User Routine Model using a Cloud-Connected Social Robot

Luís Santos[1] and Jorge Dias[1,2]

Abstract— **In this manuscript we propose a distributed classifier to perform inference on a person daily behaviour routine, based on multi-modal input data. The model is implemented on a social robot and allows to efficiently fuse locally perceived information with data classified remotely on a cloud. Unlike the dominant multi-class approaches, where each class is classified separately, the multi-label scheme estimates all classes simultaneously from the available input instances. This method enables a robot to capture user typical behaviour and provides a simple scheme of regulation that allows the identification of abnormal situations. We propose to solve our problem in two steps based on the principles of Binary Relevance and Label Power-set: (1) a label classification is used to filter input instances into independent labels; (2) the algorithm will map the labels into an hyper-label space, where each hyper-label represents the behaviour which maximizes input instance correlations. Results show the proposed multi-label model to achieve a highly accurate comprehension of the user behaviour even within more demanding test scenarios. As for the regulatory experiments, initial results show that the proposed behaviour model allows to identify unexpected events, that can be used to trigger care giver interventions.**

Index Terms— **multi-label classification, service robots, robot perception, multi-modal interface, human robot interaction.**

I. INTRODUCTION

The purpose of this work is to pave the way into enabling robotic platforms with means to establish person behaviour patterns using any type of information available, beyond the classical, self contained recognition problems, thus allowing to better respond to different events that occur on a daily basis. The current state of the art on context analysis often addresses classification problems of different types of data separately and often implicitly [1], in what is called the multi-class problem. However, there is some experimental evidence supporting that the context questions may be more reliably answered if they are answered in groups of two or three using the information extracted from multimodal input streams [2]. The state of the art in human behavior understanding is characterized by the study of machine learning methods, which are usually applied to specific, single instantiation problems (multi-class classification) such as object recognition [3], action recognition [4], emotion and facial recognition [5], speech recognition [6], etc. The commonly addressed paradigm concerns processing sets of data to generate features, and associate them to classes for posterior identification (classification). Additionally, these methods do not usually consider context, they fail to handle the subtle changes for naturalistic (not deliberate) behavior observed in real scenarios and they do not consider long time scales [1], [7].

One popular way of associating different types of information, is to use ontology representations [8] or semantic models. These have been applied successfully in multiple research areas, e.g. business analysis [9], [10], analysis of body motion [11] or to define a grammar-based representation for manipulation actions [12]. Specifically in the AAL area, recent advances show data driven models, which start with seed models and expand them by allowing them to learn new instances in activity patterns for smart homes [13]. One other rapidly growing approach in the research community to aggregate different types of data, is to address the problem of multi-label classification [14], which beyond multi-class problems, can interpret different types of information towards a simultaneous and broader understanding of the user and the environment.

Therefore, we aim to explore multi-label classification to enable a social robot to understand user behaviour by fusing locally perceived data (e.g. emotions, navigation) with information from a cloud, which can include data as diverse as its preferences, personal data, etc.

In Section II the problem is introduced and several relevant definitions explained. Section III is divided two-fold: 1) the perception of a multi-modal behaviour state, using a multi-label approach; 2) the state transition which regulates the robots actions and reactions according to the inferred behaviour state. The multi-label model and the action regulation module experiments are presented in Section IV. This manuscript concludes with a discussion and future work with Section V.

II. PROBLEM FORMULATION AND DEFINITIONS

In this manuscript we aim to address the problem of creating a mathematical representation of a person's daily routine, such that it encodes different types of information that can be gathered from different sources, either local (robot) or remote (cloud). To address this problem we propose a multi-label classification using a distributed classification framework using the principles of Binary Relevance [15] and Label Power-set [16]. This approach is used to infer the current behaviour state, simplifying a complex multi-label classification problem into a multi-class one. The behaviour model is represented by a state transition matrix, representing the possible behaviour state sequences that define a person's typical routine. The following definitions are required to understand the proposed solution.

