$

where $\mathbf{r}$ are predictor variables, x are outputs or responses, g are *input dependent* positive gates. g are normalized to sum to 1 for consistency, by construction, for any given input $\mathbf{r}$. In the model, p are Gaussian distributions (3) with variance $\sigma^2\mathbf{I}$, centered at linear regression predictions given by models with weights $\mathbf{w}$. Whenever possible, we drop the index of the experts *(but not the one of the gates)*. The weights of experts have Gaussian priors, controlled by hyperparameters $\boldsymbol{\alpha}$:

$$p(\mathbf{w}|\boldsymbol{\alpha}) = (2\pi)^{-D/2} \prod_{d=1}^{D} \alpha_d^{1/2} \exp\{-\frac{\alpha_d w_d^2}{2}\} \tag{4}$$

with $\dim(\mathbf{w}) = D$. The parameters of the model, including experts and gates are collectively stored in $\boldsymbol{\theta} = \{(\mathbf{w}_i, \boldsymbol{\alpha}_i, \sigma_i, \boldsymbol{\lambda}_i) \mid i = 1 \dots M\}$.

To learn the model, we design iterative, approximate Bayesian EM algorithms. In the E-step we estimate the posterior:

$$h_j \equiv h_j(x, \mathbf{r}|\mathbf{w}_j, \sigma_j, \boldsymbol{\lambda}_j) = \frac{g_j(\mathbf{r}) p_j(x)}{\sum_{k=1}^{M} g_k(\mathbf{r}) p_k(x)} \tag{5}$$

and let $h_j^{(i)} = h_j(x^{(i)}, \mathbf{r}^{(i)})$ be the probability that the expert j has generated datapoint i. Parenthesized superscripts index datapoints. In the M-step we solve two optimization problems, one for each expert and another for its gate. The first learns the expert parameters, based on training data weighted according to the current membership estimates

h. The second optimization trains the gates g to predict h. The complete log-likelihood (Q-function) for the conditional mixture of Bayesian experts can be derived as [10]:

$$Q = \sum_{i=1}^{N} \log P(x^{(i)}|\mathbf{r}^{(i)}) = \tag{6}$$

$$= \sum_{i=1}^{N}\sum_{j=1}^{M} h_j^{(i)}(\log g_j^{(i)} + \log p_j^{(i)}) = \tag{7}$$

$$= L_g + L_p \tag{8}$$

The likelihood decomposes into two factors, one for the gates and the other for the experts. The experts can be fitted independently using sparse Bayesian learning, under the change of variables $\mathbf{r}^{(t)} \leftarrow \sqrt{h^{(t)}}\mathbf{r}^{(t)}$ and $x^{(t)} \leftarrow \sqrt{h^{(t)}}x^{(t)}$. The equations for the gates are coupled and require iteration *during each* M-step. Although the problem is convex, it is computationally expensive to solve because the cost is not quadratic and the inputs are high-dimensional. A classical iteratively reweighted least squares (IRLS), or a naive Newton implementation, requires $\mathcal{O}(N(MD)^2 + (MD)^3)$ computation, multiple times during each step which is prohibitive for large problems (*e.g.* for 15 experts and 10000 training samples with 1000 input dimension, the computational cost becomes untenable even on today's most powerful desktops). Note that the cost of computing the Hessian (the first complexity term above) becomes higher than the one of inverting it (the second term) when the number of training samples is very large.

2.2. Training the Experts

For Bayesian learning with Gaussian priors and observation likelihoods, the expert posterior and predictive uncertainty (marked with '*') are computable in closed form:

$$\boldsymbol{\mu} = \sigma^2\boldsymbol{\Sigma}\mathbf{R}\mathbf{X}, \boldsymbol{\Sigma} = (\sigma^{-2}\mathbf{R}\mathbf{R}^\top + \mathbf{A})^{-1} \tag{9}$$

$$x^* = \boldsymbol{\mu}^\top\mathbf{r}, (\sigma^2)^* = \mathbf{r}^\top\boldsymbol{\Sigma}\mathbf{r} \tag{10}$$

where $\mathbf{A} = \text{diag}[\alpha_1, \ldots, \alpha_D]$, $\mathbf{R}$ stores the training set inputs columnwise and $\mathbf{X}$ their corresponding vector of x-outputs (see fig. 1 for illustration). The marginal likelihood of the experts is:

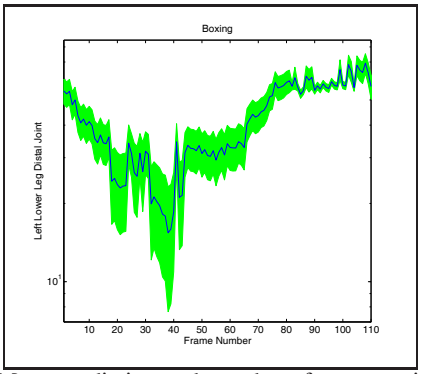

Figure 1. Mean prediction and errorbars for one variable of our Bayesian model (see (9) for derivations).

