Context Feature Learning through Deep Learning for Adaptive Context-Aware Decision Making in the Home

Alexis Brenon, François Portet, Michel Vacher

Context Feature Learning through Deep Learning for Adaptive Context-Aware Decision Making in the Home

Alexis Brenon, François Portet, Michel Vacher
Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG, F-38000 Grenoble France
alexis.brenon@imag.fr,francois.portet@imag.fr, michel.vacher@imag.fr

***Abstract*—In Intelligent Environments, prediction and decision must take the context of interaction into account to adapt themselves to the evolving environment. If most of the approaches to deal with this problem have used a formal representation of context, we present in this paper a direct extraction of the context from raw sensor data using deep neural network and reinforcement learning. Experiments undertaken in a voice controlled smart home showed which elements are useful to perform context-aware decision-making in the home and the adequacy of reinforcement learning to tackle an evolving environment.**

***Index Terms*—smart home, context-aware computing, Ambient Intelligence, reinforcement learning**

I. Introduction

Intelligent Environments, such as smart homes, are fit with numerous sensors and actuators to provide services that enhance users' experience and control over their environment [1]. To make this enriched control possible, these systems perceive their environment and decide which actions to apply to the environment. This decision can be made after some specific events such as a user's request, a scheduled event (e.g., programmed task) or a detected risky situation. This perception is not only useful to trigger a decision but also to adapt the decision making to the current circumstances in which such decision must be performed. These circumstances are called *the context* and systems that explicitly take the context into account, are said *context-aware*. However, defining explicitly what pieces of information is playing an important role in the context of a decision is a difficult and tedious task. Indeed, if a general definition of context exists in the field [2], the large amount of existing models [3], [4] demonstrates not only the importance of this notion for research but also the lack of a unified way of modelling context in intelligent environments.

In this paper, we present an approach to perform context-aware decision for voice command in a smart home that does not use an explicitly defined context. Rather, the context information is directly learned from data using deep neural network [5], [6]. Indeed, one of the main advantages of Deep Learning (DL) is to learn attributes from raw data at the same time as the classification model is trained. This ability of DL has been demonstrated in numerous classification tasks reaching superhuman accuracy in the case of image classification [6]. Apart from the identification of necessary elements of the context, another aspect of context awareness is to be able to capture a changing context over time. Indeed, sensors, activity places, dwellers, etc. can change and any system that targets a lifelong support must be able to take these changes into account. To adapt continuously to a changing environment, the decision model we present in this paper uses reinforcement learning to update its decision policy.

The paper is structured as follows. Section II presents the method to represent sensor data for neural network and the reinforced adaptation loop with the decision model acquired using deep learning. A smart home is fit with many heterogeneous sensors that are highly different in term of semantic and dynamic (e.g., temperature, contact-door, speech), thus, this section describes the method used to project this raw information into a unique representation: an image. Moreover, two decision models using different abstraction levels of contextual information are detailed. In Section III, the methods are experimented and compared using a real smart home corpus. The results give an insight of the capability of DL to make decision based on raw data and shed light on the most important contextual information used by the decision model. The approach is then positioned with respect to related work in Section IV before the paper ends with a short conclusion.

II. Methods

This section presents the adopted data representation (II-A) and the reinforced adaptation loop with the decision model acquired using deep learning (II-B).

A. A common representation for heterogeneous information: a raw image approach

In smart-homes, data are multi-modal: from continuous time series, such as the overall water consumption to event based semantic information such as voice commands or the state change of a device. Hence, finding a representation accommodating low-level continuous values (e.g. water consumption) and high-level categorical events (user current activity) is a problem that has not found a consensual solution in the community. We propose to use a **graphical representation**, projecting data on a two-dimensional map of the smart-home any time the environment state changes. Then, relevant features from this raw representation are extracted by Convolutional Neural Networks (CNN). This approach provides some advantages:

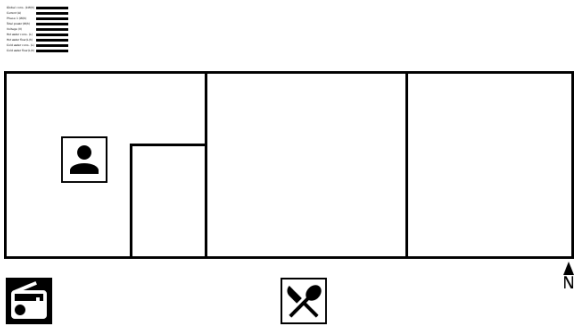

Fig. 1. Example of generated image from annotations

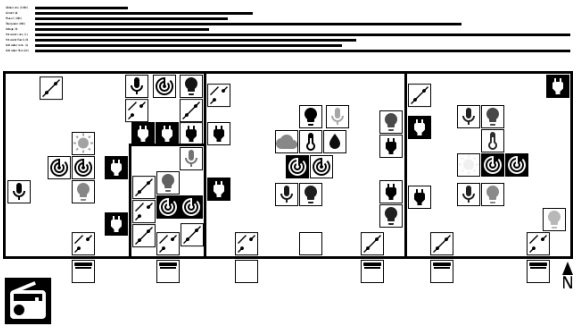

Fig. 2. Example of generated image from raw data

1) Image generated can be used by the user to monitor what the system sees or understands;
2) Resulting image size is only loosely linked to the number or type of sensors (adding, removing, or changing some sensors will not change the size of the image), hence images accommodate a variable number of data sources (sensors can come and go) a property that classical input vector machine learning approaches do not *naturally* handle;
3) Image *naturally* convey spatial information (here an *a priory* information about the home room organisation);
4) High redundancy of sensor values can be efficiently used by deep learning.

