2008 American Control Conference
Westin Seattle Hotel, Seattle, Washington, USA
June 11-13, 2008

Multimodal Operator Decision Models

Nisar Ahmed and Mark Campbell
Sibley School of Mechanical and Aerospace Engineering
Cornell University, Ithaca, NY 14853, USA
{nra6,mc288}@cornell.edu

Abstract— **This paper develops the multimodal softmax (MMS) model, a probability distribution for multimodal discrete random variables with continuous conditioning random variables. MMS is motivated by the problem of learning multimodal probabilities for categorical human decisions in Bayes Net models of semi-autonomous systems. The MMS model is then derived vis-a-vis softmax and softmax mixture distribution models. MMS training is discussed in the context of maximum likelihood estimation. Finally, decision classification results using experimental data from Cornell's RoboFlag human-robotic interaction testbed are presented.**

I. INTRODUCTION

Joint human-autonomous vehicle teams are envisioned for many applications, such as search and rescue, forest fire-fighting, planetary exploration, agriculture, and security. Human operators are unique elements of these teams because they provide critical strategic input under many uncertainties.

Probabilistic models of these teams can be used to improve coordination between multiple robots and humans. Recent research in human-robotic interaction has fused methods from cognitive engineering [6], control systems and estimation theory [13]. Ref. [3] proposed human decision models via Bayesian Networks/Dynamic Belief Networks (BNs/DBNs); these are combined with vehicle models to form a comprehensive joint human-vehicle modeling methodology [1]. This approach models humans and autonomous vehicles in a unified probabilistic framework, thus naturally connecting human decision-making with control and estimation theory. Accurate probabilistic human decision models will enable both accurate prediction of operator decisions given sensor measurements and vehicle trajectories, and accurate estimation of hidden state variables via known operator decisions.

In the BN/DBN framework, categorical human decisions (e.g. goals, strategies, robot type selections, rooms to search, etc.) are modeled as discrete random variables. The 'parent random variables' influencing them can be a mix of discrete (e.g. target identity) and continuous (e.g. command trajectory) variables. 'Discrete parent-discrete child' dependencies are easily modeled with probability tables [7]. However, 'continuous parent-discrete child' dependencies pose certain difficulties [2], [7], [9]. While softmax and mixture of softmax models are tractable for BN inference [9], [12], they either fail to learn multimodalities in training data or are inefficient at learning such features.

This work was supported by AFOSR, contract number FA9550-05-1-0118.

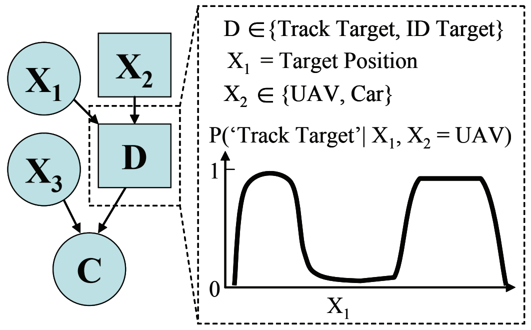

Fig. 1. BN model of an operator tasking a pair of robot vehicles. The inserted plot shows a multimodal probability distribution for the likelihood that the operator will task the UAV to track a target given its current position.

The multimodal softmax (MMS) model proposed here is a general probability distribution for discrete random variables with continuous parents. It will be shown how the MMS model emerges logically from the softmax model, and that the MMS model naturally describes multimodal discrete random variables with continuous parent variables. Finally, applications of the MMS model will be given in the context of operator decision prediction/classification for a joint human-robotic team experiment.

II. PROBABILISTIC HUMAN DECISION MODELS

A. Bayesian Networks

Bayesian Networks (BNs) are intuitive represention of probability distributions as directed acyclic graphs [2]. Complex joint distributions (including discrete and continuous random variable mixtures) are easily found through the conditional dependencies entailed by the directed graph structures [7]. Figure 1 shows a simple BN model for an operator tasking a pair of robot vehicles (one car, one UAV). Square and circular nodes represent discrete and continuous random variables, respectively. Arrows connecting nodes represent the direction of probabilistic 'influences' between variables. The $X_{(.)}$ represent parent variables for the discrete and continuous operator decision variables, D and C. X_1 is the position of a local target, while X_2 and X_3 are the tasked vehicle type and the target bearing, respectively. The discrete decision variable D is the tasked vehicle's operator-assigned strategic function mode, and C is the vehicle's operator-assigned continuous waypoint.

Realizations of D probabilistically depend on X_1 and X_2, while realizations of C depend on X_3 and D. This entails conditional distributions $P(D|X_1, X_2)$ and $f(C|D, X_3)$, as well as 'prior' distributions $f(X_1)$, $P(X_2)$, and $f(X_3)$[1]. The product of these conditional distributions gives the BN's joint probability distribution [2],

$$f(Joint) = f(X_1)P(X_2)f(X_3)P(D|X_1,X_2)f(C|D,X_3). \quad (1)$$

Probability distributions for subsets of variables are found via the joint distribution. However, finding 'accurate' yet 'tractable' conditional distributions for BNs with mixed discrete and continuous variables is challenging [8]. Complex distribution functions are difficult to use for BN inferencing; simpler distribution functions may be very inaccurate.

