Contextual Bandit Learning for Activity-aware Things-of-Interest Recommendation in an Assisted Living Environment

May S. Altulayan[1], Chaoran Huang[1] Lina Yao[1], Xianzhi Wang[2], and Salil Kanhere[1]

[1] The University of New South Wales, Sydney, NSW 2052, Australia
{m.altulyan, lina.yao, chaoran.huang, salil.kanhere}@unsw.edu.au
[2] The University of Technology Sydney, Ultimo, NSW 2007, Australia
xianzhi.wang@uts.edu.au

Abstract. Recommendation systems are crucial for providing services to the elderly with Alzheimer's disease in IoT-based smart home environments. Therefore, we present a Reminder Care System to help Alzheimer patients live safely and independently in their homes. The recommendation system is formulated based on a contextual bandit (CB) approach to tackle dynamicity in human activity patterns. Correct recommendations aimed at meeting user needs without their feedback is achieved. Our experimental results show the feasibility and effectiveness of RCS in real-world IoT-based smart home applications.

Keywords: Contextual bandit · IoT · Recommender System.

1 Introduction

Alzheimer's disease (AD) is the most common type of dementia with severe implications on the day-to-day activities of numerous people world-wide [1]. In the US, 6.08 million elderly people reportedly suffer from clinical AD or mild cognitive impairment in 2017 with a potential escalation of this figure to 15.0 million by 2060 [2]. Meanwhile, the cost implication in the areas of providing care for Americans with AD and other dementia is equally of major concern as $290 billion was estimated in 2019 [3] opposed to the 2018 figure which stood at $277 billion [4].

The various stages of AD are generally grouped into three, mild, moderate, and late or severe stages. Each of these stages present different symptoms with the mild and moderate stages capable of lasting for about 3 years while the late or severe stage could last throughout the remainder of the patient's life. In the mild stage, patients begin to lose only short-term memory where they forget their ability to remember people's names or recent events. This stage is best manageable with technological aids. However, patients at moderate stage may have acute memory loss which could affect the ability of handling some simple tasks, language problems, time consideration, and some changes in their personality which may become emotional. For the last stage, patients lose their ability

of talking, understanding, swallowing, and walking. Consequently, intensive care from family members or professional caregivers are required [1].

Recommender systems could be adapted to help patients live safely and independently especially in IoT-based smart-home environments during the mild stage, and they are increasingly employed to provide critical services, e.g. reminder care services. A reminder care system is especially suitable for the mild stage because at this stage, patients just begin to lose short-term memory (e.g. difficulty in remembering people's names and recent events) [1, 5] without losing the ability to use such a system. A reminder care system in a smart-home environment would generally exploit sensory data from various sources e.g., environmental sensors, wearable sensors, and appliance sensors to deliver reminder recommendations to patients of items that they might need. This does not necessarily have to wait for the feedback to improve the quality of recommendations. For instance, using the following scenario to demonstrate the importance of a reminder care system for Alzheimer patients: Aris is a 79-year-old woman living alone with mild AD. The reminder care system automatically monitors and recognizes Aris' activity patterns and recommends items that might be needed based on Aris' status and activity patterns. For example, if she completes cooking but forgets to turn off the stove, the system is set to remind her of that timely. Afterwards, the system checks her acceptance for the recommendation automatically to improve the quality without needing Aris' explicit feedback.

Several studies have focused on reminder recommender systems aimed at providing assistance to elderly people diagnosed with AD. Oyeleke et al. [6] propose a recommendations system to monitor the daily indoor activities of seniors with mild cognitive impairment while Ahmed et al. [7] design a smart biomedical assisted system to assist Alzheimer patients. Several studies [8–11] use smart phone applications to provide care services to AD patients. The dynamicity in human activity patterns have not been given desired attention in most previous works, thus delivering low-quality recommendations. Another notable issue is the increased focus on monitoring which gives reminder to patients while the system has to wait for patients' feedback to update itself. From the illustration of Aris scenario above, suppose Aris has the following pattern when preparing a cup of coffee in the morning: (1) turning on the coffee machine, (2) bringing a cup, (3) putting some milk, and (4) then adding sugar. Then, if she brings a cup and forgets it, what is next? the system should remind her of grabbing the cup. However, if one day, she changes this pattern and decides not to add milk to her coffee in the future, the system should also cope with that. From Aris' standpoint, she expects the system to be a caregiver, which helps only when needed without actively requesting feedback. Therefore, the system should be capable of assessing the quality of recommendations without requiring user feedback.