Examples of generated images for the smart-home considered in the study are provided in Figures 1 and 2 which represent the same situation: the user asks to turn off the radio while she is eating in the kitchen. Black lines represent the walls of the home so that spatial information is provided to the system. To capture context, two representations were chosen. The **annotated** one, represented Figure 1, uses icons to display humanly annotated contextual information such as location of the user, activity and voice command. In the **raw** one (Fig. 2), only raw sensor data are represented. For instance usage of light is represented by a black lamp on white background while a turned off lamp is a white lamp on dark background. The highest part presents some gauges for electricity and water consumption while the icon at the bottom left is the uttered command. Images and icons were chosen to maximize the contrast and redundancy for the CNN, to be understandable by human and to be easy to generate on the fly.

B. Decision adaptation through reinforcement learning

The method to continuously adapt the decision model to the situation and the user follows a Reinforcement Learning (RL) strategy. In a RL problem, two components are interacting with each other: the *environment* and the *agent*. At each interaction, the agent (here the decision-making system) is fed with the current environment state $s_t \in \mathcal{S}$. From s_t and the past, it chooses an action $a_t \in \mathcal{A}$ to be applied to the environment. The environment (here the house and the user) state is modified by the action (from s_t to s_{t+1}) and reacts by providing a reward to the agent. This reward $R(s_t) = r_t$ can be positive if the action is appropriate, negative if the action is not adequate, or null. Based on this reward, the agent adapts its decision policy.

While dynamic programming [7] can solve the problem of finding the optimal policy, it cannot be used in real settings due to missing information about the environment model and the reward function. Instead, the *Q-Learning* method [8], can be used to compute and update a policy which will converge toward the optimal one. In Q-Learning, a function, named Q-*function*, associates a value, named Q-*value*, to each possible pair of state-action, named Q-state: $Q : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$, where $\mathcal{S}$ is the set of possible states of the environment, and $\mathcal{A}$ is the set of possible actions to perform. Beginning with an initial Q-function (Q_0), it can be updated after each interaction $\langle s_t, a_t, r_t, s_{t+1} \rangle$, following Equation 1 (where γ is the discounting factor of the reward function and α is a learning rate). This updating rule has been proved to converge toward the actual Q-value of each Q-state, making it possible to compute the optimal policy.

$$Q_{k+1}(s_t, a_t) = \alpha\big(r_t + \gamma max_a Q_k(s_{t+1}, a)\big) + (1-\alpha) Q_k(s_t, a_t) \quad (1)$$

The learning method is highly dependent on the way the Q-function is actually represented. For instance, classical implementations choose a tabular representation: a two-dimensional array of Q-values, indexed by states (rows) and by actions (columns). However, this approach does not scale to big or continuous sets of states or actions and does not make it possible to use redundancy in the learning.

Instead of a tabular representation, a deep neural network can be used to approximate the Q-function. In [9], a new approach, Deep Q-Network (DQN), has been developed to model the Q-function and applied to a game decision problem given a sequence of video game image frames. Although the work in [9] is not the first attempt to use neural network to model the Q-function [10], it is the first which succeeded to implement an efficient learning. Thanks to different optimisation techniques such as the use of a target network, experience replay [11], mini-batch learning or RMSProp gradient optimisation [12], the learning can converge in reasonable time and is able to scale to large images[1]. This is the approach we adopted and

[1]Due to space constraint the reader is refereed to Mnih *et al.* [9] for details.

adapted to context aware smart-home decision-making and that is succinctly exposed in the remaining of this section.

The decision model is based on a Convolutional Neural Network (CNN) to process the images followed by a fully connected neural network to compute the Q-value for each action. Figure 3 presents the architecture of our model, the Q-network. The neural network is composed of four *convolutional layers*, reducing a 256×256 input image to features maps of sizes: $16 \times 64 \times 64$, $32 \times 21 \times 21$, $64 \times 10 \times 10$, and $64 \times 8 \times 8$. The features maps of the last layer are then rearranged in a vector of dimension $64 \times 8 \times 8 = 4096$. This vector passes through *fully connected linear layers* of width 512 and 33 which is the number of possible actions. A *rectified linear unit* follow each layer except the last one.

Thus, this model provides a way to process the generated images presented in II-A. After preprocessing (scaling and reducing the depth of the image), the image is passed forward in the Q-network. The network builds and extracts relevant features used to compute the Q-value of each possible action. Thus the network can be seen as a the function $Q_{-network} : [0, 255]^{256\times256} \rightarrow \mathbb{R}^{33}$, where 33 is the number of actions and 256 the size of one side of the image. Then, Equation 1 detailed earlier can be used as a loss function to update the weights of the Q-network, as in typical deep learning approaches.

The overall learning of the decision-making module (the agent) is decomposed into two phases: a *pre-training phase* (i.e., initialisation of the agent), during which the model is learned from scratch and an *adaptation phase* during which the model is tuned toward the environment. In practice, the pre-training phase is performed off-line and is mostly devoted to the learning of the CNN and the subsequent neural network. It is well-known that CNN needs a high amount of data ; that is why, in our work, we used artificially generated data to feed the CNN with raw input to learn the CNN weights. In the adaptation phase, the CNN is not supposed to evolve but the last part of the neural network (i.e., the decision part) is biased towards the adaptation data. These data can be either acquired on-line or performed on corpus if available.

