Variational Bayesian Learning of Probabilistic Discriminative Models With Latent Softmax Variables

Nisar Ahmed, *Student Member, IEEE*, and Mark Campbell, *Member, IEEE*

***Abstract*—This paper presents new variational Bayes (VB) approximations for learning probabilistic discriminative models with latent softmax variables, such as subclass-based multimodal softmax and mixture of experts models. The VB approximations derived here lead to closed-form approximate parameter posteriors and suitable metrics for model selection. Unlike other Bayesian methods for this challenging class of models, the proposed VB methods require neither restrictive structural assumptions nor sampling approximations to cope with the problematic softmax function. As such, the proposed VB methods are also easily extendable to more complex softmax-based hierarchical discriminative models and regression models (for continuous outputs). The proposed VB methods are evaluated on benchmark classification data and a decision modeling application, demonstrating good results.**

***Index Terms*—Bayesian networks, latent variables, mixture of experts, model selection, subclasses, variational Bayes (VB).**

I. Introduction

A PROBABILISTIC DISCRIMINATIVE MODEL (PDM) is a direct estimate of the conditional probability distribution $P(D\,|\,X)$ for a K-valued random variable D that depends on a continuous vector $X \in \mathbb{R}^{M+1}$. PDMs find wide use in complex stochastic hybrid system models [27], [22], [26], [25], and related applications such as statistical pattern classification [5], [6], and signal detection [30]. Multiclass PDMs (i.e., $K \geq 2$) are often based on the softmax model (also known as the multinomial logistic function), which produces linear class boundaries with respect to X in its most basic form. To approximate more complex distributions with nonlinear class boundaries, the softmax model can be enhanced either via nonlinear basis transformations or by introducing latent switching variables. The former approach yields nonlinear softmax PDMs in which class boundaries are implicitly defined in X (e.g., [23] and [33]). The latter approach yields latent softmax variable PDMs, in which boundaries can be explicitly defined in X via mixtures of basic softmax distributions. Examples of this latter model class include the well-known mixture of experts (ME) model [20] and the multimodal softmax (MMS) model [2], which probabilistically generalizes determinstic "subclass" classifiers (e.g., [16] and [38]).

Manuscript received May 23, 2010; revised October 19, 2010 and March 31, 2011; accepted April 03, 2011. Date of publication April 21, 2011; date of current version June 15, 2011. This work was supported by a National Science Foundation Graduate Research Fellowship and by an Air Force Office of Scientific Research Award FA9550-09-1-0462.

The authors are with the Autonomous Systems Laboratory, Cornell University, Ithaca, NY 14853 USA (e-mail: nra6@cornell.edu; mc288@cornell.edu.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2011.2144587

In addition to being useful classifiers, latent softmax variable PDMs are particularly useful in hybrid Bayesian networks, which can model mixed continuous and discrete probabilistic relationships for a wide variety of applications [24], [25]. Hybrid Bayesian nets generally feature unobserved random variables in the parent vector X that must be estimated or marginalized out via Bayesian inference at run time. As discussed in [1] and [27], PDMs defined via basic softmax models advantageously permit closed-form variational Bayesian (VB) inference approximations, which can be much more efficient than either the Monte Carlo or discretization approximations required with nonlinear softmax models. Hence, there is ample motivation to develop reliable and efficient learning methods for latent softmax variable PDMs.

This paper addresses the related challenges of parameter estimation and structure estimation for ME and MMS discriminative models. Although previous work examined these issues for ME models via fully Bayesian learning approximations, the methods studied thus far are either: i) slow and inefficient due to the use of sequential Monte Carlo methods [29]; or ii) limited to only learning ME models that do *not* use the softmax function, which is analytically nonintegrable [36], [34], [7]. Compared to softmax-based models, softmax-free ME models are often far more structurally complex due to their limited gating/output abilities, which makes them more expensive and difficult to learn as a result. In contrast, Bayesian methods have not yet been applied to subclass models such as MMS, due to their recent development.

This paper derives new VB learning approximations for both MMS and ME models which (unlike previously proposed Bayesian learning methods) cope analytically with softmax functions and thus do *not* require any restrictive structural/parametric constraints or sampling approximations. The proposed variational Bayes methods yield computationally efficient closed-form approximate posteriors for Bayesian parameter estimation and suitable metrics for Bayesian model selection, which are evaluated here experimentally on benchmark and application data. In addition, the proposed VB methods can be easily generalized to more complex hierarchical ME discriminative and regression models.

Section II provides background on Bayesian MMS and ME model learning and variational Bayes methods. Section III derives VB learning for MMS and describes a compressive search procedure for tackling the difficult problem of MMS model selection. Section IV derives VB learning for ME discriminative models. In Section V, the proposed VB approximations are evaluated on benchmark classification data and decision data from a human-robot interaction application.

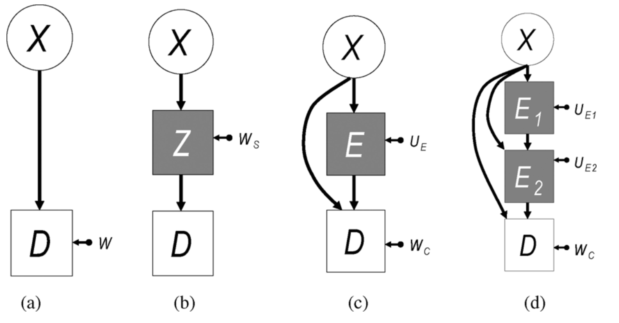


Fig. 1. Probabilistic graph structures for softmax-based models (square nodes are discrete, round nodes are continuous, shaded nodes are hidden, point nodes are deterministic). (a) Basic softmax. (b) MMS. (c) ME. (d) 2-level HME with 2 hidden gating nodes. The model weights shown here as deterministic for ease of illustration.

II. Preliminaries

A. Model Definitions

Let D take class label realizations $d \in \{1,\ldots,K\}$ and let $X \in \mathbb{R}^{M+1}$. The conditional probability of $D=d$ given $X=x$ under the "basic" softmax (i.e., multinomial logistic) model is

$$P(D=d\,|\,X=x;W) = \frac{e^{w_d^T x}}{\sum_{h=1}^{K} e^{w_h^T x}} \equiv f(d;x,W) \quad (1)$$

where $x = [x_1,\ldots,x_M,1]^T$, w_d is an $(M+1)$-dimensional weight parameter for the dth discrete class, and $W = \{w_1,\ldots,w_K\}$ is the set of all class weights.[1] The log-odds ratio between any two classes is easily shown to be an $(M-1)$-dimensional hyperplane in X, so the 'probabilistic boundaries' between classes are linear for (1). Fig. 1(a) shows the probabilistic graphical model for (1), for which two latent variables generalizations are considered.

1) MMS Model: The MMS model assumes each class d has s_d mutually exclusive and exhaustive *subclasses*, where $s_d \in \mathbb{N}^+$ and $\sum_{d=1}^{K} s_d = S$ for $S \geq K$. Let latent variable Z take values in $\Psi = \{1,\ldots,S\}$ given X, with $P(Z\,|\,X)$ given by (1), and let $\sigma(d) \subset \Psi$ be the set of s_d subclasses for d. Furthermore, define $P(D=d\,|\,Z=i) = I(i \in \sigma(d))$ as the indicator function, which is 1 if $i \in \sigma(d)$ and 0 otherwise. Assuming $P(D,Z\,|\,X,W_S) = P(D\,|\,Z)P(Z\,|\,X,W_S)$, the total probability theorem then gives the MMS class probability as the sum over subclass probabilities

$$\begin{aligned} P(D=d\,|\,x;W_S) &= \sum_{i=1}^{S} P(D=d\,|\,Z=i)\cdot P(Z=i\,|\,x,W_S) \\ &= \sum_{i=1}^{S} I(s \in \sigma(d))\cdot f(i;x,W_S) \quad (2)\end{aligned}$$

where W_S is the set of S softmax weights for $P(Z\,|\,X,W_S)$. The MMS graph model is shown in Fig. 1(b). Equation (2) produces piecewise linear log-odds boundaries for the original K classes with respect to X. This also yields probabilistic "soft subclass" labels for known class data; from Bayes' rule, the posterior MMS subclass responsibility for the point $y_n = (D = d_n, X = x_n)$ is

$$P(Z=i\,|\,y_n;W_s) = \frac{T_{\text{in}} e^{w_i^T x_n}}{\sum_{j=1}^{S} T_{jn} e^{w_j^T x_n}} \equiv g(i;x_n,W_S) \quad (3)$$

where $T_{\text{in}} \equiv I(i \in \sigma(d_n))$. The MMS model structure is given by the *subclass configuration* $[s_1,\ldots,s_K]$ (the number of subclasses per class label). See [2] for further details on MMS.

2) ME/HME Models: In the mixture of experts (ME) model, the outputs of G softmax "expert" models over the K classes are mixed together by a latent gating variable E, where $P(E\,|\,X)$ also follows (1). For the ME graph model shown in Fig. 1(c), the class probability is

$$\begin{aligned} P(D=d\,|\,X;U_e,W_c) &= \sum_{e=1}^{G} P(E=e\,|\,x)\cdot P(D=d\,|\,E=e,x) \\ &= \sum_{e=1}^{G} f(e;x,U_e)\cdot f(d;x,W_c) \quad (4)\end{aligned}$$

where $U_e = \{u_1,\ldots,u_G\}$ are the G gating weights and $W_c = \{w_{11},\ldots,w_{1K},\ldots,w_{G1},\ldots,w_{GK}\}$ are the GK expert weights over all K classes. Equation (4) yields "soft piecewise linear" class boundaries in X, as a result of the blended expert model outputs. More complex hierarchical mixture of experts (HME) models [20] are obtained by connecting additional hidden gating nodes (dependent on X) to lower level gating and expert nodes, as Fig. 1(d) shows. Clearly, HME models can have many possible structures for enhancing the nonlinear boundary approximation abilities of (4) [37]. See [20] for further details on ME and HME models; for ease of explanation, this paper refers primarily to ME models, without loss of generality.