In our previous work [12, 13], we developed a prototype system capable of detecting complex activities to enable the system to cope with the dynamicity in human activity patterns. Here, we extend the previous work by developing a recommender system that can not only learn the dynamicity of human activity pattern, but also remind patients about the correct item when the patients need

it without requiring their feedback. To this end, we formulate our problem using a contextual bandit approach, which focuses on context as input to produce the next action. The main contributions of this paper are as follows:

- We propose a recommender system based on contextual bandit by fusing context information from past activities, current activity, and items to recommend the correct item.
- We update our system automatically without needing feedback from users to improve the recommendations.
- We evaluate the model using a public dataset and our experimental results demonstrate the feasibility and effectiveness of our approach.

2 Related work

2.1 Recommender system for the IoT

Generally, recommendation systems can be used to assist users in selecting their preferences of things in IoT enriched environments. Some of these works exploit the traditional recommender system approaches: collaborative filtering [14], content-based [15] and hybrid-based approach [16] to build their systems. Authors in [17] propose a unified collaborative filtering model based on probabilistic matrix factorization recommender system that exploits three kinds of relations to extract the latent factors among these relations. In [15], the authors adapt a content-based solution for the recommender engine in their AGILE project which aims to improve the health conditions of users. Authors in [16,18,19] built their recommender system engine using a hybrid recommendation algorithm.

Reinforcement learning approach (RL) has also been adapted in building recommender system for the IoT environments. RL deals with dynamic environments and learn a policy that maximizes the long-term reward particularly for continuous record update. Massimo et al. [20, 21] adopt inverse reinforcement learning to model user behaviours. Oyeleke et al. [6] design a system that monitors daily indoor activities for people with mild cognitive impairment. Most RL algorithms particularly dealing with dynamic environment focus on matching each state for an action using different policies sequentially. This is achieved by observing how the taken action could affect next state by considering the future rewards. However, RL cannot handle a system that needs to learn the best action in different scenarios and treat each state independently while not allowing one action affect the next stat. Consequently, we formulate our recommender system with a contextual bandit approach.

2.2 Contextual bandit approach for recommendation

Contextual bandit approach combines two common features of RL by using policy to take an action based on the context of each state and of multi armed bandit (MAB) by focusing on the immediate reward. Some studies have adapted CB for their recommender systems. Li et al. [22] adopt contextual bandit for news

article recommendation. The proposed algorithm, LinUCB, shows the ability to deal with sparse and large data combined with other algorithms such as ϵ-greedy. In [23], CB is adapted to build an online learning recommender system where information for history students learning and current student are used as context to conduct the learning recommendations to the student. Zhang et al. [24] propose a novel contextual bandit method named SAOR for online recommendations. It deals with sparse interactions by distinguishing between negative response and non-response to improve recommendation quality. We adapt CB approach to formulate the problem tackling two main challenges in our system, the dynamicity of human activity pattern to recommend correct item and then knowing the feedback automatically without waiting for feedback from user.

3 Contextual-Bandit-based Reminder Care System

Our proposed system for reminder recommendation has three major stages (see Fig. 1): (1) Complex activity recognition stage, where we exploited three data sources, wearable sensors, environmental sensory data, and the usage of home appliances; (2) Prompt detection stage, which determines if an ongoing activity requires an item recommendation using data mining approaches; (3) reminder recommendation, which uses contextual bandit approach to extract context from the previous two stages and recommend items to the user during an activity. We further discuss the stages in the subsections below.