[1]Luís Santos and Jorge Dias are with the Institute of Systems and Robotics, University of Coimbra, 3030-290 Coimbra, Portugal, luis@isr.uc.pt, jorge@deec.uc.pt

[2]Jorge Dias is also with the Khalifa University of Science, Technology & Research (KUSTAR), Abu Dhabi, UAE, jorge.dias@kustar.ac.ae

- INPUT INSTANCE - this corresponds to a random variable $x \in \mathbf{X}$ representing a measurable property of observable *label*.
- LABEL - a Label $l \in \mathbf{L}$ is defined as a state belonging to a classifiable random variable $z_r \in Z$. For example, it corresponds to an emotion, a face, an activity, *etc*.
- HYPER-LABEL - a Hyper-Label $y \in \mathbf{Y}$, is a combination of observed labels $l \in \mathbf{L}$ given an input vector $\mathbf{X}$. For example, given an input $\mathbf{X}_k$ that generates a label set $\mathbf{L}_k$, such that $\mathbf{L}_k = [Run, Afternoon, Park]$, then a single y will be used to represent this label power set.
- ROUTINE MODEL - The behaviour/routine model is described as an ordered sequence of Hyper-Labels, that will encode the user's typical routine and associated attributes, such as $\Gamma = [y_2, y_5, y_1, y_{10}]$.

The proposed hyper-label classification is based on Bayesian methods [17]. We justify the use of these methodologies for its possibility to use the prior distribution to influence the decision of the classified data but also to penalize unlikely hyper-labels. Unlike other approaches, which rely on the *frequentist* approximations [18], we assume the possibility that established prior knowledge (such as a person agenda) might contradict data, which must be factored in.

III. BEHAVIOUR MODEL

A. HYPER-LABEL INFERENCE

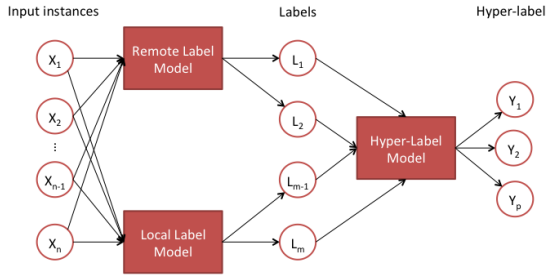

Fig. 1. Hyper-Label Learning Framework

Multi-modal input instances $\mathbf{X} = [x_1, x_2, \ldots, x_n]$ are pre-filtered in a distributed classification framework, using classification techniques to generate independent label data $\mathbf{L} = [l_1, l_2, \ldots, l_m]$. The labels are used as input to encode the so called hyper-labels $\mathbf{Y} \equiv \{y_1, y_2, \ldots, y_p\}$ as described in the next paragraph.

TABLE I
HYPER-LABEL TRAINING ALGORITHM

	Training Algorithm	
1	pre-processing	map each label-set $\{l_m\}$ to a hyper-label y_p
2	training	learning a multi-class label model from $\{x_n, y_p\}$
3	classification	given input instances $\{x_n\}$ find y_p

1) ***Hyper-Label Pre-processing****:* A Binary Relevance based approach is used to encode a hyper-labels y_p from the set of labels l_m. The encoding scheme is based on the presence ("1") or absence ("0") of a given label, therefore generating a number 2^m for m different labels $\in L$. The following Table II illustrates an encoding example for 3 different labels, with all possible combinations.

TABLE II
ENCODING TABLE FOR P=3 DIFFERENT LABELS

p	Labels	Binary Relevance			Power-set
		L_1	L_2	L_3	
1	$\emptyset$	0	0	0	$y_1 = \emptyset$
2	l_1	0	0	1	$y_2 = \{l_1\}$
3	l_2	0	1	0	$y_3 = \{l_2\}$
4	l_1, l_2	0	1	1	$y_4 = \{l_1, l_2\}$
5	l_3	1	0	0	$y_5 = \{l_3\}$
6	l_1, l_3	1	0	1	$y_6 = \{l_1, l_3\}$
7	l_2, l_3	1	1	0	$y_7 = \{l_2, l_3\}$
8	l_1, l_2, l_3	1	1	1	$y_8 = \{l_1, l_2, l_3\}$

Despite the previous table illustrates all possible combinations, the hyper-label set only considers combinations that have been previously observed, in order to reduce the set dimension. When a new hyper-label is created, it is added to the power-label set, which is mapped using a bijection function B, such that,

$$B : \mathbf{L}_k \mapsto y_k \tag{1}$$

where, $\mathbf{L}_k$ is a vector combining different labels l, and y_k the corresponding hyper-label.