B. ML/MAP Learning

The MMS learning problem is to estimate the subclass configuration $[s_1,\ldots,s_K]$ and weights W_S in (2) from a given set of labeled training data. Similarly, the ME learning problem is to estimate the number of experts G and weights U_e and W_c in (4). Note that unique maximum likelihood (ML) or MAP estimates for W in (1) can be found efficiently from training data using Newton-type optimization, since the softmax likelihood function is concave [28], [6]. This does not depend on $p(X)$, which can be any pdf.

While this latter property also holds for MMS and ME, the likelihood functions for fixed $[s_1,\ldots,s_K]$ or G are unfortunately nonconcave due to the presence of hidden variables, so that weight estimates will always convege to saddle points or local maxima. The MMS log-likelihood Hessian can be tractably computed for efficient direct optimization with fixed $[s_1,\ldots,s_K]$ [2]. Direct optimization can also be applied to ME for fixed G [20], although the Hessian is typically quite expensive to compute. Parameters can also be obtained for both models via expectation maximization (EM) [20], [13],

[1]The last element of x is defined as 1 to introduce a bias term via w_d's last element, although the notation "$X = x$" is retained for convenience.

[2], which is useful for large parameter spaces. Structural estimation of $[s_1, \ldots, s_K]$ or G can be performed via asymptotic ML/MAP-based model selection metrics such as the Aikake Information Criterion or the Bayes Information Criteron [6], [14]. However, these metrics can be unreliable with insufficiently large data sets since they tend to over/underpenalize parametric complexity [4].

C. Bayesian Model Learning

Selection of an appropriate $[s_1, \ldots, s_K]$ for MMS or G for ME is quite important, as there is a danger of overfitting (underfitting) if too complex (too simple) a model is used. It is also important to consider that ML/MAP parameter estimates are often inappropriate for small training sets, as overfitting can still occur even when the best structures are known. In such cases, it is often desirable to account for parameter uncertainty in these models to do fully probabilistic inference, e.g., for making predictions/forecasts with new data [6] or for learning within larger models such as Bayesian networks [4].

These issues can be addressed by fully Bayesian methods, in which both the discrete model structures and associated model parameters are treated as unknown random variables with prior pdfs in the presence of training data $\mathbf{Y} = \{y_1, \ldots, y_N\} = \{(x_1, d_1), \ldots, (x_N, d_N)\}$. If Ω is the model structure and Θ is the set of unknown model parameters (and prior hyperparameters) under Ω, Bayes' rule gives the posterior distribution

$$p(\Theta \mid \Omega, \mathbf{Y}) = \frac{p(\mathbf{Y}, \Theta \mid \Omega)}{\int p(\mathbf{Y}, \Theta \mid \Omega) d\Theta} = \frac{p(\mathbf{Y} \mid \Omega, \Theta) p(\Theta \mid \Omega)}{p(\mathbf{Y} \mid \Omega)} \tag{5}$$

where $p(\Theta \mid \Omega)$ is the *prior* over Θ, $p(\mathbf{Y} \mid \Omega, \Theta)$ is the *data likelihood* and $p(\mathbf{Y} \mid \Omega)$ is the *model likelihood.* For fixed Ω, (5) can be used to compute a Bayesian point estimate of Θ (e.g., MMSE or MAP) or form a fully Bayesian predictive output density for new inputs. To select Ω from a set of candidate models $\mathcal{M}$, the Bayesian posterior for Ω is considered

$$p(\Omega \mid \mathbf{Y}) = \frac{p(\mathbf{Y} \mid \Omega) p(\Omega)}{\sum_{\Omega \in \mathcal{M}} p(\mathbf{Y} \mid \Omega) p(\Omega)} = \frac{p(\mathbf{Y} \mid \Omega) p(\Omega)}{p(\mathbf{Y})} \tag{6}$$

where $p(\mathbf{Y})$ is a constant independent of Ω. In practice, it is common for $\mathcal{M}$ to be finite (e.g., by restricting model complexity to some upper bound) and for $p(\Omega)$ to be uniform over $\mathcal{M}$ (i.e., to assume all $\Omega \in \mathcal{M}$ are equally likely). As such, $p(\Omega \mid \mathbf{Y}) \propto p(\mathbf{Y} \mid \Omega)$ is the consistent inference for Bayesian model comparison, and thus Bayesian model selection can be performed by choosing the most probable Ω according to $p(\mathbf{Y} \mid \Omega)$ alone [4], [6], [7], [12], [32], [34].

As in many Bayesian learning problems, the main challenge here is marginalization over Θ in $p(\mathbf{Y}, \Theta \mid \Omega)$ to find $p(\mathbf{Y} \mid \Omega)$ in (5) and (6): since $\mathbf{Y}$ induces complex depedencies in Θ, the required integrals/sums are intractable to compute exactly for even modest N. Furthermore, softmax terms in $p(\mathbf{Y} \mid \Theta, \Omega)$ cannot be integrated in closed form with respect to $p(\Theta \mid \Omega)$ for MMS or ME models. Therefore, approximations to (5) and $p(\mathbf{Y} \mid \Omega)$ are needed, e.g., via Markov Chain Monte Carlo (MCMC) sampling or the Laplace approximation, which have both been applied to ME in [29] and [36], respectively. While MCMC methods are quite flexible and highly accurate with large sample sizes, they can be computationally inefficient and thus very slow to converge to reliable estimates. The Laplace method gives much faster posterior approximations. However, due to its crude nature, it cannot handle certain types of hyperparameters (e.g., strictly positive scale parameters) and is quite inaccurate with asymmetric posteriors.

D. Variational Bayes Approximations

Alternatively, variational Bayes (VB) approximations can be used. In the present context, VB seeks to minimize the Kullback-Leibler divergence (KLD) functional

$$\mathrm{KL}(q \,\|\, p) = -\int q(\Theta \mid \mathbf{Y}, \Omega) \log \left\{ \frac{p(\Theta \mid \mathbf{Y}, \Omega)}{q(\Theta \mid \mathbf{Y}, \Omega)} \right\} d\Theta \tag{7}$$

between an integrable and closed-form *variational posterior* parameter distribution $q(\Theta \mid \mathbf{Y}, \Omega)$ and the true posterior parameter distribution $p(\Theta \mid \mathbf{Y}, \Omega)$, for $\Omega \in \mathcal{M}$. Since (7) can be viewed as a 'distance measure' between two distributions, VB seeks a principled probabilistic approximation q to the true intractable posterior p, where the conditional independence properties of q are chosen *a priori* to ensure tractability. Equation (7) is minimized in practice by maximizing a lower bound $\mathcal{L}$ to $\log p(\mathbf{Y} \mid \Omega)$, where it can be shown via Jensen's inequality that [6]

$$\log p(\mathbf{Y} \mid \Omega) = \mathcal{L} + \mathrm{KL}(q \,\|\, p) \tag{8}$$

$$\text{where } \mathcal{L} = \int q(\Theta \mid \mathbf{Y}, \Omega) \log \left\{ \frac{p(\Theta, \mathbf{Y} \mid \Omega)}{q(\Theta \mid \mathbf{Y}, \Omega)} \right\} d\Theta \tag{9}$$

The property $\mathcal{L} \leq \log p(\mathbf{Y} \mid \Omega)$ follows from the fact that $\mathrm{KL}(q \,\|\, p) \geq 0$ for any q and p. The VB posterior approximation q can be determined iteratively with monotonically increasing $\mathcal{L}$, such that q is guaranteed to converge on a local minimizer of $\mathrm{KL}(q \,\|\, p)$ [6]. Furthermore, since the maximizer of $p(\mathbf{Y} \mid \Omega)$ also maximizes $\log p(\mathbf{Y} \mid \Omega)$ and neither term is available in practice, $\mathcal{L}$ can be used to approximate $\log p(\mathbf{Y} \mid \Omega)$ and thus serve as the metric to maximize for Bayesian model selection [4], [6], [12], [17], [32]. Note that p can be multimodal due to complex probabilistic dependencies between variables or nonidentifiability issues, so that q (which is often unimodal) can converge towards any one of p's true modes to (locally) minimize the KLD. Therefore, a common strategy to finding the best KLD minimizer (also used in this work) is to apply multiple initial guesses for q, e.g., using ML/MAP solutions.

References [34] and [7] proposed VB approximations for ME *regression* models (i.e., with continuous expert outputs in (4) instead of discrete ones), although their methods have important drawbacks that severely limit their applicability to ME and MMS *discriminative* models. In particular, the method of [34]: 1) cannot use the softmax model; 2) cannot estimate expert model parameters from data; and 3) requires density estimation of the joint input-output distribution $p(X, D)$, which is generally challenging and computationally expensive. The method of [7] overcomes only the last two drawbacks, since it can only be applied to ME models specified via trees of *binary logistic* gating and expert functions. This requires $\mathcal{M}$ to contain all possible binary trees for fixed G, which raises the overall cost of learning (especially if $K > 2$, since each expert must be also be expressed as a "one-versus-all" binary classification tree). The limitations of both these VB methods stem directly from analytical difficulties in handling the softmax function. As such, they are ill-suited for learning general models with latent softmax terms and are particularly unusable for learning MMS, which is intrinsically defined by the softmax model.

To overcome these limitations, new VB approximations are proposed here to handle softmax functions analytically using a bound proved in [8]. These approximations place no restrictions on learning with latent softmax variables, and are therefore easier to use and more broadly applicable than either of the aforementioned VB methods for restricted ME models. In particular, it is straightforward to extend the proposed VB approximations to hierarchical mixture of expert discriminative and regression models, although the exact formulas for these extensions are not given here due to limited space.