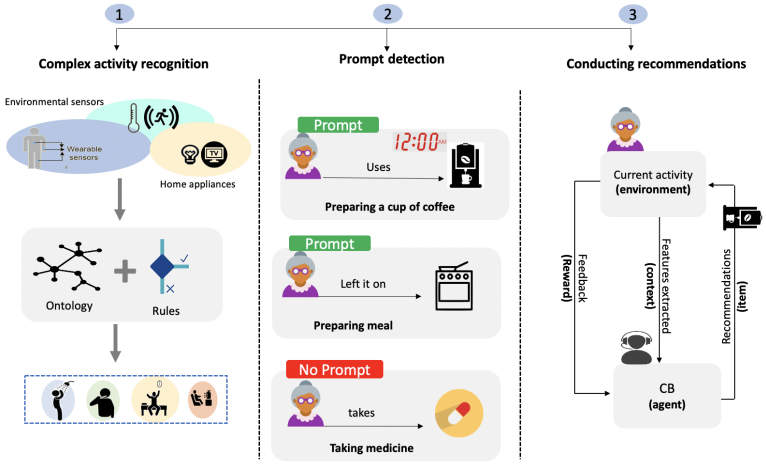


Fig. 1: Overview of the proposed methodology.

3.1 Complex activity detection

A system needs to detect what activity a user is performing before it can recommend an appropriate item to the user. Although Human Activity Recognition (HAR) has been studied extensively, most existing studies focus on detecting simple activities using wearable sensors, which are inadequate to support the detection of complex activities. Exploiting further sources, such as environmental sensors and home appliance sensors, helps the system to detect the complex activity accurately.

In our previous work [12], we designed a preliminary reminder care system that provides reminder recommendations based on complex activities detection. We conduct recommendations via three main steps:

- Elementary activity recognition. We use the common configuration of DeepConvLSTM as the classifier to detect elementary activities. DeepConvLSTM has 4 convolutional layers with feature maps and 2 LSTM layers with 128 cells. The result shows that DeepConvLSTM archives a promising accuracy of 77.2%.
- Ontology for complex activity recognition. After detecting elementary activities, we build an OWL (Ontology Web Language3) ontological models, which include the artifacts, environment, locations, and activities required to define things involved in the interaction.
- Rule-based orchestration. This step uses the output from the two previous steps to detect complex activities. It consists of a set of rules that are produced based on the previous ontological models.

3.2 Prompt detection

This part uses the collected data from the previous stage to identify if an activity needs a prompt or not. A prompt is defined in two main situations: (1) when the user has been stuck within an activity for some time without taking an action; (2) when the user uses a wrong item that does not belong to this activity. Various learning models can be adopted at this stage to determine when the user needs a prompt during her activities. For example, Das et al. [25] test several classification methodologies on the PUCK dataset, including Support Vector Machines(SVM) [26], Decision Tree [27] and Boosting [28]. In particular, Boosting applies a classification algorithm to re-weight the training data versions sequentially and then extracted a weighted majority vote of the previous sequentially classifiers. And it generally outperforms the other two methods.

3.3 Conducting recommendations

When the system is defined that the user's activity needs a prompt, the system at this stage should decide which item is suitable to be recommended at this moment based on the user situation. One of the main challenges as we mentioned above is that each activity could be done with different way. The system has to

consider which is a correct item to be recommended even it is the same activity by considering the user situation. This stage represents our main contributions in this paper.

Problem definition When a complex activity that needs a prompt is received by the agent G at time t, our algorithm extracts the context x and nominates an appropriate item a for the current activity. Then, the agent receives a feedback as reward r for the recommended item. Finally, the system is be updated based on the received reward.

Algorithm 1: Our procedure to recommended a correct item for user's activity. It takes context x as input, and returns a recommended item as output a

Input: x
Output: a
Procedure $agentrecommend(a)$
for $x_t \in X$ **do**
  $x_t \leftarrow PAC, CAC, IC$
  $a \leftarrow x_t$ agent G uses a policy to match the context for a correct item
  waiting for T_r
  compute $V(x)$
  put x_t, r_t, a into experience pool
  update the system
end for
return

Method We formulate our problem as a contextual bandit approach to tackle the dynamicity of human activity patterns and to recommend the correct item without having to wait for the user's feedback. Contextual bandit provides a learning model based on context. Three kinds of context are extracted at this stage:

- **Past activities context (PAC).** Since each activity can have a different pattern, for each activity, the system extracts the path/sequences of items used in the past (recorded in the log file) as a type of context. We use the recorded paths of each activity as an experience pool based on which the agent can decide which item to recommend at a specific state.
- **Current activity Context (CAC).** When the system receives data from the previous two stages, it extracts the context about the current state. For example, when the system receives that the user needs a prompt for preparing coffee, the context of the current activity (locations, previous items, user position, and time.) will be extracted.