In any of these phases, reinforcement learning method is the same. Starting from an initial agent, the system makes $maxSteps$ interactions, in which the observable state of the environment is passed to the agent, before it chooses an action applied back to the environment which releases a reward. All these interactions are recorded by the agent which uses them during a learning procedure. This training epoch (interactions plus training) is realised a number $maxEpoch$ of times.

Since during the pre-training phase the agent explores artificially created data, the environment is not able to evolve consequently to a wrong action. Thus, to multiply the number of situations and explore wrong decisions, up to $triesThreshold$ actions can be successively tried to simulate the patience of the user, repeating the same command until the system finally behaves correctly. For the *pre-training phase*, the agent executes a near greedy policy named ϵ-greedy policy [13]. Given $\epsilon \in [0; 1]$, this policy returns a random action with probability ϵ for the action with the highest Q-value given the current state.

To compute the Q-value of each action, the agent uses its approximated Q-function, modelled by the deep neural network Q-Network. However, to ensure a stable learning, the method does not use the latest interaction to compute the error but a set of past interactions from which a (mini-) batch is randomly drawn to train the network. Using this batch, an updated Q-value named *target-value* is computed and the difference between current Q-values (immediates) and *target-values* (from a set of past interactions) is used as a loss and propagated through the Q-Network [9].

III. Experiments

The models were trained in two phases: the **pre-training phase**, where the agent is trained on artificial data and the **validation phase** which performs a cross-validation on a real corpus. Then, the approach learned from **raw** sensor data is analysed to explore the kind of sensors and contextual information that are used to make decision. Before entering into the details of the experiment, the input data are presented in the following section.

A. Input data

1) Smart-home Corpus: The dataset used for the validation phase is part of the publicly available Sweet-Home corpus [14]. It was recorded in the DOMUS smart-home designed by the Laboratoire d'Informatique de Grenoble (LIG) [15]. This 30m^2 home includes a bathroom, a kitchen, a bedroom, and a study as shown in Figure 4. All these rooms are equipped with sensors and actuators such as infrared motion sensors, touch sensors, video cameras (only used for annotation purposes), etc. In addition, seven microphones in the ceiling capture the audio. More than 150 sensors are managed in the flat to provide different services (e.g. light, opening/closing shutters, media management, etc.).

In this study, 15 naïve participants (9 women, 6 men) were recruited to play scenarios of the daily life in the flat. To make the data more natural, the participants were asked to enact the following situations:

- to clean up the flat;
- to prepare and eat a meal;
- to converse via video conference;
- to do leisure activities (reading);
- to take a nap.

In each situation, participants were asked to issue a number of voice commands to activate the actuators in the smart home. The objective of this experiment was to test a smart controller in real situations corresponding to voice commands pronounced by the user.

2) State and action sets: A total 11 hours of data of every sensor and actuator were recorded. The smart-home corpus was synchronized and annotated [16]. The annotations consisted in a set of situations, composed of the user's uttered command, the user's location and the user's current activity. This input state is associated to the expected output

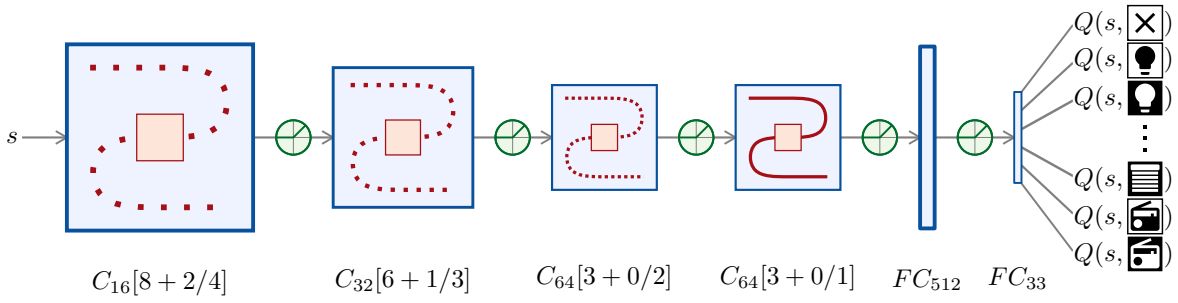

Fig. 3. Architecture of the Q-network. s is the input state (image) and $Q(s,x)$ is the Q-value for action x given state s. $C_n[f+p/k]$ denotes a convolution layer (using n filters of size f, with a padding of p, and using a stride of k) and FC_n a fully connected linear layer (of width n).

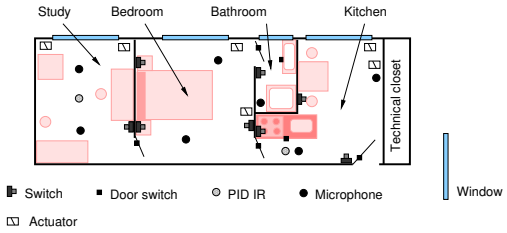

Fig. 4. The Domus smart-home

decomposed in command type and a location about where to execute the command. For instances, the following excerpt from the dataset shows that a voice command to open the blinds in the kitchen should be followed by the opening of all the blinds of the kitchen. On the second line, when the light is asked to be turned on while cooking in the kitchen, the only possible action is to light on the lamp above the sink.

```
blind - open   kitchen   none -> blind - open   kitchen
light - on     kitchen   cook -> light - on     kitchen - sink
```

This results in a state space of size $324 = 12$ voice commands $\times 3$ locations $\times 9$ activities.

