Chapter 4

Personalization with Partial Observability

4.1 Overview

With the current integration of machine learning, the adoption of smart homes is increasingly becoming popular. Devices such as smart thermostats are often limited to ***personalized*** schedules, which may not be ideal for multiple occupants with different thermal preferences, which in turn may lead to discomfort. Accordingly, the lack of complete environment information by the smart home may create uncomfortable scenarios for users (lack of personalization). In most cases, smart homes do not have the ability to identify occupants with varying preferences due to lack of ubiquitous biometric technologies in smart homes as well as privacy issues which can be encountered as a result of using sensing systems like cameras.

In this chapter, we address this by improving our smart home model to take effective decisions when the current user is not explicitly observable. This is possible through identification of users based on their detailed thermal preferences or learning the temporal relations between the sequence of actions taken for a given activity and TH. We simulate a series of experiments with an HRL-based human model capable of

learning activities and setting its thermal preferences and an SHS that aims to learn its preferences. In this setup, the SHS agent does not have the complete information about the state of the environment (i.e. the occupant) and the SHS agent has to rely on the observation it receives at each time step. To accomplish the task of learning a policy with uncertainty, our agent maintains a belief over the human model that is pursuing its activity in the environment. Our experiments show that the SHS can learn to accurately identify the current user in the environment based on its thermal preferences given the current activity. Beyond identifying the occupant in the environment, SHS also learns the thermal preference of each activity of each identified user which can be used for further personalization.

In the next section, we describe the architecture of the smart home agent that can *a*) learn to recognize the current user and to set the TH according to its preferences, and *b*) learn the temporal relation between the embedding of the sequence of thermal preferences for multiple users. We then compare the performance of each agent for two to five human occupants. Finally, we discuss the limitation of each SHS agent and where they fail to learn the preferences. Our results show that the smart home can accurately identify the user based on their thermal preferences with high accuracies. We run our experiments with small and moderate differences in metabolism rate and observe that models with similar thermal preferences lowers the accuracy in prediction but does not increase the time-steps required to set the temperature and humidity settings by the human model.

4.2 Methods

In this section, we extend the RL-based SHS system from Chapter 3 with the ability to estimate who is the current occupant using current TH and the human model's activity and actions only. The assumption for this problem is that there is only one human model in the environment at a given time. In this section, we first describe the general framework for the problem. Next, we will describe our proposed solution. Finally we will describe an alternate (baseline) solution.

4.2.1 Partial Observable Markov Decision Process

When there is uncertainty about the current state state of the environment (in this case, who the current occupant is, or what its preference are), a common framework for RL is the Partial Observable Markov Decision Process (POMDP) [65–68]. A POMDP is defined by the tuple $< \mathcal{S}, \mathcal{A}, \Omega, T, R, O, \gamma, b_0 >$, where $\mathcal{S}$ is the set of discretized states, $\mathcal{A}$ is the set of actions, and Ω is the set of observations. Moreover, $T(s_t, a_t, s_{t+1}) : [S \times A \times S] \rightarrow [0, 1]$ is the transition function that defines the probability of ending in state s_{t+1} after taking action a_t in state s_t, R is the reward function that specifies the immediate reward received after taking action a in state s_t, and $O(a_t, o_{t+1}, s_{t+1}) : [A \times \Omega \times S] \rightarrow [0, 1]$ is the observation function that defines the probability of observing o_{t+1} after taking action a_t and ending in state s_{t+1}. Lastly, γ is the discount factor and b_0 is the initial belief state that defines a probability over the observable state of the environment.

To calculate the belief about the current state, POMDP uses the Bayes theorem, which is defined as:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}, \tag{4.1}$$

where A and B are the events, $P(A)$ is the prior, $P(B)$ is the evidence, $P(A|B)$ is the posterior probability of event A given that event B has occurred, $P(B|A)$ is the probability of event B given that event A has occurred (likelihood). Here, we set A as a possible human model $\mathcal{H}_i$ in the home, and B as the next TH and activity o_{t+1} observed by the smart home. Note that here the state is the concatenation of the observation o_{t+1} and the current occupant $\mathcal{H}_i$ ($s_t = [o_t, \mathcal{H}_i]$), but the belief only needs to be over the occupants (since the TH and activity are fully observable). Accordingly, $P(B|A)$ is the probability of observing TH preferences given user $\mathcal{H}_i$ from the observation function O. $P(A|B)$ is the posterior probability of the human occupant $\mathcal{H}_i$ given the current activity and TH of the environment and previous belief. Finally $P(A)$ is the prior probability of human occupant $\mathcal{H}_i$ derived from the transition function T, while $P(B)$ is the marginal probability of the current observation received from the environment. Given our problem setup, the smart home agent takes an action a_t at every time-step after which it receives an observation o_{t+1} (consisting of the current activity of the occupant, temperature, and humidity) from the environment. With this observation, a POMDP agent would update its belief over each possible occupant by the following equation:

$$\overbrace{b_{t+1}(\mathcal{H}_i)}^{P(A|B)} = \frac{\overbrace{O(a_t, o_{t+1}, [o_{t+1}, \mathcal{H}_i])}^{P(B|A)} \overbrace{\sum_{\mathcal{H}_j \in \mathcal{H}} T([o_t, \mathcal{H}_j], a_t, s_{t+1}) b_t(\mathcal{H}_j)}^{P(A)}}{\underbrace{Pr(o_{t+1}|b_t, a_t)}_{P(B)}}, \tag{4.2}$$

where $b_t(\mathcal{H}_i)$ is the previous belief (probability) that the occupant is human model $\mathcal{H}_i$, $b_{t+1}(\mathcal{H}_i)$ is the updated belief for the next time step, and $Pr(o_{t+1}|b_t, a_t)$ is the probability of observing o_{t+1} when action a_t is taken with the belief vector b_t over all

possible human models (or occupants) $\mathcal{H}$. The denominator is given by:

$$Pr(o_{t+1}|b_t, a_t) = \sum_{\mathcal{H}_k \in \mathcal{H}} O(a_t, o_{t+1}, [o_{t+1}, \mathcal{H}_k]) \sum_{\mathcal{H}_j \in \mathcal{H}} T([o_t, \mathcal{H}_j], a_t, s_{t+1}) b_t(\mathcal{H}_j). \quad (4.3)$$

It should be noted that in the context of our problem, the transition function T and observation function O are unknown. Therefore, in the next sections we will describe our solution around this issue.

4.2.2 Partial Observable Smart Home System

Here, we describe a modified SHS that implements a Bayesian model to learn multiple user's personalized TH preferences when the user information is limited. Accordingly, we define a Partial Observable Smart Home System (POSHS) as a Bayesian model [65, 66] of the SHS where the state of the environment is not fully observable as discussed in the previous section.We will further assume that the model of the occupant does not change during an episode (a set of 3 activities). We will then use the Bayes theorem and the learnt occupants' TH preferences to infer the current human model in the environment, thus identifying it.