III. Variational Bayes Learning for MMS Models

The graphical model for Bayesian MMS learning is shown in Fig. 2(a), for fixed $\Omega = [s_1, \ldots, s_K]$ with labeled observations $\mathbf{Y} = \{y_1, \ldots, y_N\} = \{(x_1, d_1), \ldots, (x_N, d_N)\}$ and hidden variables $\Theta = \{Z_{1:N}, w_{1:S}, \alpha_{1:S}\} = \{\mathbf{Z}, \mathbf{w}, \boldsymbol{\alpha}\}$. Defining $\mathbf{D} = \{d_1, \ldots, d_N\}$ and $\mathbf{X} = \{x_1, \ldots, x_N\}$, it follows from Fig. 2(a) that

$$p(\mathbf{w}, \boldsymbol{\alpha}, \mathbf{Z}, \mathbf{D} \,|\, \mathbf{X}) = p(\boldsymbol{\alpha}) p(\mathbf{w} \,|\, \boldsymbol{\alpha}) \prod_{n=1}^{N} p(Z_n \,|\, X_n, \mathbf{w}) p(D_n \,|\, Z_n). \quad (10)$$

For $i \in \{1, \ldots, S\}$, $n \in \{1, \ldots, N\}$, and $d_n \in \{1, \ldots, K\}$, the conditional distributions are assumed to be

$$p(\alpha_i; a_0, b_0) = \mathcal{G}_{\alpha_i}(a_0, b_0) \quad (11)$$
$$p(w_i \,|\, \alpha_i) = \mathcal{N}_{w_i}\left(0, \alpha_i^{-1} \cdot \mathrm{I}\right) \quad (12)$$
$$p(D_n = d_n \,|\, Z_n = i) = T_{\mathrm{in}} \quad (13)$$
$$p(Z_n = i \,|\, X_n = x_n, \mathbf{w}) = f(i; x, \mathbf{w}) \quad (14)$$

where $\mathcal{G}$ is the Gamma distribution with fixed shape and scale parameters a_0 and b_0. Note that each subclass weight w_i has a spherical zero-mean Gaussian prior with a Gamma-distributed precision hyperparameter α_i; these conjugate priors effectively enable regularization with an unknown penalty parameter and are commonly used in Bayesian learning (e.g., see [6], [7], [32], and [34]). Also note that α_i need not be identical for all w_i, although this is assumed here for simplicity.

From (5), the posterior over Θ is (dropping Ω for simplicity)

$$p(\Theta \,|\, \mathbf{D}, \mathbf{X}) = \frac{p(\Theta, \mathbf{D} \,|\, \mathbf{X})}{p(\mathbf{D} \,|\, \mathbf{X})} = \frac{p(\mathbf{Z}, \mathbf{w}, \boldsymbol{\alpha}, \mathbf{D} \,|\, \mathbf{X})}{\sum_{z_{1:N}} \int\int p(\mathbf{Z}, \mathbf{w}, \boldsymbol{\alpha}, \mathbf{D} \,|\, \mathbf{X}) d\mathbf{w} d\boldsymbol{\alpha}}. \quad (15)$$

Upon substitution of (10)–(14) into (15), it is clear that (15) is intractable since $p(\mathbf{D} \,|\, \mathbf{X})$ requires an exponentially large sum over integrals that have no closed form. To obtain a tractable approximate posterior $q(\Theta \,|\, \mathbf{Y}, \Omega) \equiv q(\Theta) = q(\mathbf{Z}, \mathbf{w}, \boldsymbol{\alpha})$ using

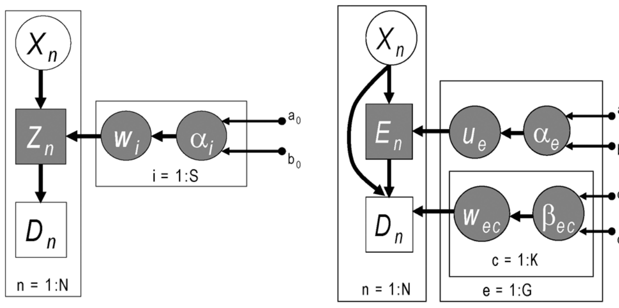


Fig. 2. Graphical plate models for Bayesian learning: (a) MMS. (b) ME.

VB, q is first specified via an assumed conditional independence factorization. The typical "mean-field" approximation (e.g., [4], [7], [12], and [32]) is used here, where

$$q(\Theta) = \prod_{n=1}^{N} q(Z_n) \prod_{i=1}^{S} q(w_i) q(\alpha_i) = \prod_{k=1}^{N+2S} q(\theta_k). \quad (16)$$

This factorization is convenient since it can be shown through a standard proof (e.g., [6, Ch. 10]) that subsequent minimization of (7) leads to the following canonical "free-form" inference formula for each posterior factor $\theta_k \in \Theta$

$$\ln q(\theta_k) = \mathbb{E}[\ln p(\Theta, \mathbf{D} \,|\, \mathbf{X})]_{q(\Theta \setminus \theta_k)} + \text{const}. \quad (17)$$

The expectations here are taken on the log of (10) with respect to the variational posterior factors $q(\theta_k)$ of all hidden variables θ_k in Θ *except* the θ_k of interest on the left hand side. Since the resulting formulas correspond to solving simultaneous nonlinear equations, they are iteratively re-solved in cyclic fashion until convergence. A single pass through (17) for each θ_k is defined as one VB cycle.

Now, the updates in (17) must be modified to handle the nonintegrable softmax terms from (14). These softmax terms are replaced by the local variational softmax lower-bound $\hat{p}(Z_n = i \,|\, X_n = x_n, \mathbf{w}) \leq p(Z_n = i \,|\, X_n = x_n, \mathbf{w})$ proved in [8], where

$$\hat{p}(Z_n = i \,|\, X_n = x_n, \mathbf{w}) = \exp\left(w_i^T x_n - \Phi_n\right) \quad (18)$$

$$\Phi_n = \gamma_n + \sum_{i=1}^{S} \frac{w_i^T x_n - \gamma_n - \xi_{\mathrm{in}}}{2} + \lambda(\xi_{\mathrm{in}}) \left[\left(w_i^T x_n - \gamma_n\right)^2 - \xi_{\mathrm{in}}^2\right] + \log(1 + e^{\xi_{\mathrm{in}}}) \quad (19)$$

$$\lambda(\xi_{\mathrm{in}}) = \frac{1}{2\xi_{\mathrm{in}}} \left[\frac{1}{1 + e^{-\xi_{\mathrm{in}}}} - \frac{1}{2}\right]. \quad (20)$$

This lower bound replaces the nonintegrable denominator of the softmax function in (14) with a product of unnormalized Gaussians, whose log is given by (19). The scalars ξ_{in} and γ_n are *local variational parameters* that control the shapes and locations of the bounding Gaussians across each training datum and across each discrete label of the softmax distribution (the subclass labels, in the MMS case). ξ_{in} and γ_n must be reoptimized within each VB cycle to tighten the expected lower-bounds to the true softmax terms being approximated; fortunately, this is done easily in closed-form with fairly tight bounds, as described in [8]. It is easy to see that these local softmax lower bounds still ensure a lower bound $\tilde{\mathcal{L}} \leq \mathcal{L}$ to $p(\mathbf{Y} \mid \Omega)$, where

$$\tilde{\mathcal{L}} = \int q(\Theta \mid \mathbf{Y}, \Omega) \log \left\{ \frac{\hat{p}(\Theta, \mathbf{Y} \mid \Omega)}{q(\Theta \mid \mathbf{Y}, \Omega)} \right\} d\Theta,$$
$$\hat{p}(\Theta, \mathbf{Y} \mid \Omega) = p(\boldsymbol{\alpha}) p(\mathbf{w} \mid \boldsymbol{\alpha}) \prod_{n=1}^{N} \hat{p}(Z_n \mid X_n, \mathbf{w}) p(D_n \mid Z_n). \tag{21}$$

Substituting (21) and (11)–(13) into (17) for (10) and computing the required expectations leads to the following closed-form variational posterior factors

$$q(w_i) = \mathcal{N}_{w_i}(\mu_i, \Sigma_i) \tag{22}$$
$$q(\alpha_i) = \mathcal{G}_{\alpha_i}(a_i, b_i), \tag{23}$$
$$q(Z_n = i) = g(i; x_n, \{\langle w_1 \rangle, \ldots, \langle w_S \rangle\}) \tag{24}$$

where $\langle \cdot \rangle$ is the expected value with respect to $q(\Theta)$, and

$$\Sigma_i^{-1} = \langle \alpha_i \rangle \cdot \mathrm{I} + 2 \sum_{n=1}^{N} \lambda(\xi_{\text{in}}) x_n x_n^T, \tag{25}$$
$$\Sigma_i^{-1} \mu_i = \sum_{n=1}^{N} \left[\langle t_{\text{in}} \rangle - \frac{1}{2} + 2 \gamma_n \lambda(\xi_{\text{in}}) \right] x_n \tag{26}$$
$$a_i = a_0 + \frac{M+1}{2}, \; b_i = b_0 + \frac{1}{2} \left\langle w_i^T w_i \right\rangle \tag{27}$$
$$\langle t_{\text{in}} \rangle = q(Z_n = i), \langle \alpha_i \rangle = a_i / b_i \tag{28}$$
$$\langle w_i \rangle = \mu_i, \langle w_i^T w_i \rangle = \text{tr}(\Sigma_i) + \mu_i^T \mu_i. \tag{29}$$

Here, $\langle t_{\text{in}} \rangle = q(Z_n = i)$ is subclass i's *expected posterior responsibility* for training datum n, which is computed via (3) using the expected subclass weights. As shown in [8], the parameters ξ_{in} and γ_n can be set to maximize the expectation of (18) with respect to $q(\Theta)$, giving

$$\xi_{\text{in}}^2 = x_n^T \Sigma_i x_n + \left(\mu_i^T x_n \right)^2 + \gamma_n^2 - 2 \gamma_n \mu_i x_n \tag{30}$$
$$\gamma_n = \frac{\frac{1}{2} \left(\frac{S+1}{2} - 1 \right) + \sum_{i=1}^{S} \lambda(\xi_{\text{in}}) w_i^T x_n}{\sum_{j=1}^{S} \lambda(\xi_{jn})}. \tag{31}$$

The lower bound $\tilde{\mathcal{L}}$ to $p(\mathbf{Y} \mid \Omega)$ can be computed to gauge convergence of the VB cycles, where it can be shown that [for T_{in} as defined in (3)]

$$\tilde{\mathcal{L}} = \sum_{n=1}^{N} \left\{ \log \left(\sum_{i=1}^{S} T_{\text{in}} \cdot \exp \left(\mu_i^T x_n \right) \right) - \langle \Phi_n \rangle \right\}$$
$$- \left\{ \sum_{i=1}^{S} \text{KL}[q(w_i) \,\|\, p(w_i \mid \langle \alpha_i \rangle)] + \text{KL}[q(\alpha_i) \,\|\, p(\alpha_i)] \right\}. \tag{32}$$

TABLE I
VB MMS LEARNING ALGORITHM FOR FIXED SUBCLASS CONFIGURATION

```
0. Given: $\mathbf{X}$, $\mathbf{D}$, initial $\xi_{in}$, $\gamma_n$,
       fixed subclass configuration with $S$ total subclasses
1. for $\xi_{in}$ and $\gamma_n$ fixed for $i \in \{1, ..., S\}$, $n \in \{1, ..., N\}$
    for $i = 1 : S$
      compute $q(w_i)$ using (22)
      compute $q(\alpha_i)$ using (23)
    end
    for $n = 1 : N$
      compute $q(Z_n)$ using (24)
    end
2. for fixed $q(\Theta|\mathbf{X})$,
    for $i = 1 : n_{lc}$
      (a) compute all $\xi_{in}$ for fixed $\gamma_n$ using (30)
      (b) compute all $\gamma_n$ for all fixed $\xi_{in}$ using (31)
    end
3. Compute $\tilde{\mathcal{L}}$ using (32). Stop if $\tilde{\mathcal{L}}$ converged; else, Repeat 1-2.
```

$\tilde{\mathcal{L}}$ represents an expected data log-likelihood penalized by the nonnegative and closed-form KLDs between the priors and variational posteriors.[2] Most of the required terms for $\tilde{\mathcal{L}}$ are already computed during the cyclic updates for $q(\Theta)$ and $\tilde{\mathcal{L}}$ is also guaranteed to monotonically increase over the VB cycles.

