Accurate Action Recommendation for Smart Home via Two-Level Encoders and Commonsense Knowledge

Hyunsik Jeon
Seoul National University
Seoul, South Korea
jeon185@snu.ac.kr

Jongjin Kim
Seoul National University
Seoul, South Korea
j2kim99@snu.ac.kr

Hoyoung Yoon
Seoul National University
Seoul, South Korea
crazy8597@snu.ac.kr

Jaeri Lee
Seoul National University
Seoul, South Korea
jlunits2@snu.ac.kr

U Kang
Seoul National University
Seoul, South Korea
ukang@snu.ac.kr

ABSTRACT

How can we accurately recommend actions for users to control their devices at home? Action recommendation for smart home has attracted increasing attention due to its potential impact on the markets of Internet of Things (IoT). However, designing an effective action recommender system is challenging because it requires handling context correlations, considering both queried contexts and previous histories of users, and dealing with capricious intentions in history. In this work, we propose SmartSense, an accurate action recommendation method for smart home. For individual action, SmartSense summarizes its device control and temporal contexts in a self-attentive manner, to reflect the importance of the correlation between them. SmartSense then summarizes sequences considering queried contexts in a query-attentive manner to extract the query-related patterns from the sequential actions. SmartSense also transfers the commonsense knowledge from routine data to better handle intentions in action sequences. As a result, SmartSense addresses all three main challenges of action recommendation for smart home, and achieves the state-of-the-art performance giving up to 9.8% higher mAP@1 than the best competitor.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

action recommendation, transformer, commonsense knowledge

ACM Reference Format:
Hyunsik Jeon, Jongjin Kim, Hoyoung Yoon, Jaeri Lee, and U Kang. 2022. Accurate Action Recommendation for Smart Home via Two-Level Encoders and Commonsense Knowledge. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM '22), October 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3511808.3557226

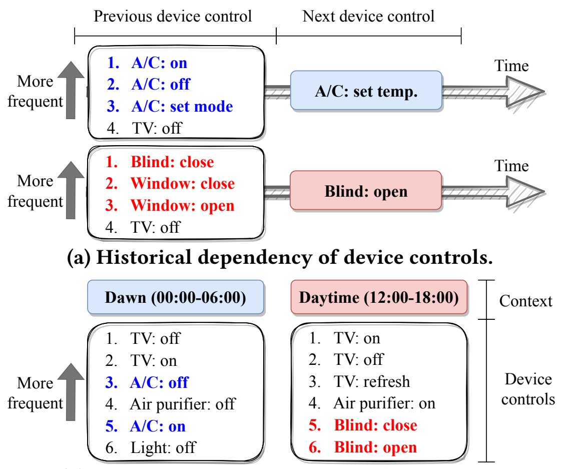

Figure 1: Historical and contextual dependency patterns of SmartThings users. (a) Frequent previous device control rankings of two different device controls. The next device controls are affected by previous ones. (b) Frequent device control rankings according to two different given hours. Temporal contexts affect users' device controls.

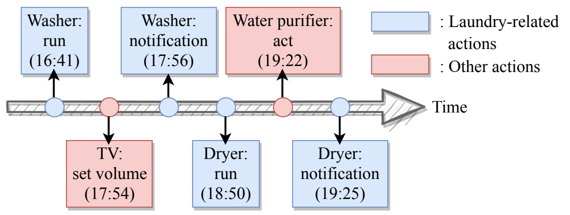
Washer: run (16:41)
Washer: notification (17:56)
Water purifier: act (19:22)
: Laundry-related actions
: Other actions
Time
TV: set volume (17:54)
Dryer: run (18:50)
Dryer: notification (19:25)

Figure 2: An example of sequential device controls performed by a *SmartThings* user. While the user is controlling the laundry-related devices (washer and dryer), other devices (TV and water purifier) are also controlled since the running times of the laundry-related devices are long. A sequence of actions contains capricious intentions.

1 INTRODUCTION

How can we accurately recommend actions for users to control their devices at home? Action recommendation for smart home has attracted increasing attention in data mining and machine learning communities due to its potential impact on the markets of virtual assistants [28] and the Internet of Things (IoT) [47]. The

Table 1: SmartSense addresses all three challenges of action recommendation for smart home, while competitors miss one or more of the challenges.

Method \ Challenges	Context correlations	Context-aware personalization	Capricious intentions
FMC [31], TransRec [7], Caser [36], SASRec [16], BERT4Rec [34]			
SIAR [29]	✓		
CA-RNN [25]	✓	✓	
SmartSense(proposed)	✓	✓	✓

problem is to predict future actions (e.g. device controls) of users and recommend the next action. It is of great value, as an accurate recommendation keeps users safe when they forget a critical action (e.g. shutting off a gas valve), and reduces the hassles of users when performing a cumbersome action (e.g. arming an alarm).

Action recommendation for smart home entails three main challenges. First, the complicated correlations of multiple contexts such as day of the week, hour affect users' device controls. For instance, laundry-related controls are highly affected by the correlation of day of the week and hour since people usually do laundry on weekend during the day. Second, both the sequential pattern of user actions and queried contextual information affects the future user actions. Figure 1a shows the frequency rankings of previously performed device controls in two different cases from SmartThings users, where SmartThings is a smart home system with 62 million active users worldwide. As shown in the figure, the next actions of people depend on the previous ones. For instance, people often perform various controls on air conditioners before setting the temperature, while they frequently control blinds and windows before opening the blinds. Figure 1b shows the frequency rankings of device controls depending on two different contexts. As shown in the figure, the device controls are affected by the current temporal contexts. For instance, people often control air conditioners before dawn since the proper temperature is an important factor for sleeping, while they frequently control blinds during the day because controlling the amount of sunlight entering the room is important for living. Figure 1 shows that both previous history and the current context are crucial when predicting users' current device controls. Third, users' action sequences contain capricious intentions. For example, if a user performs a series of actions: TV off, blind off, and light off, then we assume that the user wants to sleep. However, users do not always do their actions with only one intention. Figure 2 shows an example of an action sequence performed by a SmartThings user. As shown in the figure, a user controls a TV and a water purifier that are not related to laundry while doing laundry-related actions such as controlling a washer and a dryer. Capricious intentions lead to a degraded performance of recommendation models since they are trained to treat the actions in a sequence as highly related actions.

Sequential recommender systems [7, 13, 16, 19, 20, 31, 34, 36, 45] exploit sequential patterns of user actions to predict the future user actions. However, they have not addressed any of the three aforementioned main challenges. Context-aware recommender systems [25, 29] utilize the contextual information to predict the future user actions. However, they have not considered both queried contexts and previous histories, and have not dealt with capricious intentions in the histories. Thus, there is room for improvement of designing an effective model because the previous works failed to address such challenges.

In this work, we propose SmartSense, a novel approach for action recommendation for smart home. We design SmartSense with the following main ideas. First, SmartSense encodes individual action with a device control and its temporal contexts by a context-aware action encoder to consider their complicated correlation. Second, SmartSense encodes an action sequence and the current context by a context-attentive sequence encoder to consider both personalization and the contextual information. Third, SmartSense transfers the knowledge from routines of various users to handle capricious intentions in action sequences. With these ideas, SmartSense accurately recommends next actions that users would prefer.

Table 1 compares our proposed SmartSense with other methods in various perspectives. SmartSense is the only method that handles all the three challenges of action recommendation for smart home: context correlations, context-aware personalization, and capricious intentions.