As there is no universal method for finding probabilities in BNs [7], the problem addressed here is batch estimation of the conditional distribution $P(D|X)$ for a discrete decision D with a q-dimensional parent variable set X, using complete labeled parent variable data. It assumed without loss of generality that all elements of X are continuous, where X has been identified within some static (i.e. time-independent) graph[2]. While finding such conditional dependencies is a complex problem in itself, several BN structural learning methods could be used [7], [8], [11]. Furthermore, D's outcomes are assumed mutually exclusive and countably finite. In general, D's outcomes may each have several modes of high probability in the X space (see Fig. 1).

BN models for $P(D|X)$ often come from statistical classification theory and are fit via supervised learning. Some popular methods include tabulation via discretization of X [7], [10], softmax modeling [8], and discriminative mixture modeling [5], [9]. The curse of dimensionality makes discretization difficult for q>2, as many data and parameters are required to model the continuous X space adequately. While softmax and 'mixture of softmax' distributions provide tractable continuous models for BN implementation [9], [12], their accuracy is limited since they do not cope easily with multimodalities in the X space.

B. Softmax Models

Suppose D has m discrete outputs d_j, $j \in \{1,...,m\}$. $P(D|X)$ can be modeled as a softmax distribution, which acts as a thresholding function for the m outcomes over the X space [3], [5]. The softmax distribution is defined as

$$P(D = d_j|X = x) = \frac{e^{\tilde{w}_j^T \tilde{x}}}{\sum_{h=1}^{m} e^{\tilde{w}_h^T \tilde{x}}}. \quad (2)$$

where $\tilde{w}_j = \frac{1}{\sigma_j}[w_j, -b_j]^T$ and $\tilde{x} = [x, 1]^T$. For the j^{th} decision variable, the vector w_j ($\|w_j\| = 1$) represents relative weightings on the q-dimensional parent variables, while the scalar σ_j represents a 'threshold' steepness, and scalar b_j represents a bias from zero. This gives $m(q+1)$ free parameters (accounting for each w_j's norm constraint).

[1]$f(.)$ is a continuous probability density function; $P(.)$ is a probability mass function for a discrete random variable

[2]Adding time dependency between nodes yields a DBN, which are basically BNs repeated over discrete time slices [8].

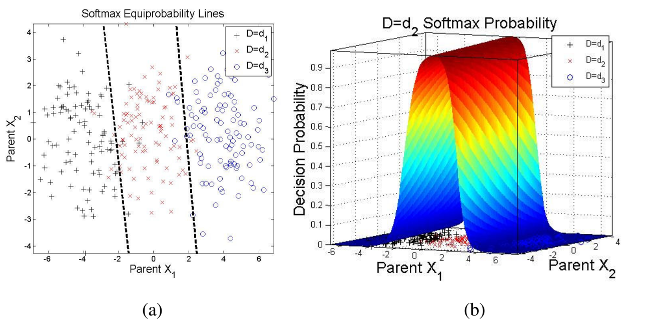

Fig. 2. Softmax distribution estimation example with synthetic data. (a) Data for $m = 3$ decisions conditioned on two parent variables, showing lines of equiprobability. (b) Probability surface for $D = d_2$.

The weights $\tilde{w}_j$ can be learned from labeled decision data using maximum likelihood estimation (MLE) [3]. Suppose n labeled i.i.d. training data points t_i are given such that $t_i = [d_i, x_i]^T$, $i = \{1,...,n\}$, where d_i represents the one-of-m decision label corresponding to a parent variable data point x_i. Let the training set be partitioned according to d_i so that there are n_j training data points for each $D = d_j$, $j \in \{1,...,m\}$. Then, the softmax likelihood $\mathcal{L}$ is defined to be

$$\mathcal{L} = \prod_{j=1}^{m} \left[\prod_{k_j}^{n_j} P(D = d_j|X = x_{k_j}) \right]. \quad (3)$$

The weights maximizing Eq. 3 can be found via numerical optimization using the Jacobian and Hessian of $\mathcal{L}$ with respect to the $\tilde{w}_j$ [3], [5]. $\mathcal{L}$ is convex for softmax, so it has a unique, globally optimum set of $\tilde{w}_j$ [5].

Fig. 2 illustrates softmax fitting results for a synthetic set of data with three discrete decisions conditioned on two parent variables (i.e. q=2), shown in Fig. 2(a). The resulting probability decision surface for $D = d_2$ using softmax is shown in Fig. 2(b). Fig. 2(a) also shows the resulting equiprobability lines, which are defined through the log-odds ratio [5] as

$$\log \frac{P(D = d_j|X = x)}{P(D = d_g|X = x)} = (\tilde{w}_j - \tilde{w}_g)^T x = 0. \quad (4)$$

These lines denote points in the parent variable space where decisions $D = d_j$ and $D = d_g$ have equal probabilities. Note that these lines are hyperplanes for q>2, and can be interpreted as probabilistic boundaries between two decisions.