– **Item context (IC).** Item context includes information about items, such as to which activity this item belongs, how long could it be in use, and how many times the user needs it for the current activity. For example, a coffee machine as an item can be used for the activity of 'preparing coffee', where it can be being used for around 2 minutes each time.

The agent receives the above contextual information as input (see Algorithm 1). The contextual bandit combined three main components: an environment, which represents the context of the user's activity $x \in X$, an agent G, which chooses an action $a \in A$ (notice that the common name in CB is Learner but we call it an agent in our case) based on the received context, and a reward $r \in \{0, 1]$, which the agent aims to maximize by recommending the correct action at each round $t = 1, 2, ..., T$. We calculate the expected reward of each policy using the following equation:

$$V(x) = \frac{1}{T} \sum_{t=1}^{T} E[r_t | x, \pi(x)] \tag{1}$$

where the agent G can choose from a set of policies $\prod \subseteq \{x \to A\}$ by employing two streaming models: Linear regression and stochastic gradient distance (to be detailed in section 4.3).

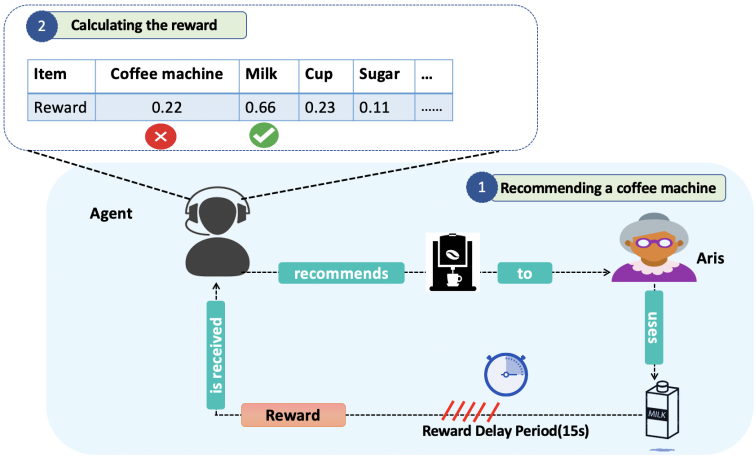


Fig. 2: (1) The agent recommends a coffee machine to Aris whereas she uses milk instead; then the system waits for 15s; (2) The feedback is received by the system as a reward and it is calculated accordingly as coffee machine is the wrong item.

Most traditional recommender systems focus on 'click' or 'not click' as feedback to calculate the reward function immediately and to update the system.

In contrast, our system recommends an item to the user and then waits for sufficient time to decide if the recommended item is used or not—by checking its status (on/off or moved/not moved). For example, if the system recommends a coffee machine to Aris (see Fig. 2) when she is preparing a cup of coffee, whereas she wants to use it later yet not immediately. This does not mean that the recommended item is incorrect, and it is better for the system to ignore this false negative feedback this time. To facilitate the above, we introduce a Reward Delay Period T_r, which accounts for the different paces of users in carrying out activities and is calculated as:

$$T_r = x_t + W_t \tag{2}$$

where x_t represents the user's activity and W_t represents the waiting time period. We consider W_t a hyperparameter (to be detailed Section 4.3).

4 Evaluation

We report our evaluation the proposed system. by introducing the dataset that we fit into our system, the feature engineering process, and finally, our experimental results.