We depict an overview of POSHS in Figure 4.1, where it is divided into two components (panels (a) and (b)). During the episode (panel (a)), at each time step, the agent estimates a probability distribution over the TH preferences of the current occupant based on the TH and activity observations. This distribution is combined with a pool of previously learned user TH preference distributions using the Bayes theorem to estimate the belief state over the possible current occupants. The belief state is then used to weight the Q-table of each possible occupant in order to select the optimal action for the current belief. At the end of the episode (panel (b)),

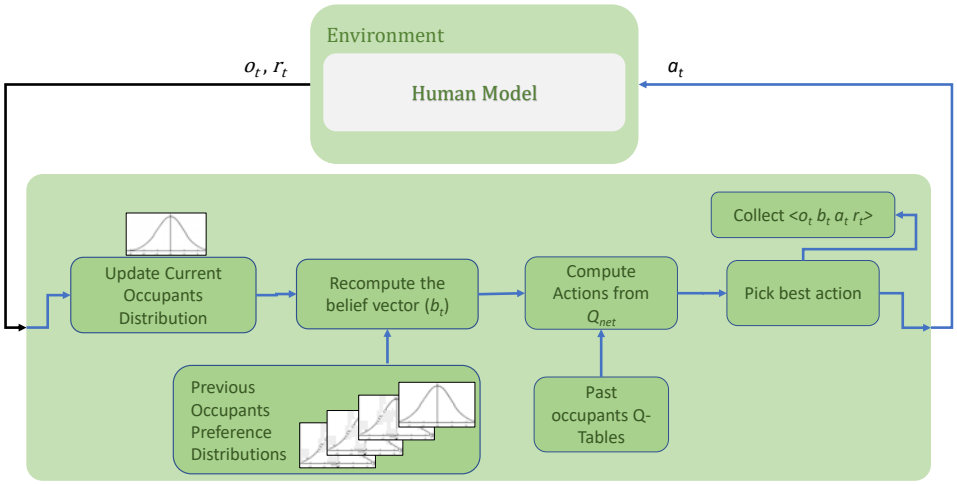
the agent compares the final estimated TH preference distribution to the pool of learned preferences. If a match is found, then the episode distribution parameters are used to update that occupant's TH preference distribution. If a match is not found, we assume this is a new user. Therefore, the estimated probability TH preference distribution is added to the pool along with a new Q-table. Finally, the Q-tables are updated with the episode data. In the next subsections, we describe these elements (TH preference distribution, belief estimation, distribution similarity, and Q update) in depth.

4.2.3 TH Preference Distribution

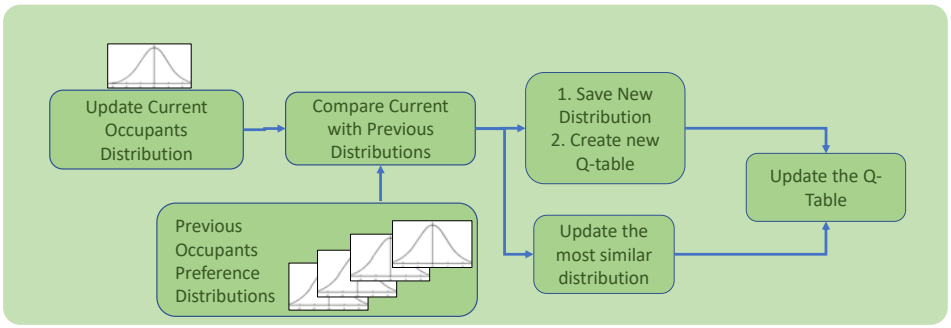
Since it is impossible to fully define T and O for an unknown set of occupants, we need a different approach to estimate the SHS belief over the set of possible occupants during the episode. To make the problem more tractable, we first assume that the occupants are the same through the whole episode. This implies that the transitional probabilities in an episode have the following property:

$$\begin{aligned} T([o_t, \mathcal{H}_i], a_t, [o_{t+1}, \mathcal{H}_j]) &= 0 \quad \forall i \neq j \\ T([o_t, \mathcal{H}_i], a_t, [o_{t+1}, \mathcal{H}_i]) &= 1. \end{aligned} \tag{4.4}$$

Accordingly, given a potential occupant $\mathcal{H}_i$, we need to define a probability distribution over the TH preference for each activity ($P(B|A)$ in Eq. 4.2). Using the Bayes rule, this will provide a way to estimate the probability of a specific occupant, given the TH observation ($P(A|B)$ in Eq. 4.2). The TH preference probability distribution $P(TH|\mathcal{H}_i)$ of a given occupant $\mathcal{H}_i$ over temperature and humidity for each activity



(a) POSHS during the episode



(b) POSHS at the end of episode

Figure 4.1: (A) POSHS during the episode: belief is computed and action is taken based on the current belief vector. The observation transitions are stored and distribution parameters are updated. (b) POSHS at the end of the episode: the current observed distribution is compared with stored distributions. If the distance between the distributions is greater than a threshold, a new Q-table is created for the new human occupant, otherwise the parameters of the distribution most similar to the current distribution are updated.

is described by a Gaussian distribution given as:

$$P(TH|\mathcal{H}_i) = \frac{1}{\sigma_t\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{TH - \mu_{\mathcal{H}_i}}{\sigma_{\mathcal{H}_i}}\right)^2\right), \tag{4.5}$$

where TH is an observed temperature (or humidity) and $\mu_{\mathcal{H}_i}$ and $\sigma_{\mathcal{H}_i}$ are the mean and standard deviations of the of the preference distribution for occupant $\mathcal{H}_i$. The above equation is defined individually for both temperature and humidity, and for each activity.

Given a complete episode, the SHS samples each observation when the human agent is not making any changes to the TH and continues with a given activity. Accordingly, the parameters of the TH preference distribution for the occupant of that episode can be estimated using best point estimators given by

$$\mu_e = \frac{1}{C} \sum_{\text{valid } t=0}^{T} TH_t \tag{4.6}$$

and

$$\sigma_e = \sqrt{\frac{1}{C} \sum_{i=0}^{T} (TH_i - \mu_t)^2}, \tag{4.7}$$

where TH_t is the observed temperature or humidity at time step t for the given activity, C is the number of valid observations, and T is the length of the episode. Therefore, a complete TH preference probability distribution for a model can be coded using a 12D vector (2 parameters $\times$ 2 TH $\times$ 3).