Qualitatively, the softmax distribution yields good probabilistic decision boundaries between discrete decisions in the X space if the training data for each d_j has a single 'cluster' that is well-separated from other decisions by at most $m-1$ hyperplanes [3]. Fig. 3(a) shows data for another synthetic set of three discrete decisions with two parent variables that are not modeled well by the softmax distribution. The challenge here is the non-convex grouping of $D = d_1$ data sitting between the disjoint clusters of $D = d_2$ data (also non-convex). As Fig.3(b) shows, softmax provides thresholds for the $D = d_2$ data using one weight set, leading to fuzzy decision probability estimates.

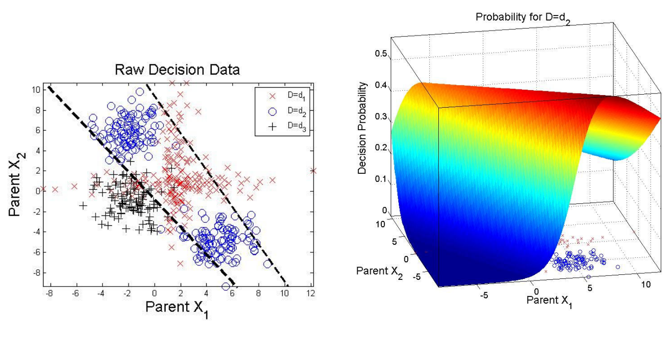

Fig. 3. Example where softmax does yields inaccurate probabilites. (a) Synthetic multimodal data with $m = 3$ decisions and two parent variables, with lines of equiprobability. (b) Softmax probability surface for $D = d_2$.

C. Mixture of Softmax Models

The mixture of softmax model [5], [9] is a weighted sum of N softmax distributions, used for modeling multimodalities. The distribution is written as

$$P(D = d_j|X = x) = \sum_{l=1}^{N} \pi_l \cdot \frac{e^{\tilde{w}_{jl}^T \tilde{x}}}{\sum\limits_{h=1}^{m} e^{\tilde{w}_{hl}^T \tilde{x}}} \tag{5}$$

where π_l represents the mixing weight of the l^{th} softmax model ($\sum\limits_l \pi_l = 1$). The expectation-maximization (EM) algorithm is used to train this model [5]. This approach uses an outer optimization loop to 'associate' the training data with the N softmax components, and N inner softmax optimization loops to learn each set of softmax weights (N is fixed in advance by the user). Training is trickier, since the resulting set of softmax parameters is not always 'minimal' for the desired distribution (the total number of parameters is $N(m[q + 1] + 1)$). Fig. 4 shows results of training an N=5 softmax mixture distribution on the data in Fig. 3(a). This results in 50 model parameters (compared with 9 for the simple softmax model). Clearly, some weights are redundant; the softmax thresholds overlap as they try to predict the outcomes of D. This 'inefficiency' dilutes the overall model's precision: good predictions in some regions are averaged together with poor ones. Two well-known improvements to basic softmax mixtures are mixture of experts and hierarchical mixture of experts [5]. The mixture of experts approach varies π_l weights over the parent variable space, while the second approach is a mixture of mixtures of experts. However, all softmax components still maintain a full weight set for all m decisions, so the training processes and resulting models remain 'inefficient'.

III. Multimodal Softmax Model

The multimodal softmax model is a 'divide and conquer' strategy: if an entire data set of (non-convex) multimodal decisions can be divided into smaller 'convex' clusters for all decisions, then a single softmax distribution can be used to model these convex clusters as a set of well-separated 'sub-decisions'. Then, this softmax distribution's outputs can

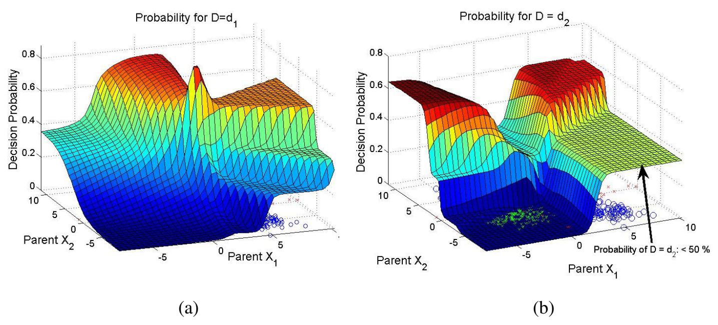

Fig. 4. Mixture of Softmax model with $N = 5$ applied to data set in Fig. 3. Decision probability surfaces for (a) $D = d_1$ and (b) $D = d_2$. Note the poor fit in (b) for the d_2 region in the lower right corner. The local minimum log-likelihood (obtained here via the EM algorithm) is -24,768.81 .

be combined into a multimodal decision distribution for the original (non-convex) decision data.