4.1 Dataset

We evaluate the proposed system on, PUCK [25], a public dataset published in 2011. The PUCK dataset collected from a Kyoto smart home testbed located in Washington State University in two-story apartments with one living room, one dining area, and one kitchen on the first floor and, one bathroom and three bedrooms on the second floor. It combines three types of sensory data: (1) environmental sensors, including motion sensors on ceilings, door sensors on room entrances, kitchen cabinet doors, microwave, and refrigerator doors, temperature sensors in rooms, power meter, burner sensor, water usage sensors, and telephone usage sensors, (2) items sensors for usage monitoring, and (3) two wearable sensors. Eight complex activities are defined: Sweep and Dust, DVD Selection and Operation, Prepare Meal, Fill Medication Dispenser, Water Plants, Outfit Selection, Write Birthday Card, and Converse on Phone. Also, activities are divided into ordered steps, which can help detect whether the activity is completed correctly.

4.2 Features Engineering

The PUCK dataset has four fields (date, time, sensor ID, and sensor value). To adapt the PUCK dataset for our system, we process it to extract the required features via the following step:

– Combining the environmental data sensors(motion, items, power/ burner/ water usage, door...etc.) with the wearable sensors for each participant.

- Labeling the complex activities for the whole dataset.
- Extracting the start and the end of each activity as a session to define when the user needs a prompt.
- Selecting only the common sensors among all participants, where the total measurement counts of each sensor greater than 25% for all the participants.
- Dividing the sensors into four groups(movement sensors, motion sensors, count sensors and, continuous values sensors) based on kind of measurements(e.g. binary value, continues value and multiple values)
  and then processing each group as follows:
  1. For the movement sensors group, extracting the following features: Mean, STD, Correlations.
  2. For motions sensors group, computing the fraction counts across the groups.
  3. Counting sensors that have on/off measurements.
  4. For the last group, calculating the average for continuous value sensors.
- After extracting all features, applying the previous group process for all the participant sessions.

We take two methods to overcome the item usage imbalance problem (i.e, only a small number of items are frequently used): 1) Dropping outliers in items. This method is simple yet effective in improving the performance; 2) Sampling other random points in activity sessions to increase the prompt points, although this does not help balance the item usages as (1).

4.3 Experiment's results

We first evaluate the effectiveness of the contextual bandit approach in recommending the correct item to a user in case the user's current activity needs a prompt. The system uses all the extracted features as context to make a recommendation of the correct item. We use one public available contextual bandit package of python for our experiments. The package provides two types of models: full batch models and streaming models. Because of the sample limitation of the PUCK dataset, we focus on the streaming models, namely SGDClassifier (SGD) and LinearRegression.

Both models are sensitive to hyperparameters such as beta_prior or smoothing. However, SGDClassifier has stochastic matrices while LinearRegression(OLS) has matrices which are closed to the solution, and it updates them incrementally. Consequently, Linear regression performed better across different policies than SGD in our system (see Fig. 3a.)

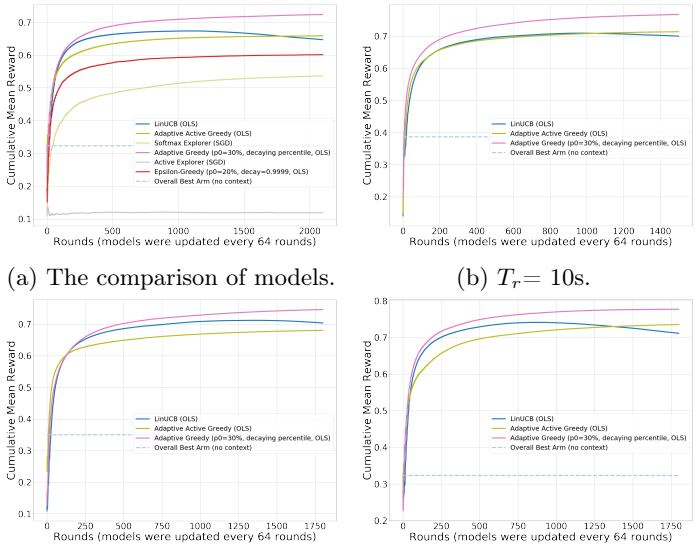
As shown in Fig. 3a, a set of policies are used for each model. Based on the results, we exclude SGD for the rest of our experiments due to its unpromising result. Details about parameters can be found in Table 1.