4.2.4 Belief Estimation

Here, we describe the mathematical formulation of the belief estimation of the smart home about the human occupant at time step t given observation TH_t. For simplicity, let us assume that we have two human agents in a given smart home environment denoted by $\mathcal{H}_\mathcal{A}$ and $\mathcal{H}_\mathcal{B}$. Restructuring Eq. 4.2 based on the Bayes

theorem, we obtain:

$$P(TH_t|\mathcal{H}_\mathcal{A}) = \frac{P(\mathcal{H}_\mathcal{A}|TH_t)P(TH_t)}{P(\mathcal{H}_\mathcal{A})} \tag{4.8}$$

and

$$P(TH_t|\mathcal{H}_\mathcal{B}) = \frac{P(\mathcal{H}_\mathcal{B}|TH_t)P(TH_t)}{P(\mathcal{H}_\mathcal{B})}, \tag{4.9}$$

which together yield:

$$\frac{P(TH_t|\mathcal{H}_\mathcal{A})}{P(TH_t|\mathcal{H}_\mathcal{B})} = \frac{P(\mathcal{H}_\mathcal{A}|TH_t)P(\mathcal{H}_\mathcal{B})}{P(\mathcal{H}_\mathcal{B}|TH_t)P(\mathcal{H}_\mathcal{A})}. \tag{4.10}$$

Solving the above equation for $\mathcal{H}_\mathcal{A}$ and $\mathcal{H}_\mathcal{B}$, we obtain:

$$P(\mathcal{H}_\mathcal{A}|TH_t) = \frac{P(TH_t|\mathcal{H}_\mathcal{A})P(\mathcal{H}_\mathcal{A})}{P(TH_t|\mathcal{H}_\mathcal{A})P(\mathcal{H}_\mathcal{A}) + P(TH_t|\mathcal{H}_\mathcal{B})P(\mathcal{H}_\mathcal{B})} \tag{4.11}$$

and

$$P(\mathcal{H}_\mathcal{B}|TH_t) = \frac{P(TH_t|\mathcal{H}_\mathcal{B})P(\mathcal{H}_\mathcal{B})}{P(TH_t|\mathcal{H}_\mathcal{A})P(\mathcal{H}_\mathcal{A}) + P(TH_t|\mathcal{H}_\mathcal{B})P(\mathcal{H}_\mathcal{B})}. \tag{4.12}$$

Given N different human models, we can define the probability of the i^{th} human model $\mathcal{H}_i$ at the t_{th} time-step as:

$$P(\mathcal{H}_i|TH_t) = \frac{P(TH_t|\mathcal{H}_i)P(\mathcal{H}_i)}{\sum_{i=0}^{N} P(TH_t|\mathcal{H}_i)P(\mathcal{H}_i)}. \tag{4.13}$$

Dividing $P(\mathcal{H}_\mathcal{A}|TH_t)$ by $P(\mathcal{H}_\mathcal{B}|TH_t)$ from Eq. 4.11 and Eq. 4.12, and then substituting $P(TH_t|\mathcal{H})$ with Eq. 4.5 we obtain:

$$\begin{aligned}\frac{P(\mathcal{H}_\mathcal{A}|TH_t)}{P(\mathcal{H}_\mathcal{B}|TH_t)} &= \frac{\frac{1}{\sigma_{\mathcal{H}_\mathcal{A}}\sqrt{2\pi}}\exp\left(-\frac{1}{2}\left(\frac{TH_t-\mu_{\mathcal{H}_\mathcal{A}}}{\sigma_{\mathcal{H}_\mathcal{A}}}\right)^2\right)P(\mathcal{H}_\mathcal{A})}{\frac{1}{\sigma_{\mathcal{H}_\mathcal{B}}\sqrt{2\pi}}\exp\left(-\frac{1}{2}\left(\frac{TH_t-\mu_{\mathcal{H}_\mathcal{A}}}{\sigma_{\mathcal{H}_\mathcal{A}}}\right)^2\right)P(\mathcal{H}_\mathcal{B})} \\ &= \frac{\sigma_{\mathcal{H}_\mathcal{B}}}{\sigma_{\mathcal{H}_\mathcal{A}}}\frac{\exp\frac{1}{2}\left(\left(\frac{TH_t-\mu_{\mathcal{H}_\mathcal{A}}}{\sigma_{\mathcal{H}_\mathcal{A}}}\right)^2-\left(\frac{TH_t-\mu_{\mathcal{H}_\mathcal{B}}}{\sigma_{\mathcal{H}_\mathcal{B}}}\right)^2\right)P(\mathcal{H}_\mathcal{A})}{P(\mathcal{H}_\mathcal{B})}.\end{aligned} \tag{4.14}$$

Assuming $\sigma_{\mathcal{H}_\mathcal{A}} \approx \sigma_{\mathcal{H}_\mathcal{B}} = \sigma$, we get:

$$\frac{P(\mathcal{H}_\mathcal{A}|TH_t)}{P(\mathcal{H}_\mathcal{B}|TH_t)} = \frac{\exp\frac{1}{2\sigma^2}\left(\left(2TH_t-\mu_{\mathcal{H}_\mathcal{A}}-\mu_{\mathcal{H}_\mathcal{B}}\right)\left(\mu_{\mathcal{H}_\mathcal{A}}-\mu_{\mathcal{H}_\mathcal{B}}\right)\right)P(\mathcal{H}_\mathcal{A})}{P(\mathcal{H}_\mathcal{B})}. \tag{4.15}$$

At the beginning of the training episode, the POSHS agent has no information about the user, thus maintains an equal initial belief for each possible human occupant. Accordingly, the prior probability of each model can be defined as $P(\mathcal{H}_\mathcal{A}) = P(\mathcal{H}_\mathcal{B})$. Hence, we can rewrite Eq. 4.15 as

$$\begin{aligned}\frac{P(\mathcal{H}_\mathcal{A}|TH)}{P(\mathcal{H}_\mathcal{B}|TH)} &= \exp\frac{1}{2\sigma^2}\left(\left(2TH_t-\mu_{\mathcal{H}_\mathcal{A}}-\mu_{\mathcal{H}_\mathcal{B}}\right)\left(\mu_{\mathcal{H}_\mathcal{A}}-\mu_{\mathcal{H}_\mathcal{B}}\right)\right) \\ \frac{P(\mathcal{H}_\mathcal{A}|TH)+P(\mathcal{H}_\mathcal{B}|TH)}{P(\mathcal{H}_\mathcal{B}|TH)} &= \frac{1}{P(\mathcal{H}_\mathcal{B}|TH)} \\ \frac{1}{P(\mathcal{H}_\mathcal{B}|TH)} &= \exp\frac{1}{2\sigma^2}\left(\left(2TH_t-\mu_{\mathcal{H}_\mathcal{A}}-\mu_{\mathcal{H}_\mathcal{B}}\right)\left(\mu_{\mathcal{H}_\mathcal{A}}-\mu_{\mathcal{H}_\mathcal{B}}\right)\right)+1.\end{aligned} \tag{4.16}$$

This gives

$$P(\mathcal{H}_b|TH) = \frac{1}{\exp\frac{1}{2\sigma^2}\left(\left(2TH_t-\mu_{\mathcal{H}_\mathcal{A}}-\mu_{\mathcal{H}_\mathcal{B}}\right)\left(\mu_{\mathcal{H}_\mathcal{A}}-\mu_{\mathcal{H}_\mathcal{B}}\right)\right)+1} \tag{4.17}$$

and

$$P(\mathcal{H}_a|TH_t) = \frac{\exp\frac{1}{2\sigma^2}\left((2TH_t - \mu_{\mathcal{H}_\mathcal{A}} - \mu_{\mathcal{H}_\mathcal{B}})\,(\mu_{\mathcal{H}_\mathcal{A}} - \mu_{\mathcal{H}_\mathcal{B}})\right)}{\exp\frac{1}{2\sigma^2}\left((2TH_t - \mu_{\mathcal{H}_\mathcal{A}} - \mu_{\mathcal{H}_\mathcal{B}})\,(\mu_{\mathcal{H}_\mathcal{A}} - \mu_{\mathcal{H}_\mathcal{B}})\right) + 1}. \tag{4.18}$$