More formally, suppose that each original decision $D = d_j$ has s_j sub-decisions, for a total of $\sum_{\mathrm{j=1}}^{\mathrm{m}} s_j = s$ sub-decisions, for $m \leq s$. Let d_{jl} denote d_j's l^{th} sub-decision, $l = 1, ..., s_j$. Let D_s be a discrete random variable, conditioned on X, that selects $r \in S = \{d_{11}, ..., d_{1s_1}, d_{21}, ..., d_{ms_m}\}$, where S is the set of all s sub-decisions. Each sub-decision d_{jl} in S is a convex cluster of decision data, so d_{jl} can be defined as the output of a softmax distribution $P(D_s|X)$, if the following substitutions are made into Eq.2: s for m, D_s for D, and index jl for j. The s_j sub-decision softmax probabilities (defined for each d_{jl} through $P(D_s|X)$) for each original decision d_j are then added together to yield the desired conditional probability distribution, $P(D = d_j|X = x)$. This is done via the theorem of total probability [2],

$$P(D = d_j|X) = \sum_{r \in \{S\}} P(D = d_j|D_s = r) \cdot P(D_s = r|X). \tag{6}$$

Here $P(D = d_j|D_s = r) = 1$ only if $r = d_{jl}$ for $l = \{1, ..., s_j\}$; otherwise $P(D = d_j|D_s = r) = 0$. Since $P(D_s = r|X = x)$ is a softmax distribution, this gives

$$P(D = d_j|X = x) = \sum_{l=1}^{s_j} P(D_s = d_{jl}|X = x) = \frac{\sum\limits_{l=1}^{s_j} e^{\tilde{w}_{jl}^T \tilde{x}}}{\sum\limits_{h=1}^{s} e^{\tilde{w}_{h}^T \tilde{x}}}, \tag{7}$$

where h indexes all s sub-decisions in S. Note that for s total sub-decisions, there are s total softmax weights $\tilde{w}_{jl}$; each original decision d_j has a corresponding set of s_j softmax weights. Thus, given $X = x$, Eq. 7 adds the relevant sub-decision softmax probabilities for $D = d_j$, as desired. Eq. 7 defines the MMS distribution probability for $D = d_j$. Figure 5(a) shows the BN representation used for the resulting MMS model. The continuous node X represents the continuous parent variables, D_s is a hidden (i.e. unobserved) softmax distribution node with s discrete outputs, and the discrete node D represents the original set of m decisions, now conditioned on D_s. Note that the MMS distribution has $s(q+1)$ parameters (whereas the mixture of softmax model has $N(m[q + 1] + 1)$ parameters).

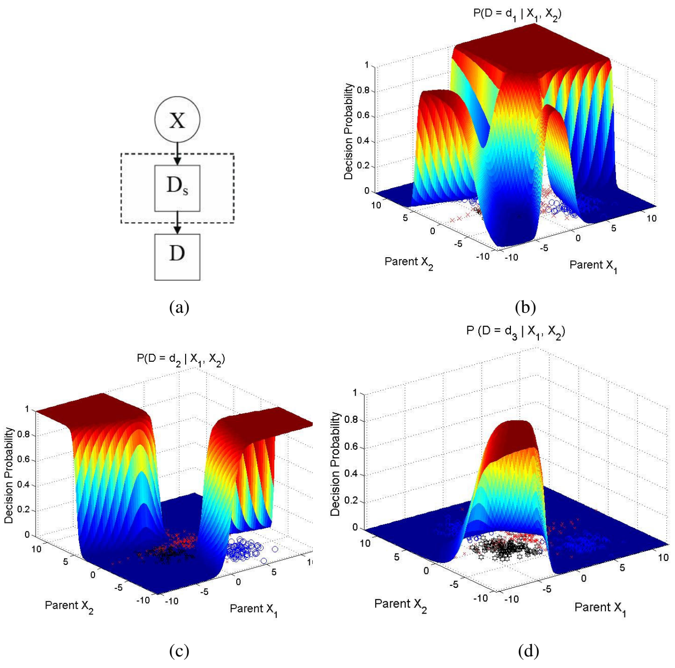

Fig. 5. (a) Conceptual BN for the MMS model, where D_s is a hidden softmax node. (b)-(d) MMS surfaces for Fig. 3(a) data are for $D = d_1$, $D = d_2$, and $D = d_3$, respectively. The log-likelihood here is -151.99.

For instance, in Fig. 3(a), one may specify $s = 7$ clusters of sub-decisions over the $m = 3$ original decisions as follows: $s_1 = 4$ for d_1 (i.e., one sub-decision for each arm of the red cross), $s_2 = 2$ for d_2 (one sub-decision for each disjoint blue cluster), and $s_3 = 1$ for d_3 (one sub-decision for the one black cluster). These $s = 7$ sub-decisions are modeled probabilistically by assuming that each sub-decision is a 'separable' outcome of single discrete decision random variable D_s, which (when conditioned on X_1 and X_2) selects an element from the set of sub-decisions $S = \{d_{11}, ..., d_{14}, d_{21}, d_{22}, d_{31}\}$. This allows the sub-decisions in S to be modeled with a softmax distribution, where each of the s weights has $q+1$ parameters. Since D_s only chooses one sub-decision from S at a time and each of the s sub-decisions descends from one of the original $m = 3$ decisions, it follows that $D = d_j$ any time $X = x$ gives $D_s = d_{jl}$. For instance, in the case of Fig. 3(a), if it is given that $D_s = d_{21}$ when $X = x$, then $D = d_2$ must occur with probability 1. Figure 5(b)-(d) show the resulting probablity surfaces fitted via an MLE procedure (described next). Clearly, these MMS decision probability estimates (using 21 total parameters) are sharper than the softmax mixture probabilities in Fig. 4 (which uses 50 parameters for N=5), since 'parameter overlap' is greatly reduced in the MMS model.