$$L_p(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \log p(x^{(i)}|\mathbf{r}^{(i)}, \boldsymbol{\alpha}, \sigma^2) = \tag{11}$$

$$= \sum_{i=1}^{N} \log \int p(x^{(i)}|\mathbf{r}^{(i)}, \mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha})\mathbf{dw} = \tag{12}$$

$$= -\frac{1}{2}\{N\log 2\pi + \log|\mathbf{K}| + \mathbf{X}^\top\mathbf{K}^{-1}\mathbf{X}\} \tag{13}$$

where $\mathbf{K} = \sigma^2\mathbf{I} + \mathbf{R}^\top\mathbf{A}^{-1}\mathbf{R}$. It can be shown that the marginal likelihood decomposes as [23]:

$$L_p(\boldsymbol{\alpha}) = L_p(\boldsymbol{\alpha}_{\setminus i}) + l(\alpha_i) \tag{14}$$

with

$$l(\alpha_i) = \frac{1}{2}\{\log\alpha_i - \log(\alpha_i + s_i) + \frac{q_i^2}{\alpha_i + s_i}\} \tag{15}$$

where $s_i = \mathbf{C}_i^\top\mathbf{K}_{\setminus i}^{-1}\mathbf{C}_i$ and $q_i = \mathbf{C}_i^\top\mathbf{K}_{\setminus i}^{-1}\mathbf{X}$, $\mathbf{C}_i$ collects the ith column from the matrix $\mathbf{R}^\top$, $\mathbf{K}_{\setminus i}, \alpha_{\setminus i}$ are the matrix and vector obtained with the corresponding entry of the input vector removed, and $L_p(\boldsymbol{\alpha}_{\setminus i})$ is the log-likeihod for the corresponding model. It is known [23] that $L_p(\boldsymbol{\alpha})$ has a unique maximum w.r.t. parameter α_i, which is either finite and equal to $s_i^2/(q_i^2 - s_i)$ if $q_i^2 > s_i$ or infinite otherwise. This forms the basis for our forward selection process that starts with one input dimension and incrementally adds one more dimension, as long as the marginal likelihood is increased or a desired sparsity level is reached. In each step we maximize across remaining dimensions and hyperparameters α_j in order to add a new input index i to the active set S, with: $i = \arg\max_{\{j\notin S\}} l(\alpha_j)$. Hyperparameters are re-estimated, hence not only new dimensions are added to the active set S, but ones already present are removed, if their values follow $q_i^2 \leq s_i$. The procedure stops when there is no increase in the marginal likelihood, or a given level of sparsity is reached. Usually, only a small fraction of the input dimensions is selected, which makes the computational cost of learning each expert significantly lower than $\mathcal{O}(ND^2 + D^3)$.

2.3. Training the Gates

The log-likelihood component that corresponds to the gates decomposes as ($\boldsymbol{\lambda}$ is the $D \times M$-dimensional vector of all gate parameters $\boldsymbol{\lambda}_i$):

$$L_g(\boldsymbol{\lambda}) = \sum_{i=1}^{N}\sum_{j=1}^{M} h_j^{(i)} \log g_j^{(i)} = \tag{16}$$

$$= \sum_{i=1}^{N}\sum_{j=1}^{M}\{h_j^{(i)}\boldsymbol{\lambda}_j^\top\mathbf{r}_i - \log\sum_{j=1}^{M}\exp(\boldsymbol{\lambda}_j^\top\mathbf{r}_i)\} \tag{17}$$

For efficiency, we use bound optimization [13, 12] and maximize a surrogate function $\mathcal{F}$ with $\boldsymbol{\lambda}^{(t+1)} \leftarrow \arg\max_{\boldsymbol{\lambda}} \mathcal{F}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(t)})$ (the upper parameter superscript indexes the iteration number in this case). This is guaranteed to monotonically increase the objective, provided that $L_g(\boldsymbol{\lambda}) - \mathcal{F}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(t)})$ reaches its minimum at $\boldsymbol{\lambda} = \boldsymbol{\lambda}^{(t)}$. A natural surrogate is the second-order Taylor expansion of the objective around $\boldsymbol{\lambda}^{(t)}$, with a bound $\mathbf{H}_b$ on its second derivative (Hessian) matrix $\mathbf{H}$, so that $\mathbf{H}(\boldsymbol{\lambda}) \succeq \mathbf{H}_b, \forall \boldsymbol{\lambda}$:

$$\mathcal{F}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(t)}) = \frac{1}{2}\boldsymbol{\lambda}^\top \mathbf{H}_b \boldsymbol{\lambda} + \boldsymbol{\lambda}^\top(\mathbf{g}(\boldsymbol{\lambda}^{(t)}) - \mathbf{H}_b \boldsymbol{\lambda}^{(t)}) \quad (18)$$