The raw states can take more different possible values. 81 sensors were recorded in the smart home:

- 49 binary sensors;
- 32 continuous sensors which can take values in different ranges (from $[0;100]$ up to $[0;2^{32}-1=4294967295]$).

This leads to more than 10^{12} possible states.

From the neural network point of view, a state is an image of size $256 \times 256 \times 3$ pixels, each taking a value between 0 to 255. The resulting state set have a size of $256^{256\times256\times3} = 2^{1572864} \approx 1.75e473479$. From which one out of 33 possible actions must be chosen.

3) Generated corpus: Unfortunately, the smart-home corpus does not contain enough relevant samples of voice commands to perform deep-learning from it. As a workaround, a data generation system was developed based on a few generation rules. Two types of corpus were generated: the annotated corpus and the raw corpus. The raw corpus was generated to use the ability of deep learning to build its own features from raw data. The generation consisted in generating annotations for the voice commands and the expected actions as well as simulating values for the raw sensor data. This was performed by producing a random *annotated* state which is then converted using a set of rules. For example, if the annotated location is the kitchen, then the presence detector sensors of the kitchen are constrained to be active. In the meantime, the bedroom being the room next to the kitchen, its presence detector sensors are constrained to be active with a defined probability. This mechanism is applied for all relevant sensors, while random values are applied to irrelevant ones.

B. Learning results

The two network architectures and image inputs have been trained on artificial data and evaluated on the real Sweet-home corpus. For the DQN with annotated data, the *pre-training phase* used annotated simulated data composed of annotations about the user's command, location and activity resulting in images with only three icons (Figure 1).

For the DQN with raw sensor data (Figure 2), images were generated during the pre-training phase using a synthetic corpus where all data streams were perfectly synchronized. However, the real smart-home corpus does not provide this quality of synchronization between heterogeneous data streams. To overcome this problem, we chose to provide to the Q-network a wider window of the last 3 seconds of the state of the environment. As a consequence, many samples where no action was required by the agent were introduced (2 samples of no action for any of the 32 other actions), leading to an unbalance dataset biased towards the *no action* class. To compensate for this learning bias, the reward function was updated so that instead of receiving a reward of (-1), a reward of (-192) is provided if the agent select the *no action* class when an action is expected.

Once a model is learned from scratch, the *evaluation phase* adapts it to new data. The **Leave One Subject Out Cross-Validation** (LOSOCV) method is adopted. This cross validation method keeps one subject file for testing and uses the remaining one for learning, each file in turn. Each fold of the LOSOCV method runs the system for $300,000$ interactions on the real annotated data. Test of the model happens every $6,000$ interactions, running $5,000$ monitored interactions.

A **baseline** system using a classical tabular Q-learning has been learned and tested by LOSOCV. However, since the classical approach uses a matrix to represent the state, this

approach would not be tractable if it was applied directly to the large amount of sensors values (cf. sec II-B). Hence, the baseline was trained on the humanly annotated location and activity labels, which favour the baseline performances.

Table I summarises the result of the learning step. Even though the reinforcement learning objective is to maximise the reward, in this paper, standard classification measures such an F-Measure have been chosen. Indeed, the task can be seen as a classification one, classifying a sample (the environment state) between 33 classes (the actions). In our experiment, a correct classification is considered only when the chosen action is composed of the right action (e.g., turn on) on the right device (e.g., light) at the right place (e.g., study). Any other choice is a wrong classification.

TABLE I
F-MEASURE OF THE LEARNING AND EVALUATION PHASE FOR THE DECISION MAKING TASK.

	Testing F-M (%) synthetic data	Validation F-M (%) real corpus
baseline*	NC	46.0
DQN & annot.*	100	79.0
DQN & raw	100	67.5

* learned and evaluated on humanly annotated data.

While the baseline only deal with humanly annotated data, it presents a low F-Measure of 46%. Regarding the DQN on annotated data the adaptation of the model lead to an average reward of 0.52 and an F-M of 79%. When a DQN is trained and validated on raw sensor data, the score reaches more than 65% of F-Measure. Both DQN approaches demonstrated results far better than the baseline system [16]. The DQN on annotated data exhibits the highest performances but was trained on perfectly humanly annotated high level contextual data (i.e., activity and location) which is not consistent with real usage in which contextual information is inferred and infected with errors. Although, the DQN on raw sensor data does not lead to the best performances, it demonstrates that a context aware decision-making model without an explicit representation of context is possible and lead to a very promising result of 67.5% of F-Measure. An interesting question is to know what kind of information this DQN extracts to make a decision.

C. Analysis of information used by the deep-model

While an associative table can be interpreted for a tabular Q-learning problem, it is more complex to analyse a neural network. This is a recurrent grief addressed to neuronal models, and thus methods have been developed to better understand how they work [17]. We present two complementary analyses. The first one consists in removing some sensor sources in the learning to test their usefulness and the capability of the learning process to overcome the sudden lack of information. The second analysis investigates which features the neural network builds, and if they match the ones usually found in the literature.