A. ML Estimation of MMS Weights

To optimize the softmax weights in Eq. 7, the sub-decision number s_j must be given for each d_j. If the softmax likelihood (Eq.3) for $P(D_s|X)$ were maximized, then the decision label d_i in t_i would be replaced with the sub-decision label d_{jl}, and the user would have to manually partition the data among the s sub-decisions while tracking the correlations between sub-decisions d_{jl} and original decisions d_j.

Fortunately, such explicit repartitioning of the data is unnecessary. As long as only the values of all s_j is specified, the 'optimum' data partition can be found automatically by directly maximizing the likelihood of the *originally labeled* training data (i.e. $t_i = [d_i, x_i]^T$) over the s softmax sub-decision weights using $P(D|X)$ in Eq. 7.

Substituting Eq. 7 into Eq. 3 for $P(D|X)$, the MMS likelihood for the original labeled training data is

$$\mathcal{L}_{MMS} = \prod_{j=1}^{m} \left[\prod_{k_j=1}^{n_j} \left(\frac{\sum_{l=1}^{s_j} e^{\tilde{w}_{jl}^T \tilde{x}_{k_j}}}{\sum_{h=1}^{s} e^{\tilde{w}_h^T \tilde{x}_{k_j}}} \right) \right], \tag{8}$$

where h indexes all s sub-decisions and l indexes only the s_j subdecisions for d_j in D_s. It is more convenient to maximize the log-likelihood,

$$\log(\mathcal{L}_{MMS}) = \sum_{j=1}^{m} \sum_{k_j=1}^{n_j} \log \left[\sum_{l=1}^{s_j} e^{\tilde{w}_{jl}^T \tilde{x}_{k_j}} \right] - \sum_{k=1}^{n} \log \left[\sum_{h=1}^{s} e^{\tilde{w}_h^T \tilde{x}_k} \right]. \tag{9}$$

A gradient-based or quasi-Newton optimization algorithm can be implemented to find the optimal sub-decision weights, which are $q+1$ dimensional. The column of the Jacobian corresponding to $\frac{\partial \log(\mathcal{L}_{MMS})}{\partial \tilde{w}_{jl}}$ is given by

$$\nabla_{\tilde{w}_{jl}} \log(\mathcal{L}_{MMS}) = \sum_{k_j=1}^{n_j} \tilde{x}_{k_j} \frac{e^{\tilde{w}_{jl}^T \tilde{x}_{k_j}}}{\sum_{l=1}^{s_j} e^{\tilde{w}_l^T \tilde{x}_{k_j}}} - \sum_{k=1}^{n} \tilde{x}_k \frac{e^{\tilde{w}_{jl}^T \tilde{x}_k}}{\sum_{h=1}^{s} e^{\tilde{w}_h^T \tilde{x}_k}}. \tag{10}$$

The Hessian is a $s(q+1) \times s(q+1)$ symmetric matrix. Let $H_{w_{jl}, w_{pg}}$ denote the $q+1 \times q+1$ block that is obtained by differentiating the $\frac{\partial \log(\mathcal{L}_{MMS})}{\partial \tilde{w}_{jl}}$ column of the Jacobian with respect to $\tilde{w}_{pg}$, which is the weight corresponding to the g^{th} sub-decision of decision $D = d_p$. Then,

$$H_{w_{jl}, w_{pg}} = \begin{cases} \sum_{k_j=1}^{n_j} \tilde{x}_{k_j} \tilde{x}_{k_j}^T \rho_{k_j}^{jl} - \sum_{k=1}^{n} \tilde{x}_k \tilde{x}_k^T \xi_k^{jl} & (jl = pg) \\ - \sum_{k_j=1}^{n_j} \tilde{x}_{k_j} \tilde{x}_{k_j}^T \alpha_{k_j}^{jl,pg} + \sum_{k=1}^{n} \tilde{x}_k \tilde{x}_k^T \beta_k^{jl,pg} & (jl \neq pg) \end{cases} \tag{11}$$