- **Method.** We propose SmartSense, an accurate method for action recommendation for smart home. SmartSense correlates a device control and its temporal contexts to capture their correlation. SmartSense then summarizes a session's history while capturing highly related actions to the current contexts. Furthermore, SmartSense deals with capricious intentions by transferring commonsense knowledge from routine data.
- **Experiment.** Extensive experiments on real-world datasets show that SmartSense provides state-of-the-art performance with up to 9.8% higher mAP@1 in action recommendation for smart home compared to the best competitors (see Table 4).
- **Case study.** We show in case studies that SmartSense successfully recommends actions taking into account the context correlations and context-aware personalization (see Figures 5 and 6).
- **Real-world dataset.** We open-source the dataset from *SmartThings* which is a worldwide Internet of Things (IoT) platform with 62 million users. This is the first dataset for studying action recommendation in smart home. We provide sequential device control histories from four countries and device routine data from three territories (see Section 4.1). The datasets are available at https://github.com/snudatalab/SmartSense.

2 RELATED WORKS

2.1 Sequential Recommendation

Given a sequence of user behaviors, sequential recommendation aims to recommend items to users by modeling sequential dependencies of the user behaviors [38]; it is different from traditional recommendation systems [14, 21, 27, 33], which model interactions in a static way to capture only the general preference of users and items. On early works, FPMC [31] combines first-order Markov chains (MCs) and Matrix Factorization [21, 33] to model both sequential behaviors and general interests of users. Besides the first-order MCs, higher-order MCs [7, 8] have been studied to consider previous items in a sequence. In recent years, deep neural networks such as

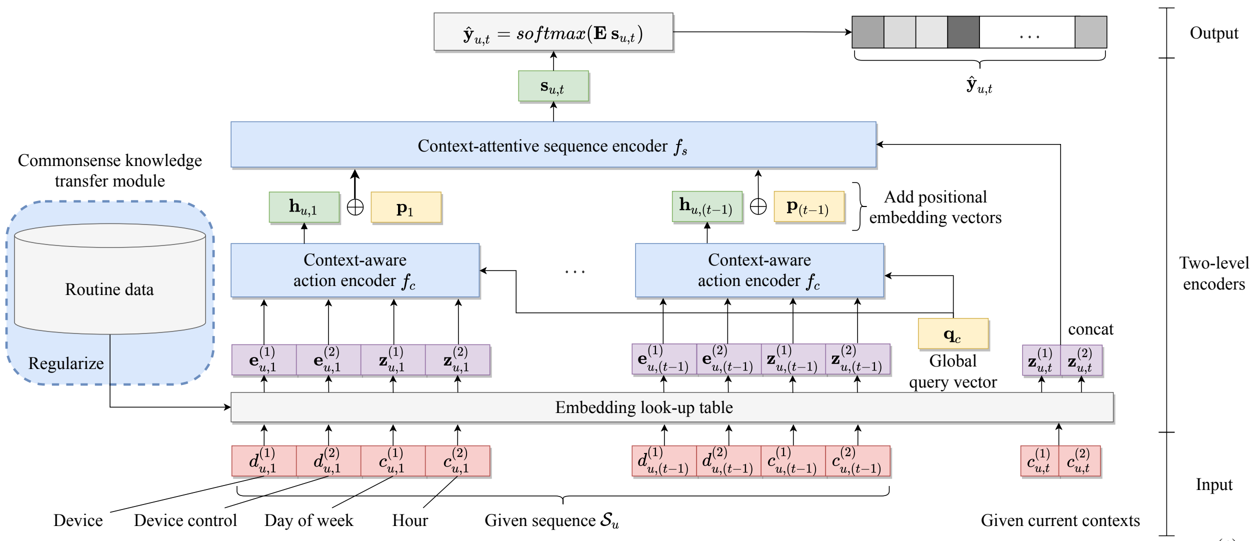

Figure 3: The overall architecture of SmartSense. Given session u'th sequence $\mathcal{S}_u$ and the current temporal context $(c^{(1)}_{u,t}, c^{(2)}_{u,t})$, SmartSense predicts the probability $\hat{y}_{u,t}$ of session u's device control at time t. SmartSense consists of context-aware action encoder (Section 3.4), context-attentive sequence encoder (Section 3.5), and commonsense knowledge transfer module (Section 3.6), where each encoder follows the structure of queried transformer encoder (Section 3.3). Context-aware action encoder interprets an individual action using the global query vector q_c. Context-attentive sequence encoder then summarizes the encoded sequence information using the current context. Commonsense knowledge transfer module regularizes device embeddings with routine data. SmartSense estimates the probability of the next device control using the summarized vector.

Recurrent Neural Networks (RNN) [4, 11], Convolutional Neural Networks (CNN) [22], and Transformers [37] have been adopted in sequential recommendation to address the complicated non-linear patterns in user behaviors. For instance, GRU4Rec [10] used Gated Recurrent Unit (GRU) [4] to model sequential patterns for session based recommendation. GRU4Rec$^+$ [9] boosted the performance of GRU4Rec for top-k sequential recommendation by improving the loss function. Caser [36] employed CNN to capture sequential patterns from both time-axis and feature-axis of sequences. SASRec [16] and BERT4Rec [34] utilized unidirectional Transformers and bidirectional Transformers, respectively, to capture sequential patterns in sequences while considering the importance of correlations between behaviors. Advanced techniques such as memory networks [3, 12], translation learning [7], hierarchical attention mechanisms [43], graph neural networks [2, 26, 39], and contrastive learning [45] have been adopted to sequential recommendation. However, such sequential recommender systems fail to achieve high performance in action recommendation for smart home since they do not handle contextual information and capricious intentions in sequences which should not be neglected for an accurate action recommendation.

2.2 Context-aware Recommendation

Context-aware recommendation aims to capture user preferences by considering contextual attributes such as time and temperature as well as interaction information between users and items [1, 23]. Early works on context-aware recommendation [32, 41] adopted Factorization Machines (FMs) [30], which model all correlations between variables, to capture the significant correlations between contexts. Recent works leverage deep neural networks and advanced technologies such as attention mechanisms [40] to model higher-order interactions and generate more meaningful representations of contexts. In recent years, context-aware sequential recommendation, which considers sequential patterns as well as contextual information, has been studied. For instance, CA-RNN [25] employs context-specific transition matrix to represent contextual information, and adopts RNN to capture sequential pattern in a history. Analogously, SIAR [29] utilizes stacked RNN to consider temporal dynamics of both contexts and actions. However, such context-aware recommender systems are still not suitable for action recommendation for smart home since they do not simultaneously consider both queried contexts and histories, or unravel the capricious intentions in histories.

2.3 Transformer

Transformer [37] is a deep learning model adopting self-attention mechanism which differentially weights the significance of each part of the input data to handle sequential tasks. The main advantages of Transformer are that it effectively learns the correlations between the input variables and effectively represents the sequential pattern of data. Transformer has attracted increasing attention in natural language processing [5], computer vision [6], stock prediction [44], and recommender systems [16, 34] due to its superior performance. The self-attention mechanism of Transformer enables us to successfully capture significant correlations between a device control and contexts, and effectively represent sequential patterns resulting in accurate action recommendation for smart home.

3 PROPOSED METHOD

We define the problem of action recommendation for smart home and propose SmartSense, an accurate method to solve the defined problem.