With the above equation, we derive the *initial* posterior for each human agent at each time step for two human occupants. At the next time step we perform the Bayesian update where the posterior ($P(\mathcal{H}_i|HT_t)$) at time step t becomes the prior ($P(\mathcal{H}_i)$) at time step $t+1$. Accordingly, we can take Eq. 4.10 and divide both numerator and denominator by the $P(TH_t|\mathcal{H}_\mathcal{A})$, and then replace the ratio in the second term of the denominator by Eq. 4.14. This yields:

$$\begin{aligned} P(\mathcal{H}_\mathcal{A}|TH_t) &= \frac{P(TH_t|\mathcal{H}_\mathcal{A})P(\mathcal{H}_\mathcal{A})}{P(TH_t|\mathcal{H}_\mathcal{A})P(\mathcal{H}_\mathcal{A}) + P(TH_t|\mathcal{H}_\mathcal{B})P(\mathcal{H}_\mathcal{B})} \\ &= \frac{P(\mathcal{H}_\mathcal{A})}{P(\mathcal{H}_\mathcal{A}) + \frac{P(TH_t|\mathcal{H}_\mathcal{B})}{P(TH_t|\mathcal{H}_\mathcal{A})}P(\mathcal{H}_\mathcal{B})} \\ &= \frac{P(\mathcal{H}_\mathcal{A})}{P(\mathcal{H}_\mathcal{A}) + \exp\left(\frac{1}{2\sigma^2}\left(2TH_t - \mu_{\mathcal{H}_\mathcal{B}} - \mu_{\mathcal{H}_\mathcal{A}}\right)\left(\mu_{\mathcal{H}_\mathcal{B}} - \mu_{\mathcal{H}_\mathcal{A}}\right)\right)P(\mathcal{H}_\mathcal{B})}. \end{aligned} \tag{4.19}$$

Similarly, for Model $\mathcal{H}_\mathcal{B}$, the posterior can be computed as:

$$P(\mathcal{H}_\mathcal{B}|TH) = \frac{P(\mathcal{H}_\mathcal{B})}{\exp\left(\frac{1}{2\sigma^2}\left(2TH_t - \mu_{\mathcal{H}_\mathcal{A}} - \mu_{\mathcal{H}_\mathcal{B}}\right)\left(\mu_{\mathcal{H}_\mathcal{A}} - \mu_{\mathcal{H}_\mathcal{B}}\right)\right)P(\mathcal{H}_\mathcal{A}) + P(\mathcal{H}_\mathcal{B})}. \tag{4.20}$$

Expanding the above equations for N human models, we can get our posterior for the i^{th} human model as:

$$P(\mathcal{H}_i|TH_t) = \frac{P(\mathcal{H}_i)}{\sum_{j=0}^{N} C(\mathcal{H}_j, \mathcal{H}_i)P(\mathcal{H}_j)}, \tag{4.21}$$

where

$$C(\mathcal{H}_j, \mathcal{H}_i) = \exp\Big(\frac{1}{2\sigma^2}\left(2TH_t - \mu_{\mathcal{H}_j} - \mu_{\mathcal{H}_i}\right)\left(\mu_{\mathcal{H}_j} - \mu_{\mathcal{H}_i}\right)\Big). \tag{4.22}$$

4.2.5 Distribution Similarity

Upon completion of an episode, the SHS agent needs to know if the current occupant belongs to the pool of previously observed occupants or if it is a new occupant in the environment. To do so, the SHS agent follows the procedure in Figure 4.1(b). Here, the best point estimators (μ_e and σ_e from Eqs. 4.6 and 4.7) for the current occupant TH distribution are computed. The resulting TH preference distribution is then compared with the distributions in the pool of previously observed occupants using the Jensen-Shannon Divergence (JSD) measure [69]. We select this particular divergence measurement since it has shown success in similar RL-related applications in prior works [70]. Moreover, this metric provides benefits over other divergence metrics, in particular, (a) when variations between distributions are small, the divergence remains smooth, and (b) the divergence between distributions is symmetric, i.e., $JSD(P_0, P_1) = JSD(P_1, P_0)$.

The Jensen-Shannon divergence between n probability distributions is formulated as:

$$JSD = H\left[\sum_{i=1}^{n} w_i p_i\right] - \sum_{i=1}^{n} w_i H(p_i), \tag{4.23}$$

where H is the Shannon entropy [71] and w_i are the weights selected for each probability distribution p_i. For our problem, we compare only two distributions at an instance, i.e., the current distribution of the occupant in question and each of the distributions from the pool of observed occupants. As a result, we set $n = 2$, and

thus $w_1 = w_2 = 0.5$. Next, we define a new term τ_{JSD} as the threshold such that

$$JSD \geq \tau_{JSD} \iff \text{Different Human Model}, \tag{4.24}$$

where JSD is the measured Jensen-Shannon divergence between the two TH distributions. If the divergence between the distributions is greater than τ_{JSD}, the current occupant's distribution is added to the pool and the SHS agent creates a new Q-table. Otherwise the distribution with the smallest divergence (distribution with highest likelihood) is updated using a moving average with the current estimated parameters to improve the long term estimation of the occupant's preference distribution.

4.2.6 Q Update

Here, we describe the computation of the Q values of state-action pairs for a given occupant, followed by optimally selecting the best action using the belief vector over the user space. During each episode as shown in Figure 4.1(a), the SHS agent updates the TH distribution estimation parameters. It then calculates the belief vector using Bayes rule after which the Q-tables of the observed occupants are weighted using their belief vector to get a summarized Q_{net} value. The SHS agent then chooses the optimal action from the Q_{net} values of all the possible actions, which is then executed in the environment. The observation o_t, computed belief b_t, action taken a_t, and the reward received r_{t+1} from the environment are all stored in the memory.

To calculate the weighted Q update, each weight is decided by the belief of the specific human model for a given state from the belief vector. This update is given

by the following equation:

$$Q_{net}(o_t, a_t) = \sum^{\mathcal{H}} b_t(\mathcal{H}) Q_{\mathcal{H}}(o_t, a_t), \tag{4.25}$$

where o_t is the observation, a_t is the chosen action, the weight $b_t(\mathcal{H}) = P(\mathcal{H}|TH_t)$ is the conditional probability given the current TH, and Q_{net} is the weighted sum of all Q values of all the occupants for a given observation.

