We describe ongoing work toward using human collaborators as providers of ‘soft sensor’ data, which can be formally combined with conventional ‘hard sensor’ data to augment robotic state estimation and model learning. Formal integration of robotic and human sensing can greatly improve the robustness of autonomous perception and decision making, especially in unstructured environments where uncertainties cannot be well-characterized in advance and must be modeled/adapted on the fly. This is particularly important for unmanned aerial system (UAS) applications such as large-scale surveillance [1], [2] and wilderness search and rescue [3], in which autonomous vehicles are subject to size, weight and power constraints that restrict onboard sensing, processing and communication abilities. Soft data integration in multi-robot teams can also allow human teammates to stay ‘in the loop’ without cognitively overloading them or undermining robotic autonomy: rather than exert high mental effort to provide direct commands, a human sensor can perform the cognitively easier task of providing helpful observations, which can be consumed at each robot’s discretion [4].

Soft data can be broadly related to either ‘abstract’ phenomena that cannot be measured by robotic sensors (e.g. labels for object categories and behaviors) or measurable dynamical physical states that must be monitored constantly (object position, velocity, attitude, temperature, size, mass, etc.) [5]. This work focuses on the latter, under the key assumption that humans are not oracles: as with any other sensor data, human observations are subject to errors, limitations and ambiguities that must be handled properly. As such, we aim to adapt widely used statistical sensor fusion and robotic state estimation algorithms, e.g. Kalman filters and the like, so that soft data can be exploited with minimal effort on the part of the robot or the human sensor. Refs. [6], [7], [8] were among the first to develop Bayesian fusion techniques allowing human sensors to directly ‘plug into’ robotic state estimation and perception algorithms. However, these works assume that humans report data in the same way robots do, and thus greatly limit the flexibility of human-robot communication. In the context of target tracking with extended Kalman filters, for instance, [6] assumes that humans provide numerical range and bearing measurement reports (‘The target is at range 10 m, bearing 45 degrees’).

Ref. [9] showed how to model and fuse flexible semantic natural language soft data to provide a much broader range of positive/negative information for Bayesian state estimation, e.g. ‘The target is parked near the tree in front of you’, ‘Nothing is next to the truck heading North’. One nice theoretical property of the resulting fusion algorithm is its ability to directly plug into Gaussian mixture filters for robotic state estimation, which can accurately represent complex posterior pdfs while avoiding the curse of dimensionality encountered by grid or particle filter methods [10], [11], [6]. However, this approach requires modeling and calibration of semantic soft data via hybrid (continuous-to-discrete) generalized softmax likelihood functions. While such likelihood functions can be theoretically constructed to model semantic descriptions in 2D/3D spatial domains and beyond (to include velocities, headings, etc.), it is unclear how they can be practically constructed, learned or modified in general dynamic state spaces. This issues is especially relevant if we wish to adapt ‘plug and play’ Bayesian semantic soft data fusion to unstructured settings (e.g. [2]), or translate semantic sensor models between different domains by exploiting ‘geometries of meaning’ [12].

To this end, we present novel strategies for learning and manipulating softmax likelihoods in arbitrary state spaces. Our main contribution is a solution to the ‘softmax synthesis’ problem: given a set of semantic labels corresponding to polytopes in a complete mutually exclusive convex decomposition of some state space, find the parameters for the softmax model(s) whose probabilistic class boundaries yield the specified polytopes. We derive a formal approach to correctly ‘reverse engineer’ the close geometric relationship between softmax parameters and class log-odds boundaries, which determine semantic regions of dominance in the state space. Thus, structural model knowledge from ‘semantic first principles’ (expected symmetries, scale invariances, contextual bindings, etc.) can be easily embedded within deformable softmax ‘templates’, using simple linear operations and no training data. Such templates are useful for model reduction and calibration/learning with sparse training data, and lead to major performance gains for maximum likelihood/MAP learning [13], [14]. By manipulating softmax parameters on the fly, semantic likelihoods can be updated to reflect information changes due to dynamic contextual shifts or environmental uncertainties (e.g. changes in reference grounding dimensions/scales). Finally, detailed cognitive ‘forward semantic models’ (e.g. [15], [16], [17], [18], [19], [20]) can be quickly approximated by flexible piecewise-convex likelihoods that avoid state space discretization and are easier to ‘invert’ for fast scalable online inference vis-a-vis [9].