2) ***Hyper-Label Training****:* The hyper-label encoding makes it possible to solve the multi-label classification problem problem by reducing it to a multi-class classification approach. Therefore, we will associate values of input vector $\mathbf{X}$ to power-label variables $y_p \in \mathbf{Y}$. Let the k^{th} input vector $\mathbf{X}_k$ generate a label set $\mathbf{L}_k$, which encodes to a label, such that $y_k = B(\mathbf{L}_k)$. For all K samples of $\mathbf{X}$ associated to one label y_p, we can relate them probabilistically using a function $y_p = F(\mathbf{X})$ from the training dataset $\{(\mathbf{X}_i, y_k)\}_{i=1}^{K}$.

In our particular case, F is a Bayesian distribution parametrised by multi-variate Gaussian distributions, such that:

$$\mathbf{Y} = F(\mathbf{X}) = P(x_1, x_2, \ldots, x_n | \mathbf{Y}) \tag{2}$$

Because labels l_m are independent, we hypothesize that we can simplify the multi-variate distribution to n independent univariate Gaussian distributions. Therefore, each being represented by:

$$P(x|\mathbf{Y}) = N(\mu_x, \sigma_x) \tag{3}$$

where μ_x and σ_x are calculated for all samples of $\mathbf{X}$ that are associated to the label $y_k \in Y$.

3) ***Hyper-Label Classification****:* Hyper-Label Model is defined as a multi-class classifier, here developed as a Bayesian Network. On the training phase, we have defined the likelihood distributions $P(x|\mathbf{Y})$ of the proposed Bayesian Model, which will be used to answer the Bayesian question. Bellow, in Figure 2 we describe our model using the Bayesian Programming Formalism [19], [20], [21]. As perceived from last subsection, we formulate the model as the estimation of a discrete variable Y using a continuous distribution over the range of $x \in \mathbf{X}$. In fact, in the process of computing the average μ_x, we will most likely obtain a real value; therefore,

in order to maintain precision, we proposed to represent it using a continuous distribution. To avoid computational complexity, the likelihood is in fact a kernel of Gaussian distributions. For each $y \in \mathbf{Y}$, exists a Gaussian distribution over $x \in \mathbf{X}$ (or in the event of multi-variate Gaussian distributions, over $\mathbf{X}$).

Bayesian Program: HYPER-LABEL CLASSIFIER

program

description

specification

Variables:
$x_t \in X$: Input variable
Decomposition:
$P(Y, x_1, \cdots x_n) \propto P(Y) \prod_{i=1}^{n} P(x_i|Y)$
Formulation:
$P(\mathbf{Y}) : \begin{cases} Uniform(\mathbf{Y}) & t = 0 \\ P(\mathbf{Y})_{t-1} & t \neq 0 \end{cases}$
$P(x_n|Y)$: Kernel of Gaussian Distributions
such that $P(x_n|Y) = P(x_n|Y = y_i)|_{i=1:p}$
and $P(x_n|Y = y_p) = N(\mu_{x_n}, \sigma_{x_n})$
Identification: Gaussian parameters μ and σ based on training dataset Ω.
Question: $P(Y|x_1, \cdots, x_n)$ answered using *Maximum A Posteriori* (Bayesian Inference).

Fig. 2. Bayesian Program for the Hyper-Label classifier.

We propose at this stage a simple strategy to define the prior $P(\mathbf{Y})$, which represents previous knowledge. In this work, we start by using a Uniform distribution at the first iteration and replace it by the last estimation of $P(Y)_{t-1}$ in subsequent iterations.

We use a *Maximum A Posterior* (MAP) approach to solve the inference problem. Hence, it can be computationally solved by the following equation.

$$y_{p_{\text{MAP}}}(\mathbf{X}) \propto \arg\max_{y \in Y} P(Y = y_p) \prod_{i=1}^{n} P(x_i|Y = y_p) \quad (4)$$

B. HYBRID STATE TRANSITION MODEL

Let the system be on a functional state associated to the current behaviour y_p. When on a state, the framework will (1) monitor only a set of relevant observable variables $\mathbf{X}_t$, which are required to trigger a state transition; and (2) will compare the predicted next state with the current observation to either allow the transition or generate an alarm.