Accordingly, we now update the Q table for a given occupant $\mathcal{H}$. Let b_t be the belief vector at time step t in the episode and $b_t(\mathcal{H})$ be the belief component of b_t for human $\mathcal{H}$. Moreover, let o_t be the received observation at time step t in the episode, $\alpha = 0.05$ be the learning rate, $\gamma = 0.98$ be the discount factor, and r_{t+1} be the reward received after action a_t from observation o_t at time step t. Accordingly, the $Q_{\mathcal{H}}$ update is given by

$$Q_{\mathcal{H}}(o_t, a_t) = (1 - \alpha) Q_{\mathcal{H}}(o_t, a_t) + \alpha[r_{t+1} * b_t(\mathcal{H}) + \gamma \underset{a'}{max} Q_{\mathcal{H}}(o_{t+1}, a')]. \tag{4.26}$$

With the updated Q_{net} value, the agent can select the optimal action. For a given observation o_t, actions are selected using a decaying ϵ-Greedy policy keeping a balance between exploration and exploitation. The policy is given by:

$$\pi(o_t) = \begin{cases} \underset{a \in \mathcal{A}}{\arg\max}\{Q_{net}(o_t, a)\} \text{with probability } 1 - \epsilon \\ a \in \mathcal{A} \text{ with probability } \epsilon / |\mathcal{A}| \end{cases}, \tag{4.27}$$

where ϵ is a threshold value between 0 and 1, and $\mathcal{A}$ is the set of all possible actions.

4.2.7 Pseudocodes

In order to better facilitate reproduction of our work, in this subsection we present the pseudocodes and detailed algorithms for computing the Jensen-Shanon divergence, action selection and belief estimation, Q table update, and POSHS, in Algorithms 2 through 5.

Algorithm 2 Jensen-Shannon Divergence

function JSD(Distribution d_a, Distribution d_b, Amplification Factor A_f)
    let $d_c = \frac{d_a + d_b}{2}$
    let $H_a = d_a \times log\frac{d_a}{d_c}$ ▷ Calculate Shannon entropy for d_a
    let $H_b = d_b \times log\frac{d_b}{d_c}$ ▷ Calculate Shannon entropy for d_b
    $\tau = \sqrt{\frac{\sum H_a + \sum H_b}{2}}$
    return $A_f * \tau$

Algorithm 3 GetAction

function GETACTION(o_t)
    let N = number of human models
    let $TH_t \leftarrow o_t$
    calculate belief $P(\mathcal{H}_i|TH_t) = \frac{P(TH_t|\mathcal{H}_i)P(\mathcal{H}_i)}{\sum_{i=0}^{N} P(TH_t|\mathcal{H}_i)P(\mathcal{H}_i)}$ ▷ Belief update
    $Q_{net}(o_t, a_t) = P(\mathcal{H}_a)Q_a(o_t, a_t) + P(\mathcal{H}_b)Q_b(o_t, a_t) + \ldots + P(\mathcal{H}_n)Q_n(o_t, a_t)$
    return $\arg\max_a Q(o_t, a)$

4.3 Baseline

In this section, we describe a baseline method that we design and implement with which to compare our proposed method (POSHS) against. This baseline includes a Recurrent Neural Network (RNN) that can remember sequences of TH preferences in order to solve a partial observable environment. Classical RNNs, however, often suffer from issues such as vanishing gradient, making it difficult for the model to

Algorithm 4 Update

function UPDATE($memory, newNode$)
  let N = number of human models
  for $\mathcal{H}$ in all human models **do**
    let $b = P(\mathcal{H})$ ▷ Belief i_{th} human model
    if newNode **then**
      $Q_{\mathcal{H}} \leftarrow 0$ ▷ initialize Q table for unobserved human model
    Endif
    for $< o_t, a_t, o_{t+1}, r_{t+1} >$ in $memory$ **do**
      $Q_{\mathcal{H}}(o_t, a_t) = (1-\alpha)Q_{\mathcal{H}}(o_t, a_t) + \alpha[r_{t+1} * b + \gamma \max_{a'} Q_{\mathcal{H}}(o_{t+1}, a')]$
    Endfor
  Endfor

remember long-term information. As a result, we use a Long Short-Term Memory (LSTM) networks as our baseline. LSTM is a type of RNN that can hold a sequence of information in its memory and learn temporal relations. To do this, LSTM uses long-term sequences in memory referred to as 'cell-state'. The output from previous time-steps are referred to as 'hidden-state'. Information in an LSTM is controlled via three gates given as (a) forget-gate, which decides the information that needs to be rejected or accepted, (b) input-gate, which is used to control the information into the cell-state, and (c) output-gate, which generates the output for each time-step.

In a POMDP environment, Deep Q Network (DQN) fails to learn the optimal policy because it assume complete information about the state which we lack in our problem. Unlike DQNs, LSTM networks are capable of remembering long sequences of TH preferences and encoding the observation onto a latent space vector, making it possible to approximate the underlying hidden state [67, 68]. Thus, unlike POSHS where the agent needs to recognize the hidden state of the human occupant using Bayes rule explicitly, LSTMs can implicitly learn the underlying states using its recurrent memory.

Algorithm 5 Partially Observable Smart Home System (POSHS)

let $\mu_{TH} = \{\ \}$ ▷ Set containing Model's mean TH
let $\sigma_{TH} = \{\ \}$ ▷ Set containing Model's deviation of TH
let $Q_{\mathcal{H}} = \{\ \}$ ▷ Set containing Model's Q Table
let $o_0 = \mathcal{O}(s_0)$ where o_t is partial observation of state s_t
for each e in *episodes* **do**
    let $C = 0$
    let $\mu_e = 0$ ▷ Current Episode mean of TH
    let $\sigma_e = 0$ ▷ Current Episode standard deviation of TH
    let $memory = [\ ]$
    while o_t is not terminal **do**
        with greedy policy $a_t = getAction(o_t)$ ▷ Algorithm 3
        execute a_t
        observe o_{t+1}, r_{t+1}
        update μ_e, σ_e for C time-steps
        append $< o_t, a_t, o_{t+1}, r_{t+1} >$ in $memory$
        $C = C + 1$
    Endwhile
    let $newNode = False$ ▷ Flag to indicate a new human user
    let $divergence = [\ ]$
    for $< \mu, \sigma >$ in μ_{TH}, σ_{TH} **do**
        let $d_a = PDF(\mu_e, \sigma_e)$
        let $d_b = PDF(\mu, \sigma)$
        let $k = JSD(d_a, d_b)$ ▷ Algorithm 2
        append k in $divergence$
    Endfor
    if $min(distance) > \tau_{JSD}$ & $!newNode$ **then**
        add μ_e, σ_e in μ_{TH}, σ_{TH} ▷ Add New Human Node
        $newNode = True$
    else if $!newNode$ **then**
        let $i = \arg\min(divergence)$
        let $m = 0.5$ ▷ Moving filter significance
        $\mu_{TH}[i] = (1 - m)\mu_e + m\mu_{TH}[i]$
        $\sigma_{TH}[i] = (1 - m)\sigma_e + m\sigma_{TH}[i]$
    Endif
    $update(memory, newNode)$ ▷ Algorithm 4
Endfor