Softmax Modeling of Piecewise Semantics in Arbitrary State Spaces for ‘Plug and Play’ Human-Robot Sensor Fusion

Nisar Ahmed and Nicholas Sweet

Authors are with the Research and Engineering Center for Unmanned Vehicles (RECUV), Department of Aerospace Engineering Sciences, University of Colorado Boulder, Boulder, CO USA 80309; e-mail: Nisar.Ahmed@colorado.edu
To give more context, we first briefly summarize the cooperative human-robot state estimation method developed in [9]. Let $X$ be a continuous random vector representing the dynamic state of interest (e.g. target location, velocity, heading) with prior pdf $p(X)$ (which may already be conditioned on hard data), and $D$ be a discrete random variable representing a human-generated semantic observation related to $X$ (e.g. ‘The target is on the bridge’, ‘The target is heading over the bridge and slowing down’, etc.). Given the likelihood function $P(D|X)$, Bayes’ rule gives the posterior pdf $p(X|D) \propto p(X)P(D|X)$, where $P(D|X)$ models the human’s ‘semantic classification error’ as a function of $X$. If $D = l$ corresponds to one of $m$ exclusive semantic categories for a known dictionary, then a softmax function can be used to model $P(D = l|X)$,

$$P(D = l|X) = \frac{\exp(w_l^T x + b_l)}{\sum_{c=1}^{m} \exp(w_c^T x + b_c)} \quad (1)$$

Fig. 1 (a) shows an important feature of this likelihood model: for a given parameter set of class weights and biases, $\Theta = \{w_c, b_c\}_{c=1}^m$, the state space $X$ is divided into $m$ convex ‘probabilistic polytopes’, i.e. subsets in $X$ where certain semantic categories occur with high probability [21]. The set of log-odds functions between classes $i$ and $j$ define the linear hyperplane boundaries of these polytopes at different relative probability levels; for equal probabilities, we have

$$\log \frac{P(D = j|X)}{P(D = i|X)} = (w_j - w_i)^T x + (b_j - b_i) = 0 \quad (2)$$

Categories in a softmax model can also be internally grouped together as ‘subclasses’ within larger semantic classes of a multimodal softmax (MMS) likelihood model [22], which can represent non-convex polytopes in $X$ via piecewise-convex subclasses $^1$. Although $p(X|D)$ is analytically intractable for any given $p(X)$, [9] showed that this pdf can be closely approximated by a Gaussian mixture, while [23] experimentally validated this fusion approach in a set of 2D autonomous target search experiments with real human sensors (using a simple codebook interface). The likelihood models for the semantic prepositions used in those studies were trained on real human sensor data, but were constrained to follow handcrafted templates that enforced the expected shapes of the 2D likelihood surfaces, such that only scale and gradient parameters were allowed to be set via maximum likelihood estimation. Fig. 1 (c)-(d) shows that such constraints can lead to significant reductions in the effective parameter space, which is advantageous for learning/calibration with sparse training data. However, such constraints are only beneficial and worthwhile in more general cases if desirable geometric properties for the likelihood function surface can be easily translated into constraints on $\Theta$. For instance, the constraints for $\Theta$ in Fig. 1 (d) were originally obtained by trial and error using fake ‘prototypical’ training data to produce the desired subclass geometries. Such labor intensive brute force approaches are not useful for higher dimensions or rapid online model adaptation/deformation.