3.1 Action Recommendation for Smart Home

Action recommendation aims to recommend actions to users to help them control their devices at home. We describe the problem definition of action recommendation for smart home as follows.

Problem 1 (Action recommendation for smart home). *For each session u, we are given a sequence of history $\mathcal{S}_u = [x_{u,1}, \ldots, x_{u,(t-1)}]$, where $x_{u,i} = (d^{(1)}_{u,i}, d^{(2)}_{u,i}, c^{(1)}_{u,i}, c^{(2)}_{u,i})$ is a quadruplet of indices of i'th device, device control, day of week, and hour, respectively. Given a previous sequence $\mathcal{S}_u$, and the current temporal contexts $c^{(1)}_{u,t}$ and $c^{(2)}_{u,t}$, the goal is to accurately predict the current device $d^{(1)}_{u,t}$ and its control $d^{(2)}_{u,t}$.*

Note that we aim to predict only the device control $d^{(2)}_{u,t}$, since $d^{(2)}_{u,t}$ also contains the information of the target device $d^{(1)}_{u,t}$. To address the problem, a method should carefully handle device controls and their temporal contexts in the previous sequence, and effectively reflect the current temporal context in the prediction. Sequential recommendation methods [7, 16, 31, 34, 36] utilize only sequences of device controls without considering any contextual information. Context-aware recommendation methods [25, 29] consider temporal contexts in the previous sequence, or deal with the current temporal context. However, they are still unsatisfactory to action recommendation for smart home since they do not consider capricious intentions in the previous sequence.

3.2 Overview

We address the following challenges to achieve a high performance of smart home recommendation.

C1. **Considering the correlations of contexts.** The complicated correlations between a device control and various contexts such as day of week and hour affect a user's future device control. *How can we find meaningful correlations in the device control and the various contexts?*

C2. **Considering both history and the current context.** Both a user's past actions and the current context are crucial for predicting the user's current action. *How can we personalize the prediction while considering the current contexts?*

C3. **Handling capricious intentions.** Capricious intentions make us difficult to learn distant representations for actions with different intentions, which leads to a degraded performance. *How can we learn representations of actions such that similar intentions are close to each other and different intentions are distant?*

To address the aforementioned challenges, we propose SmartSense with the following main ideas.

I1. **Context-aware action encoder (Sections 3.3 and 3.4).** We encode a device control and its temporal contexts in a self-attentive manner to capture significant correlations between them.

I2. **Context-attentive sequence encoder (Sections 3.3 and 3.5).** We encode a sequence of actions in a self-attentive way to capture the correlations between the actions. We then summarize the sequence by a query-attention mechanism to consider the current context in personalization.

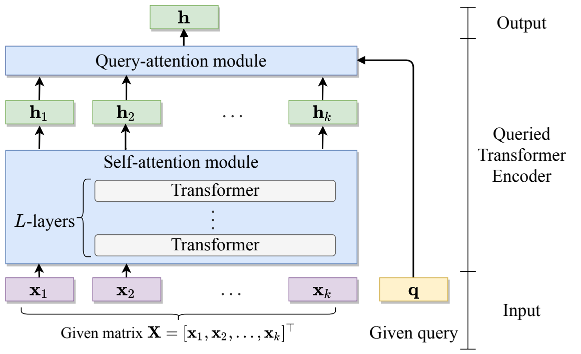

Figure 4: The queried transformer encoder (QTE) of SmartSense. Given a set of input vectors and a query vector, QTE summarizes the input vectors into a single output vector while considering the correlations of the input vectors and their relation to the query.

I3. **Commonsense knowledge transfer (Section 3.6).** We transfer commonsense knowledge from routine data. Each routine is intentionally defined by a user, as an explicit sequence of actions. As a result, the knowledge transfer from the routine data enables us to learn more meaningful representations of actions.

Figure 3 shows the overall structure of SmartSense, which consists of context-aware action encoder, context-attentive sequence encoder, and common sense knowledge transfer module. The context-aware action encoder encodes an action of each time step, and then the sequence encoder predicts the current action of the session based on encoded previous actions and the current context. The commonsense knowledge transfer module regularizes device embeddings to capture hidden relationships between them.

3.3 Queried Transformer Encoder

The aim of action recommendation for smart home is to predict a current device control given a previous history and a current context. To achieve the goal, we divide the task into two subtasks: 1) encoding individual action, and 2) encoding the sequential history with the current context. The two subtasks require the two common functionalities as follows. First, we need to consider the correlations between given variables since multiple variables are intricately related. Second, we need to consider the significance of each variable because the importance of each variable is different. Then, how can we design a model to embody the two functionalities? We propose a queried transformer encoder (QTE), which is used for both two subtasks: action encoding and sequence encoding. Our main ideas of QTE are 1) correlating given variables in a self-attentive manner, and 2) capturing significant variables for a query by a query-attentive mechanism. QTE is defined as follows:

$$\mathbf{h} = f(\mathbf{X}, \mathbf{q}), \tag{1}$$

where $\mathbf{h} \in \mathbb{R}^d$ is the summarized vector, $f(\cdot)$ is QTE, $\mathbf{X} \in \mathbb{R}^{k\times d}$ is the input matrix, k is the number of input vectors, d is the dimensionality of the input vectors, $\mathbf{q} \in \mathbb{R}^{d'}$ is the query vector, and d' is the dimensionality of the query vector. $\mathbf{x}_k \in \mathbb{R}^d$ denotes the k'th row of $\mathbf{X}$ representing the k'th input vector. Figure 4 depicts the structure of QTE. QTE consists of self-attention module and query-attention module which correspond to functionalities of correlating given variables and capturing significant variables, respectively. We describe the self-attention module and the query-attention module in detail.

Self-attention module. The goal of self-attention module is to correlate given variables. We employ transformer encoder [37] for the self-attention module since it represents all pair-wise correlations between the given variables by learning different query, key, and value weight matrices for each variable. Given an input matrix $\mathbf{X}$, we obtain query, key, and value matrices as follows:

$$\mathbf{Q} = \mathbf{X}\mathbf{W}^{Q}, \mathbf{K} = \mathbf{X}\mathbf{W}^{K}, \mathbf{V} = \mathbf{X}\mathbf{W}^{V}, \tag{2}$$

where $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{k \times d}$ are query, key, and value matrices, respectively; $\mathbf{W}^{Q}, \mathbf{W}^{K}, \mathbf{W}^{V} \in \mathbb{R}^{d \times d}$ are learnable weight matrices for query, key, and value, respectively. We compute the transformed matrix as follows:

$$\bar{\mathbf{X}} = \mathbf{A}\mathbf{V} \quad \text{where} \quad \mathbf{A} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^{\top}}{\sqrt{d}}\right). \tag{3}$$

$\bar{\mathbf{X}} \in \mathbb{R}^{k \times d}$ is the transformed matrix, $\mathbf{A} \in \mathbb{R}^{k \times k}$ is an attention score matrix for all pairs between the variables, and softmax($\cdot$) indicates the row-wise softmax function. We then adopt a position-wise feed forward network (FNN) and residual connections as follows, to impose nonlinearity in the transformation and enable it to learn an identity function if needed:

$$\mathbf{H} = \text{Trans}(\mathbf{X}) = \mathbf{X} + \bar{\mathbf{X}} + \text{FNN}(\mathbf{X} + \bar{\mathbf{X}}), \tag{4}$$

where $\mathbf{H} \in \mathbb{R}^{k \times d}$ is the hidden representation matrix of the input variables, Trans($\cdot$) is the transformer, FNN($\cdot$) is a 2-layered position-wise feed forward network with the structure $\mathbb{R}^{d} \to \mathbb{R}^{4d} \to \mathbb{R}^{d}$. We adopt multi-head attention in $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ because the multi-head attention shows better performance than a single-head attention in various works such as recommender systems [34], natural language processing [5], and computer vision [17]. We also adopt dropout and layer normalization after the attention and FNN to improve the generalization performance. We stack the transformer layer (Equation 4) L times to represent the complicated relationships between the input variables.