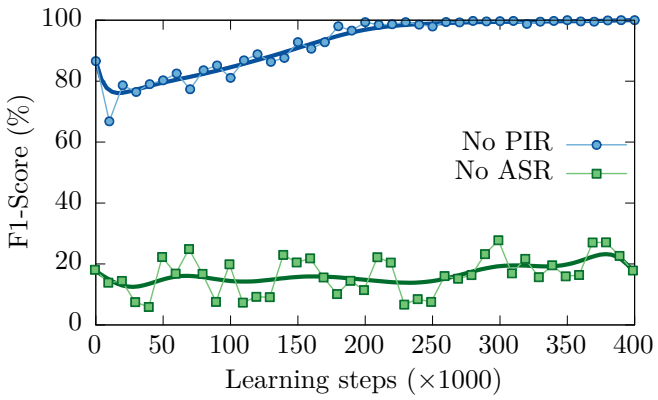

Fig. 5. Learning with a missing information: in blue kitchen motion sensors information (from PIR — Presence Infra-Red — sensors), in green voice command information (from the ASR — Automatic Speech Recognition— system).

1) Sensors usage by the deep model: Four experiments were performed, based on a pre-trained (with synthetic raw data) model, in which different kinds of sources of information were removed. As shown on Figure 5, in case of removal of the kitchen motion sensors and during the adaptation phase, the performances are impacted but the system quickly adapts its behaviour to reach back its initial performances. Thus, although this location source is important, the network uses the other available sources to counterbalance this loss of information. Rather, when the speech source is removed, a dramatic fall of the performances appears. However, the same Figure 5 shows that the system still acts with an higher F-Measure (around $20 + -10\%$) than randomly (a random choice would get about 3% of F-M). This behaviour could be explained by the fact that the system still gets images 2 seconds before an action is expected. This historical information seems to have been used during the pre-training to know when an action is expected. However, the oscillations of the score indicate that the system is not able to infer the user intention only from the sensor values (i.e. from contextual information). The speech signal is thus highly necessary to get user's intention. Two others sensors were removed in the kitchen: the sound detectors and the door switches for kitchen cupboard and fridge. The former are particularly active when the user tidies the home, and one would expect a drop in the performances. But, no impact in the F-Measure performances were noticed while this information is of particular importance for activity recognition.

2) State features: During its forward pass through the network, the input image is transformed to extract relevant features, resulting in high-dimensional vectors. One way to study these vectors is to project them in a two-dimensional space while preserving some inter-vectors relations such as the euclidean distance. One of the most used projection methods is the t-SNE (t-distributed Stochastic Neighbour Embedding) projection [18] which performs very well to project vectors issued by a convolutional neural network.