We show that this issue can be resolved by applying eq. (2) to transform the normal vectors for a desired set of polytope boundaries into a system of linear difference equations for $\Theta$, which are quickly solved by singular value or QR decomposition. Once established, these parameters and their relationships can be further manipulated to deform the likelihood model’s embedded probabilistic polytopes as needed. However, surprisingly subtle caveats to this approach imply that not all decompositions of $X$ into $m$ convex polytopes can be directly translated into softmax parameters $\Theta$ for $m$ classes – in most cases, extra ‘subclasses’ are needed to obtain MMS models that can embed the desired semantic geometries. The stems from the fact that the hyperplane normals are generally not unit vectors and are specified in terms of differences between elements of $\Theta$, which induces a latent consistency condition analogous to Kirchoff’s voltage law $^2$. The Rouche-Capelli theorem and a simple graphical constraint checking technique can be used to find the minimal set of polytope boundary normal specifications for general softmax synthesis. This approach could be extended to set invariant geometrical features (e.g. angles, directions in $\mathbb{R}^n$) and induce embedded polytope priors for learning. Preliminary results for a dynamic target search and tracking application will also be discussed.

\[^1\]e.g. a range-only semantic MMS model can be obtained from Fig. 1 (a) by grouping together classes with the same range labels

\[^2\]in this case: the sum of hyperplane normals around any ‘loop’ starting from class $i$ and ending at class $i$ must equal zero
A. Example Results

Two simple examples are described next to demonstrate the proposed model synthesis approach, and to illustrate the mathematical relationship between softmax parameters and hybrid likelihood functions for semantic observations.

1) Arbitrary polygon: As a first example, consider the convex irregular 5-sided polygon shown in 2D space shown in Figure 2(a). Suppose it is desired to construct a softmax model describing the likelihood function of being either ‘inside’ or ‘outside’ the polygon. As stated earlier, the conventional approach to building a softmax likelihood for this semantic model requires labeling a densely sampled set of 2D points, and then feeding these training data to a maximum likelihood/MAP estimation algorithm to identify the appropriate softmax model parameters. Since the ‘outside’ region is non-convex, this requires the use of an MMS model featuring at least 5 softmax ‘subclasses’ that together make up the entire ‘outside’ class. Each ‘outside’ subclass corresponds to one face of the polygon; for convenience, denote these subclasses with the labels ‘top left’, ‘top right’, ‘bottom left’, ‘bottom right’, and ‘bottom’. The likelihoods for the ‘outside’ subclasses are given by the softmax likelihood (1), where \( j \) is a subclass label in the set \{ ‘top left’, ‘top right’, ‘bottom left’, ‘bottom right’, and ‘bottom’ \} (which we will also denote \{ \( j_1, ..., j_5 \) \}). Each subclass is associated with its own softmax weight vector \( w_j \) and a bias term \( b_j \). A separate single subclass \( i \) is responsible for the polygon interior and thus all of the ‘inside’ class with weight \( w_i \) and bias \( b_i \). The combined set of subclasses are modeled within the same softmax likelihood, and the MMS likelihoods for the classes ‘inside’ and ‘outside’ are given by

\[
P(D = \text{‘outside’}|X) = \sum_{o=j_1}^{j_5} \exp(w_o^T x + b_o) / \sum_{c=i,j_1, ..., j_5} \exp(w_c^T x + b_c)
\]

(3)

and

\[
P(D = \text{‘inside’}|X) = \exp(w_i^T x + b_i) / \sum_{c=i,j_1, ..., j_5} \exp(w_c^T x + b_c)
\]

(4)

i.e. the class probabilities are obtained by summing over the softmax subclass probabilities. Hence, obtaining the correct parameters for the subclasses leads to the correct class probability surfaces, such that all ‘outside’ subclasses start becoming less likely than the ‘inside’ (subclass at the polygon perimeter. In other words, the log-odds boundaries between each ‘outside’ subclass and the ‘inside’ subclass must correspond to the edges of the polygon.