The Reward Delay Period T_r, as we mentioned above, plays the main role in defining when the agent receives the reward as a feedback of the recommended item. Tuning this parameter is important, as decreasing T_r could consider that the recommended item is not used while increasing T_r could confuse the agent

Table 1: Tuning hyperparameters for the OSL model policies

Policy	Hyperparametrs						
	beta_prior	alpha	smoothing	decay	refit_buffer	active_choice	decay_type
LinUCB	None	0.1	.	.	.	.	.
AdaptiveGreedy(Active)	((3./nchoices, 4), 2)	.	None	0.9997	.	weighted	percentile
AdaptiveGreedy	None	.	(1,2)	0.9997	.	.	percentile
SoftmaxExplorer	None	.	(1,2)	.	50	.	.
EpsilonGreedy	None	.	(1,2)	None	.	.	.
ActiveExplorer	((3./nchoices, 4), 2)	.	None	.	50	.	.

specifically when the user starts to use other items before receives the feedback about the recommended one. The results in Fig. 3 show that when the T_r has a large value, it improves the accuracy to become around 0.78. Here, we treat T_r as a hyperparameter that can be adjusted based on each item; we will leave it to our future work. In addition, we can see here the system does not need to any feedback from the user to receives the reward. Consequently, it is calculated automatically after the Reward Delay Period. We focus on this feature because our system deals Alzheimer's patients which is difficult for them holding a smart phone and confirm their response for recommendations.



(a) The comparison of models. (b) T_r= 10s.

(c) T_r= 15s. (d) T_r= 20s.

Fig. 3: Cumulative mean reward with the comparison of online contextual bandit models (Streaming data mode, see Fig. 3a) and selected models with different Reward Delay Periods T_r (see Fig. 3b, 3c, 3d).

5 Conclusion

In this work, we explore the feasibility of building a system that makes reminder recommendations to Alzheimer's patients only when they need a reminder. We take advantage of the contextual bandit approach to formulate our problem and tackle two main issues: dynamicity of human activity pattern and recommending the correct item without needing explicit user feedback. Experiments demonstrate the effectiveness of our recommender system. One limitation lies in our evaluation of the system is that our experiments are still not comprehensive enough because the only suitable dataset that we use does not include time labels, which are, however, one important and critical type of context. In the future, we will create our own test-bed to collect inclusive and adequate data for complex experiments, and testing our framework in real-life scenarios.

References

1. "Alzheimer's society." accessed 2019-01-07.
2. R. Brookmeyer, N. Abdalla, C. H. Kawas, and M. M. Corrada, "Forecasting the prevalence of preclinical and clinical alzheimer's disease in the united states," *Alzheimer's & Dementia*, vol. 14, no. 2, pp. 121–129, 2018.
3. A. Association, "2019 alzheimer's disease facts and figures," *Alzheimer's & Dementia*, vol. 15, no. 3, pp. 321–387, 2019.
4. A. Association *et al.*, "2018 alzheimer's disease facts and figures," *Alzheimer's & Dementia*, vol. 14, no. 3, pp. 367–429, 2018.
5. L. Yao, X. Wang, Q. Z. Sheng, S. Dustdar, and S. Zhang, "Recommendations on the internet of things: Requirements, challenges, and directions," *IEEE Internet Computing*, vol. 23, no. 3, pp. 46–54, 2019.
6. R. O. Oyeleke, C.-Y. Yu, and C. K. Chang, "Situ-centric reinforcement learning for recommendation of tasks in activities of daily living in smart homes," in *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, vol. 2, pp. 317–322, IEEE, 2018.
7. Q. A. Ahmed and A. Q. Al-Neami, "A smart biomedical assisted system for alzheimer patients," in *IOP Conference Series: Materials Science and Engineering*, vol. 881, p. 012110, IOP Publishing, 2020.
8. N. Armstrong, C. Nugent, G. Moore, and D. Finlay, "Developing smartphone applications for people with alzheimer's disease," in *Proceedings of the 10th IEEE International Conference on Information Technology and Applications in Biomedicine*, pp. 1–5, IEEE, 2010.
9. L. Choon, "Helper system for managing alzheimer's people using mobile application," *Universiti Malaysia Pahang*, 2015.
10. S. Alharbi, A. Altamimi, F. Al-Qahtani, B. Aljofi, M. Alsmadi, M. Alshabanah, D. Alrajhi, and I. Almarashdeh, "Analyzing and implementing a mobile reminder system for alzheimer's patients," pp. 444–454, 2019.
11. S. S. Aljehani, R. A. Alhazmi, S. S. Aloufi, B. D. Aljehani, and R. Abdulrahman, "icare: Applying iot technology for monitoring alzheimer's patients," in *1st International Conference on Computer Applications & Information Security*, IEEE, 2018.