Table I summarizes the VB learning procedure for fixed Ω. Note that an additional convergence loop of n_{lc} iterations is required here for the local variational parameters ξ_{in} and γ_n, which are coupled in (30)–(31); $n_{lc} = 15$ was sufficient for this paper's implementation. Since $q(\Theta)$ converges to a local KLD minimizer, the VB algorithm should be run with multiple intializations to ensure $\tilde{\mathcal{L}}$ is maximized for fixed Ω (this is the typical approach for avoiding poor local solutions in VB learning, e.g., see [4], [7], [12], [32], and [34]). In this paper's implementation, ξ_{in} and γ_n were initialized via (30) and (31) by replacing each μ_i with maximum likelihood weight estimates and Σ_i with various spherical covariances.

A. Bayesian MMS Model Selection

For each $\Omega = [s_1, \ldots, s_K]$ in a set of candidate models $\mathcal{M}$, the VB approximation can be applied to find $\log p(\mathbf{Y} \mid \Omega) \approx \tilde{\mathcal{L}}^*$, where $\tilde{\mathcal{L}}^* = \tilde{\mathcal{L}} - \ln(\prod_{i=1}^{S} s_d!)$ is the penalized lower bound obtained at convergence.[3] $\mathcal{M}$ could be defined by placing an upper bound r on s_d for each class d. However, this leads to r^K possible MMS models, which can be impractical to assess. Hence, heuristics are generally needed within the Bayesian model selection scheme (or any other model selection strategy) to define $\mathcal{M}$ within practical limits for MMS.

A simple but principled "compressive search" (CS) heuristic is proposed here to define $\mathcal{M}$ iteratively via the posterior responsibilities $\langle t_{\text{in}} \rangle$ from the VB approximation. This uses the idea that subclasses whose existence is not well supported by data are likely irrelevant and can probably be discarded. First, the VB approximation is applied to the first r uniform subclass MMS models, which have $s_d = c \; \forall d \in \{1, \ldots, K\}$ and

[2]See Appendix B of [32] for the necessary Gaussian-Gaussian and Gamma-Gamma KLDs in each prior/posterior pairing.

[3]The penalty is needed since parameters are not identifiable if any $s_d > 1$; the approximate model likelihood probability must be divided among all possible label permutations to correctly penalize model complexity [14].

TABLE II
COMPRESSIVE SEARCH FOR MMS MODEL SELECTION

```
0. Given: $s_d$ upper bound $r$, max number of iterations $g_{max} \geq 1$,
      PSP threshold probability $\epsilon$, number of iterations $g = 0$
1. Set $\mathcal{M}$ to all uniform subclass configurations between 1 and $r$;
2. Define $\mathcal{M}_u$ to be the set of models not yet learned in $\mathcal{M}$;
3. Set $g = g + 1$ and for each configuration in $\mathcal{M}_u$:
      perform VB learning using Table 1;
      compute $\tilde{\mathcal{L}}^* = \tilde{\mathcal{L}} - \ln(\prod_{i=1}^{S} s_d!)$;
      compute $p(i)$ for all subclasses using (33);
      threshold $p(i)$ against $\epsilon$ (subclass is relevant if p(i)$\geq \epsilon$);
      count number of relevant subclasses $s_d^{rel}$ per class label $d$;
      set spawned configuration to $[s_1^{rel}, ..., s_K^{rel}]$ and add to set $\mathcal{S}_g$
4. If $g < g_{max}$ and $\mathcal{S}_g \cup \mathcal{M} \neq \mathcal{M}$, let $\mathcal{M} \leftarrow \mathcal{M} \cup \mathcal{S}_g$ and repeat 2-3;
   Otherwise, stop;
5. Select $\Omega \in \mathcal{M}$ with largest $\tilde{\mathcal{L}}^*$.
```

$c \in \{1, \ldots, r\}$. *Posterior subclass probabilities* (PSPs) are then estimated within each model for each original class label d and relevant subclass $i \in \sigma(d)$ by computing

$$p(i) = \frac{1}{N_d} \sum_{n=1}^{N} \langle t_{\text{in}} \rangle \tag{33}$$

which is the expected number of times i appears for all observed labels $d_n = d$ divided by N_d, the number of training data with $d_n = d$. Note that $\langle t_{\text{in}} \rangle = 0$ if $i \notin \sigma(d_n)$, so $\sum_{i \in \sigma(d_n)} \langle t_{\text{in}} \rangle = 1$ and $\sum_{i \in \sigma(d)} \sum_{n=1}^{N} \langle t_{\text{in}} \rangle = \sum_{i \in \sigma(d)} \sum_{\{n \,|\, d = d_n\}} \langle t_{\text{in}} \rangle = N_d$; hence, $\sum_{i \in \sigma(d)} p(i) = 1$ for each d. Next, the PSPs are thresholded against a user-defined probability $\epsilon \in [0, 1]$ to determine the number of "relevant" subclasses per class in each model. This results in an additional subclass configuration that can be learned by VB on the next search iteration. The CS procedure stops when either no new subclass models are formed or $g_{\max}$ iterations are reached. The CS method thus avoids brute force searches over r^K MMS models by looking for "persistently useful" subclasses over a smaller uniform block of models. Note that the PSPs (33) resemble expected component responsibilities for Gaussian mixtures [6], which is not surprising given the resemblences between MMS and Gaussian mixture classifiers [2].

Table II summarizes the CS method, where r can be initialized to a modest value (e.g., 4 or 5) and then increased to scan more models, if required. Setting ϵ too large leads to overly conservative softmax models, while too small an ϵ can spawn no models. Hence, ϵ should be in the range between which these two extremes start occuring (which can be quickly estimated via bisection searches), although this will be problem dependent and sensitive to small N_d. In the experiments here, $\epsilon \in [0.01, 0.1]$ produced consistent results, though this range may not be suitable in special cases (e.g., highly imbalanced data sets, which may require a separate ϵ for each class). The CS procedure forms at most r new models per iteration g, so $\mathcal{M}$ has at most $rg_{\max}$ models, which is linear in r and independent of K. As the CS method is essentially top-down, the highest learning cost is incurred by the r uniform subclass model, which can be used as a basis for setting $g_{\max}$. In practice, CS should be run with multiple ϵ values in Step 3 to scan as many 'interesting' models as possible.

IV. Variational Bayes Learning for ME Models

The graphical model for Bayesian mixture of experts (ME) learning is shown in Fig. 2(b) for fixed $\Omega = G$ with labeled observations $\mathbf{Y}$ and hidden variables $\Theta = \{E_{1:N}, u_{1:G}, w_{11:GK}, \alpha_{1:G}, \beta_{11:GK}\} = \{\mathbf{E}, \mathbf{u}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}\}$. There are two key changes to the multimodal softmax VB approximation for ME learning: (1) the softmax lower bound is applied $G + 1$ times: once for each of the G softmax expert models and once for the softmax gating function, (2) separate priors, hyperparameters and hyperpriors are specified for the gating weights and the G sets of expert weights.

The joint pdf for the ME model Fig. 2(b) given $\mathbf{X}$ is

$$\begin{aligned} p(\mathbf{E}, \mathbf{D}, \mathbf{u}, \mathbf{w}, \boldsymbol{\alpha}, \beta_{1:G} \,|\, \mathbf{X}) &= p(\boldsymbol{\alpha})p(\boldsymbol{\beta})p(\mathbf{u} \,|\, \boldsymbol{\beta})p(\mathbf{w} \,|\, \boldsymbol{\alpha}) \\ &\quad \times \prod_{n=1}^{N} \prod_{g=1}^{G} [p(E_n = g \,|\, X_n, \mathbf{u}) \\ &\quad \times \prod_{d=1}^{K} [p(D_n = d \,|\, X_n, E_n, \mathbf{w})]^{s_{dn}} \Bigg]^{t_{gn}} \end{aligned} \tag{34}$$

where s_{dn} and t_{gn} are binary indicator variables denoting membership of datum n to the states $D_n = d$ and $E_n = g$, respectively (s_{dn} is observed in the training data through $\mathbf{D}$; t_{gn} is hidden since E_n is hidden). For $g \in \{1, \ldots, G\}$, $c \in \{1, \ldots, K\}$, and $n \in \{1, \ldots, N\}$, the distributions are

$$p(\alpha_g) = \mathcal{G}_{\alpha_g}(a_0, b_0) \tag{35}$$

$$p(\beta_{gc}) = \mathcal{G}_{\beta_{gc}}(c_0, d_0) \tag{36}$$

$$p(u_g \,|\, \alpha_g) = \mathcal{N}_{u_g}\left(0, \alpha_g^{-1} \cdot \mathrm{I}\right) \tag{37}$$

$$p(w_{gc} \,|\, \beta_{gc}) = \mathcal{N}_{w_{gc}}\left(0, \beta_{gc}^{-1} \cdot \mathrm{I}\right) \tag{38}$$

$$p(E_n = g \,|\, X_n = x, \mathbf{u}) = f(g; x, \mathbf{u}) \tag{39}$$

$$p(D_n = d \,|\, X_n = x, E_n = g, \mathbf{w}) = f(d; x, w_{g1:gK}). \tag{40}$$

From (5), the posterior over the hidden variables $\Theta = \{\mathbf{E}, \mathbf{u}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}\}$ takes exactly the same form as (15), where the denominator is now given by marginalization of (34) over the Θ variables. Once again, substitution of (35)–(40) into (15) leads to an intractable posterior, as in the MMS case. The VB approximation thus proceeds as before, although now the following factorized approximate posterior is assumed

$$\begin{aligned} q(\Theta) &= \prod_{g=1}^{G} \left[q(\alpha_g)q(u_g) \prod_{c=1}^{K} q(\beta_{gc})q(w_{gc}) \right] \\ &\quad \times \prod_{n=1}^{N} q(E_n)q(D_n) \\ &= \prod_{k=1}^{2N+2G(K+1)} q(\theta_k) \end{aligned} \tag{41}$$

which again leads to the variational posterior factor update formulas from (17), where the joint pdf is now given by (34). However, to obtain closed-form expectations, the softmax lower bound (18) is now applied $G + 1$ separate times to replace the