Let \( n_{ji} \) denote the normal vector for the edge between ‘outside’ subclass \( j \) to the inside subclass \( i \); by definition, each 2D point \( x \) on this edge must satisfy \( n_{ji}^T x + c_{ji} = 0 \), where \( c_{ji} \) is an origin offset constant and \( n_{ji} \) is not necessarily a unit vector. Figure 2 shows the components of \( n_{ji} \) for each edge (black arrows). Equating all \( x \) on the log-odds boundaries to \( x \) points on the polygon edge, it follows that the softmax parameters must satisfy

\[
(w_j - w_i)^T x + (b_j - b_i) = n_{ji}^T x + c_{ji},
\]

and hence \( (w_j - w_i) = n_{ji} \in \mathbb{R}^2 \) and \( (b_j - b_i) = c_{ji} \in \mathbb{R} \) for all ‘outside’ subclasses \( j \in \{ \text{‘top left’, ‘top right’,...‘bottom’} \} \) and the ‘inside’ subclass \( i \). This leads to a system of 15 difference equations in 18 unknown softmax parameters. Since we can always arbitrarily assign one set of subclass parameters to be the zero vector, assume \( w_i = 0 \) and \( b_i = 0 \), so that there are now only 15 unknown softmax parameters in 15 difference equations. Define the stacked subclass parameter vector

\[
\tilde{\theta} = [w_{j_1}(:) ; b_{j_1} ; w_{j_2}(:,) ; b_{j_2} ; ... ; w_{j_5} ; b_{j_5}]
\]

(5)

where indices \( j_1 \) through \( j_5 \) correspond to the ‘outside’ subclasses and the stacked vector of desired edge parameters

\[
\tilde{\beta} = [n_{j_1, i}(:) ; c_{j_1, i} ; n_{j_2, i}(:,) ; c_{j_2, i} ; ... ; n_{j_5, i} ; c_{j_5, i}]
\]

(6)

where ‘;’ and ‘,:’ denote vertical array concatenation and vectorization, respectively. Then, the required difference equations can be written

\[
M \tilde{\theta} = \tilde{\beta}.
\]

(7)

where matrix \( M \in \mathbb{R}^{15 \times 15} \) performs the appropriate differencing operations on the elements of \( \tilde{\theta} \). In this case, we are very lucky indeed, since all difference equations are specified relative to the inside class for which \( w_i = 0 \) and \( b_i = 0 \), so \( M = I \) (the identity matrix). Therefore, \( \tilde{\theta} = \tilde{\beta} \) and the softmax model parameters for each the 5 ‘outside’ subclasses can be taken directly from the corresponding polygon edge equations. Fig. 3(a) shows a top view of the resulting probability surfaces, showing that the log-odds boundaries coincide with the polygon edges as desired. Fig. 3(b) shows the same softmax subclass surfaces from the side when each \( n_{ji} \) normal vector has unit magnitude; Fig. 3(c) shows that the resulting surfaces become steeper when the magnitudes of each \( n_{ji} \) are scaled up by a factor of 80. This shows that the boundary normals \( n_{ji} \) not only control the direction of the log-odds boundaries in the softmax model, but also the relative gradients between regional probabilities. While not shown here, altering the \( c_{ji} \) terms lead to dilations (expansions/contractions) relative to the origin, thus allowing the area of the polygon to shrink and grow while preserving relative orientations of the log-odds boundaries. Fig. 3(d) shows the surface of the ‘outside’ MMS likelihood function (3) obtained from the softmax model parameters in Fig. 3(c); notice that the internal boundaries between each of the ‘outside’ subclasses disappear, leaving only the boundaries between ‘outside’ and ‘inside’. However, it is worth noting that these particular ‘internal’ boundaries change automatically as a function of \( \tilde{\beta} \), since they are linearly dependent on the desired ‘inside’/‘outside’ subclass log-odds boundaries.

This result readily generalizes to the formation of semantic regions inside and outside of other arbitrary convex polygons in 2D and convex polytopes in n-dimensions: assuming the weights and biases for (sub)class corresponding to the
interior of the polygon/polytope are set to 0, the remaining softmax parameters for the ‘outside’ subclasses are directly obtained from the normal vectors and offset parameters of the polygon/polytope hyperplane boundaries.

2) Voronoi decomposition: Consider next a slightly more complicated modeling problem for which the solution of the parameter vector \( \theta \) is far less trivial. Figure 4 (a) shows the Voronoi decomposition of a 2D space about 5 randomly generated landmarks, which is to be transformed into a softmax/MMS model containing one class for each of the 5 convex Voronoi cells that dominates the other classes in the same part of the 2D state space. Voronoi decompositions are widely used in robotics, e.g. for coordinated planning/control [24] and qualitative spatial mapping/navigation [25], and have also attracted the attention of cognitive researchers such as Gardenfors for modeling generalized spatial language, since they provide a convenient means for semantically quantizing complex geometric spaces around ‘contextual anchors’ [12].