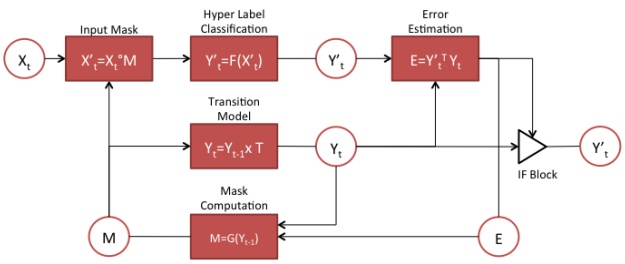

Fig. 3. Transition Model

*1) **Input Instance Filtering**:* The $\mathbf{X}_t$ is the input instance vector at the current time and $\mathbf{M}$ a mask vector that is used to filter which variables of $\mathbf{X}_t$ are relevant for the state transition. Therefore, the filtered input vector $\mathbf{X}'_t$ is obtained by applied the Hadamard Product (Element wise multiplication) of the input vector with the hyper-label mask, such that:

$$\mathbf{X}'_t = \mathbf{X}_t \circ \mathbf{M} \quad (5)$$

where $\mathbf{M}$ is a vector computed as a function of $G(\mathbf{Y}_{t-1})$. The result is a vector of binary elements, which by applying the Hadamard product, can filter out elements of $\mathbf{X}_t$, by assigning them to "0".

*2) **Transition Model**:* The transition from one state to the other is estimated using a Transition Model. Such model encodes a **sequence of behaviour states**, which defines on its own the user's **routine model**. When detecting a user behaviour state, the robot will remain on the associated **functional state**. The functional State is responsible for selecting **appropriate services to meet users needs**, when at a current behaviour state. Let $\mathbf{T}$ be a transition model, the predicted behaviour state $\mathbf{Y}_t$ is computed as

$$\mathbf{Y}_t = \mathbf{Y}_{t-1} \times \mathbf{T} = [y_1\ y_2\ \ldots\ y_p]_{t-1} \begin{bmatrix} t_{11} & t_{12} & . & t_{1p} \\ t_{21} & t_{22} & . & t_{2p} \\ . & . & . & . \\ t_{p1} & t_{p2} & . & t_{pp} \end{bmatrix} \quad (6)$$

where, each element t_{ij} contains the normalized number of observations for each transition as described in the next equation.

$$t_{pp} = \frac{t_{pp}}{\sum_{i=1}^{p} t_{pi}} \quad (7)$$

*3) **Transition Error**:* To verify if the observed state Y'_t is a possible solution given the transition model and the expected state Y_t, a transition error E is computed. This error is computed using the Dot Product, such that:

$$E = Y_t \cdot Y_{t-1} \quad (8)$$

Because the sum of all elements of Y_t and Y_{t_1} adds up to 1, then it is easy to demonstrate that $0 < E < 1$. When E is close to 1, it means the observed state is consistent with the expected state. On the opposite, for low values of E, it means that the observed state is not what is expected, therefore the system has an indication that something might be wrong.

C. HYPER LABEL INCREMENTAL LEARNING

Label classifiers are responsible for receiving multi-model input instances X_n and classify independent labels L_m. Consider multiple variables $z_r \in Z$, each pertaining a different data type, such as faces or emotions, where each recognizable face, emotion, object, activity, etc. belongs to the multi-modal label space L_m. Each variable can be classified using its own independent method, as long as it retrieves the inferred class.

*1) **Hyper-Label Learning Mechanism**:* The system is assumed to be constantly evolving, which means it needs to adapt to new observed data and/or behaviours. Therefore, we propose a simple scheme of incremental learning, which acts on two events:

- When an input instance $\mathbf{X}_k$ generates a new label $\hat{y} \notin \mathbf{Y}$ the system needs to add it to the training set and generate a new likelihood model to include it.
- When an existing label $y \in \mathbf{Y}$ is generated from a new input instance $\mathbf{X}_k$, the system will add this data to the existing likelihood model to include it.

On the first case, the system will verify the absence of label $\hat{y}$ by running through the whole set $\mathbf{Y}$, and upon confirmation, it adds a new distribution $P(\mathbf{X}|Y = \hat{y})$ to the likelihood kernel. The procedure is described in the following algorithm of Table III.