The gradient and Hessian of L_g can be computed analytically:

$$\mathbf{g}(\boldsymbol{\lambda}) = \sum_{i=1}^{N} (\mathbf{U}_i - \mathbf{v}_i(\boldsymbol{\lambda})) \otimes \mathbf{r}_i \quad (19)$$

with $\mathbf{U}_i = [h_1^{(i)}, \ldots, h_M^{(i)}]^\top$, $\otimes$ the Kronecker product, and $\mathbf{v}_i(\boldsymbol{\lambda}) = [g_1(\mathbf{r}_i), \ldots, g_M(\mathbf{r}_i)]^\top$. The Hessian of L_g is:

$$\mathbf{H}(\boldsymbol{\lambda}) = -\sum_{i=1}^{N} (\mathbf{V}_i(\boldsymbol{\lambda}) - \mathbf{v}_i(\boldsymbol{\lambda})\mathbf{v}_i(\boldsymbol{\lambda})^\top) \otimes (\mathbf{r}_i \mathbf{r}_i^\top) \quad (20)$$

where $\mathbf{V}_i(\boldsymbol{\lambda}) = \text{diag}[g_1(\mathbf{r}_i), \ldots, g_M(\mathbf{r}_i)]$ (the dimensionality of the Hessian is $D \times M$). The Hessian is lower bounded by a negative definite matrix which depends on the input, but *remarkably*, is independent of $\boldsymbol{\lambda}$ [4]:

$$\mathbf{H}(\boldsymbol{\lambda}) \succeq \mathbf{H}_b \equiv -\frac{1}{2}[\mathbf{I} - \frac{\mathbf{1}\mathbf{1}^\top}{M}] \otimes \sum_{i=1}^{N} \mathbf{r}_i \mathbf{r}_i^\top \quad (21)$$

where $\mathbf{1} = [1, 1, \ldots, 1]^\top$. The parameter update is based on the standard Newton step:

$$\boldsymbol{\lambda}^{(t+1)} \leftarrow \boldsymbol{\lambda}^{(t)} - \mathbf{H}_b^{-1} \mathbf{g}(\boldsymbol{\lambda}^{(t)}) \quad (22)$$

To fit the gates we use a forward greedy algorithm that

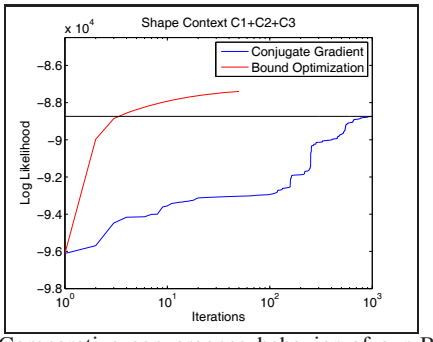

Figure 2. Comparative convergence behavior of our Bound Optimization (BO) and the Conjugate Gradient (CG) method when fitting the gates on a training set of 35,000 datapoints. Notice the rapid convergence of BO and that after *significantly more iterations* CG has not yet converged to the maximum of the log-likelihood.

combines gradient boosting and bound optimization. It selects the variables according to functional gradient boosting [7] and optimizes the resulting sub-problems using bound optimization, as described above. To compute the functional gradient, we rewrite the objective in terms of functions $F_j(\mathbf{r}^{(i)})$. This method is applicable to any differentiable log-likelihood:

$$L_g = \sum_{i=1}^{N} \sum_{j=1}^{M} \{h_j^{(i)} F_j(\mathbf{r}^{(i)}) - \log \sum_{j=1}^{M} \exp(F_j(\mathbf{r}^{(i)})\} \quad (23)$$

The functional gradient corresponding to one component of F_j is:

$$d_j^{(i)} = \frac{\partial L_g(F_j(\mathbf{r}^{(i)}))}{\partial F_j(\mathbf{r}^{(i)})} = \quad (24)$$

$$= h_j^{(i)} - \frac{\exp(F_j(\mathbf{r}^{(i)}))}{\sum_{j=1}^{M} \exp(F_j(\mathbf{r}^{(i)}))} \quad (25)$$

with the full gradient of the jth gate assembled as $\nabla \mathbf{f}_j = [d_j^{(1)}, \ldots, d_j^{(N)}]^\top$ – the steepest descent direction in function space. For feature selection, we choose the row vector $\mathbf{v}$ of $\mathbf{R}$ with weight index not already in the active set S, and most correlated (collinear) with the gradient [7]:

$$i = \arg\max_{k \notin S, j=1\ldots M} |\mathbf{v}_k^\top \nabla \mathbf{f}_j| \quad (26)$$

We initialize $\boldsymbol{\lambda} = \mathbf{0}$ and select the ith variable, incrementally, based on the gate parameter estimates at the previous round of selection. Once the ith variable is selected, we optimize (16) with respect to all pre-selected i variables using bound optimization. We use the solution of the previous iteration to quick-start the current optimization problem (this is convex but a good initialization spares iterations). The advantage of bound optimization in a greedy forward selection context is that we can efficiently update the Hessian bound using the Woodbury inversion identity. Thus, the cost of each iteration is $\mathcal{O}(cNMD)$ where c is a small constant, and the total cost of selecting the k variables is $\mathcal{O}(kNMD)$. When the specified number of variables is reached, we terminate. Unlike gradient boosting where the only current selected variable is optimized, we also perform back-fitting [24], *i.e.* optimize all selected variables in each round. To speed-up computation, it is possible to optimize the weights of the gating networks sequentially–fix the weights of other gating networks than the one currently optimized–the problem in (24). This requires the solution to a sequence of k-dimensional problems (usually $k << D$) and can be significantly cheaper than updating all gate parameters simultaneously, especially when denser (less sparse) models are desired. To sparsify the gating network, one can consider forward selection ideas based on maximizing the marginal likelihood, along the same lines as used for experts. However, the computational cost of this approach is high even for fast Bayesian approximations to multinomial classification. Differently from Bayesian regression, there is no analytical expression for the marginal likelihood, hence we

have to resort on Laplace approximation. But this only works around the maximized posterior point, so we have to recompute the most probable weight and the corresponding Hessian matrix after adding or deleting an input entry (or basis function). For large problems this operation is computationally prohibitive.