(39) and (40) terms in (34). To this end, let the lower bounding softmax replacements be denoted by

$$\hat{p}(E_n = g \,|\, x, \mathbf{u}) = \exp\left(u_g^T x - \Phi_n^0\left(\gamma_n^0, \xi_n^{1:G}\right)\right) \quad (42)$$

$$\hat{p}(D_n = d \,|\, x, g, \mathbf{w}) = \exp\left(w_{gd}^T x - \Phi_n^g\left(\gamma_n^g, \xi_n^{g1:gK}\right)\right) \quad (43)$$

where the relations from (19)–(20) still apply for $g \in \{1, \ldots, G\}$, $d \in \{1, \ldots, K\}$, and where $\{\gamma_n^0, \xi_n^{1:G}\}$ and $\{\gamma_n^g, \xi_n^{g1:gK}\}$ are the corresponding sets of variational parameters for the gating lower bound and the gth expert's lower bound, respectively (note that there are now a total of $2N + NG(K+1)$ local variational parameters).

Substituting (42), (43), and (35)–(38) into (34) and computing the required expectations in (17) via (41) leads to the following closed-form variational posterior factors

$$q(u_g) = \mathcal{N}_{u_g}(\mu_g, \Sigma_g), \quad (44)$$

$$q(\alpha_g) = \mathcal{G}_{\alpha_g}(a_g, b_g), \quad (45)$$

$$q(w_{gc}) = \mathcal{N}_{w_{gc}}(\mu_{gc}, \Sigma_{gc}), \quad (46)$$

$$q(\beta_{wg}) = \mathcal{G}_{\beta_{gc}}(c_{gx}, d_{gc}), \quad (47)$$

$$q(E_n = g) = f(g; [x_n, 1]^T \{v_{n1}, \ldots, v_{nG}\}), \quad (48)$$

where

$$\Sigma_g = \langle \alpha_g \rangle \cdot \mathrm{I} + 2 \sum_{n=1}^{N} \lambda\left(\xi_n^g\right) x_n x_n^T \quad (49)$$

$$\Sigma_g^{-1} \mu_g = \sum_{n=1}^{N} \left[\langle t_{gn} \rangle - \frac{1}{2} + 2\gamma_n^0 \lambda\left(\xi_{gn}^g\right) \right] x_n \quad (50)$$

$$\Sigma_{gc} = \langle \beta_{gc} \rangle \cdot \mathrm{I} + 2 \sum_{n=1}^{N} \langle t_{gn} \rangle \lambda\left(\xi_n^{gc}\right) x_n x_n^T \quad (51)$$

$$\Sigma_{gc}^{-1} \mu_{gc} = \sum_{n=1}^{N} \langle t_{gn} \rangle \left[s_{dn} - \frac{1}{2} + 2\gamma_n^g \lambda\left(\xi_n^{gc}\right) \right] x_n \quad (52)$$

$$v_{ng} = \left[\mu_g + \mu_{gh}, \, -\langle \Phi_n^g \rangle\right]^T, \text{where } h = d_n \quad (53)$$

$$a_g = a_0 + M/2, \; b_g = b_0 + \frac{1}{2}\left\langle u_g^T u_g \right\rangle \quad (54)$$

$$c_{gc} = c_0 + M/2, \; d_{gc} = d_0 + \frac{1}{2}\left\langle w_{gc}^T w_{gc} \right\rangle \quad (55)$$

$$\langle t_{gn} \rangle = q(E_n = g), \langle \alpha_g \rangle = a_g / b_g, \langle \beta_{gc} \rangle = c_{gc} / d_{gc} \quad (56)$$

$$\left\langle u_g^T u_g \right\rangle = \mathrm{tr}(\Sigma_g) + \mu_g^T \mu_g, \; \left\langle w_{gc}^T w_{gc} \right\rangle = \mathrm{tr}(\Sigma_{gc}) + \mu_{gc}^T \mu_{gc} \quad (57)$$

The updates for the local gating and expert variational parameters are the same as in (30) and (31) with μ_i and Σ_i replaced: the updates for ξ_n^g and γ_n^0 use μ_g and Σ_g, while the updates for ξ_n^{gn} and γ_n^g use μ_{gc} and Σ_{gc}. To assess convergence of VB iterations, $\tilde{\mathcal{L}}$ can be computed as

$$\begin{aligned} \tilde{\mathcal{L}} = & \sum_{n=1}^{N} \sum_{g=1}^{G} \left[\langle t_{gn} \rangle \mu_g^T x_n + \sum_{c=1}^{K} \langle t_{gn} \rangle s_{dn} \mu_{gn}^T x_n \right] \\ & + \mathcal{H}[q(\mathbf{E})] - \left(\sum_{n=1}^{N} \left[\langle \Phi_n^0 \rangle + \sum_{g=1}^{G} \langle t_{dn} \rangle \langle \Phi_n^g \rangle \right] \right) \\ & - \mathrm{KL}[q(\boldsymbol{\alpha}) \,\|\, p(\boldsymbol{\alpha})] - \mathrm{KL}[q(\boldsymbol{\beta}) \,\|\, p(\boldsymbol{\beta})] \\ & - \mathrm{KL}[q(\mathbf{u}) \,\|\, p(\mathbf{u} \,|\, \langle \boldsymbol{\alpha} \rangle)] - \mathrm{KL}[q(\mathbf{w}) \,\|\, p(\mathbf{w} \,|\, \langle \boldsymbol{\beta} \rangle)] \end{aligned} \quad (58)$$

TABLE III
VB ME LEARNING ALGORITHM FOR FIXED G

```
0. Given: X, D, number of experts $G$, initial $\xi_n^g$, $\gamma_n^0$, $\xi_n^{gd}$, and $\gamma_n^g$
1. for $\xi_n^g$, $\xi_n^{gd}$, $\gamma_n^0$ and $\gamma_n^g$ fixed for $g \in \{1, ..., G\}$, $d \in \{1, ..., K\}$,
   and $n \in \{1, ..., N\}$
   for $g = 1 : G$
     compute $q(u_g)$ using (44)
     compute $q(\alpha_g)$ using (45)
       for $d = 1 : K$
         compute $q(w_{gd})$ using (46)
         compute $q(\beta_{gd})$ using (47)
       end
   end
   for $n = 1 : N$
     compute $q(E_n)$ using (48)
   end
2. for fixed $q(\Theta|\mathbf{X})$,
   for $i = 1 : n_{lc}$
     (a) update all $\xi_n^g$ and $\xi_n^{gd}$ for all fixed $\gamma_n^0$ and $\gamma_n^g$
     (b) update all $\gamma_n^0$ and $\gamma_n^g$ for all fixed $\xi_n^g$ and $\xi_n^{gd}$
   end
3. Compute $\tilde{\mathcal{L}}$ using (58). Stop if $\tilde{\mathcal{L}}$ converged; else, Repeat 1-2.
```

where $\mathcal{H}[q(\mathbf{E})]$ is the Shannon entropy of $q(\mathbf{E}) = \prod_{n=1}^{N} q(E_n)$ (note that each $q(E_n)$ is a discrete distribution and the KLDs are again closed-form, as per footnote 2). Table III summarizes the VB learning procedure for ME, which should be run with different initializations to avoid poor local solutions. As with MMS learning, maximum likelihood ME weight estimates are used here to intialize the local variational parameters for (42) and (43) for multiple runs. As before, $n_{lc} = 15$ was sufficient for the experiments here.

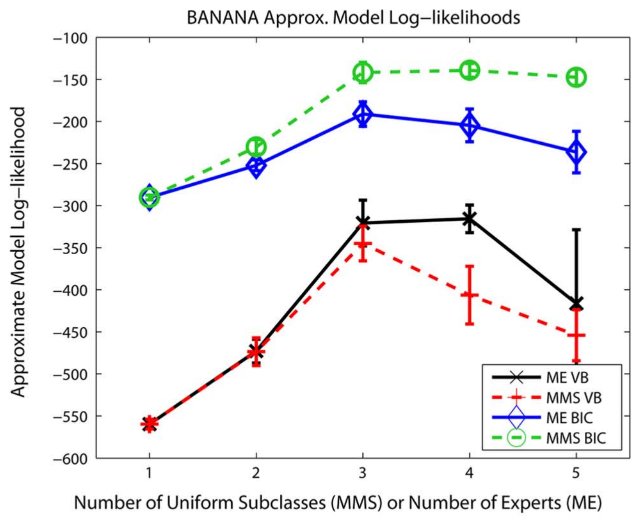
A. Bayesian ME Model Selection

Compared with the MMS model, the ME model space is much simpler to consider; a natural choice for $\mathcal{M}$ is the set of ME models with $\Omega = G$ between 1 and some upper bound. The VB approximation of Table III can be applied to each $\Omega \in \mathcal{M}$ to obtain the penalized VB model log-likelihood lower bound $\tilde{\mathcal{L}}^* = \tilde{\mathcal{L}} - \ln G!$ (which accounts for hidden expert label permutations). The Ω with the largest $\tilde{\mathcal{L}}^*$ can then be selected as the best fit.