Query-attention module. We have the hidden representation matrix $\mathbf{H} \in \mathbb{R}^{k \times d}$ and a query vector $\mathbf{q} \in \mathbb{R}^{d'}$. We denote i'th row vector in $\mathbf{H}$ as $\mathbf{h}_i \in \mathbb{R}^{d}$, and each $\mathbf{h}_i$ corresponds to the hidden representation vector of $\mathbf{x}_i$. The goal of query-attention module is to summarize the hidden representation matrix $\mathbf{H}$ into a single vector $\mathbf{h} \in \mathbb{R}^{d}$ while capturing significant variables depending on the query vector $\mathbf{q}$. In other words, we need to give more weight to information relevant to the query since the input variables do not equally contribute to the result. We propose a query-attention module as follows:

$$\begin{aligned} \mathbf{h} &= \text{QueryAtt}(\mathbf{H}, \mathbf{q}) = \sum_{i=1}^{k} \alpha_i \mathbf{h}_i, \quad \text{where} \\ \alpha_i &= \frac{\exp(\beta_i)}{\sum_{j=1}^{k} \exp(\beta_j)}, \quad \beta_i = \mathbf{q}^{\top} \tanh\left(\mathbf{W}^{H}\mathbf{h}_i + \mathbf{b}^{H}\right). \end{aligned} \tag{5}$$

$\mathbf{h} \in \mathbb{R}^{d}$ is the summarized vector, $\mathbf{H} \in \mathbb{R}^{k \times d}$ is the hidden representation matrix, $\mathbf{q} \in \mathbb{R}^{d'}$ is the query vector, $\mathbf{h}_i \in \mathbb{R}^{d}$ is i'th row vector of $\mathbf{H}$, $\alpha_i, \beta_i \in \mathbb{R}$ are a normalized and an unnormalized scores for $\mathbf{h}_i$, respectively, $\mathbf{W}^{H} \in \mathbb{R}^{d' \times d}$ and $\mathbf{b}^{H} \in \mathbb{R}^{d'}$ are learnable parameters, and tanh($\cdot$) is the hyperbolic-tangent function. We first apply a linear projection using the trainable parameters $\mathbf{W}^{H}$ and $\mathbf{b}^{H}$ to $\mathbf{h}_i$, to transform it into the space of the query vector $\mathbf{q}$. The hyperbolic tangent function makes the transformed vector give element-wise weights to the query vector $\mathbf{q}$. Finally, we perform the weighted sum of the hidden vectors while considering the importance of each hidden vector $\mathbf{h}_i$. As a result, we obtain the summarized vector $\mathbf{h}$ while considering the correlations between the input vectors of $\mathbf{X}$ using the self-attention module, and capturing the significant input vectors depending on the query vector $\mathbf{q}$ using the query-attention module.

3.4 Context-aware Action Encoder

The objective of the context-aware action encoder is to encode the information of an individual action. To effectively encode each action, the encoder requires two functionalities as follows. First, it is necessary to correlate a device control and its temporal contexts. For instance, assume one opens a blind on Monday morning. We need to correlate opening the blind with morning since the correlation is more important than each of them. Second, it is required to give more weight to the significant information in an action. For example, suppose a user turns on an air conditioner on Monday night. In the action, turning on the air conditioner is more important than the other information because the user is likely to turn off the air conditioner in the future. Then, how can we consider the two functionalities when encoding an action?

For i'th action in session u, we have a quadruplet of indices $(d_{u,i}^{(1)}, d_{u,i}^{(2)}, c_{u,i}^{(1)}, c_{u,i}^{(2)})$, where each of them are indices of device, device control, day of week, and hour, respectively. We firstly gain each embedding vectors $\mathbf{e}_{u,i}^{(1)}, \mathbf{e}_{u,i}^{(2)}, \mathbf{z}_{u,i}^{(1)}, \mathbf{z}_{u,i}^{(2)} \in \mathbb{R}^{d}$ for the given indices which correspond to the embedding vectors of $d_{u,i}^{(1)}, d_{u,i}^{(2)}, c_{u,i}^{(1)}$, and $c_{u,i}^{(2)}$, respectively. To embody the aforementioned functionalities, we propose to employ the queried transformer encoder (QTE) for the context-aware action encoder as follows:

$$\mathbf{h}_{u,i} = f_c(\mathbf{X}_{u,i}, \mathbf{q}_c), \tag{6}$$

where $\mathbf{h}_{u,i} \in \mathbb{R}^{d}$ is the hidden representation vector of i'th action in session u, $f_c(\cdot)$ is the context-aware action encoder, $\mathbf{X}_{u,i} \in \mathbb{R}^{4 \times d} = [\mathbf{e}_{u,i}^{(1)}, \mathbf{e}_{u,i}^{(2)}, \mathbf{z}_{u,i}^{(1)}, \mathbf{z}_{u,i}^{(2)}]^{\top}$ is a matrix of stacked vectors, and $\mathbf{q}_c \in \mathbb{R}^{d}$ is a trainable global query vector. The context-aware action encoder f_c uses the structure of QTE. We use the same query vector $\mathbf{q}_c$ for all actions in the all sessions because we aim to learn global representations for the significance of device controls and temporal contexts. As a result, for i'th action of session u, we obtain the encoded vector $\mathbf{h}_{u,i}$ which considers the correlations between the device control and the temporal contexts, and gives more weight to the significant information.

3.5 Context-attentive Sequence Encoder

The objective of the context-attentive sequence encoder is to encode a sequence while considering the sequential patterns and the current context. It is important to consider the correlations

Table 2: Statistics of Log Datasets.

Name	Region	Time period (Y-M-D)	# Sessions	# Instances	# Devices	# Device controls
KR	Korea	2021-11-20 ~ 2021-12-20	12,992	285,409	38	272
US	USA	2022-02-22 ~ 2022-03-21	4,764	67,882	40	268
SP	Spain	2022-02-28 ~ 2022-03-30	1,506	15,665	34	234
FR	France	2022-02-27 ~ 2022-03-25	388	4,423	33	222

Table 3: Statistics of Routine Datasets.

Name	Region	# Routines	# Devices
AP	Asia-Pacific	17,773	36
NA	North America	26,241	35
EU	Europe	23,781	28

between actions to capture the meaning of the sequence. For instance, locking the door comes after closing the window if the user wants to go out. This example shows that the meaning of sequence changes even if only one action changes. It is also critical to be aware of the current context to predict the next action. For example, assume a user has turned off the light. After that, the user is likely to open the blind to brighten up the room at daytime, while is likely to close the blind to sleep at night. Then, how can we consider sequential patterns and the current contexts simultaneously when encoding the sequence?