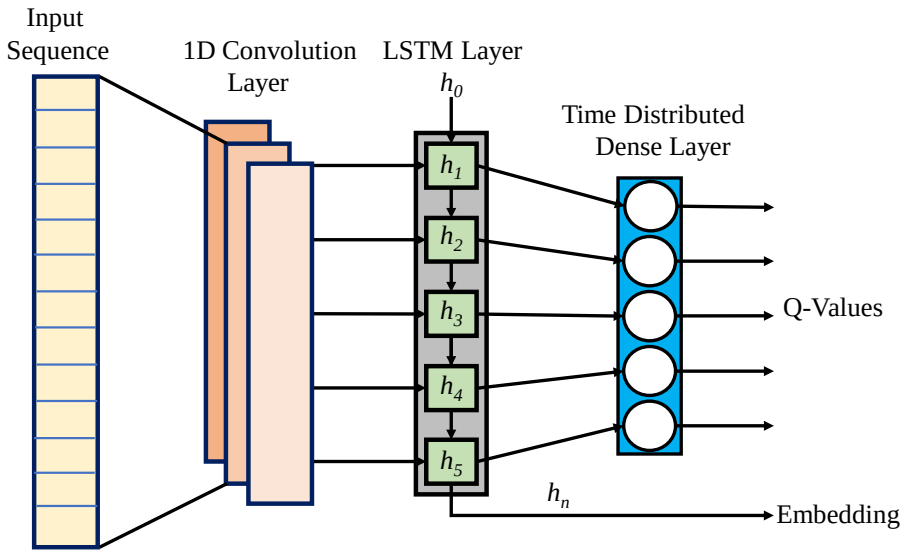


Figure 4.2: LSTM architecture: Input is a variable length sequence with 5d features which is convoluted in 1-dimension. The convoluted outputs is fed to the LSTM layer which is connected with a Time distributed dense layer outputting five Q-values for five actions to be taken by the SHS.

To use LSTM as a baseline, we design two networks namely *train* and *target*. Both networks have the same architecture and are initialized with the same weights and biases. The inputs are sequences of TH, activity, and human action of the occupant. The input layer of the model is a 1D convolutional layer with 32 filters, a kernel size of 1, a stride of 1, and a Rectified Linear Unit (ReLU) activation function. The output of this layer is fed to an LSTM layer with 175 cells followed by a tanh activation function. In the end, the output of this layer is fed to a dense layer of 5 neurons and ReLU activation to output the Q-values for each possible action (total of 5 actions: increase/decrease T and H, and no action). The training procedure for this baseline is described in detail in Algorithm 6 with its architecture shown in Figure 4.2.

In this algorithm, in contrast to POSHS where we update the Q-values directly, we update the parameters of the network with which we predict the next Q-value. The

Algorithm 6 Deep Reccurent Q-Learning for POMDP Environment

Initialize $train$ = train network with weights θ
Initialize $target$ =target network with initial weights $\theta^- = \theta$
let $stateMemory$ = [ ]
let $C = 0$
let $o_t = \mathcal{O}(s_t)$ where o_t is partial observation of state s_t at time step t
while $i < episodes$ **do**
    while o_t is not terminal **do**
        with probability ϵ select random action
        else select $a_t = \arg\max_a Q_t(o_t, h_{t-1}, a|\theta)$
        execute a_t
        observe o_{t+1}, r_t
        Store the transition $< o_t, a_t, o_{t+1}, r_t >$ in $stateMemory$
        $C = C + 1$
    Endwhile
    for $< o_t, a_t, o_{t+1}, r_t >$ in $stateMemory$ **do**
        $Q_t \leftarrow$ Predict $Q|\theta$ for o_{t+1} from $train$
        $Q^- \leftarrow$ Predict $Q|\theta^-$ for s_{t+1} from $target$
        $Q^-_{max_{t+1}} = \max Q^-$
        if o_{t+1} is terminal **then**
            $y_t \leftarrow r_t$
        else
            $y_t \leftarrow r_t + \gamma Q^-_{max_{t+1}}$
        Endif
        calculate loss $\mathcal{L} = (y_t - Q_t)^2$
    Endfor
    fit $train$
    After C steps $\theta^- \leftarrow \theta$
Endwhile

Q-values of the current state from the *train* network are updated using the Q-values of the next state from the *target* network. Using the same network (*train*) for predicting the target Q-value would cause instability in the target Q-value. The weights for the *train* and *target* networks are represented by θ and θ^- respectively. Target weights θ^- are updated after every C iterations with θ, where C is a hyper-parameter set to 7 empirically. We use MSE as our loss function and ADAM [72] optimizer. We define our loss function similar to the update in table-based Q-learning which is the difference between the target and current Q-values. Accordingly, the loss function is given by

$$\mathcal{L}(s_t, a_t|\theta) = (r_t + \gamma \underset{a}{max} Q(s_{t+1}, a|\theta^-) - Q(s_t, a_t|\theta))^2. \tag{4.28}$$

4.4 Experiments and Results

In this section, we present our results using both the proposed POSHS and the baseline LSTM. In the first experiment, we evaluate the POSHS performance by its accuracy in identifying the occupant in the environment. In the second experiment, similar to the previous chapter, we evaluate the impact of the SHS on the occupant by measuring the time-steps required by the occupant to set TH with and without the SHS. We further compare those results to the LSTM baseline.

4.4.1 Experiment A

In this experiment, we measure the performance of the POSHS in identifying occupants while learning the thermal preferences of each user. We train each human model separately for 350 episodes such that each model can complete each activity

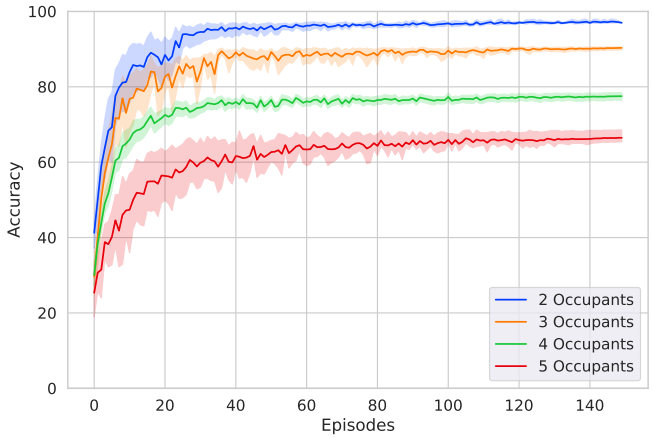


Figure 4.3: Training accuracy of 2-5 human models with POSHS agent using Jensen-Shannon Divergence to determine distribution similarity.

while also setting the optimal TH. Next, we train the POSHS with the trained occupant for 150 episodes. During the training of the POSHS, a human model is chosen randomly before each episode. Finally, each human model is run with the POSHS for 50 episodes to evaluate the final performance of the system.