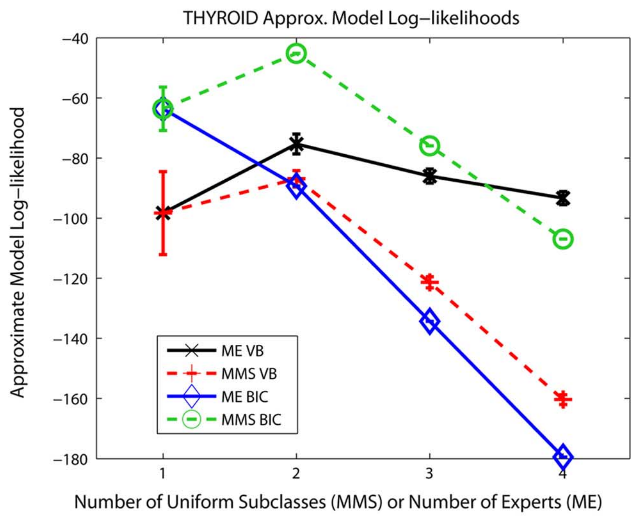
V. Experimental Results

This section presents VB learning results for MMS and ME on four benchmark data sets from classification literature and two data sets from an experimental application. Ten randomized training sets were created in each case to simulate multiple learning instances with the 'raw' X space, using no feature processing other than simple normalization. In all cases, the shape and scale prior hyperparameters for VB learning were set to $a_0 = b_0 = c_0 = d_0 = 1$,[4] and the convergence tolerance on $\tilde{\mathcal{L}}$ was set to 1×10^{-3} over 600 maximum update cycles. Maximum likelihood estimates were used to initialize all VB learning algorithms (as described in Sections III and IV) except for the second phase of compressive searches (CS) for MMS,

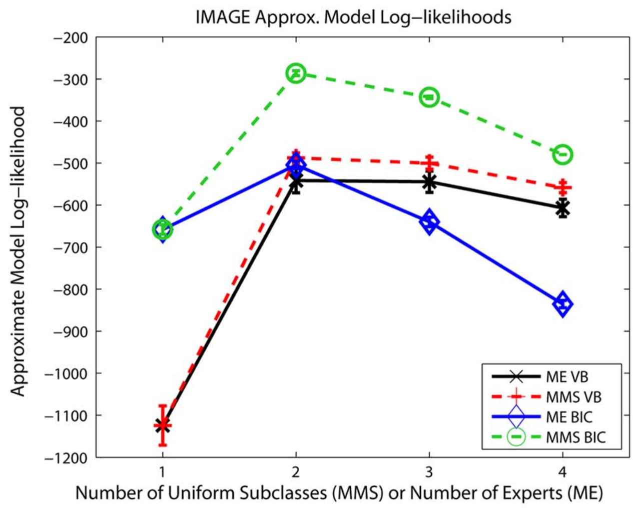
[4]The results were insensitive to these values, i.e., similar learning results were obtained with much broader gamma hyperpriors.



(a)



(b)



(c)

Fig. 3. Selected MMS and ME learning results on benchmark data (uniform MMS configurations shown only). (a) Banana. (b) Thyroid. (c) Image.

in which relevant parent subclass weights were used instead. Models learned by VB were also compared to those obtained by maximum likelihood (ML) and the Bayes Information Criterion (BIC) score

$$\mathrm{BIC}(\Omega) = \log(L_\Omega) - \frac{1}{2}k_\Omega \log(N) - \frac{1}{2}C(\Omega) \qquad (59)$$

where L_Ω is the maximized likelihood for model Ω, k_Ω is the free parameter count and C_Ω is the model permutation penalty. BIC is a popular metric for probabilistic model selection with hidden variables [4], [11], [14], [32]; as [4] notes, (59) asymptotically approximates $\log p(\mathbf{Y} \mid \Omega)$ and can be seen as a limiting case of the VB lower bound for $N \to \infty$. Thus, (59) is a suitable fitness metric to compare against the VB lower bound $\tilde{\mathcal{L}}^*$. All experiments were run in MATLAB (2.41 GHz AMD Athlon processor, Windows XP, 1.96 GB of RAM).

Since the generating models for the data are unknown, more sophisticated nonlinear kernel classifier models were used as 'high baselines' to assess the quality of the learned MMS and ME models by comparing holdout classification error rates. It is emphasized here that model selection is the focus of this study and these error rates are included only for completeness.

A. Benchmark Data

Four binary class data sets were taken from the Rätsch benchmark set [31], which is often used to compare sparse nonlinear kernel classifiers (e.g., [33] and [19]). Hence, these data offer a good challenge for Bayesian learning of 'standard' MMS and ME models that can only construct piecewise linear log-odds boundaries (cf. Section 2). The benchmarks used here are[5]: *Banana* (2 dimensions, 5400 total data points, 400 training points), *Thyroid* (5 dimensions, 215 total, 140 training), *Image* (20 dimensions, 2310 total, 1300 training), and *Twonorm* (20 dimensions, 7000 total, 400 training). In all cases, ME models were learned for all $G \in [1, 5]$ and MMS models were learned for all $s_d \leq 5$. The compressive search (CS) in Table II was also used separately with $r = 5$ and $g_{\max} = 2$ (so that at most 10 models could be assessed in each case). All sets used $\epsilon = 0.05$ except *Thyroid*, where $\epsilon = 0.10$ to due to less available data. Table IV summarizes the benchmark learning results, where classification error rates for the VB-learned models with the highest average $\tilde{\mathcal{L}}^*$ are computed with MAP parameters (errors via (1) with ML weights are also shown for a "low-baseline" comparison). Also shown for the CS results are: the number of times nonuniform VB-selected MMS models are scanned over all instances (#sc.); the average number of parents for the best-fit VB model ($\bar{pa}$); and the maximum number of models assessed by CS across all instances ($(|\mathcal{M}|^{\max})$). Fig. 3 plots some of the average $\tilde{\mathcal{L}}^*$ and BIC model scores for ME and MMS.

Fig. 3 shows that both scores generally behave similarly for MMS and ME as a function of complexity, as expected from theory. Interestingly, BIC is more conservative for the ME *Twonorm* case and less conservative for the MMS *Thyroid* case. In the former case, the complexity of the ME model dominates the BIC score, leading to a very stiff penalty. In the latter case, complex MMS models are often able to overfit with large enough L_Ω in (59) to counteract small fixed k_Ω and C_Ω penalties. VB avoids these issues and behaves more consistently in all cases due to $\tilde{\mathcal{L}}^*$'s adaptive penalty structure,

[5]The training/validation splits were tabulated previously in a standard catalog.

TABLE IV
BENCHMARK LEARNING RESULTS (MMS: BEST $[s_1, \ldots, s_K]$ FROM BRUTE FORCE (BF) AND CS SHOWN; ME: BEST G SHOWN)

Data	MMS				ME			Eq. (1)
	BIC-BF	VB-BF	VB-CS (#sc., $\bar{pa}$, $\lvert\mathcal{M}\rvert^{max}$)	VB-CS % Error	BIC	VB	VB %Error	ML %Error
Banana	[4,3]	[3,3]	[3,3] (-,1.5,6)	13.01±0.73	3	4/3	12.60±0.98 (G=4)	47.63 ± 4.17
Thyroid	[2,1]	[2,1]	[2,1] (10,4,6)	5.60±2.65	1	2	5.20±3.17	12.53 ± 4.71
Image	[3,1]	[3,1]	[3,2] (10,1.8,6)	3.40±0.57	2	3/2	3.69±0.88 (G=3)	48.81 ± 9.71
Twonorm	[1,1]/[1,2]/[2,1]	[1,1]	[1,1] (-,4,5)	3.06±0.36	1	1	3.06±0.36	4.26 ± 0.68

TABLE V
MMS/ME LEARNING RESULTS FOR ROBOFLAG DATA (MMS BIC/VB RESULTS FOR CS-GENERATED MODELS)

	BIC	VB (#sc., $\bar{pa}$, $\lvert\mathcal{M}\rvert^{max}$)	VB %Error	BIC	VB	VB %Error	%Error	ML %Error
1	[1,2,2]	[1,2,2] (10, 3.6, 8)	16.33±0.34	1	2	16.02±0.35	15.71±0.48	19.09±0.63
2	[1,2,2]	[1,2,2]/[1,3,2] (10/5,2.1/0.9,10)	18.44±0.45 / 17.92±1.05	1	3	18.10±1.03	18.53±0.83	23.23±0.68

which also accounts for estimated parameter *magnitude*. The MMS and ME models selected by VB in Table IV also show better agreement with each other, as the maximum s_d closely matches G in each case. This is reassuring, given the similarities between MMS and ME [2].

The error rates can be compared to the following literature results for nonlinear kernel classifiers: *Banana*: 10.8% and *Image*: 3.9% (both RVMs) [33]; *Thyroid*: 4.3% and *Twonorm*: 2.6% (both Laplacian classifiers) [19]. From these baselines, it is easy to see that the models learned by VB are reasonable and accurate. Since MMS and ME only use piecewise-linear class boundaries here, the errors in Table IV are expected to be somewhat higher than these baselines (which used much more highly optimized complex class boundaries), although they are still fairly close for these data. The models learned by VB and ML/BIC generally performed similarly (i.e., <1% error difference), with some exceptions. The [1,2] MMS model 'selected' by ML/BIC for *Twonorm* leads to 12.27% error, while the ML/BIC-selected $G = 2$ ME models for *Image* and *Thyroid* give errors of 7.33% and 13.37%, respectively. The former result stems from poor BIC penalization. The latter results are due to ML overfitting, as evidenced by the fact that VB selects $G = 2$ in both sets while achieving smaller errors (the ML weights are two orders of magnitude larger).