For each session u, we have a set of vectors $[\mathbf{h}_{u,1}, ..., \mathbf{h}_{u,(t-1)}]$, where $\mathbf{h}_{u,i} \in \mathbb{R}^d$ is the hidden representation vector of i'th action in the session; note that $\mathbf{h}_{u,1}, ..., \mathbf{h}_{u,(t-1)}$ are from the context-aware action encoder. To encode the session u, we propose to utilize the queried transformer encoder (QTE) for the context-attentive sequence encoder as follows:

$$\mathbf{s}_{u,t} = f_s(\mathbf{H}_u + \mathbf{P}, \text{concat}(\mathbf{z}^{(1)}_{u,t}, \mathbf{z}^{(2)}_{u,t})), \tag{7}$$

where $\mathbf{s}_{u,t} \in \mathbb{R}^d$ is the encoded vector for the current time t, $f_s(\cdot)$ is the context-attentive sequence encoder, $\mathbf{H}_u \in \mathbb{R}^{(t-1)\times d} = [\mathbf{h}_{u,1}, \dots \mathbf{h}_{u,(t-1)}]^\top$ is a matrix of stacked vectors, $\mathbf{P} \in \mathbb{R}^{(t-1)\times d}$ is a positional embedding matrix, concat$(\cdot)$ is the concatenation, and $\mathbf{z}^{(1)}_{u,t}, \mathbf{z}^{(2)}_{u,t} \in \mathbb{R}^d$ correspond to the embedding vectors of the current temporal contexts $c^{(1)}_{u,t}, c^{(2)}_{u,t}$ where $c^{(1)}_{u,t}$ and $c^{(2)}_{u,t}$ are day of week and hour at the current time t, respectively. The context-attentive sequence encoder f_s uses the structure of QTE. The positional embedding matrix $\mathbf{P}$ allows us to identify the position of the input variable, which leads us to learn sequential patterns. We define the $\mathbf{P}$ as a trainable matrix as in previous works [34, 37] for better generalization performance.

After obtaining $\mathbf{s}_{u,t}$, we compute the probabilities of device controls as follows:

$$\hat{\mathbf{y}}_{u,t} = \text{softmax}(\mathbf{E}\, \mathbf{s}_{u,t}), \tag{8}$$

where $\hat{\mathbf{y}}_{u,t} \in \mathbb{R}^{N_d}$ is the predicted probabilities of device controls at the current time t for user u, softmax$(\cdot)$ is the softmax function, $\mathbf{E} \in \mathbb{R}^{N_d \times d}$ is the matrix of device controls for the prediction, and N_d is the number of device controls.

3.6 Commonsense Knowledge Transfer

The aim of commonsense knowledge transfer module is to refine the embedding vectors based on the intention of actions. As shown in Figure 2, histories of actions often contain capricious intentions. This misleads the model to learn false relationship between two unrelated actions which co-occurred in the same sequence but are performed for different purposes. To solve this problem, we utilize the routine data that include massive routines of multiple users. Routine data are collections of frequently used device patterns triggered by various contextual backgrounds. Devices of each routine share the common intention since users submit routines to conveniently execute multiple actions for specific tasks such as doing the laundry or cooling off the room. Our idea is to adopt transfer learning mechanism, which is widely used in various domain adaptation tasks [15, 24, 35, 42, 46], to utilize the knowledge of the routine data. Specifically, we perform regularization such that similarities between devices in the same routine to be high while those not in the same routine to be low. Thus, regularization from the routine data makes embedding vectors of related devices sharing the same intention to be closer to each other.

Let i'th routine instance $\mathcal{R}_i$ be $[d_1, d_2 \dots, d_r]$, where d_j is j'th device of the instance. Let $\mathbf{e}_j \in \mathbb{R}^d$ be the embedding vector of device d_j. Note that the commonsense knowledge transfer module shares the device embeddings with the context-aware action encoder and the context-attentive sequence encoder. We define the regularization loss to minimize as follows:

$$\mathcal{L}_{reg} = -\sum_i \sum_{d_j \in \mathcal{R}_i} \left(\log\Big(\sigma(\mathbf{e}_j^\top \mathbf{e}_{j+1})\Big) + \sum_{d_k \in p(\mathcal{R}_i)} \log\Big(\sigma(-\mathbf{e}_j^\top \mathbf{e}_k)\Big) \right), \tag{9}$$

where $\sigma(\cdot)$ is the sigmoid function and $p(\mathcal{R}_i)$ is the negative samples of $\mathcal{R}_i$. For each device d_j, we randomly select a set of negative devices of size m, which are not in routine $\mathcal{R}_i$.

3.7 Objective Function

We train SmartSense to minimize the cross-entropy loss and the regularization loss as follows:

$$\mathcal{L}(\mathcal{X}, \mathbf{Y}) = -\frac{1}{n} \sum_u \sum_i \mathbf{y}_{u,t}(i) \log \hat{\mathbf{y}}_{u,t}(i) + \mathcal{L}_{reg}, \tag{10}$$

where $\mathcal{X} \in \mathbb{R}^{n \times l \times 4}$ is an input tensor, and $\mathbf{Y} \in \mathbb{R}^{n \times N_d}$ is a matrix of the ground-truth label; n and l are the numbers of sessions and time steps, respectively, and N_d is the number of device controls. $\mathbf{y}_{u,t} \in \mathbb{R}^{N_d}$ is the one-hot vector of the ground-truth label for session u, and $\mathbf{y}_{u,t}(i) \in \mathbb{R}$ is the i'th element in the vector. Similarly, $\hat{\mathbf{y}}_{u,t} \in \mathbb{R}$ is the predicted probability for session u.

4 EXPERIMENTS

We perform experiments to answer the following questions:

Q1. **Accuracy.** Does SmartSense show higher accuracy in action recommendation for smart home than baselines?

Q2. **Ablation study.** How do main ideas of SmartSense help improving the performance of recommendation?

Q3. **Case study.** Does SmartSense capture meaningful correlations from each action? How do the recommendation results of SmartSense change according to given contexts?

Q4. **Embedding space analysis.** Does SmartSense successfully learn useful embedding vectors of contexts and devices?

4.1 Experimental Setup

We introduce our experimental setup: datasets, baselines, evaluation metrics, experimental process, and the hyperparameters.

Datasets. We use real-world SmartThings datasets collected by Samsung from various regions to evaluate the performance. There are four log datasets and three routine datasets. Log datasets contain histories of device controls executed by users of Bixby. These are used to generate sequential instances for general sequential recommendation. As explained in Section 3.6, routine datasets are collections of frequently used device patterns triggered by various contextual backgrounds. Those instances are submitted by users of SmartThings. These datasets are used to gain commonsense knowledge in the corresponding log datasets.

Tables 2 and 3 show the statistics of log datasets and routine datasets, respectively. In the routine datasets, AP and NA correspond to KR and US in log datasets, respectively. EU corresponds to SP and FR in log datasets.

Baselines. We compare SmartSense with existing methods of sequential recommendation and context-aware recommendation.

- **POP** recommends device controls based on their popularity.
- **FMC [31]** uses two item embeddings to build an item transition matrix which predicts next device control.
- **TransRec [7]** represents relationships between consecutive items as a vector operation to perform sequential recommendation.
- **Caser [36]** employs CNN [22] in both time-axis and feature-axis to capture temporal dynamics in a sequential recommendation.
- **SASRec [16]** uses time-growing directional transformer encoder to consider sequential patterns of user actions.
- **BERT4Rec [34]** utilizes BERT architecture [5] into sequential recommendation.
- **SIAR [29]** stacks RNN layers of contexts and actions to consider temporal dynamics of both contexts and actions.
- **CA-RNN [25]** uses context-specific transition matrix in RNN cell to consider context-dependent features in a sequential recommendation.