TABLE III
NEW HYPER-LABEL LEARNING ALGORITHM

	Learning a new Hyper-Label
1	IF $\hat{y} \notin Y$
	GO TO 2
	ELSE
	GO TO 4
2	COMPUTE new $N(\mu_x, \sigma_x)$ for $\{(\mathbf{X}_i, \hat{y}\}$
3	ADD new $N(\mu_x, \sigma_x)$ to LIKELIHOOD KERNEL $P(\mathbf{X}\|\mathbf{Y})$
4	END

On the second case, it will linearly fuse the new data to the parameters of the Gaussian distribution for all inputs $x \in \mathbf{X}$, such that

$$\mu'_x = \frac{\alpha\mu_x + \beta x_k}{\alpha + \beta};\ \tau'_x = \frac{\alpha(\sigma_x + \mu_k) + \beta(\sigma_x + x_k)}{\alpha + \beta} \quad (9)$$

where α and β are weight factors for existing and new data respectively, such that $\alpha + \beta = 1$.

IV. EXPERIMENTS

Our experiments are divide two-fold, to test:

(A) the hyper-label inference model;
(B) the state transition, which will regulate the functional robot state.

On the first case (A), synthetic data is used to provide a significant amount of data to validate the proposed methodology for large amounts of data and different model configurations. The Hyper-Label Model estimates a **behaviour state** that is used to regulate the **transition model**. On the latter case (B), a small realistic scenario is presented on a mobile robotic platform, with the purpose of demonstrating the concept of the state transition model in a real application.

A. Hyper-Label Model Experiments

Let us describe the case where we assume a set of $r = 5$ different variables $Z = \{z_1, z_2, z_3, z_4, z_5\}$, such that they could represent Faces, Emotions, Locations, Objects and Actions respectively. From Z, a set of labels are generated, e.g. Faces variable can have $q = 2$ different identities, such that $z_1 = \{l_1, l_2\}$. Similarly, we assume the remaining variables $z_r \in Z$ all have the same number q of possible states, which generates a label set $l_m \in L$ where $m = rq = 10$. For a set of this size, there should $2^m = 1024$ possible hyper-labels. However, our problem presents a restriction, labels of the same variable are exclusive (e.g. a person cannot be "Happy" and "Sad" at the same time), which means that instead of 2^m possible hyper-labels, there are $q^r = 32$, which means only 32 out of 1024 possible combinations are valid hyper-labels, presenting a sparsity of

$$\eta = \frac{\#Hyper\_labels}{\#TotalCombinations} = \frac{32}{1024} = 0.03125. \quad (10)$$

Let there be also 5 different input instances $\mathbf{X} = [x_1, x_2, x_3, x_4, x_5]$, which are in these experiments modelled using Bayesian Networks[1].

$$P(x_i|z_i = l_j) = N(j, 1) \text{ for } i = 1 : r \text{ and } j = 1 : q \quad (11)$$

We seed $k = 10000$ random input $\mathbf{X}$ sample combinations with $x_n \in [0, 6]\ \forall\ n$, which in turn generate a vector of labels $\mathbf{L} = [l_1\ l_2\ \ldots\ l_{10}]$, by applying Bayesian inference with the likelihoods described in equation 11, as written next.

$$P(Z|\mathbf{X}) \propto P(Z)P(\mathbf{X}|Z) = P(Z)\prod_{i=1}^{n} P(x_i|Z) \quad (12)$$

The outcome of estimated label l_m for each variable z_r will result in a label vector $\mathbf{L}$, which using a bijection function B yielding $y_p = B(\mathbf{L})$. This will generate $k = 10000$ pairs $\{\mathbf{X}_k, y_p\}|_{k=1:10000}$ that constitute our Hyper-Label training data-set, trained according to equation (2).

A **new** random input stream of $k = 10000$ samples is seeded for classification, where we take the Label vector $\mathbf{L}$ combination for ground truth and the classified Hyper-Label $y_p \in \mathbf{Y}$ from equation (4). Results are analysed in terms of precision, recall and Hamming Loss (HL). Precision and Recall are classic measurements in classification problems, whereas the latter measures the fraction of wrong labels between the classified and the ground truth Hyper-Label.

$$\text{HL} = \frac{1}{k}\sum_{i=1}^{k}\frac{\sum_{j=1}^{m}(l_{j,i} \oplus \hat{l}_{j,i})}{m} \quad (13)$$

To compute the Hamming Loss, we decode the label set $\mathbf{L}_k = \{l_m\}$ from the estimated y_p as $\mathbf{L}_k = B^{-1}(y_p)$. This set of experiments (with a variable number of q states) is summarized in the following Table IV.