First, we only experiment with two human models, namely $\mathcal{H}_a$ and $\mathcal{H}_b$. We set the metabolism indices for $\mathcal{H}_a$ to $[1, 1.2, 1.4]$ based on [49, 64]. For $\mathcal{H}_b$, slightly different indices are required. We therefore add 0.05 to each of the indices for $\mathcal{H}_a$ to achieve $[1.15, 1.25, 1.45]$ for $\mathcal{H}_b$ for the PMV range of $d_{0.25}$. We use $\tau_{JSD} = 0.13$ which empirically showed the best results in preliminary tests. Figure 4.3 shows the accuracy of identifying the current model correctly during the training of POSHS. As discussed earlier, the occupant identification is performed by comparing the TH preference distribution estimated by the POSHS from the episode to the ones stored in the pool of distributions using the JSD, and selecting the best match. It can be seen on the Figure 4.3 that the performance quickly achieves an accuracy of 85% and

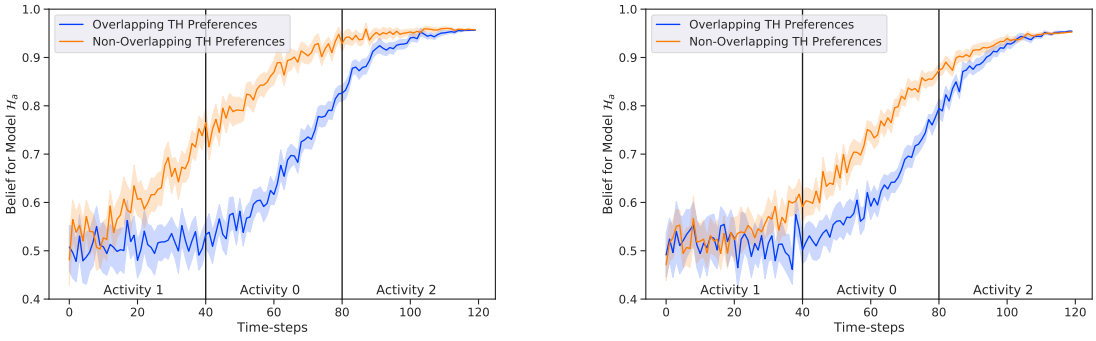


Figure 4.4: Model $\mathcal{H}_a$ belief update for overlapping ($d_{0.5}$) and non-overlapping ($d_{0.25}$) TH preferences (with Model $\mathcal{H}_b$). In plot (a), POSHS learns the activity (12D) specific TH distributions of the occupant while in plot (b), it learns the episode (4D) specific TH distribution of the occupant.

converges to nearly 100% accuracy.

To further evaluate the POSHS performance, we repeat the experiment using a wider PMV range of $d_{0.50}$, leading the human models to have more overlapping TH preference distributions. Figure 4.4(a) shows the average belief of the POSHS over the occupant during the test episodes. As we can see, when the human model TH preference distributions overlap, it takes more time-steps for the POSHS to properly identify the model's distribution. To further determine if maintaining the per-activity TH preference distributions (12 dimensional: μ and σ for T and H given each activity) is an important factor in the POSHS's performances, we repeat the same experiment once more, but using a TH preference distribution that does not encode activity-specific preferences (4 dimensional: μ and σ for T and H for each episode). The results are shown in Figure 4.4(b). As we can see, compared to Figure 4.4(a), the two curves are now relatively slow at identifying the current occupant, even when their preference distributions do not overlap as much. For a 12d representation, we

re-run preliminary experiments and find $\tau_{JSD} = 0.20$ which empirically showed the best results.

Finally, we repeat the same experiment, using 3, 4, and 5 different human models with a PMV range of $d_{0.25}$ (and the 12D TH preference representation). To do so, we increase the indices of $\mathcal{H}_A$ by small fixed amounts to get the metabolism indices for the new human models $\mathcal{H}_C$, $\mathcal{H}_D$ and $\mathcal{H}_E$. The final values are $[1.15, 1.22, 1.35]$, $[1.15, 1.25, 1.4]$, and $[1.05, 1.3, 1.45]$, respectively. The final accuracies as well as its test F1 scores are reported in Table 4.1. We observe that with only two human models, the POSHS is able to perform identification with high accuracy based on the thermal preferences. We also observe that with the integration of more human models, the accuracy drops to due to the increased overlap between TH preferences. Nonetheless, strong results (68%) above chance level (20%) are still obtained for higher number of human models, for instance 5.

4.4.2 Experiment B

Here, we evaluate how both approaches, the POSHS and the LSTM model, perform in terms of rewards and time steps required for the human occupant to set TH as we increase the number of humans models in the home. Note that both POSHS and LSTM must form an internal representation of the underlying hidden state. POSHS does it by explicitly estimating the current occupant's TH preference, while the LSTM builds an internal representation based on the sequence of observed actions (TH and activities).

Similar to Experiment A, we train the LSTM with the trained human occupant for 175 episodes with a learning rate of 0.001. During the training, a human model is

Table 4.1: POSHS mean Accuracy (in %) and F1 Score for up to 5 human models.

	Model	POSHS	
		Accuracy	**F1 Score**
2 Models	Model $\mathcal{H}_a$	0.96	0.98
	Model $\mathcal{H}_b$	1.00	0.98
	Mean	0.98	0.96
3 Models	Model $\mathcal{H}_a$	0.92	0.89
	Model $\mathcal{H}_b$	0.96	0.94
	Model $\mathcal{H}_c$	0.92	0.88
	Mean	0.90	0.90
4 Models	Model $\mathcal{H}_a$	0.60	0.67
	Model $\mathcal{H}_b$	0.83	0.83
	Model $\mathcal{H}_c$	0.90	0.78
	Model $\mathcal{H}_d$	0.77	0.77
	Mean	0.76	0.76
5 Models	Model $\mathcal{H}_a$	0.67	0.63
	Model $\mathcal{H}_b$	0.60	0.60
	Model $\mathcal{H}_c$	0.88	0.78
	Model $\mathcal{H}_d$	0.70	0.70
	Model $\mathcal{H}_e$	0.62	0.70
	Mean	0.68	0.66

chosen randomly at the beginning of the episode. We repeat the experiment for up to 5 human models. For testing purposes, we test the LSTM model for 50 episodes to evaluate the final performance. Therefore, both POSHS and LSTM were tested under the same training and testing conditions.

The performance in terms of the occupant's rewards for both SHS models are shown in Table 4.2. There is a slight increase in the observed mean reward for the occupants with the SHS in comparison to mean reward without the SHS, suggesting that the human occupants spend less time changing the TH in the presence of the SHS model. We then measure the actual number of time-steps required by the occupants to set their preferences with the SHS models compared to the time-steps required by

Table 4.2: Mean Reward for 2 Human Models with POSHS and LSTM. (Higher is better)

	Model	No SHS	POSHS	LSTM
2 Models	Model $\mathcal{H}_a$	286 ± 1.12	293 ±0.67	291 ±1.21
	Model $\mathcal{H}_b$	282 ± 1.03	289 ±0.92	288 ±1.07
3 Models	Model $\mathcal{H}_a$	286 ± 1.12	293 ±0.87	288 ±1.51
	Model $\mathcal{H}_b$	282 ± 1.03	289 ±1.22	285 ±1.77
	Model $\mathcal{H}_c$	281 ± 0.98	291 ±1.04	285 ±1.53
4 Models	Model $\mathcal{H}_a$	286 ± 1.12	291 ±1.77	285 ±1.81
	Model $\mathcal{H}_b$	282 ± 1.03	288 ±1.92	282 ±1.78
	Model $\mathcal{H}_c$	281 ± 0.98	288 ±1.42	283 ±2.07
	Model $\mathcal{H}_d$	281 ± 0.66	281 ±1.82	287 ±2.10
5 Models	Model $\mathcal{H}_a$	286 ± 1.12	288 ±2.07	284 ±3.21
	Model $\mathcal{H}_b$	282 ± 1.03	287 ±2.02	282 ±2.74
	Model $\mathcal{H}_c$	281 ± 0.98	283 ±1.90	280 ±3.77
	Model $\mathcal{H}_d$	281 ± 0.66	283 ±1.67	277 ±3.87
	Model $\mathcal{H}_e$	283 ± 1.35	283 ±2.52	278 ±4.17

the occupants without the SHS. This is shown in Figure 4.5 where we present the results for 2, 3, 4, and 5 human occupants in the SHS. It is observed that the number of extra time steps taken by the occupants are close for POSHS and LSTM for up to 3 occupants. However, when compared to Figure 4.5(a) and (b) in Figure 4.5(c) and (d), the difference between the time-steps of LSTM and POSHS increases with more human occupants. Furthermore, referring back to Table 4.2, we observe that POSHS slightly outperforms the LSTM model as the amount of reward received by the occupant with POSHS is higher, and the difference increases as the number of human models increases.