Table IV also shows that CS searches over no more than 6 models to consistently find MMS models that are the same as/very close to the best models obtained by VB with a more expensive brute force search. The results for $\bar{pa}$ also show that complex higher order models are frequently compressed to lower orders (e.g., for *Banana*, $\bar{pa} = 1.8$ is reasonable for [3,3] since it can only be spawned by two models: [4,4] and [5,5]). Varying ϵ between $[0.01, 0.1]$ and/or setting $g_{\max} = 3$ did not change the set of scanned models significantly (*Thyroid* was similarly insensitive for $\epsilon \in [0.01, 0.36]$). The slight disagreement between brute force and CS for *Image* gives a small 0.12% difference in classification error, though it is worth noting that CS never spawns the [3,1] model (altering ϵ or g did not change this).

B. Roboflag Data

The application data come from an experimental human-robotic interaction study, in which 16 human operators were trained to play 20 games of 'capture the flag' with robots in the RoboFlag simulator program [10]. In each game, a single player assigned waypoint locations for three robots over a search field in order to locate/identify two enemy targets, while avoiding collisions and a mobile enemy chaser. All telemetry and waypoint data from each game was recorded. These were post-processed by hand-labeling each assigned waypoint as one of several discrete high-level operator strategies that were commonly observed during the game. The ones considered here are (with number of total instances):

- *Strat*: move robot to strategic place for later use (1277)
- *Dcy*: use robot to decoy chaser (1156)
- *Srch*: use robot to search for targets (1034)
- *ID/Loc*: use robot to identify/localize targets (1674)
- *Evas*: avoid collision between robot and enemy (736).

These data are used to learn $P(D \mid X)$ via latent softmax variables, where X is a 10-dimensional vector of continuous telemetry data (relative positions/orientations of all vehicles) and D is a discrete strategy given X. Ten training instances of two different decision data subsets were used with 300 training data points per class: Case 1 with classes [*Dcy*, *Srch*, *ID/Loc*], and Case 2 with classes [*Evas*, *Strat*, *Srch*]. Further details on the experimental data and the motivation for learning $P(D \mid X)$ in the context of hybrid Bayesian network modeling can be found in [9] and [10]. ME models were learned for all G between 1 and 6. CS was applied to each case with $r = 6$, $\epsilon = 0.05$ and $g_{\max} = 2$ to generate $\mathcal{M}$ for MMS.

Table V summarizes the learning results, with classification error rates for high-baseline nonlinear kernel SVM classifiers (N-SVM, with Gaussian kernels whose bandwidths were set via cross-validation) and low-baseline basic softmax models (trained via ML). Fig. 4 plots some $\tilde{\mathcal{L}}^*$ values from MMS and ME learning. The VB-learned MMS and ME models are comparably accurate to each other and the N-SVM classifiers (the modest error rates for the N-SVMs reflect the inherent difficulty of separating the decision classes using telemetry data alone). Thus, the VB models are indeed reasonable and unlikely to be overfit. The models selected by ML/BIC show good agreement with $\tilde{\mathcal{L}}^*$ in both cases for the MMS model. This is not the case for the ME model: ML/BIC heavily overpenalizes model complexity and always selects the basic softmax model ($G = 1$), which is least accurate in both cases Fig. 4 shows that

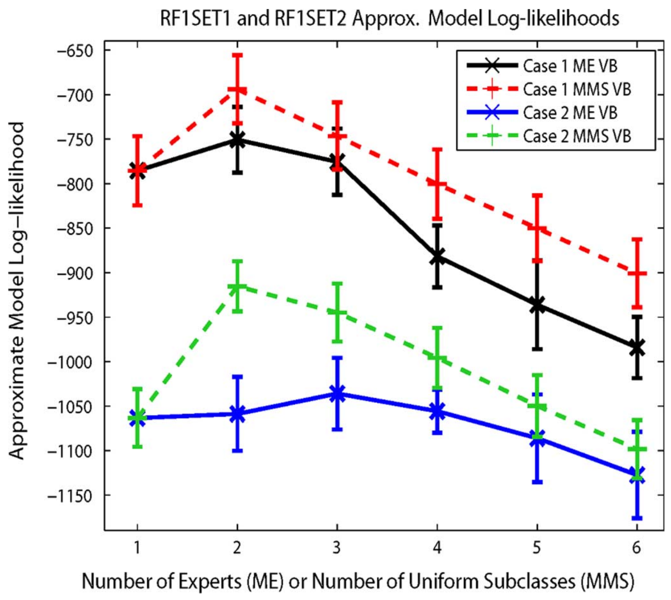


Fig. 4. VB model log-likelihoods for RoboFlag cases 1 and 2 (uniform MMS configurations shown only).

TABLE VI
MEAN LEARNING TIMES FOR MODELS SELECTED BY VB (S)

Data	MMS (CS selection)		ME	
	ML	VB	ML	VB
Banana	0.25	5.91	0.84	4.83
Thyroid	0.20	1.49	2.92	1.64
Image	39.98	24.95	209.70	14.91
RF 1	1.79	12.92	32.24	10.83
RF 2	1.16	16.30	27.59	10.84

$\tilde{\mathcal{L}}^*$ is generally higher for MMS than for ME. This is due to the dimension of X, which leads to heavier penalties in (58). Table V also shows that CS searches over 10 or fewer models in each case. While the [1,2,2] model is scanned in all instances by CS for both Case 1 and 2, the [1,3,2] model in Case 2 (which has slightly lower classification error and slightly higher $\tilde{\mathcal{L}}^*$) is scanned in only half the instances (altering ϵ and $g_{\max}$ did not force [1,3,2] to appear in the other half or change $\bar{pa}$). The CS results for Cases 1 and 2 were consistent for $\epsilon \in [0.015, 0.11]$ and $\epsilon \in [0.02, 0.08]$, respectively.

C. Performance Times

Table VI shows average computation times for VB and ML/BIC learning for the "nonbasic" models selected via $\tilde{\mathcal{L}}^*$ in Tables IV and V.[6] The ML results used direct nonlinear optimization via Matlab's quasi-Newton `fminunc` routine (the Hessian was computed explicitly for MMS and automatically approximated for ME). It is clear that direct ML is faster than VB on low-dimensional problems such as *Banana*, although the times increase (mainly for ME) as the problem dimensions increase.[7] The times required for the VB approximations are reasonable, especially for the higher dimensional problems (as the factorized mean-field approximation advantageously decouples complex parametric dependencies).

VI. DISCUSSION

A. Performance Considerations

While a full analysis cannot be given here due to space limitations, some comments are in order regarding sample size sensitivity for the proposed VB approximations. Further trials (not presented here) on the data of Section V, on synthetic truth data and on more complex/highly sparse benchmark data (e.g., the *Forensic Glass* set [3]) suggest that the proposed mean field VB approximations can perform badly with very sparse data sets and are instead most reliable for intermediate and large sample sizes (i.e., relative to the number of parameters to be estimated). This agrees with the findings of previous empirical and theoretical studies on the asymptotic properties of mean field VB approximations in other related models [15], [18], [35], which show that VB estimates are generally biased with respect to the true posterior statistics (as an unavoidable consequence of 'deliberate model misspecification' via the mean field assumption). While such biases can be quite significant at small sample sizes, they generally become much less significant as the proportion of observed to unobserved variables in the training set increases (or equivalently, as the number of training data increases when the number of hidden parameters can be fixed, as in the models studied here). As noted in [4], sparse data can cause significant difficulties even for state-of-the-art sampling algorithms, which typically require more effort to tune and are more computationally expensive than VB for fully Bayesian learning.

Since latent softmax models are useful for pure classification as well as general hybrid probabilistic modeling (cf. Section I), it should also be noted that models selected to maximize the VB lower bound $\tilde{\mathcal{L}}^*$ are not necessarily guaranteed to minimize classification error rate (as these metrics are not the same [6]). For instance, it was found that the [4,3] model learned by VB had the best overall MMS error rate for *Banana* (12%), despite having a slightly lower $\tilde{\mathcal{L}}^*$ score than the [3,3] model. Thus, the proposed VB approximations should be used appropriately. If the goal is to optimize classifier accuracy, then estimated classification error should be used as the fitness metric, if possible [23]. While VB can always be used to approximate (5) for Bayesian classification, $\tilde{\mathcal{L}}^*$ should only be used for classifier selection if reliable error rate estimates are infeasible/expensive to obtain [7]. In contrast, $\tilde{\mathcal{L}}^*$ should always be used in general Bayesian network identification problems, since (approximate) model log-likelihoods are of direct interest [4], [26].

Regarding MMS model selection, it should be emphasized that the proposed compressive search (CS) is only a simple suboptimal heuristic for avoiding brute force comparisons over an exponentially large model space. As such, it has no formal guarantee of finding an MMS model to maximumize $\tilde{\mathcal{L}}^*$. Although CS does well here overall at scanning suitable MMS models

[6]A comparison of the times for the baseline nonlinear kernel models is not feasible, as these times are either not available in the literature or cannot be directly compared to m-code execution times (e.g., the N-SVM used highly optimized compiled C code).

[7]Turning off Hessian computations sped up ML greatly, although it converged to poor local maxima much more frequently.

with limited data, the results nevertheless point to some limitations. First, as (33) suggests, CS requires each class to have sufficient data for the estimates in (33) to be reliable. Second, CS can get trapped with extra subclasses due to highly nonlinear class boundaries in X (e.g., CS is unable to compress [3,2] to [3,1] in the *Image* results). Therefore, while CS generally searches in the "best neighborhood" of models (given enough data), it may not fully explore this neighborhood. Running CS with multiple ϵ values per class can sometimes remedy this, though *post-hoc* analysis of scanned models often yields better insight for improving searches.[8] Although beyond the scope of this paper, it is also possible to consider $\tilde{\mathcal{L}}^*$ as a score for other model search strategies (e.g., greedy searches [34]).

B. Extensions

The proposed VB approximations can be readily extended to hierarchical ME models (cf. Section II), where the local variational softmax bound can be applied to every softmax gating/expert function in a mean-field approximation. The proposed methods can also be extended to softmax-gated ME *regression* models by replacing the softmax experts in (4) with linear-Gaussian output models (e.g., see [7]). This replacement is particularly nice as it eliminates the local softmax bound G times (it must only be applied for the gating function). The proposed VB approximations can also be used if X is replaced by an arbitrary nonlinear input map $\phi(X)$ (e.g., from feature selection or basis changes). As such, the lower bound $\tilde{\mathcal{L}}^*$ can also be very useful for comparing models with different $\phi(X)$.