TABLE IV
HYPER-LABEL CLASSIFICATION RESULTS FOR $r = 5$ DIFFERENT VARIABLES $\in \mathbf{Z}$, AND DIFFERENT NUMBER OF q STATES IN EACH.

q	m	p	η	k	precision	recall	Hamming Loss
2	10	32	31.25×10^{-3}	10^5	96.72%	96.69%	0.00668
3	15	243	7.41×10^{-3}	10^5	75.95%	81.45%	0.02133
4	20	1024	0.98×10^{-3}	10^5	80.02%	84.54%	0.01459
5	25	3125	93.13×10^{-6}	10^6	75.65%	78.26%	0.01184

It is clear to observe that for an increasing number of states, that there is an exponential growth in the cardinality of Y (verified in p column). However, these states are also exponentially sparse (η) in the label set domain. It is also clear that there is a decrease in both *precision* and *recall* in the classification of hyper-labels when the number of possible states y_p increases. In fact, from $p = 20$ to $p = 25$, we had to increase the number of samples from $k = 10^5$ to $k = 10^6$ in order to have enough training samples for each of the hyper-labels. This allows us to state that the amount of training data will play a significant role in the performance of the hyper-label classifier. However,

[1]Despite using Bayesian Networks, the relation between $\{X, L\}$ could be modelled using any other classification method.

interestingly, the HL measurement indicates that, although hyper-label y_p precision decreases, it does not reflect in the fraction of misclassified labels (HL). The maximum fraction of misclassified labels l_m is 2.133% for $q = 3$. It indicates that, although the number of misclassified y_p increases, it is by the misclassification of 1 or 2 labels l_m, for this particular case in the set of $m = 15$. This is a positive indicator, even more noticeably if we consider that HL decreases for increasing values of m.

To further complement these experiments, the number r of variables was made dynamic $\in \mathbf{Z}$, whereas a static value of $q = 2$ was used.

TABLE V

HYPER-LABEL CLASSIFICATION RESULTS VARIABLE r VARIABLES $\in \mathbf{Z}$, AND STATIC $q = 2$ STATES IN EACH.

r	m	p	η	k	precision	recall	Hamming Loss
6	12	64	15.63×10^{-3}	10^5	93.35%	93.29%	0.01143
7	14	128	7.81×10^{-3}	10^5	89.42%	89.34%	0.01590
8	16	256	3.90×10^{-3}	10^5	81.90%	81.74%	0.02246
9	18	512	1.95×10^{-3}	10^6	90.08%	89.99%	0.01148
11	22	2048	0.49×10^{-3}	10^6	75.79%	75.55%	0.02452
12	24	4096	0.24×10^{-3}	10^6	64.11%	63.73%	0.03518

The presented results illustrate an extended refinement in the values of p and also the effect of varying the number of variables z_r rather than the number of states q. We have tested from using $r = 6$ to $r = 12$ variables with binary states. Results are in-line with the ones resulting from the first set of experimental simulation. In fact, only for a very large number of possible y_p states, we have a precision of 64.11%, but the HL tells that samples are misclassified by an implicit single label in a range of $m = 24$.

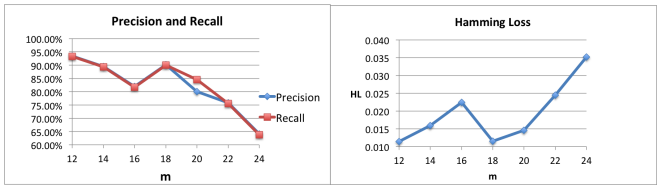

Fig. 4. Precision, Recall and Hamming Loss plot function of size of $\mathbf{L}$.

In Figure 4 we can observe a *quasi* linear decay in *precision* and *recall*, which is compensated from $m = 16$ to $m = 18$ with an increase of the training samples. A similar behaviour is observed for the Hamming Loss. However in this latter case, one must factor in the fact that HL represents a fraction of misclassified samples $\in L = \{l_m\}$, which does not reflect such linear decay in the actual number of misclassified l_m, which is on average $\lesssim 1$.