The LSTM training loss curve is presented in Figure 4.6, showing that the LSTM struggles to learn the Q-values accurately as the number of occupants increases. This is particularly highlighted with 4 and 5 human models, where the LSTM seems to suffer from considerable forgetting [73] (depicted by the peaks in the training loss

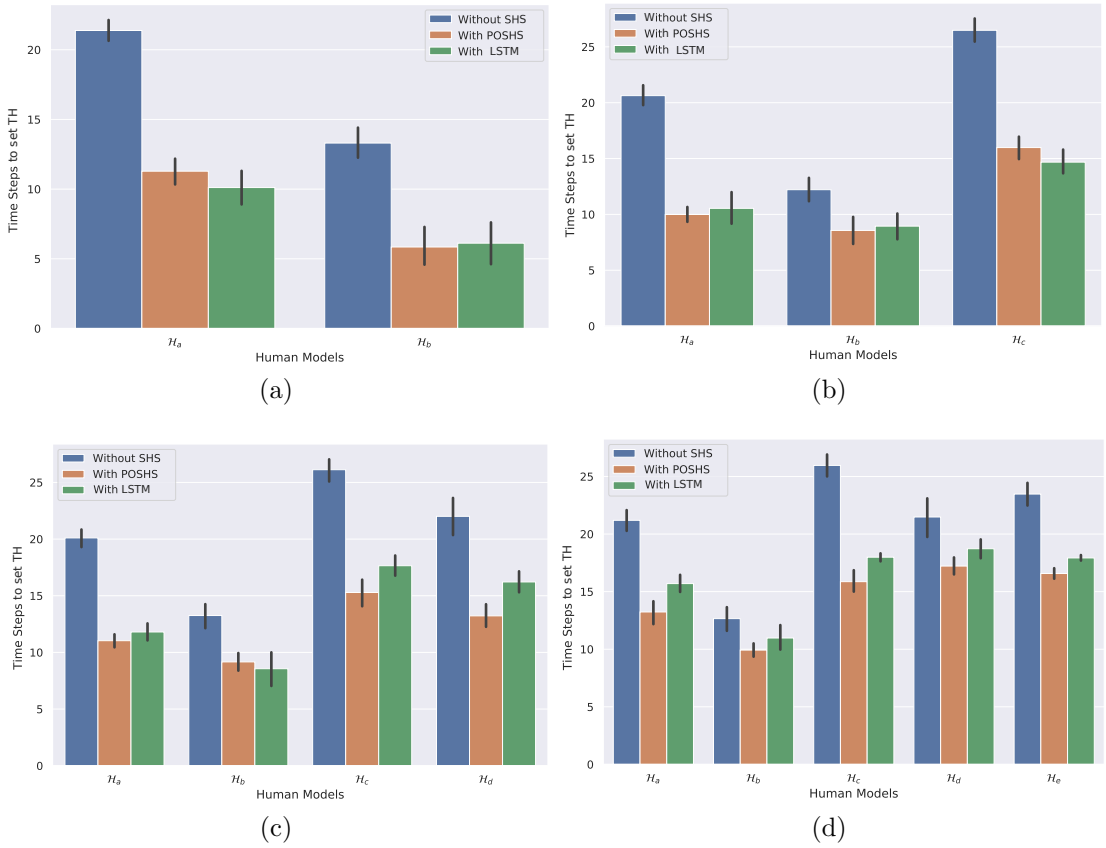


Figure 4.5: Mean time-steps taken to set TH without SHS, with POSHS, with LSTM for a)two, b)three, c)four, and d)five human models in the environment is shown here. (Lower is better)

curve) due to the changing weights of the network required to learn the sequences of different TH preferences.

4.4.3 Conclusion

In this chapter, we investigate the scenario where limited information about the occupants is available to the smart home, resulting is often sub-optimal actions by the smart home in a given state. In order to tackle this problem we modify our smart home by allowing it to learn to recognize the occupant's thermal preference distributions

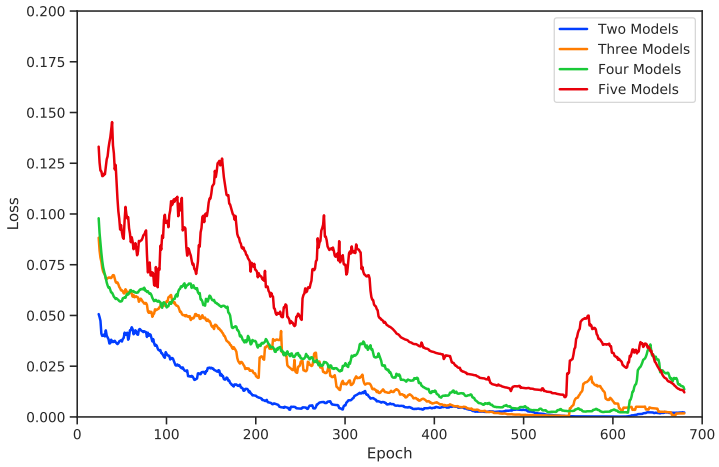


Figure 4.6: Mean training loss of LSTM with a combination of two, three, four, and five human models in the environment.

using Bayesian modelling, which helps maintain a belief over the hidden user state that the smart home uses to improve its policy. As a baseline for comparison, we design an approach in which the temporal relations between the TH sequences are learned to approximate the hidden state. We integrate both smart home models with up to 5 human models and evaluate them based on their ability to identify the underlying state (or occupant's TH preferences), as well as the overall human performance in terms of human rewards and time steps spent changing the TH with the SHS. Finally, we compare the POSHS with the baseline model that aims to learn its own representation of underlying states of the partially observable environment. Our simulations show good performance with POSHS when the number of human occupants are low without an increase in time required to set the thermal preferences by the occupant. Similar performance was observed with the baseline. With more human models integrated in the environment, the time-step difference between the baseline and POSHS increases where the occupants now take more time to set the TH

with the baseline compared to the POSHS model. In the end, With our simulated experiments, we demonstrate that in an environment where the user information is not fully observable it is possible to approximate the user's hidden state with multiple occupants without having an impact on the user's behavior even with sub-optimal approximations.