Note that these baselines are not aware of commonsense knowledge of action recommendation for smart home. Thus, baseline methods use only log datasets during evaluation.

Evaluation metrics. We evaluate the performance of competing models with two evaluation metrics: hit ratio (HR@k) and mean average precision (mAP@k). Both metrics compare the recommendation list of the model with the ground truth value. For the k size of recommendation list, HR@k treats every item in them equally important, while mAP@k treats higher-ranked items more importantly. We vary k in $\{1, 3, 5\}$ for all datasets.

Experimental process. For each session in a log dataset, we create sequential instances with a window of length 10. The first nine events of the window are input of the sequential recommendation model. Each input event is a pair of a temporal context and a device control information. Temporal context of an event is a combination of a day of week and hour based on the event's timestamp. The hour in a context is one of the 8 time ranges of 3 hours length: 0-3, 3-6, 6-9, 9-12, 12-15, 15-18, 18-21, and 21-24. Device control information is composed of a device and its control. The device control of the last event is the ground truth for the window. We randomly split sequential instances into a training, a validation, and a test sets in 7:1:2 ratio. We train the model until the validation accuracy is maximized, and report the test accuracy.

Hyperparameters. All models are trained with Adam optimizer [18] with learning rate 0.001 and l_2 regularization coefficient 0.00001. For fair comparison, we set the size of embedding vectors to 50 and the size of mini batch 1024 for all models. For both the context-aware action encoder and the context-attentive sequence encoder, we set the numbers of both transformer layers and the heads as 2. We set the dropout ratio to 0.1, and the size of negative samples $m = 5$ for the commonsense knowledge transfer module.

4.2 Recommendation Accuracy (Q1)

We measure the accuracy of SmartSense and baselines in four real world datasets. Table 4 shows the result in terms of mAP. Note that SmartSense consistently outperforms baselines in all cases. Table 5 shows the result in terms of HR@k. Note that SmartSense shows the best performance in most cases, for various k. These results show that SmartSense is an accurate method for action recommendation for smart home.

4.3 Ablation Study (Q2)

We verify the effectiveness of our three main ideas, context-aware action encoder, context-attentive sequence encoder, and commonsense knowledge transfer. We compare SmartSense with SmartSense-*Act*, SmartSense-*Seq*, SmartSense-*Reg*, and SmartSense-*All*. SmartSense-*Act* is a SmartSense without a context-aware action encoder. This model encodes each pair of a temporal context and a device control as the mean value of corresponding embeddings. Thus, the model is not aware of correlations of temporal context and device control. SmartSense-*Seq* is a SmartSense without a context-attentive sequence encoder. The sequence encoder of this model summarizes the output of transformer layers as a mean value instead of a weighted sum. This is equivalent to the context-attentive sequence encoder where the given query is a zero vector instead of the current temporal context embedding. In this way, the model is not aware of the current temporal context during prediction. SmartSense-*Reg* is a SmartSense without the regularization from commonsense knowledge transfer module. This model cannot access information gathered from routine datasets. SmartSense-*All* is a SmartSense without any of the main ideas.

Table 6 shows that SmartSense is the most accurate model among competitors, while SmartSense-*All* shows the worst accuracy. In summary, all three main ideas are helpful to improve performance of action recommendation for smart home.

4.4 Case Study (Q3)

We analyze cases to observe how SmartSense deals with context information.

First, we observe the attention scores of the context-aware action encoder to find out how SmartSense reacts to the given pair of

Table 4: MAP@k of SmartSense and competitors for smart-home recommendation on four real-world datasets. SmartSense outperforms all competitors in all cases, demonstrating its superiority of action recommendation for smart home. Bold and underlined values indicate the best and the second-best accuracies, respectively.

Model	mAP@k											
	Korea			USA			Spain			France		
	@1	@3	@5	@1	@3	@5	@1	@3	@5	@1	@3	@5
POP	0.3416	0.4918	0.5045	0.1886	0.3146	0.3737	0.4973	0.6337	0.6455	0.4949	0.5955	0.6114
FMC [31]	0.5075	0.6391	0.6569	0.4581	0.6082	0.6270	0.4102	0.5953	0.6015	0.4427	0.6330	0.6477
TransRec [7]	0.3854	0.5637	0.5830	0.3351	0.5240	0.5426	0.3819	0.6149	0.6209	0.4255	0.6238	0.6393
Caser [36]	0.5676	0.7064	0.7213	0.5535	0.7051	0.7177	0.7906	0.8548	0.8616	0.7706	0.8249	0.8295
SASRec [16]	0.5763	0.7064	0.7212	0.5657	0.7098	0.7228	<u>0.7929</u>	0.8570	0.8630	0.7740	0.8286	0.8389
BERT4Rec [34]	0.5927	<u>0.7253</u>	<u>0.7393</u>	0.5630	0.7121	0.7254	0.7887	<u>0.8610</u>	<u>0.8662</u>	<u>0.7776</u>	<u>0.8475</u>	<u>0.8507</u>
CA-RNN [25]	0.5703	0.6958	0.7095	0.4860	0.6315	0.6459	0.6748	0.7253	0.7350	0.5141	0.5650	0.5767
SIAR [29]	<u>0.5936</u>	0.7248	0.7381	<u>0.5718</u>	<u>0.7163</u>	<u>0.7288</u>	0.7913	0.8560	0.8628	0.7706	0.8258	0.8311
SmartSense (proposed)	**0.6515**	**0.7650**	**0.7760**	**0.6247**	**0.7541**	**0.7639**	**0.8101**	**0.8707**	**0.8756**	**0.7944**	**0.8544**	**0.8578**

Table 5: HR@k of SmartSense and competitors for smart-home recommendation on four real-world datasets. SmartSense outperforms all competitors in most cases, demonstrating its superiority of action recommendation for smart home. Bold and underlined values indicate the best and the second-best accuracies, respectively.

Model	HR@k											
	Korea			USA			Spain			France		
	@1	@3	@5	@1	@3	@5	@1	@3	@5	@1	@3	@5
POP	0.3416	0.6527	0.7095	0.1886	0.4872	0.7493	0.4973	0.7916	0.8426	0.4949	0.7243	0.7955
FMC [31]	0.5075	0.7921	0.8683	0.4581	0.7994	0.8811	0.4102	0.7966	0.8226	0.4427	0.8529	0.9161
TransRec [7]	0.3854	0.7649	0.8478	0.3351	0.7596	0.8405	0.3819	0.8825	0.9085	0.4255	0.8586	0.9247
Caser [36]	0.5676	0.8711	0.9345	0.5535	0.8886	0.9429	0.7906	0.9295	0.9591	0.7706	0.8859	0.9073
SASRec [16]	0.5763	0.8603	0.9240	0.5657	0.8862	0.9420	<u>0.7929</u>	0.9320	<u>0.9682</u>	0.7740	0.8938	0.9377
BERT4Rec [34]	0.5927	<u>0.8825</u>	<u>0.9424</u>	0.5630	<u>0.8932</u>	**0.9502**	0.7887	**0.9461**	**0.9691**	<u>0.7776</u>	**0.9303**	**0.9460**
CA-RNN [25]	0.5703	0.8428	0.9020	0.4860	0.8096	0.8718	0.6748	0.7906	0.8324	0.5141	0.6294	0.6814
SIAR [29]	<u>0.5936</u>	0.8800	0.9369	<u>0.5718</u>	0.8918	0.9453	0.7913	0.9314	0.9607	0.7706	0.8893	0.9119
SmartSense (proposed)	**0.6515**	**0.8983**	**0.9454**	**0.6247**	**0.9079**	<u>0.9495</u>	**0.8101**	<u>0.9413</u>	0.9623	**0.7944**	<u>0.9232</u>	<u>0.9379</u>

Table 6: Ablation study of SmartSense. We report the performances by MAP@k. Note that SmartSense without at least one of the three main ideas decreases the performance, while SmartSense with all the ideas shows the best performance.