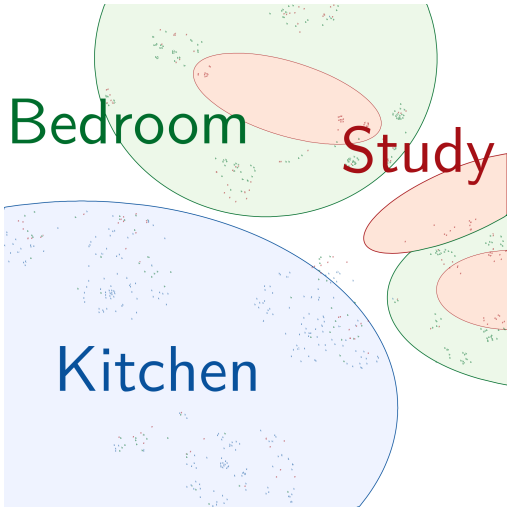

Fig. 6. Post-convolution t-SNE projections: User location

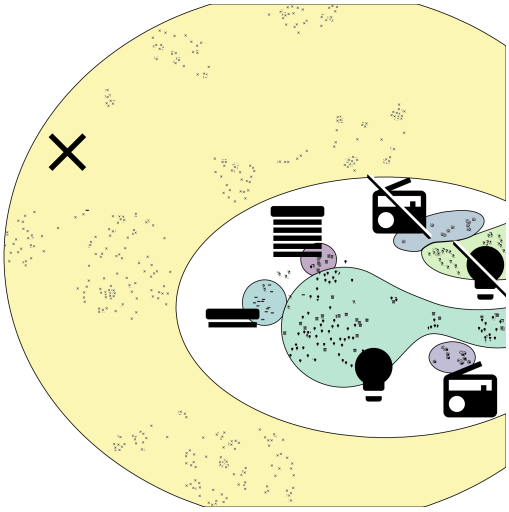

Fig. 8. Post-convolution t-SNE projections: Uttered command

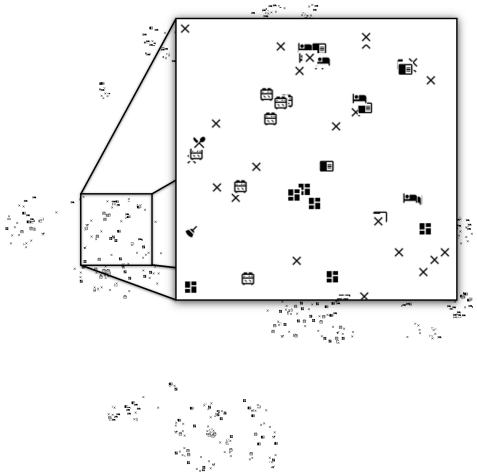

Fig. 7. Post-convolution t-SNE projections: User activity

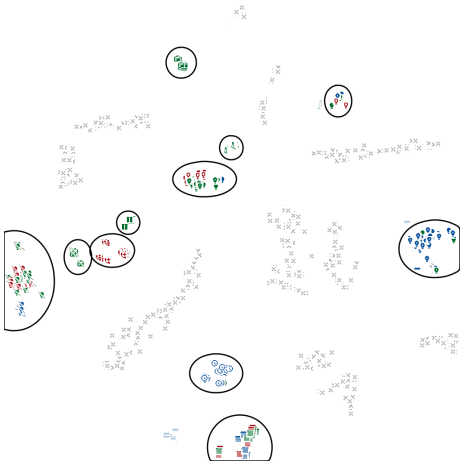

Fig. 9. Pre-classification t-SNE projection representing the expected action

In the following experiment, we provide some states (images) to a pre-trained model, while recording vectors after the last convolutional layer, and the one before the last fully connected layer. These vectors are also linked to the information characterising the input state (location, activity, uttered command, expected action, sensors activity). Projecting the vectors provide a bi-dimensional coordinate $(0,0) \leq (x,y) \leq (1,1)$ able to represent any of the linked information.

Results presented in Figures 6 and 8 show that some information is identified and extracted by the convolutional part of the network. For example, the network projects input data into blobs that are coherent both with respect to the users' location (colored area in Fig. 6) and with respect to uttered commands (colored area in Fig. 8). By contrast, no group are spotted for human activities (Figure 7). On the zoomed zone very different activities such as `do nothing` (the cross), `eating` (crossed knife and spoon) and `cooking` (oven) are projected in the same area. These results confirm the interest of user location as a contextual information but user activity does not seem to be a relevant feature for the neural network to estimate decision. However, if semantic labelling of the activity is not exploited by the network, more low-level activity measures, such as agitation level, could be studied.

The last projection, presented in 9, provides information about the way the neural network transforms the space to make each class linearly separable. While there is a majority of *Do Nothing* action, we can identify some groups of common action type (icon type) and action place (icon colour), proving that images of state contain enough information to decide which decision to make.

IV. Related works

Typical smart-homes considered in the study are the ones that permit voice based interaction. This kind of smart homes is particularly suitable to people in loss of autonomy [19], [20]. Although it recently gained interest in the speech processing community [21]–[26] none of these work formalises the voice command recognition as a context-aware decision problem. In that case, identifying what contextual information is needed

to understand the intention of the user for it utterance is of particular importance.

Regarding contextual information and context extraction, research on context aware systems has generated a vast amount of knowledge representations. The main trend for knowledge representation in smart environments has become ontology, in particular those based on Description Logic [27], [28]. This approach is also popular to model the current context for mobile applications [29], [30]. However, Description Logic has not been originally designed to model imprecision in sensors data or inferences that is why extension with fuzzy representation had been proposed [31]. Another important contribution to make ontologies capable of dealing with uncertainty is the use of probabilistic models in ontology inference. A notable application of such approach has been presented by [32] and by [33].

In all of these approaches, the context modelling is considered from a top-down perspective, where elements of the context are identified by expertise. In this paper, we propose a different approach: a data-driven constitution of the context where all the sensor data are unified into a common representation: an image. The closest work to this approach is the recent one by [34] which transforms pressure sensor data into image domain to perform gait identification. In this work, a pressure sensor matrix is converted into an image to perform CNN and RNN learning. However, in their work the data were already sequences of 2-dimensional values while in our case we dealt with a set of monodimensional sensor data of variable type (Boolean, categorical, numerical) projected in the schema of the home. Our approach is thus more adapted to context-awarness in smart home and provide the user with an understandable data representation.

Regarding the decision-making process, several logic-based approaches are presented in the literature using description logic [35], [36] or fuzzy logic [37]. However, in these proposals the system is set *a priori* to execute an action given a specific configuration (the condition) and consequently the system does not adapt its behaviour to a changing situation. Some research work on context aware systems have relied on Bayesian Network [38]–[40] or Markov Logic Network [41] which brings the advantage of being learned from data. However, these approaches do not include mechanism to adapt the decision model to new situations. Furthermore, if part of the model can be learned from data, the logic part is acquired by human expertise which has the same drawback as the logic based knowledge representations in smart-home.

To tackle the problem of adaptation, some researches in smart-home have designed the decision problem as a reinforced decision-making problem. One of the most famous of these works is *The Neural Network House* [42] where a smart home was controlled by the ACHE system [43], which commanded the lights in order to reduce the overall energy consumption without impairing the inhabitant comfort based on the perceived environment. Although this project showed promising results and gave rise to other research [44], [45], the RL technique employed poorly scales to large real cases sets. As demonstrated in [9], a way to solve this issue, is to change the data structure, relying for example on a deep neural network [5], [6]. Following the work of [9] who proposed *deep reinforcement learning* called Deep Q-Network (DQN), we adapted this approach for the first time to the smart home domain and used it to exhibits the kind of context information that is useful for adaptive decision-making.

V. Conclusion

The results of the experiments of a reinforcement learning method with a deep model for automatic adaptive decision-making in smart-homes show promises. In particular, the experiments show that:

- the deep model can learn the relevant contextual information from raw data;
- the system can adapt its model to an evolving set of sensors;
- graphical representation of smart-home multi-modal heterogeneous data can be interpreted by the system to make correct decisions;
- the method can scale from small input spaces (annotated data) to big input space (raw sensor data).