12. M. S. Altulyan, C. Huang, L. Yao, X. Wang, S. Kanhere, and Y. Cao, "Reminder care system: An activity-aware cross-device recommendation system," in *International Conference on Advanced Data Mining and Applications*, Springer, 2019.
13. L. Yao, Q. Z. Sheng, B. Benatallah, S. Dustdar, X. Wang, A. Shemshadi, and S. S. Kanhere, "Wits: an iot-endowed computational framework for activity recognition in personalized smart homes," *Computing*, vol. 100, no. 4, pp. 369–385, 2018.
14. B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*, pp. 285–295, 2001.
15. M. J. Pazzani and D. Billsus, "Content-based recommendation systems," in *The adaptive web*, pp. 325–341, Springer, 2007.
16. L. M. De Campos, J. M. Fernández-Luna, J. F. Huete, and M. A. Rueda-Morales, "Combining content-based and collaborative recommendations: A hybrid approach based on bayesian networks," *International journal of approximate reasoning*, vol. 51, no. 7, pp. 785–799, 2010.
17. L. Yao, Q. Z. Sheng, A. H. Ngu, H. Ashman, and X. Li, "Exploring recommendations in internet of things," in *the 37th international ACM SIGIR conference on Research & development in information retrieval*, pp. 855–858, ACM, 2014.
18. K. G. HamlAbadi, A. M. Saghiri, M. Vahdati, M. D. TakhtFooladi, and M. R. Meybodi, "A framework for cognitive recommender systems in the internet of things (iot)," in *2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI)*, pp. 0971–0976, IEEE, 2017.
19. A. M. Saghiri, M. Vahdati, K. Gholizadeh, M. R. Meybodi, M. Dehghan, and H. Rashidi, "A framework for cognitive internet of things based on blockchain," in *The 4th International Conference on Web Research*, pp. 138–143, IEEE, 2018.
20. D. Massimo, "User preference modeling and exploitation in iot scenarios," in *23rd International Conference on Intelligent User Interfaces*, pp. 675–676, 2018.
21. D. Massimo, M. Elahi, and F. Ricci, "Learning user preferences by observing user-items interactions in an iot augmented space," in *Adjunct Publication of the 25th Conference on User Modeling, Adaptation and Personalization*, ACM, 2017.
22. L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proceedings of the 19th international conference on World wide web*, pp. 661–670, 2010.
23. W. Intayoad, C. Kamyod, and P. Temdee, "Reinforcement learning based on contextual bandits for personalized online learning recommendation systems," *Wireless Personal Communications*, pp. 1–16, 2020.
24. C. Zhang, H. Wang, S. Yang, and Y. Gao, "A contextual bandit approach to personalized online recommendation via sparse interactions," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 394–406, Springer, 2019.
25. B. Das, D. J. Cook, M. Schmitter-Edgecombe, and A. M. Seelye, "Puck: an automated prompting system for smart environments: toward achieving automated prompting—challenges involved," *Personal and ubiquitous computing*, vol. 16, no. 7, pp. 859–873, 2012.
26. B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152, 1992.
27. J. R. Quinlan, "Induction of decision trees," *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.
28. J. Friedman, T. Hastie, R. Tibshirani, *et al.*, "Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors)," *The annals of statistics*, vol. 28, no. 2, pp. 337–407, 2000.