3. Experiments

We analyze the HumanEva dataset [19], which contains a number of sequences that include walking, jogging, throw catch, gestures, and boxing. Our tests are quite extensive, but we stress that our primary goal is to demonstrate algorithm's performance for large training sets and with high-dimensional input and output, rather than to comprehensively study a particular dataset. The part and structure of the dataset we use is given in table 1.[2]

Dataset	Action	S1	S2	S3	Total
Training set	Walking	612	437	490	1539
	Jog	251	396	440	1087
	Throw/Catch	0	560	0	560
	Gestures	405	400	568	1373
	Box	403	351	508	1262
	Total	1671	2144	2006	5821
Test set	Walking	585	433	443	1461
	Jog	362	393	396	1151
	Throw/Catch	0	545	0	545
	Gestures	390	493	528	1411
	Box	380	377	507	1264
	Total	1717	2241	1874	5832

Table 1. Number of training and test samples from HumanEva-1, for each motion category (due to format compatibility considerations, we use HumanEva's designated validation set as our test set). Models are trained for both separate viewpoints and for all viewpoints together, and for some of the larger models we will borrow samples from the test set for training (in this case we will *only* test on the remaining test samples! The reason we borrow test samples is to demonstrate the ability of *f*BME to build large models, for which we wouldn't otherwise have enough training / labeled data available). No matter what model we report on, the samples we test on are *never* used for validation or training.

We study the performance of three types of image descriptors including histograms of shape contexts sampled on human silhouettes (both internal and external contours), histograms of SIFT features, again sampled on the silhouette, and hierarchical multilevel, multi-scale hyperfeature encodings [11] that repeatedly accumulate / average template matches to prototypes (local histograms) across layers, instead of winner-takes-all MAX operations followed by template matching to prototypes. For all descriptors the image bounding box is obtained using the extents of the silhouettes computed using background subtraction based on non-parametric models [6]. The bounding box is automatically adjusted, to keep the human centered, by adding borders that maintain the 320x200 pixel aspect ratio.

For shape context features (HistoSC), edges are extracted from the silhouette image and 400 points are sampled on edges. The shape context descriptor at each image location is computed based on 15 angular bins and 8 radial bins. The SC at each of the 400 points per image are computed every 15th image in each training sequence and used to generate a codebook that consists of 300 clusters, learned using k-means (hence the descriptor size is 300). The SIFT histogram descriptor is obtained similarly with the exception that SIFT (as opposed to shape context) descriptors are sampled and computed, using 6x6 pixels per cell, 4x4 cells. The final dimensionality is 300, again given the codebook size resulting from clustering SIFT descriptors across subsampled images from the training set. Hyperfeatures are computed at 6 scales, (1.0000, 0.8333, 0.6667, 0.5000, 0.4167,0.2500) at 3 pyramid levels (level 0 has 6 scales, 1–4 scales, 2–2 scales) based on SIFT descriptor size 4x4 in each cell, 4x4 cells per block, 4 angular bins of gradient orientations (0–180), unsigned. The sampling grid is placed at every 16th pixel in the object bounding box. The next pyramid level is generated by combining neighboring scales and neighboring SIFT blocks, for a total of 9x3 SIFT blocks summed to next level. The number of cluster centers for levels (0, 1, 2) is (400, 200, 100) respectively, forming a descriptor of size 700.

We predict 3D joint centers and construct the skeleton using 'torsoDistal' as root joint. All poses are preprocessed by subtracting the root joint location from all the joint centers in every frame. The normalized pose contains 15 joint centers (each has X, Y and Z co-ordinates) to form an output dimension of 45. For prediction we use models that do not use temporal information, run independently, at each frame (the use of temporal priors is likely to improve performance, and we currently study this). The models we use are: Nearest Neighbor (NN), Ridge Regression (RR), BME and *f*BME (see our companion *Twin Gaussian Process prediction method, TGP* [3], for additional results). We train a family of models based on 5828, 8734, 11653, 17470, 26209, 34966 and 106473 samples that include 3 subjects, with different body proportions (limb lengths) and 3 viewpoints, C1, C2 and C3 (see tables 3 and 5). We train models with 10 experts for the single view data and 15 experts for those jointly trained on all viewpoints. (These are treated as additional monocular views–there is no voting among multiple predictions from different viewpoints of the same 3D frame.) The performance is in the range of 10-50 mm per joint position (see tables 2–5). Running times are re-

[2]Notice the difference in experimental settings, subsets of the database, and types of error function used when comparing different reports on HumanEva. *E.g*. training/testing on the same subject or motion can lead to lower errors although the averages over all motions are not significantly different (contrast tables 2 and 3). For time-series, training/testing on interleaved or subsampled frames of a sequence rather than on compact, separate blocks can further decrease error, but is methodologically infeasible.