In light of these extensions, it is important to note that VB approximations are always prone to poor local KLD minimizers. The implementations here largely mitigated this via multiple initializations derived from maximum likelihood (ML) estimates (e.g., see [32]). Although ML is also prone to poor local solutions, it can often be quickly re-run many times using direct optimization methods to obtain a good set of a priori initial guesses for VB. However, this strategy can be infeasible for problems involving very high-dimensional parameter spaces and/or hierarchical models. While avoidance of poor local solutions can never be fully guaranteed, it is straightforward to slightly modify the general mean-field VB updates in (17) to accommodate the deterministic annealing approach of [21], which can effectively mitigate VB's sensitivity to poor initializations (a similar method is also defined in [17] for VB learning).

VII. Conclusion

This paper derived new VB learning approximations for two latent softmax variable models: the multimodal softmax model and the mixture of expert discriminative models. The proposed VB solutions overcome the limitations of previous Bayesian methods for learning latent softmax variable models, and are also readily extendable to hierarchical mixture of expert models for classification and nonlinear/non-Gaussian regression. Experiments with benchmark and application data showed that the proposed VB approximations are effective in practice. Comparisons to models learned via the Bayes Information Criterion and to baseline nonlinear kernel classifiers confirmed the soundness of the proposed VB approximations for Bayesian parameter estimation and model selection.

[8]E.g., if $s_d = 1$ is always spawned for certain classes, then a constrained brute force search may be feasible over other classes.

Acknowledgment

The authors thank the anonymous reviewers for their many valuable comments. The authors also gratefully acknowledge the contribution of the Air Force Office of Scientific Research (AFOSR) and Dr. S. Galster.

References

[1] N. Ahmed and M. Campbell, "Variational Bayesian data fusion of multi-class discrete observations with applications to cooperative human-robot estimation," in *Proc. 2010 Int. Conf. Robot. Autom. (ICRA 2010)*, Anchorage, AK, pp. 186–191.

[2] N. Ahmed and M. Campbell, "On estimating probabilistic discriminative models with subclasses," Expert Syst. Appl. under review.

[3] A. Asuncion and D. Newman, UCI Machine Learning Repository 2007 [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html

[4] M. Beal and Z. Gharamani, "Variational Bayesian learning of directed graphical models with hidden variables," *Bayesian Anal.*, vol. 1, no. 4, pp. 793–832, 2006.

[5] C. Bishop, *Neural Networks for Pattern Recognition*. New York: Oxford Univ. Press, 1995.

[6] C. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer, 2006.

[7] C. Bishop and M. Svensen, "Bayesian hierarchical mixture of experts," in *Proc. 19th Conf. Uncertainty in Artif. Intell. (UAI03)*, 2003.

[8] G. Bouchard, "Efficient bounds for the softmax function and applications to approximate inference in hybrid models," in *Proc. NIPS 2007 Workshop for Approximate Bayesian Inference in Continuous/Hybrid Syst.*, Whistler, British Columbia, Canada, 2007.

[9] F. Bourgault, N. Ahmed, D. Shah, and M. Campbell, "Probabilistic operator-multiple robot modeling using Bayesian network representation," in *Proc. 2007 Guid., Navigat., Contr. Conf. (GNC'07)*, Hilton Head, SC, 2007.

[10] M. Campbell, F. Bourgault, S. Galster, and D. Schneider, "Toward probabilistic operator-multiple robot decision models," in *Proc. 2007 Int. Conf. on Robot. Autom. (ICRA 2007)*, Rome, Italy, 2007, pp. 4373–4379.

[11] A. Carvalho and M. Tanner, "Modeling nonlinearites with mixtures-of-experts of time series models," *Int. J. Math. Math. Sci.*, vol. 2006, pp. 1–22, 2006.

[12] M. Chappell, A. Groves, B. Whitcher, and M. Woolrich, "Variational Bayesian inference for a nonlinear forward model," *IEEE Trans. Signal Process.*, vol. 57, no. 1, pp. 223–236, 2009.

[13] K. Chen, L. Xu, and H. Chi, "Improved learning algorithms for mixture of experts in multiclass classification," *Neural Netw.*, vol. 12, pp. 1229–1252, 1999.

[14] D. Chickering and D. Heckerman, "Efficient approximations for the marginal likelihood of Bayesian networks with hidden variables," *Mach. Learn.*, vol. 29, pp. 181–212, 1997.

[15] G. Consonni and J.-M. Marin, "Mean-field variational approximate Bayesian inference for latent variable models," *Computat. Statist. Data Anal.*, vol. 52, pp. 790–798, 2007.

[16] S. Escalera, D. Tax, O. Pujol, P. Radeva, and R. Duin, "Subclass problem-dependent design for error-correcting output codes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 6, 2008.

[17] Z. Gharamani and G. Hinton, "Variational learning for switching state-space models," *Neural Comput.*, vol. 12, no. 4, pp. 963–996, 2000.

[18] P. Hall, K. Humphreys, and D. Titterington, "On the adequacy of variational lower bound functions for likelihood-based inference Markovian models with missing values," *J. Royal Statist. Soc.: Series B (Statist. Method.)*, no. 3, pp. 549–564, 2002.

[19] R. Jenssen, D. Erdogmus, J. Principe, and T. Eltoft, "The Laplacian classifier," *IEEE Trans. Signal Process.*, vol. 55, no. 7, pp. 3262–3271, 2007.

[20] M. Jordan and R. Jacobs, "Hierarchical mixtures of experts and the EM algorithm," *Neural Comput.*, vol. 6, pp. 181–214, 1994.

[21] K. Katahira, K. Watanabe, and M. Okada, "Deterministic annealing variant of variational Bayes method," in *Proc. Int. Workshop Statist.-Mechan. Informat. 2007*, 2007, vol. 95, J. Phys.: Conf. Ser.

[22] D. Koller, U. Lerner, and D. Angelov, "A general algorithm for approximate inference and its application to hybrid Bayes nets," in *Proc. 15th Conf. Uncertainty in Artif. Intell. (UAI99)*, 1999.

[23] B. Krishnapuram, L. Carin, M. Figueiredo, and A. Hartemink, "Sparse multinomial logistic regression: Fast algorithms and generalization bounds," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 6, 2005.

[24] H. Langseth, T. Nielsen, R. Rumi, and A. Salmeron, "Inference in hybrid Bayesian networks," *Reliabil. Eng. Syst. Safety*, vol. 94, pp. 1499–1509, 2009.

[25] U. Lerner, "Hybrid Bayesian networks for reasoning about complex systems," Ph.D. Dissertation, Stanford Univ., Stanford, CA, 2002.

[26] S. Monti and G. Cooper, "Learning hybrid Bayesian networks from data," in *Learning in Graphical Models*, M. Jordan, Ed. Cambridge, MA: MIT Press, 2001.

[27] K. Murphy, "A variational approximation for Bayesian networks with discrete and continuous latent variables," in *Proc. 15th Conf. Uncertainty in Artif. Intell. (UAI99)*, 1999.

[28] I. Nabney, "Efficient training of RBF networks for classification," in *Proc. 9th Int. Conf. on Artificial Neural Netw.*, 1999.

[29] F. Peng, R. Jacobs, and M. Tanner, "Bayesian inference in mixtures-of-experts and hierarchical mixtures-of-experts models with an application to speech recognition," *J. Amer. Statist. Assoc.*, vol. 91, no. 435, pp. 953–960, 1996.

[30] H. Poor, *An Introduction to Signal Detection and Estimation*. New York: Springer, 1994.

[31] G. Rätsch, T. Onoda, and K.-R. Müller, "Soft margins for adaboost," *Mach. Learn.*, vol. 42, pp. 287–320, 2001.

[32] S. Roberts and W. Penny, "Variational Bayes for generalized autoregressive models," *IEEE Trans. Signal Process.*, vol. 50, no. 9, pp. 2245–2257, 2002.

[33] M. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, 2001.

[34] N. Ueda and Z. Gharamani, "Bayesian model search for mixture models based on optimizing variational bounds," *Neural Netw.*, vol. 15, pp. 1223–1241, 2002.

[35] B. Wang and D. Titterington, "Convergence properties of a general algorithm for calculating variational Bayesian estimates for a normal mixture model," *Bayesian Anal.*, no. 3, pp. 625–650, 2006.

[36] S. Waterhouse, D. MacKay, and T. Robinson, "Bayesian methods for mixture of experts," in *Advances in Neural Information Processing Systems*, D. Touretzky, M. Mozer, and M. Hasselmo, Eds. Cambridge, MA: The MIT Press, 1996, vol. 8, pp. 351–357.

[37] S. Waterhouse and A. Robinson, "Constructive algorithms for hierarchical mixtures of experts," in *Advances in Neural Information Processing Systems*, D. Touretzky, M. Mozer, and M. Hasselmo, Eds. Cambridge, MA: The MIT Press, 1996, vol. 8, pp. 584–590.

[38] M. Zhu and A. Martinez, "Subclass discriminant analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 8, pp. 1274–1286, 2006.

Nisar Ahmed (S'07) received the B.S.E. degree from Cooper Union in 2006 and the M.S. degree in mechanical engineering from Cornell University, Ithaca, NY, in 2009.

He is currently working toward the Ph.D. degree in the field of dynamics, systems, and controls at Cornell University. In 2007, he received a National Science Foundation Graduate Research Fellowship. His research interests include estimation, machine learning, control systems, and human-robotic interaction.

Mark Campbell (M'00) received the B.S. degree in mechanical engineering from Carnegie Mellon, Pittsburgh, PA, in 1990, and the M.S. and Ph.D. degrees in control and estimation from the Massachusetts Institute of Technology, Cambridge, in 1993 and 1996, respectively.

He is an Associate Professor of Mechanical and Aerospace Engineering with Cornell University, Ithaca, NY. His current research interests are in estimation and control of autonomous vehicles.

Dr. Campbell received a NASA award for the MACE experiment flown on STS-67, Best Paper awards from the AIAA and FIE, and teaching awards from Cornell University, University of Washington, and the ASEE. He is an Associate Fellow of the AIAA, Australian Research Council International Fellow, and Associate Director of the AACC board.