B. Transition Model Experiments

The proposed hybrid transition model was implemented and tested with a real robotic platform, where the observed behaviour state $\mathbf{Y}'$ is estimated using the Hyper-Label model. In Figure 5 we illustrate the functional diagram of the implemented framework. When the framework detects the

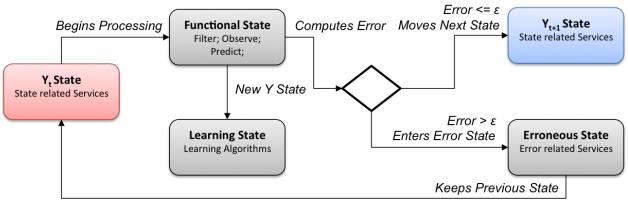

Fig. 5. Functional Diagram.

user on a given behaviour state it will allow the robot to transit into an associated functional state where it can operate services associated to such state. For example, if the robot detects that a person is in a physiotherapy session, it will execute the exercise guiding service, however the specification of the execution of these services is out of the scope of this work.

To decide whether the robot can move from one functional state to the other, an error is computed based on the behaviour state the robot is expecting and the one it is actually observing. In case there is no error is detected, the system advances to the next functional state, otherwise it will operate on an error state, from which it will execute error-related services and return to the previous state. It is worthy to mention that this state classification and transition are asynchronous.

For 3 different users, we created a routine that is composed of 4 different activities from a set of 5, which could be done at either Location A or Location B.

TABLE VI

VARIABLES AND ASSOCIATED LABELS.

Z	User	Activity	Location
l_n	Luis	Sitting	A
	Paulo	Walking	B
	Goncalo	Watching TV	
		Eating	
		Hygiene	

Let the following Figure 6, show an example of a 4 activity sequence for one of these users.

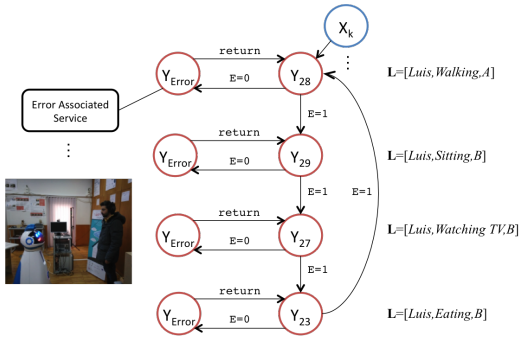

Fig. 6. Luis Behaviour State Model.

The diagram show the expected routine of each of these users. At the moment, the transition matrix is deterministic, which means that $t_{ij} = 1$ for $ij \in \{11, 12, ...\}$ and "0" for

all other ij pairs. At a given time, the robot will attempt to classify the current user behaviour state by collecting all relevant input for the expected behaviour state transition. In the current implementation of the Error parameter, the system only knows that, at least, one of the labels $l_n \in \mathbf{L}$ is wrong. In the tested model, it means that either the user identity or the location or the activity are wrong. In the presence of one or more of these situations, the robot will issue an error message. The robot is instructed to navigate to the correct place, therefore errors were caused by either detecting the wrong person or misclassifying the action and allowed the robot to issue a sound alarm.

V. CONCLUSIONS

The proposed Hyper-Label inference model and the associated experiments allow for the following conclusions. The implemented framework, if given a statistically significant training data will provide a precision/recall performance over 90% for a large numbers of possible states $y_p \in Y$. However, results also show that on the event these two indicators do not perform so well, if we decode the Hyper-Label y_p state into its correspond label set vector L, we are still able to retrieve over 96% of usable information ($= 1-\text{HL}$) of accurate information. This means that we are still able to select an appropriate reaction of the system based on the decoded information vector. Instead of having multiple classifiers, we have one that fuses different input instance types, from different sources (local and remote) to retrieve sparse multi-label information. The downside of the proposed approach is that for large numbers of labels we need very large training samples.

On the proposed hybrid state transition model, it is possible to state that the robot is able to correctly identify states that go out of the user's typical routine deterministically, causing it to enter an error state and selecting an appropriate response. However, in cases where Y_t vectors are stochastic, it might be useful to compute an error matrix rather than an error value. Based on the sparsity of the error matrix, we expect to be able to develop a error regulation model that will select from a range of possible interactions.

Also as a future work, we intent to evaluate different methodologies for the classification of the Hyper-Labels and to extend our experimental set-up with long run experiments. Also, to overcome the training sample size, we will study methods of reducing the dataset without compromising classifier performance.

ACKNOWLEDGMENT

This work has been supported by the European Union's Horizon 2020 research and innovation programme - Societal Challenge 1 (DG CONNECTH) under G.A. No 643647.

This work has been partially supported by Institute of Systems and Robotics from University of Coimbra, Portugal and by the Khalifa University, Abu Dhabi, UAE.