As Voronoi cells are convex, one might naively assume that the synthesis can be performed with a simple softmax model with 5 classes (one class for each Voronoi region). This leads to a total of 15 parameters (or 12 if again a single class is set to have 0 weights and biases), if we were to apply the same logic in the previous example to equate the \( n_{ji} \) and \( c_{ji} \) parameters for all edges of the Voronoi cells to the log-odds boundary parameters \( w_j, w_i, b_j \) and \( b_i \) for all appropriate softmax classes \( i \) and \( j \) (which in this case correspond to different Voronoi cell labels). However, since there are 7 boundary edges separating these Voronoi cells, we quickly find that we must specify 7 vector difference equations and thus obtain a \( \beta \) vector with 21 elements for a linear system of difference equations \( M\hat{\theta} = \beta \). In this case, \( M \in \mathbb{R}^{21 \times 12} \) is no longer a square matrix, and in fact represents an overconstrained system of equations. Since \( \text{rank}(M) < \text{rank}(M, \beta) \), the Rouche-Capelli theorem implies that the system of equations is inconsistent, and hence there is no solution vector \( \hat{\theta} \) that exactly satisfies this system of equations. In other words, even though the Voronoi region polytopes are all convex, there is no way to embed these within a softmax model so that each class label has a one-to-one correspondence with a Voronoi cell and also generates log-odds boundaries consistent with the edges to neighboring cells, since the model does not have enough degrees of freedom.

Note that a solution to this system of equations could be approximated via the Moore-Penrose pseudo-inverse \( M^+ \) to obtain the ‘least squares’ parameter vector \( \hat{\theta}_{LS} = M^+\beta \); this leads to the not-quite-right softmax surface shown in Fig. 4(b), where the attempt to minimize the overall model fit error leads to deviations from nearly all boundary specifications. This solution represents the closest one can get (in a least-squares sense) to meeting the desired boundary specifications with the simple softmax model, and directly underscores the subtle challenge of the softmax synthesis problem: all softmax models yield a set of embedded convex (probabilistic) polytopes, but not all convex polytopes can be easily embedded probabilistically within softmax models.

In cases like this, the desired polytope embeddings can be realized by assigning additional subclasses to the log-odds boundary specification for certain classes, which leads to an MMS model synthesis problem with additional parameters and degrees of freedom. In this case, an exact feasible solution can be obtained by introducing 3 subclasses, or 9 additional weight and bias parameters, which means that now \( \hat{\theta} \in \mathbb{R}^{21} \) and we can specify \( M \in \mathbb{R}^{21 \times 21} \) to be a well-posed system of equations, where some subset of the original softmax classes for the nominal Voronoi regions are now split into two, three or four subclasses. This approach requires us to split up the nominal Voronoi regions into a suitable set of subclasses and to revise the original log-odds boundary specifications with respect to \( \beta \) accordingly. Fig. 4(c) shows the result of one such splitting assignment, where the softmax class for Voronoi cell 2 is assigned all 3 new subclasses and thus gets split into 4 subclasses total. The combined subclass probabilities thus allow the original boundary specifications to be met without any training data. Although some hand-tuning of the magnitudes of \( n_{ij} \) was needed here for fine adjustment of the boundary positions, the computation time for this synthesis (which involves using QR-decomposition to solve a linear system of equations) is much smaller compared to the typical time needed to perform maximum likelihood/MAP parameter estimation on a densely sampled set of synthetic training data (e.g. using non-linear quasi-Newton search).

REFERENCES


Fig. 3. Resulting probability surfaces for polygon region modeling example.

Fig. 4. Voronoi cell region modeling example: (a) sample Voronoi cell centers and resulting boundary edges with normals; (b) probability surfaces for simple softmax subclasses assigned to cell region 2, showing that log-odds boundaries between softmax classes now align with Voronoi cell boundaries.