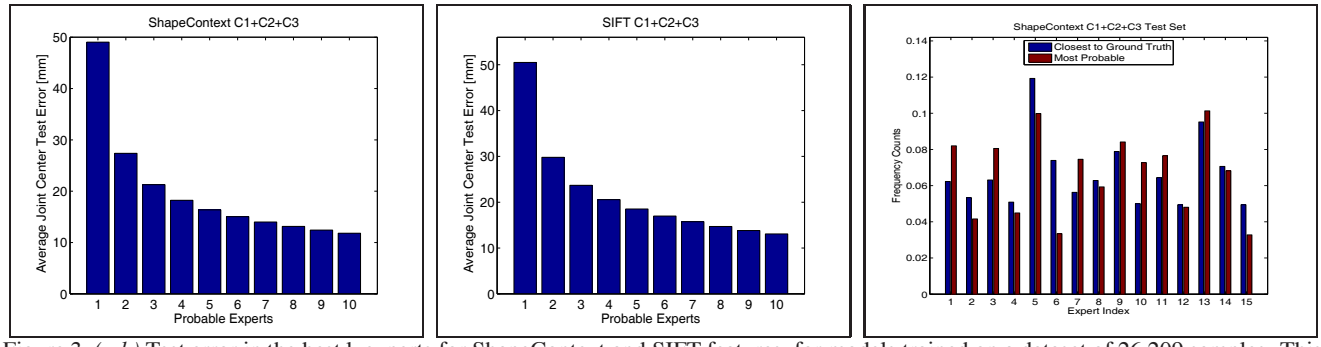

Figure 3. *(a,b)* Test error in the best k experts for ShapeContext and SIFT features, for models trained on a dataset of 26,209 samples. This corresponds to *f*BME-B10 in table 3, but notice that this is an *ideal* prediction that offers general intuition on the quality of experts, but cannot be computed when the output is unknown (instead, realistically consider *f*BME-M5, or the less accurate *f*BME-B1). *(c)* Frequency counts illustrate the accuracy of ranking produced by the gates on the test set, showing how many times each expert is closest to the ground truth vs. how many times its corresponding gate function predicts it. There is a degree of error in the rankings of some experts, *e.g.* the 5th. This occurs not only in testing but also in training, suggesting that split and merge methods may be useful to improve performance.

ported for a PC with 3GhZ processor and 8Gb of RAM. In fig. 3a,b) we show the decay in the test error of the best-k experts for different feature types and large training sets that integrate data from all viewpoints and subjects. The decay is fast, under 15mm when using more than the best 5 experts, suggesting that the method can be efficiently used either standalone (*f*BME-B1 or *f*BME-M5), or as a proposal mechanism for initializing more intensive search and feedback-based algorithms (*f*BME-B5 or *f*BME-B10). The ranking accuracy provided by the gates on the test set is shown in fig. 3c). This is not perfect but the distribution is consistent to the one we see in training, suggesting that merge-and-split training methods can potentially be used to obtain better fits. The advantage of Bound Optimization (BO) as opposed to alternative optimization methods like *e.g.* Conjugate Gradient (CG) is empirically demonstrated in fig. 2, where, within less than 5 iterations BO achieves a likelihood level that is not reached by CG even after 1000 steps! The gate optimization is a difficult, coupled high-dimensional problem, which, as the plot shows, is very ill-conditioned. Hence, first order methods have slim chances to operate efficiently. Comparisons between *f*BME and a previously proposed conditional Bayesian method [22] is given in table 4. This shows that *f*BME is accurate and more than one order of magnitude faster to train.

We conclude with fig. 4, where we show accurate qualitative results obtained by running *f*BME on the HumanEva-1 test set. The top row shows original images, whereas the bottom row shows the 3D reconstruction given by *f*BME-M5 (best-5 weighted experts), rendered from a variety of synthetic viewpoints (not only the one corresponding to the camera from where the image was captured), for diversity.

4. Conclusions

We have presented a conditional Expectation Maximization algorithm that combines several innovations based on forward feature selection and bound optimization in order to make large scale training of probabilistic sparse conditional Bayesian Mixture of Experts models practical. The proposed algorithm, referred to as *f*BME can accurately handle datasets of 100,000 training datapoints and up (one order of magnitude larger than existing methods) in time that is more than one order of magnitude faster than existing algorithms. Besides human pose prediction, we hope that this method will be useful to a larger community including researchers studying 3D reconstruction and 3D object recognition.

Future Work: We plan to explore alternative predictive models in tandem with dimensionality reduction. We also work on better initialization algorithms, in particular expert

HistoSC	Test error (mm)		Training time (s)	
	BME [22]	*f*BME	BME [22]	*f*BME
C1	21.1	21.2	33223.5	844.4
C2	19.2	19.0	33135.7	837.5
C3	17.7	17.8	33177.9	828.3

Table 4. Analysis of three algorithms applied to an arm model with 5 output dimensions and shape context histograms-based input descriptors. BME [22] uses backward elimination for feature selection and Laplace approximation for gate fitting. *f*BME uses forward feature selection and gate fitting based on bound optimization. The number of training samples is 5828 for C1, and 5821 for C2 and C3.