where

$$\rho_{k_j}^{jl} = \frac{\sum_{l=1, l \neq jl}^{s_j} e^{(\tilde{w}_{jl} + \tilde{w}_{jl})^T \tilde{x}_{k_j}}}{\left[\sum_{l=1}^{s_j} e^{\tilde{w}_l^T \tilde{x}_{k_j}} \right]^2}, \xi_k^{jl} = \frac{e^{\tilde{w}_{jl}^T \tilde{x}_{k_j}} + \sum_{h=1, h \neq jl}^{s} e^{(\tilde{w}_h + \tilde{w}_{jl})^T \tilde{x}_k}}{\left[\sum_{h=1}^{s} e^{\tilde{w}_h^T \tilde{x}_k} \right]^2}$$

$$\beta_k^{jl,pg} = \frac{e^{(\tilde{w}_{jl} + \tilde{w}_{pg})^T \tilde{x}_k}}{\left[\sum_{h=1}^{s} e^{\tilde{w}_h^T \tilde{x}_k} \right]^2}, \alpha_{k_j}^{jl,pg} = \frac{e^{(\tilde{w}_{jl} + \tilde{w}_{pg})^T \tilde{x}_{k_j}}}{\left[\sum_{l=1}^{s_j} e^{\tilde{w}_{jl}^T \tilde{x}_{k_j}} \right]^2} \cdot \delta_{jg}. \tag{12}$$

Note that δ_{jg} is the Kronecker delta function on j and g.

Unlike the softmax likelihood (Eq.3), $\mathcal{L}_{MMS}$ is not guaranteed to have a global maximum for $s_j > 1$, due to symmetry of the sub-decision weights $\tilde{w}_{jl}$ for $D = d_j$ (if all $s_j = 1$, then the softmax model is obtained and a global maximum exists). While $\mathcal{L}_{MMS}$ may have saddle points, it never has minima since the Hessian is never positive definite.

TABLE I

FIGURE 6 WEIGHTS AND OPTIMUM LOG-LIKELIHOOD.

$[s_1, s_2, s_3]$	Number of Weights	$log\,(\mathcal{L}_{MMS})$
$[2, 1, 1]$	12	-1176.4647
$[2, 1, 2]$	15	-719.6104
$[2, 2, 2]$	18	-185.3046
$[2, 5, 2]$	27	-180.5428
$[3, 3, 3]$	27	-180.3805

B. *Choosing s_j and Preventing ML Overfit*

Since it may be impossible to visually obtain s_j (as done above) when q is large, clustering methods can be used to give initial guesses for s_j. Taking data only for $D = d_j$, k-means clustering [5] can be used to find the best number of sub-decision clusters in the X space. In particular, some software packages enable visualization of the 'goodness of fit' of data to proposed k_j clusters (e.g. silhouette plots in Matlab's Statistics Toolbox). This helps determine if significant (possibly disjoint) decision modes exist for d_j (e.g. in terms of Mahalanobis distance 'membership sizes'), and the approximate number of sub-decisions s_j needed to discriminate d_j from among the m decisions in the X space (e.g. using centroid distances between all proposed s sub-decision clusters). Note, a similar approach is often used to intialize components in Gaussian mixture modeling [5].

It seems worthwhile to numerically explore s_j to find the largest $\mathcal{L}_{MMS}$. However, as with all MLE methods, there is great risk of overfitting to the data. Table I lists the number of weights and optimum log-likelihood values for various choices of $[s_1, s_2, s_3]$ using the synthetic data set shown in Fig. 6(a). The decision surfaces for $D = d_2$ are shown in Fig. 6(b)-(d). Table I shows that setting $[s_1, s_2, s_3]$ to $[2, 5, 2]$ or $[3, 3, 3]$ results in overfit; the log-likelihood does not improve much from $[2, 2, 2]$. Fig. 6(d) shows that excess s_j sub-decisions in the $[2, 5, 2]$ case are fit to outliers with very sharp thresholds. Note that the 'optimum' $\mathcal{L}_{MMS}$ always increases as s_j increases. Hence, it is recommended to initialize s_j through clustering. Then, each s_j can be increased (or decreased) until the 'optimum' log-likelihood stops (starts) changing significantly at convergence.

IV. EXPERIMENTAL RESULTS

The MMS model was compared with the softmax mixture model and the nonparametric Parzen classifier [14] in a classification task using real human operator decision data. The data were collected from human-robotic studies on Cornell's RoboFlag testbed [1], in which humans tasked a team of robotic vehicles in a reconaissance mission with an 'enemy' team of robot agents. The mission objective was to locate and identify the enemy team's stationary target vehicles without getting tagged by the enemy chaser vehicle or colliding with the targets. The human's team had two fast-moving search vehicles that located targets, and a slower ID vehicle that identified targets. Tagged vehicles returned to a home base before returning into play. Operator tasking decisions and simulation state information were recorded. The method of [4] was used to find parent variables X for paired sets of discrete operator decisions D. See [3], [4] for a complete description of the RoboFlag experiments.