Model	Korea			USA		
	@1	@3	@5	@1	@3	@5
SmartSense-*Act*	0.5925	0.7256	0.7389	0.5802	0.7228	0.7350
SmartSense-*Seq*	0.6484	0.7631	0.7743	0.6194	0.7489	0.7592
SmartSense-*Reg*	0.6461	0.7608	0.7721	0.6189	0.7497	0.7600
SmartSense-*All*	0.5941	0.7265	0.7396	0.5752	0.7198	0.7321
SmartSense	**0.6515**	**0.7650**	**0.7760**	**0.6247**	**0.7541**	**0.7639**

temporal context and a device control. Figure 5 visualizes the attention scores in the last layer of the encoder when the given temporal context is Monday with the hour of 9 to 12. Each row of the figure corresponds to a device control at a specific temporal context; the colors show how much attention the encoder gives to day of the week, hour, device, and its control information. Note that on the first row, the attention score between hour and blind is relatively high since the degree of control for incoming sunlight depends on the time. The attention score between a computer and hour is also high, because computers are turned on and off according to work hours. However, dishwasher and robot cleaner do not have strong correlations with hour. Both dishwasher and robot cleaner

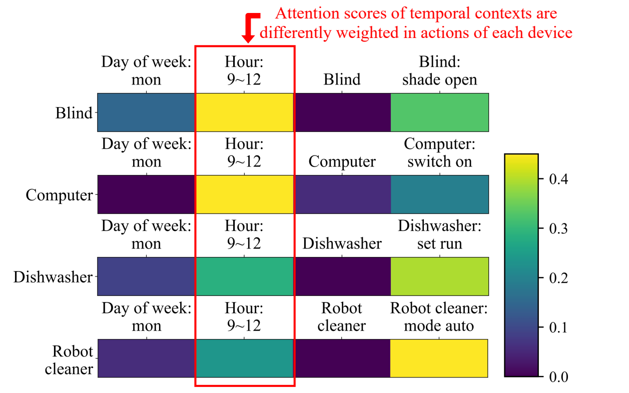

Figure 5: Attention scores in the context-aware action encoder (see Section 4.4 for details).

can be controlled when users are not paying attention to the device, so they can be controlled at any time. In summary, SmartSense successfully captures the important correlation from given inputs with a context-aware action encoder.

Second, we observe how the current temporal context affects recommendation results. Figure 6 shows top-5 recommendation results and attention scores of each action in a history depending on the current temporal context. The attention scores of the sixth and the seventh actions are high in case (A), while attention scores of

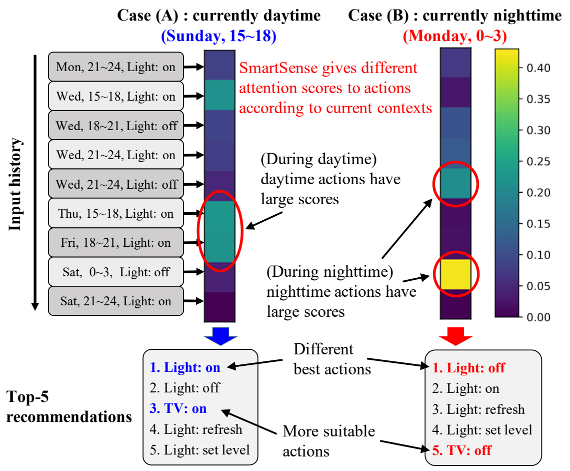

Figure 6: Top-5 recommendation lists and attention scores for each action depending on the current temporal context (see Section 4.4 for details).

the fifth and the eighth actions are high in case (B). They are actions with the most similar contexts compared to the current context in each case. SmartSense recommends turning on a light the most in case (A) but it recommends turning off a light the most in case (B), which are device controls of the most attentive actions. This shows that SmartSense dynamically collects important information from the input sequence depending on the current temporal context. Moreover, SmartSense recommends turning on a television in case (A) but it recommends turning off a television in case (B). In case (A), it is appropriate to recommend turning on the television because a user would be active since the given context is daytime. However, the given context is nighttime in case (B) so it is reasonable to recommend turning off the television because a user would like to sleep. This shows that SmartSense understands not only the difference between contexts but also the hidden meanings of each context. In summary, SmartSense is aware of the current temporal context while prediction, so it flexibly generates an accurate recommendation for the given situation.

4.5 Embedding Space Analysis (Q4)

We observe the embeddings of devices and temporal contexts to analyze how they reflects the real world.

First, we observe the device embedding space to verify the impact of commonsense knowledge transfer in the representation learning. Figure 7 shows that the cosine similarities between embedding vectors of devices in SmartSense-*Reg* (SmartSense without the commonsense knowledge transfer regularization), and SmartSense. The standard deviation of cosine similarities in SmartSense is 0.21 while that of SmartSense-*Reg* is 0.11, which shows that device embedding vectors in SmartSense-*Reg* have more complex patterns as the regularizations are applied. Furthermore, embedding vectors of similar devices in SmartSense such as sensors get closer to each other in SmartSense compared to SmartSense-*Reg*. This shows that routine data are useful to find close relationships between devices due to its task oriented composition.

Second, we visualize the hour embeddings to see whether SmartSense successfully captures the characteristic of temporal contexts. Figure 8 shows that embeddings of close hours are more similar

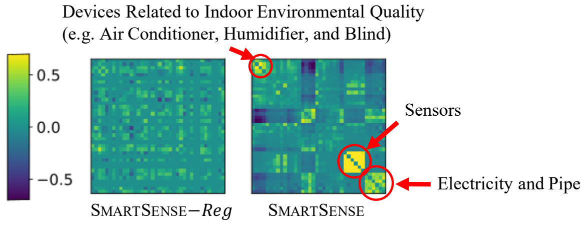

Figure 7: Matrix of cosine similarities between different device embedding vectors in SmartSense-*Reg* (left) and SmartSense (right). Device embedding vectors of similar devices are closer to each other in SmartSense compared to SmartSense-*Reg*, thanks to the common sense knowledge transfer from routine data.

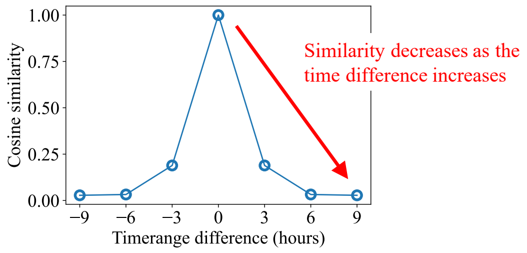

Figure 8: Average cosine similarity between two hour embedding vectors depending on their time difference. Embeddings of close hours have similar values. This shows that embeddings of temporal contexts successfully represent the real world characteristics of corresponding entities.

to each other compared to farther hours. This shows SmartSense extracts the crucial contextual information from temporal contexts.