Despite the model learned from raw data shows a slightly weaker F-Measure than the one learn from annotated high-level contextual input, this former model make it possible to do without specialised external inference models. Indeed, in typical settings, user's location, activity and decision are processed independently to form a processing pipeline. While this approach has its own virtue by making the information formally defined and shareable, this processing chain must deal with the uncertainty propagated along the pipeline and the difficulty of adapting the whole chain to changes in the smart-home. With a deep model and reinforcement learning, the context is automatically learned and tuned toward the decision task and is thus more robust to change in the home.

However, Deep Learning is very demanding in training data and computation time. For instance, in the raw data settings, it took about 8000000 interactions (which correspond to 100 days in real time) before the system acquires a coherent behaviour. It took more than 4 days and 16 hours to compute this experiment on the recorded corpus. While the learning is far too long for a home, we have shown that using artificially created data, this learning phase can be subsequently adapted to new data in a time comparable to tabular Q-learning [16].

In conclusion, Deep Learning generates great expectations given its success in many tasks in data processing and reasoning. But, beyond the shift in performance, what is probably the most interesting from a scientific perspective is its ability to learn features from raw data to inform us about which information comes into play when solving a problem such as context-aware decision-making. Although the model used in this study (neural network) does not reach the level of expressiveness of logic based approach, its ability to extract features from raw data can play a role in defining what the most important context elements might be in intelligent environment applications.

REFERENCES

[1] J. C. Augusto, V. Callaghan, D. Cook, A. Kameas, and I. Satoh, "Intelligent environments: a manifesto," *Human-centric Computing and Information Sciences*, vol. 3, no. 1, p. 12, Jun 2013.

[2] A. K. Dey, "Understanding and using context," *Personal Ubiquitous Computing*, vol. 5, no. 1, pp. 4–7, 2001.

[3] C. Bolchini, C. A. Curino, E. Quintarelli, F. A. Schreiber, and L. Tanca, "A data-oriented survey of context models," *SIGMOD Rec.*, vol. 36, no. 4, pp. 19–26, 2007.

[4] U. A. Ibarra, J. C. Augusto, and T. Clark, "Engineering context-aware systems and applications: A survey," *Journal of Systems and Software*, vol. 117, pp. 55–83, 2016.

[5] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[6] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[7] R. Bellman, *Dynamic Programming*. Princeton University Press, 1957.

[8] C. J. C. H. Watkins, "Learning from delayed rewards," Ph.D. dissertation, King's College, 1989.

[9] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[10] J. Tsitsiklis and B. V. Roy, "An analysis if temporal-difference learning with function approximation," *IEEE Trans. Automat. Contr.*, pp. 674–690, 1997.

[11] L.-J. Lin, "Reinforcement learning for robots using neural networks," DTIC Document, Tech. Rep., 1993.

[12] G. E. Hinton. (2015) Lecture 6e – rmsprop: Divide the gradient by a running average of its recent magnitude. Online courses. [Online]. Available: http://www.cs.toronto.edu/ tijmen/csc321/slides/lecture_slides_lec6.pdf

[13] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. MIT Press, 2017. [Online]. Available: http://incompleteideas.net/sutton/book/the-book-2nd.html

[14] M. Vacher, B. Lecouteux, P. Chahuara, F. Portet, B. Meillon, and N. Bonnefond, "The sweet-home speech and multimodal corpus for home automation interaction," in *The 9th edition of the Language Resources and Evaluation Conference*, 2014, pp. 4499–4506.

[15] M. Gallissot, J. Caelen, F. Jambon, and B. Meillon, "Une plateforme usage pour l'intgration de linformatique ambiante dans l'habitat. l'appartement domus," *Technique et Science Informatiques (TSI)*, vol. 32, no. 5, pp. 547–574, 2013.

[16] A. Brenon, F. Portet, and M. Vacher, "Preliminary study of adaptive decision-making system for vocal command in smart home," in *12th International Conference on Intelligent Environments*, 2016, pp. 218–221.

[17] A. Karpathy. (2015) Cs231n: Convolutional neural networks for visual recognition. Web site. Standford. [Online]. Available: https://cs231n.github.io/

[18] L. J. van der Maaten and G. Hinton, "Visualizing high-dimensional data using t-sne," *Journal of Machine Learning Research*, pp. 2579–2605, 2008. [Online]. Available: https://lvdmaaten.github.io/tsne/

[19] F. Portet, M. Vacher, C. Golanski, C. Roux, and B. Meillon, "Design and evaluation of a smart home voice interface for the elderly: acceptability and objection aspects," *Personal and Ubiquitous Computing*, vol. 17, pp. 127–144, 2013.

[20] M. Vacher, S. Caffiau, F. Portet, B. Meillon, C. Roux, E. Elias, B. Lecouteux, and P. Chahuara, "Evaluation of a context-aware voice interface for ambient assisted living," *ACM Transactions on Accessible Computing*, vol. 7, no. 2, pp. 1–36, 2015.

[21] D. Istrate, M. Vacher, and J.-F. Serignat, "Embedded implementation of distress situation identification through sound analysis," *The Journal on Information Technology in Healthcare*, vol. 6, pp. 204–211, 2008.

[22] A. Badii and J. Boudy, "Companionable - integrated cognitive assistive & domotic companion robotic systems for ability & security," in *SFTAG 2009*, Troyes, 2009, pp. 18–20.