	Training time (sec)	Error (mm)
*f*BME	9857.3	19.013

Table 5. Training using a 106,473 sample dataset (500 input dimensions, one output dimension), that combines samples from multiple views in HumanEva (real images), together with synthetically rendered image data of a graphics model animated with motion capture. The experiment is designed to show how *f*BME scales–notice that HumanEva provides only around 36,000 samples of motion capture with corresponding images.

		Subject 1					Subject 2					Subject 3				
		NN	RR	*f*BME-B1	*f*BME-B5	*f*BME-M5	NN	RR	*f*BME-B1	*f*BME-B5	*f*BME-M5	NN	RR	*f*BME-M1	*f*BME-B5	*f*BME-M5
Train/Test /per Subject	Walking	29.0	39.4	27.2	12.1	26.2	16.5	31.0	16.6	9.4	16.6	50.7	49.5	45.3	15.3	43.2
	Jog	67.9	62.2	39.6	16.1	37.0	38.2	35.1	26.6	13.7	25.5	24.3	30.8	27.6	12.3	26.8
	Gestures	7.3	19.0	6.4	3.8	6.2	50.5	49.5	46.2	16.3	41.8	14.0	21.2	12.7	6.4	12.6
	Box	38.8	39.4	29.6	13.4	28.4	49.5	49.6	41.7	18.8	40.0	37.0	40.4	32.3	14.7	31.3
	Throw/Catch	/	/	/	/	/	64.4	50.6	45.9	17.6	42.5	/	/	/	/	/
Train/Test /per Motion	Walking	28.9	28.7	23.2	12.2	23.0	15.5	21.4	13.7	9.0	13.7	50.5	43.9	41.4	19.2	40.3
	Jog	43.7	37.2	35.5	17.1	34.6	34.4	24.5	25.6	14.0	24.2	24.1	24.3	22.0	12.3	21.4
	Gestures	7.3	5.8	5.9	2.6	5.6	61.9	53.3	55.1	31.6	54.9	13.7	12.5	12.6	7.6	12.4
	Box	28.3	27.3	25.4	10.7	24.4	48.0	38.9	37.7	17.3	37.5	36.8	36.7	30.2	13.5	29.3
	Throw/Catch	/	/	/	/	/	60.1	58.2	47.4	22.4	46.2	/	/	/	/	/

Table 2. *f*BME trained / tested per-subject and per-motion, using models based on 5 experts, and shape context feature extracted from video camera C3 (error in mm/3D joint). '/' indicates that values are not available (*e.g*. throw and catch gestures [20, 15] for some of the subjects). Models based on 10 experts have smaller best error, *e.g*. for the first row: *f*BME-B10: 8.9 (S1), 7.6 (S2), 11.3 (S3).