5 CONCLUSION

In this paper, we propose SmartSense, an accurate action recommendation method for smart home. To reflect the importance of correlations, SmartSense introduces context-aware action encoder which captures significant correlations between the device control and the temporal context. The context-attentive sequence encoder of SmartSense summarizes users' sequential pattern and queried contextual information to consider both personalization and the current context. Commonsense knowledge transfer in SmartSense enables the model to successfully consider user intentions. As a result, SmartSense shows the state-of-the-art accuracy giving up to 9.8% higher mAP@1 in action recommendation for smart home than the best competitor in extensive experiments. Through case studies, we also show that SmartSense successfully captures the correlations among actions and contexts.

ACKNOWLEDGMENTS

This work was supported by Samsung Electronics. The Institute of Engineering Research at Seoul National University provided research facilities for this work. The ICT at Seoul National University provides research facilities for this study. We thank Inchul Hwang, Sanghee Kim, and Hyunju Seo for their help and insightful discussion. U Kang is the corresponding author.

REFERENCES

[1] Linas Baltrunas, Bernd Ludwig, and Francesco Ricci. 2011. Matrix factorization techniques for context aware recommendation. In *RecSys*. ACM.
[2] Jianxin Chang, Chen Gao, Yu Zheng, Yiqun Hui, Yanan Niu, Yang Song, Depeng Jin, and Yong Li. 2021. Sequential Recommendation with Graph Neural Networks. In *SIGIR*. ACM.
[3] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential Recommendation with User Memory Networks. In *WSDM*. ACM.
[4] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *EMNLP*. ACL.
[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*. ACL.
[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2020. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *CoRR* (2020).
[7] Ruining He, Wang-Cheng Kang, and Julian J. McAuley. 2017. Translation-based Recommendation. In *RecSys*. ACM.
[8] Ruining He and Julian J. McAuley. 2016. Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation. In *ICDM*. IEEE Computer Society.
[9] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent Neural Networks with Top-k Gains for Session-based Recommendations. In *CIKM*. ACM.
[10] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *ICLR*.
[11] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* (1997).
[12] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y. Chang. 2018. Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks. In *SIGIR*. ACM.
[13] Dietmar Jannach and Malte Ludewig. 2017. When Recurrent Neural Networks meet the Neighborhood for Session-Based Recommendation. In *RecSys*. ACM.
[14] Hyunsik Jeon, Bonhun Koo, and U Kang. 2019. Data Context Adaptation for Accurate Recommendation with Additional Information. In *IEEE BigData*.
[15] Hyunsik Jeon, Seongmin Lee, and U Kang. 2021. Unsupervised multi-source domain adaptation with no observable source data. *Plos one* (2021).
[16] Wang-Cheng Kang and Julian J. McAuley. 2018. Self-Attentive Sequential Recommendation. In *ICDM*. IEEE Computer Society.
[17] Salman H. Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. 2021. Transformers in Vision: A Survey. *CoRR* (2021).
[18] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.
[19] Bonhun Koo, Hyunsik Jeon, and U Kang. 2020. Accurate News Recommendation Coalescing Personal and Global Temporal Preferences. In *PAKDD*. Springer.
[20] Bonhun Koo, Hyunsik Jeon, and U Kang. 2021. PGT: news recommendation coalescing personal and global temporal preferences. *Knowledge and Information Systems* (2021).
[21] Yehuda Koren, Robert M. Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* (2009).
[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *NIPS*.
[23] Saurabh Kulkarni and Sunil F. Rodd. 2020. Context Aware Recommendation Systems: A review of the state of the art techniques. *Comput. Sci. Rev.* (2020).
[24] Seongmin Lee, Hyunsik Jeon, and U Kang. 2021. Multi-EPL: Accurate multi-source domain adaptation. *Plos one* (2021).
[25] Qiang Liu, Shu Wu, Diyi Wang, Zhaokang Li, and Liang Wang. 2016. Context-Aware Sequential Recommendation. In *ICDM*. IEEE Computer Society.
[26] Chen Ma, Liheng Ma, Yingxue Zhang, Jianing Sun, Xue Liu, and Mark Coates. 2020. Memory Augmented Graph Neural Networks for Sequential Recommendation. In *AAAI*. AAAI Press.
[27] Haekyu Park, Jinhong Jung, and U Kang. 2017. A comparative study of matrix factorization and random walk with restart in recommender systems. In *IEEE BigData*.
[28] Dimitrios Rafailidis and Yannis Manolopoulos. 2019. Can Virtual Assistants Produce Recommendations?. In *WIMS*. ACM.
[29] Lakshmanan Rakkappan and Vaibhav Rajan. 2019. Context-Aware Sequential Recommendations withStacked Recurrent Neural Networks. In *WWW*. ACM.
[30] Steffen Rendle. 2010. Factorization Machines. In *ICDM*, Geoffrey I. Webb, Bing Liu, Chengqi Zhang, Dimitrios Gunopulos, and Xindong Wu (Eds.). IEEE Computer Society.
[31] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing Personalized Markov Chains for Next-Basket Recommendation. In *WWW*. ACM.
[32] Steffen Rendle, Zeno Gantner, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2011. Fast context-aware recommendations with factorization machines. In *SIGIR*. ACM.
[33] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic Matrix Factorization. In *NIPS*.
[34] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *CIKM*. ACM.
[35] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. 2018. A Survey on Deep Transfer Learning. In *ICANN*. Springer.
[36] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *WSDM*. ACM.
[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NIPS*.
[38] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z. Sheng, and Mehmet Orgun. 2019. Sequential Recommender Systems: Challenges, Progress and Prospects. In *IJCAI*. AAAI Press.
[39] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-Based Recommendation with Graph Neural Networks. In *AAAI*. AAAI Press.
[40] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional Factorization Machines: Learning the Weight of Feature Interactions via Attention Networks. (2017).
[41] Xin Xin, Bo Chen, Xiangnan He, Dong Wang, Yue Ding, and Joemon M. Jose. 2019. CFM: Convolutional Factorization Machines for Context-Aware Recommendation. In *IJCAI*. AAAI Press.
[42] Huiwen Xu and U Kang. 2021. Transfer alignment network for blind unsupervised domain adaptation. *Knowledge and Information Systems.* (2021).
[43] Haochao Ying, Fuzhen Zhuang, Fuzheng Zhang, Yanchi Liu, Guandong Xu, Xing Xie, Hui Xiong, and Jian Wu. 2018. Sequential Recommender System based on Hierarchical Attention Networks. In *IJCAI*. AAAI Press.
[44] Jaemin Yoo, Yejun Soun, Yong-chan Park, and U Kang. 2021. Accurate Multivariate Stock Movement Prediction via Data-Axis Transformer with Multi-Level Contexts. In *KDD*. ACM.
[45] Shengyu Zhang, Dong Yao, Zhou Zhao, Tat-Seng Chua, and Fei Wu. 2021. CauseRec: Counterfactual User Sequence Synthesis for Sequential Recommendation. In *SIGIR*. ACM.
[46] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. 2019. A Comprehensive Survey on Transfer Learning. *CoRR* (2019).
[47] Adam Zielonka, Marcin Wozniak, Sahil Garg, Georges Kaddoum, Md. Jalil Piran, and Ghulam Muhammad. 2021. Smart Homes: How Much Will They Support Us? A Research on Recent Trends and Advances. *IEEE Access* (2021).