REFERENCES

[1] M. Pantic, A. Pentland, A. Nijholt, and T. Huang, "Human computing and machine understanding of human behaviour: a survey," in *8th International Conference on Multimodal Interfaces*. ACM, 2007, pp. 239–248.

[2] A. Batliner, K. Fisher, R. Huber, J. Spilker, and E. Noth, "How to find trouble in communication," *Speech Communication*, vol. 40, pp. 117–143, 2003.

[3] M. J. Choi, A. Torralba, and A. S. Willsky, "A tree-based context model for object recognition," *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 34, pp. 240–252, 2012.

[4] A. Iosifidis, A. Tefas, and I. Pitas, "View-invariant action recognition based on artificial neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, pp. 412–424, 2012.

[5] M. F. Valstar, M. Mehu, B. Jiang, M. Pantic, and K. Scherer, "Meta-analysis of the first facial expression recognition challenge," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 42, pp. 966–979, 2012.

[6] M. el Ayadi, M. S. Kamel, and F. Karray, "Survey on speech emotion recognition: Features, classification schemes and databases," *Elsevier Pattern Recognition*, vol. 44, pp. 572–587, 2011.

[7] K. Karpouzis, G. Caridakis, L. Kessous, N. Amir, A. Raouzaiou, L. Malatesta, and S. Kollias, "Modeling naturalistic affective states via facial, vocal, and bodily expressions recognition," *Artifical Intelligence for Human Computing*, vol. LNAI 4451, pp. 91–112, 2007.

[8] D. L. McGuinness, R. Fikes, J. Rice, and S. Wilder, "An environment for merging and testing large ontologies," in *Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR2000)*, 2000.

[9] A. Osterwalder and Y. Pigneur, "An ontology of e-business models," *Currie WL, Value creation from e-business models*, vol. Elsevier Amsterdam, pp. 65–97, 2004.

[10] e. a. Andre Schiltz, "Business model analysis of ehealth use cases in europe and in japan," *Journal of the International Society for Telemedicine and eHealth*, vol. 1, pp. 30–43, 2013.

[11] Y. Aloimonos, "Sensory grammars for sensor networks," *JAISE*, vol. 1, no. 1, pp. 15–21, 2009. [Online]. Available: http://dx.doi.org/10.3233/AIS-2009-0002

[12] D. R. Faria, R. Martins, J. Lobo, and J. Dias, "Extracting data from human manipulation of objects towards improving autonomous robotic grasping," *Elsevier: Special Issue on Autonomous Grasping*, vol. 60, pp. 396–410, 2012.

[13] L. Chen, C. Nugent, and G. Okeyo, "An ontology-based hybrid approach to activity modelling in smart homes," *IEEE Transactions on Human-Machine Systems*, vol. 44, pp. 92–105, 2014.

[14] M. Aly, "Survey on multiclass classification methods," 2005.

[15] E. A. Cherman, M. C. Monard, and J. Metz, "Multi-label problem transformation methods: a case study," *CLEI Electronic Journal*, vol. 14, no. 1, pp. 1–10, 2011. [Online]. Available: http://www.clei.cl/cleiej/paper.php?id=215

[16] F. Tai and H.-T. Lin, "Multilabel classification with principal label space transformation," *Neural Comput.*, vol. 24, no. 9, pp. 2508–2542, Sept. 2012. [Online]. Available: http://dx.doi.org/10.1162/NECO_a_00320

[17] K. P. Murphy, "Dynamic bayesian networks: representation, inference and learning," Ph.D. dissertation, University of California, Berkeley, 2002.

[18] C. Friedman, "The frequency interpretation in probability," *Advances in Applied Mathematics*, vol. 23, no. 3, pp. 234 – 254, 1999. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S019688589990653X

[19] J. F. Ferreira, J. Lobo, P. Bessière, M. Castelo-Branco, and J. Dias, "A bayesian framework for active artificial perception," *IEEE Transactions on Cybernetics (Systems Man and Cybernetics, part B)*, vol. 43, pp. 699–711, 2013.

[20] L. Santos, K. Khoshhal, and J. Dias, "Trajectory-based human action segmentation," *Pattern Recognition*, vol. 48, no. 2, pp. 568 – 579, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S003132031400329X

[21] P. Bessiere, J.-M. Ahuactzin, K. Mekhnacha, and E. Mazer, *Bayesian Programming*. Taylor & Francis, 2012.