Features/ View	Train Set	Test Set	Train error(mm)		Test Errors (mean absolute error in mm)							Training time(s)		Test time (s)		
			RR	*f*BME-B1	NN	RR	*f*BME-B1	*f*BME-B5	*f*BME-B10	*f*BME-Bar1	*f*BME-M5	RR	*f*BME	NN	RR	*f*BME-M5
HistoSC/C1	5828	5832	32.6	12.3	41.6	47.3	37.6	15.1	11.2	73.2	35.0	0.2	7445.8	359.7	0.05	45.4
	8741	2919	34.8	14.7	39.4	45.7	35.2	14.7	11.1	68.4	32.9	0.3	10633.8	263.3	0.03	22.8
	11660	0	35.7	16.2	/	/	/	/	/	/	/	0.4	13230.8	/	/	/
HistoSC/C2	5821	5832	29.6	10.5	38.2	43.2	31.4	13.6	10.1	77.2	29.7	0.2	7385.7	353.6	0.05	46.0
	8734	2919	31.3	12.4	36.6	42.3	31.0	13.4	10.0	67.4	29.0	0.3	10485.2	267.7	0.03	22.8
	11653	0	32.4	14.0	/	/	/	/	/	/	/	0.4	13606.9	/	/	/
HistoSC/C3	5821	5832	30.5	10.5	36.3	42.8	30.6	13.5	10.0	66.1	29.1	0.2	7251.1	352.1	0.05	45.8
	8734	2919	32.0	12.3	39.9	42.4	31.1	13.4	10.1	66.4	29.3	0.3	10086.8	265.3	0.03	22.8
	11653	0	32.9	13.6	/	/	/	/	/	/	/	0.4	13502.9	/	/	/
HistoSC/ C1+C2+C3	17470	17496	48.0	23.5	54.8	60.8	49.9	16.1	11.5	97.5	47.5	1.5	63915.9	5189.1	0.2	146.5
	26209	8757	49.4	26.8	54.9	59.2	47.0	16.0	11.4	99.3	45.0	2.1	104222.2	3884.8	0.1	72.7
	34966	0	50.0	29.5	/	/	/	/	/	/	/	2.8	142391.8	/	/	/
HistoSIFT/C1	5828	5832	40.7	20.3	56.2	51.1	43.2	17.1	12.6	79.5	40.4	0.2	7708.2	360.3	0.05	45.7
	8741	2919	42.1	22.7	56.7	50.3	42.8	17.2	12.7	64.1	39.7	0.3	9893.0	270.1	0.03	22.9
	11660	0	42.8	24.6	/	/	/	/	/	/	/	0.4	12431.6	/	/	/
HistoSIFT/C2	5821	5832	37.3	15.3	46.9	48.8	37.8	15.8	11.8	80.0	35.5	0.2	7713.7	357.5	0.2	45.6
	8734	2919	39.4	18.0	48.6	47.1	36.1	15.8	11.8	89.2	34.1	0.3	10001.1	267.8	0.1	22.8
	11653	0	39.9	19.7	/	/	/	/	/	/	/	0.4	13174.1	/	/	/
HistoSIFT/C3	5821	5832	38.0	16.1	50.2	49.1	38.6	16.1	12.0	78.6	36.3	0.2	7103.9	389.8	0.2	45.7
	8734	2919	39.6	18.3	49.7	48.6	37.0	15.5	11.7	86.2	35.0	0.3	10298.9	265.6	0.1	22.9
	11653	0	40.4	20.1	/	/	/	/	/	/	/	0.4	12680.4	/	/	/
HistoSIFT/ C1+C2+C3	17470	17496	54.0	32.0	61.0	61.9	50.1	18.9	13.3	96.9	48.6	1.5	57144.1	5296.4	0.2	146.4
	26209	8757	54.9	34.3	60.9	59.7	48.5	18.5	13.1	88.6	47.3	2.1	81230.9	3949.6	0.1	73.3
	34966	0	55.2	35.5	/	/	/	/	/	/	/	2.8	126992.5	/	/	/
Hyper/C1	5828	5832	22.7	10.0	59.0	49.3	43.9	17.9	12.6	72.9	42.6	1.1	24497.1	953.1	0.1	53.2
	8741	2919	25.8	12.0	56.4	46.7	41.5	17.2	12.6	70.4	42.6	1.5	28494.6	625.6	0.1	26.9
	11660	0	27.9	13.4	/	/	/	/	/	/	/	1.8	35187.8	/	/	/
Hyper/C2	5821	5832	23.1	9.8	49.2	47.4	40.5	16.2	11.7	73.3	39.2	1.1	24730.8	939.3	0.1	52.9
	8734	2919	26.1	11.5	48.0	45.5	38.7	16.1	11.9	72.5	38.0	1.5	29092.9	597.9	0.1	26.3
	11653	0	28.4	13.1	/	/	/	/	/	/	/	1.8	35360.9	/	/	/
Hyper/C3	5821	5832	23.9	9.9	52.3	48.0	40.1	16.5	11.8	71.2	38.7	1.1	25632.6	922.3	0.1	55.0
	8734	2919	26.8	11.8	50.5	46.3	39.4	15.9	11.6	73.5	38.3	1.5	30126.7	603.7	0.1	27.7
	11653	0	28.8	13.4	/	/	/	/	/	/	/	1.8	36146.2	/	/	/
Hyper/ C1+C2+C3	17470	17496	39.8	15.7	63.6	57.7	47.1	18.9	13.1	94.0	43.7	5.8	149091.9	10780.0	0.5	163.1
	26209	8757	41.9	17.8	64.0	54.7	46.9	18.4	12.9	96.1	43.2	8.1	217268.7	8041.6	0.3	83.4
	34966	0	44.3	19.6	/	/	/	/	/	/	/	10.7	288256.6	/	/	/

Table 3. Evaluation of *f*BME on HumanEva-1 (models based on 10 experts). In the table, '/' show that values are not available – these entries correspond to models trained on all data, in order to show how *f*BME scales. *f*BME-B1 is the error in the most probable expert, *f*BME-B5 is the error in the best 5 experts (ideal prediction, assuming the output is known and the expert closest to the ground truth is selected), *f*BME-Bar1 is the error of the expert with lowest predictive uncertainty (errorbar), hence ranking information from the gates is not used. In this (latter) case, the performance degrades which is expectable because the model (the equivalent of a mixture of predictors with fixed proportions) lacks the probabilistically consistent ranking necessary for conditional prediction. *f*BME-M5 gives the error with respect to the weighted prediction of the 5 most probable experts (this always decreases as more experts are added but typically saturates beyond 5). RR and NN are models based on Ridge Regression and Nearest Neighbor, respectively. Training models with 20 experts decreases the error somewhat, but not substantially, *e.g*. HistoSC/C3 5821 (*f*BME-B10:9.0, *f*BME-B20: 6.5, *f*BME-M5: 27.1).