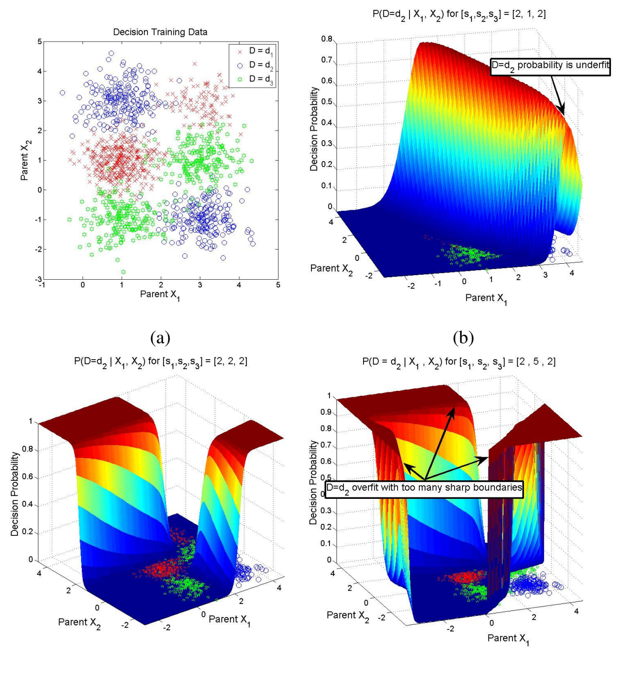

Fig. 6. Effects of varying s_j on the MMS probability estimates for the decision data in (a). MMS surfaces for $D = d_2$ (blue circles in (a)) are shown for $[s_1, s_2, s_3] =$ (b) $[2, 1, 2]$,(c) $[2, 2, 2]$, and (d) $[2, 5, 2]$.

TABLE II

10-TRIAL CLASSIFIER HOLDOUT ERROR STATISTICS.

Data Set	Model	mean % error	std.dev.	max error
'ID/Loc'	Parzen	16.83	1.24	19.34
vs.	Softmax Mix	23.96	8.61	42.57
'Evasive'	MMS	17.07	0.83	18.11
'Decoy'	Parzen	4.02	0.37	4.72
vs.	Softmax Mix	21.17	32.24	93.16
'Strategic'	MMS	3.84	0.28	4.23

Each data set for the paired decisions was randomly split into a training set and a holdout set 10 different times for the three classifier models. MMS and softmax mixture classifiers were trained with the aforementioned procedures, while Parzen classifiers were trained to minimize the leave-one-out cross-validated Bayes' error (with flat priors) [14]. Holdout classification error statistics (mean percent error, standard deviation, maximum percent error) for each classifier were then obtained.

Fig. 7(a) shows data for two high-level, strategic operator tasking decisions on all three friendly vehicles: D='ID/Localize' (move vehicle near enemy target to obtain identity/location) and D='Evasive Maneuver' (change trajectory to avoid tag or collision). The parent variables are the tasked vehicle's range to chaser (X_1) and the vehicle's range to the closest target (X_2). Fig. 7(b) shows the MMS probability surface for D='Evasive Maneuver', with $s_1 = 1$ for D='ID/Localize'and $s_2 = 2$ for 'Evasive Maneuver'. The MMS model clearly captures the increased likelihood of 'Evasive Maneuver' around the two modes that are correlated to the tasked vehicle being either too close to either an enemy target or the chaser. The X_1 and X_2 values for the decision surface's 'corner' in Fig. 7(b) can be interpreted as the average minimum distances that the operators kept

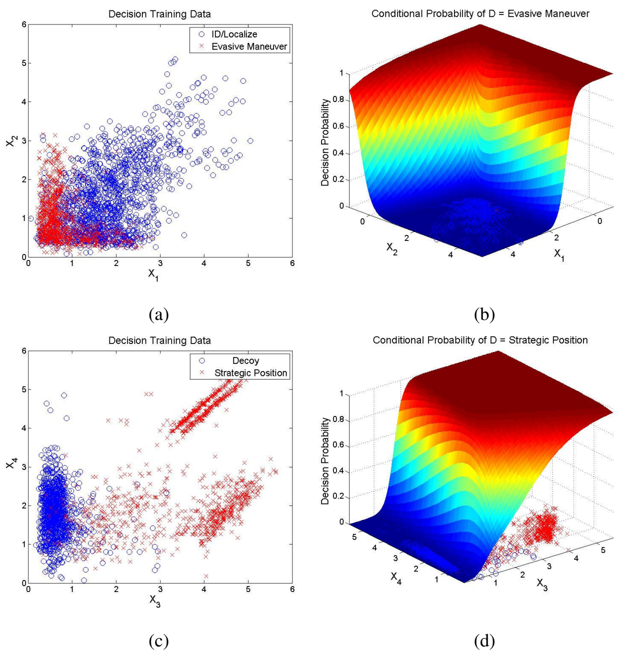

Fig. 7. (a) Data for D='Evasive Maneuver' and D='ID/Localize'. (b) MMS surface for D='Evasive Maneuver'. (c) Data for D='Decoy' and D='Strategic Position' . (d) MMS surface for D='Strategic Position'.

from each type of enemy vehicle while trying to identify or localize enemy targets. Fig. 7(c) shows data for a pair of operator decisions on the search vehicles: D='Decoy' (lure chaser away from rest of team) and D='Strategic Position' (place vehicle in useful position until needed later). The parent variables X_3 and X_4 are, respectively, the tasked vehicle's range to the chaser and the ID vehicle's waypoint distance to the chaser (i.e. the distance from the chaser to the field waypoint assigned by the operator to the ID vehicle). X_4 is relevant since the chaser can get a lead on the slower ID vehicle and intercept it en route to the waypoint. Fig. 7(d) shows the MMS probability surface for D='Strategic Position', with $s_1 = 2$ for D='Decoy'and $s_2 = 1$ for 'Strategic Position'. The soft decision boundary parallel to the X_3 axis and the sharper decision boundary parallel to the X_4 axis indicate how strongly the decisions depend on the chaser's distance to the ID and search vehicles[3].