[23] B. Lecouteux, M. Vacher, and F. Portet, "Distant speech recognition in a smart home: Comparison of several multisource asrs in realistic conditions," in *Interspeech 2011*, Florence, Italy, 2011, pp. 2273–2276.

[24] M. Ravanelli, L. Cristoforetti, R. Gretter, M. Pellin, A. Sosi, and M. Omologo, "The DIRHA-ENGLISH corpus and related tasks for distant-speech recognition in domestic environments," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2015, Scottsdale, AZ, USA, December 13-17, 2015*, 2015, pp. 275–282.

[25] I. Rodomagoulakis, A. Katsamanis, G. Potamianos, P. Giannoulis, A. Tsiami, and P. Maragos, "Room-localized spoken command recognition in multi-room, multi-microphone environments," *Computer Speech & Language*, vol. 46, pp. 419 – 443, 2017.

[26] J. Barker, R. Marxer, E. Vincent, and S. Watanabe, "Multi-microphone speech recognition in everyday environments," *Computer Speech & Language*, vol. 46, pp. 386–387, 2017.

[27] N. D. Rodríguez, M. P. Cuéllar, J. Lilius, and M. D. Calvo-Flores, "A survey on ontologies for human behavior recognition," *ACM Computing Surveys*, vol. 46, no. 4, pp. 1–33, Mar. 2014.

[28] H.-C. Liao and C.-C. Tu, "A rdf and owl-based temporal context reasoning model for smart home," *Information Technology Journal*, vol. 6, no. 8, pp. 1130–1138, 2007.

[29] J. Attard, S. Scerri, I. Rivera, and S. Handschuh, "Ontology-based situation recognition for context-aware systems," in *Proceedings of the 9th International Conference on Semantic Systems*, ser. I-SEMANTICS 2013. New York, NY, USA: ACM, 2013, pp. 113–120.

[30] O. Yilmaz and R. C. Erdur, "iConAwa an intelligent context-aware system," *Expert Systems with Applications*, vol. 39, no. 3, pp. 2907–2918, 2012.

[31] N. D. Rodríguez, M. P. Cuéllar, J. Lilius, and M. D. Calvo-Flores, "A fuzzy ontology for semantic modelling and recognition of human behaviour," *Knowledge-Based Systems*, vol. 66, pp. 46–60, 2014.

[32] R. Helaoui, D. Riboni, and H. Stuckenschmidt, "A probabilistic ontological framework for the recognition of multilevel human activities," in *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2013, pp. 345–354.

[33] P. Chahuara, A. Fleury, F. Portet, and M. Vacher, "On-line human activity recognition from audio and home automation sensors: comparison of sequential and non-sequential models in realistic smart homes," *Journal of Ambient Intelligence and Smart Environments*, vol. 8, no. 4, pp. 399–422, 7 2016.

[34] M. S. Singh, V. Pondenkandath, B. Zhou, P. Lukowicz, and M. Liwicki, "Transforming sensor data to the image domain for deep learning - an application to footstep detection," in *International Joint Conference on Neural Networks, IJCNN 2017*, Anchorage, USA, 2017, pp. 2665–2672.

[35] M. J. Kofler, C. Reinisch, and W. Kastner, "A semantic representation of energy-related information in future smart homes," *Energy and Buildings*, vol. 47, pp. 169–179, Apr. 2012.

[36] J. Gómez-Romero, M. A. Serrano, M. A. Patricio, J. Garca, and J. M. Molina, "Context-based scene recognition from visual data in smart homes: an information fusion approach," *Personal and Ubiquitous Computing*, vol. 16, no. 7, pp. 835–857, Oct. 2012.

[37] P. Moore, B. Hu, and M. Jackson, "Rule strategies for intelligent context-aware systems: The application of conditional relationships in decision-support," in *International Conference on Complex, Intelligent and Software Intensive Systems*, Seoul, Korea, 2011, pp. 9–16.

[38] K. Mitra, A. B. Zaslavsky, and C. hlund, "A probabilistic context-aware approach for quality of experience measurement in pervasive systems," in *SAC*, 2011, pp. 419–424.

[39] T. Nishiyama, S. Hibiya, and T. Sawaragi, "Development of agent system based on decision model for creating an ambient space," *AI & Society*, vol. 26, no. 3, pp. 247–259, 2011.

[40] B. De Carolis and G. Cozzolongo, "C@sa: Intelligent home control and simulation," in *International Conference on Computational Intelligence*, Istanbul, Turkey, 2004, pp. 462–465.

[41] P. Chahuara, F. Portet, and M. Vacher, "Context-aware decision making under uncertainty for voice-based control of smart home," *Expert Systems with Applications*, vol. 75, pp. 63–79, 2017.

[42] M. C. Mozer, "The neural network house: An environment that adapts to its inhabitants," *Proceedings of the American Association for Artificial Intelligence*, 1998.

[43] ——, *Smart Environments: Technology, Protocols and Applications*. John Wiley & Sons, Inc., 2005, ch. 12 - Lessons from an Adaptive Home, pp. 271–294.

[44] S. Zaidenberg and P. Reignier, "Reinforcement learning of user preferences for a ubiquitous personal assistant," in *Advances in Reinforcement Learning*, A. Mellouk, Ed. Intech, 2011, pp. 59–80.

[45] M. Hassan and M. Atieh, "Action prediction in smart home based on reinforcement learning," in *Smart Homes and Health Telematics*, 2015, pp. 207–212.