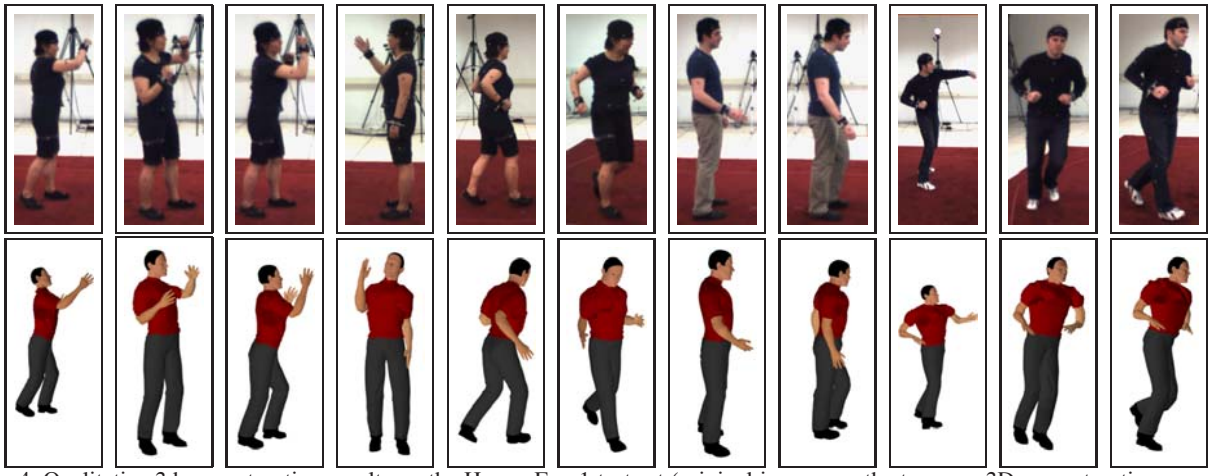

Figure 4. Qualitative 3d reconstruction results on the HumanEva-1 test set (original images on the top row, 3D reconstructions seen from different viewpoints on the second row).

fitting methods based on split and merge heuristics.

Acknowledgements: This work was supported, in part, by the NSF and the EC, under awards 0535140 and MCEXT-025481.

References

[1] A. Agarwal and B. Triggs. 3d human pose from silhouettes by Relevance Vector Regression. In *CVPR*, 2004.

[2] C. Bishop and M. Svensen. Bayesian mixtures of experts. In *UAI*, 2003.

[3] L. Bo and C. Sminchisescu. Twin Gaussian Processes for Structured Prediction. *Snowbird Learning*, April 2008.

[4] D. Böhning. Multinomial logistic regression algorithm. *Annals of Inst. of Stat. Math.*, 44:197–200, 2001.

[5] D. Edwards and S. Lauritzen. The TM algorithm for maximising a conditional likelihood function. *Biometrika*, 88(4):961–972, 2001.

[6] A. Elgammal, R. Duraiswami, D. Harwood, and L. Davis. Foreground and background modeling using non-parametric kernel density estimation for visual surveillance. *Proc.IEEE*, 2002.

[7] J. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001.

[8] I. Guyon. An Introduction to Variable and Feature Selection. *JMLR*, 3:1157–1182, 2003.

[9] T. Jebara and A. Pentland. On reversing Jensen's inequality. In *NIPS*, 2000.

[10] M. Jordan and R. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, (6):181–214, 1994.

[11] A. Kanaujia, C. Sminchisescu, and D. Metaxas. Semi-Supervised Hierarchical Models for 3D Human Pose Reconstruction. In *CVPR*, 2006.

[12] B. Krishnapuram, L. Carin, M. T. Figueiredo, and A. J. Hartemink. Sparse multinomial logistic regression: Fast algorithms and generalization bounds. *PAMI*, 27(6):957–968, 2005.

[13] K. Lange, D. Hunter, and I. Yang. Optimization transfer using surrogate objective functions. *J. Computational and Graphical Statistics*, 9:1–59, 2001.

[14] R. Navaratnam, A. Fitzgibbon, and R. Cipolla. The Joint Manifold Model for Semi-supervised Multi-valued Regression. In *ICCV*, 2007.

[15] R. Poppe. Evaluating example-based human pose estimation: Experiments on HumanEva sets . In *HumanEva Workshop CVPR*, 2007.

[16] R. Rosales and S. Sclaroff. Learning Body Pose Via Specialized Maps. In *NIPS*, 2002.

[17] M. Seeger, C. Williams, and N. Lawrence. Fast forward selection to speed up sparse gaussian process regression. In *AISTATS*, 2003.

[18] G. Shakhnarovich, P. Viola, and T. Darrell. Fast Pose Estimation with Parameter Sensitive Hashing. In *ICCV*, 2003.

[19] L. Sigal and M. Black. HumanEva: Synchronized Video and Motion Capture Dataset for Evaluation of Articulated Human Motion. Technical Report CS-06-08, Brown University, 2006.

[20] L. Sigal and M. Black. Predicting 3d people from 2d pictures. In *IV Conference on Articulated Motion and Deformable Objects, AMDO*, 2006.

[21] C. Sminchisescu, A. Kanaujia, Z. Li, and D. Metaxas. Discriminative Density Propagation for 3D Human Motion Estimation. In *CVPR*, volume 1, pages 390–397, 2005.

[22] C. Sminchisescu, A. Kanaujia, and D. Metaxas. BM^3E: Discriminative Density Propagation for Visual Tracking. *PAMI*, 2007.

[23] M. Tipping and A. Faul. Fast marginal likelihood maximisation for sparse bayesian models. In *AISTATS*, 2003.

[24] P. Vincent and Y. Bengio. Kernel matching pursuit. *Machine Learning*, 48, 2002.

[25] J. Zhu and H. Trevor. Kernel logistic regression and the import vector machine. *Journal of Computational and Graphical Statistics*, 14:185–205, 2005.