Table II shows holdout error statistics for these data sets using Parzen, softmax mixture (N=2 for both cases) and MMS (with the above s_j). For both decision data sets, 500 points from each of the two decision classes were used to train each classifier. For the 'ID/Localize' vs. 'Evasive Maneuver' trials, 989 'ID/Localize' points and 585 'Evasive Maneuver' points were used for validation. For the 'Decoy' vs. 'Strategic Position' trials, 670 'Decoy' points and 631 'Strategic Position' points were used for validation. The results show that the MMS model achieves good accuracy and consistency with only 9 parameters for both decision sets; the Parzen classifier only achieves about the same performance by storing all training data points, so that it effectively carries over 1000 parameters. The softmax mixture exhibits higher error rates and variance on both decision sets (with 14 parameters) than either MMS or Parzen. Note that the results for the 'Decoy' vs. 'Strategic Position' trials contain three substantial outlier errors (e.g. the maximum at 93%). Without these, the mean softmax mixture holdout error is $5.33\% \pm 1.1\%$ (7% max); MMS still outperforms this.

[3]Note: all distances are in meters.

V. Conclusions and Future Work

The MMS distribution was developed for multimodal operator decision modeling. The MMS model can be easily fit to multimodal data to produce crisp conditional probability estimates; clustering methods can aid the MLE fitting process. Results from experimental decision data showed that MMS can outperform standard probabilistic models in multimodal classification while using a sparser weight set.

MMS models are clearly suitable for many classification problems. Future work includes further theoretical investigation of MMS and the adaptation of a Bayesian fitting procedure (akin to a method of softmax training where priors are placed on model weights [5]). Since the simplicity of MMS makes it an attractive model to use in BNs/DBNs, inference in larger scale hybrid BNs will be studied using variational [12] and Monte Carlo techniques [8].

VI. Acknowledgments

The authors gratefully acknowledge the contribution of the Air Force Office of Scientific Research (AFOSR) and Dr. Scott Galster of the Air Force Research Laboratory (AFRL).

References

[1] M. Campbell, F. Bourgault, S. Galster, and D. Schneider, "Toward Probabilistic Operator-Multiple Robot Decision Models", in *Proc. ICRA*, 2007.
[2] R. Neapolitan, "Learning Bayesian Networks", Prentice-Hall, Upper Saddle River, NJ; 2003.
[3] M. Campbell, S. Sukkarieh, and A. Goktogan, 'Operator Decision Modeling in Cooperative UAV Systems", in *Proc. GNC*, Keystone, CO, August 2006.
[4] F. Bourgault, N. Ahmed, D. Shah, and M. Campbell, "Probabilistic Operator-Multiple Robot Modeling Using Bayesian Network Representation", in *Proc. GNC*, Hilton Head, SC, August 2007.
[5] C. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York: 2006.
[6] R. Parasuraman, T. Sheridan, and C. Wickens, "A Model for Types and Levels of Human Interaction with Automation," *IEEE Trans. on Man, Sys. and Cyb.*, vol. 30, no.3, May 2000.
[7] F. Jensen, *Bayesian Networks and Decision Graphs*, Springer Verlag, New York: 2001.
[8] K. Murphy, "Dynamic Bayesian Networks: Representation, Inference, and Learning," Ph.D. Dissertation, UC Berkeley, 2002.
[9] D. Koller, U. Lerner, and D. Angelov, "A General Algorithm for Approximate Inference and Its Application to Hybrid Bayes Nets," in *Proc. UAI*, 1999.
[10] M. Neil, M. Tailor, and D. Marquez, "Inference in Hybrid Bayesian Networks Using Dynamic Discretization," Statistics and Computing vol.3, no.17, September 2007.
[11] S. Monti and G. Cooper, " Learning Hybrid Bayesian Networks from Data," in *Learning in Graphical Models*, MIT Press, Cambridge, MA; 2001.
[12] K. Murphy, A Variational Approximation for Bayesian Networks with Discrete and Continuous Latent Variables , in *Proc. UAI*, 1999.
[13] T. Kaupp, A. Makarenko, S. Kumar, B. Upcroft, and S. Williams,"Operators as Information Sources in Sensor Networks", in *Proc. IROS*, 2005.
[14] G. McLachlan, *Discriminant Analysis and Statistical Pattern Recognition*, Wiley, Brisbane: 1992.