IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 4 October 2022, accepted 4 November 2022, date of publication 9 November 2022, date of current version 17 November 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3220781

RESEARCH ARTICLE

Unsupervised Learning of Domain-Independent User Attributes

YUICHI ISHIKAWA[1,2], ROBERTO LEGASPI[2], KEI YONEKAWA[2], YUGO NAKAMURA[1], (Member, IEEE), SHIGEMI ISHIDA[3], (Member, IEEE), TSUNENORI MINE[1], AND YUTAKA ARAKAWA[1], (Member, IEEE)

[1]Faculty of Information Science, Kyushu University, Nishi-ku, Fukuoka-shi, Fukuoka 819-0395, Japan
[2]KDDI Research Inc., Fujimino-shi, Saitama 356-8502, Japan
[3]Department of Media Architecture, School of Systems Information Science, Future University Hakodate, Hakodate, Hokkaido 041-8655, Japan

Corresponding author: Yuichi Ishikawa (yi-ishikawa@kddi-research.jp)

ABSTRACT Learning user attributes is essential for providing users with a service. In particular, for e-commerce portals which deal in variety of goods ranging from clothes to foods to home electronics, it is especially important to learn ''domain-independent'' attributes such as age, gender, and personality that affect people's behavior across various domains of daily life (e.g., clothing, eating and housing) because these attributes can be used for personalization in diverse domains their service covers. Thus far, researchers have proposed approaches to learn user representation (UR) from user-item interactions, trying to embed rich information about user attributes in UR. However, very few can learn URs that are domain-independent without confounding them with domain-specific attributes (e.g., food preferences). This could consequently undermine the former's utility for personalizing services in other domains from which the URs are not learned. To address this, we propose an approach to learn URs that exclusively reflect domain-independent attributes. Our approach introduces a novel multi-layer RNN with two types of layers: *Domain Specific Layers* (*DSLs*) for modeling behavior in individual domains and a *Domain Independent Layer* (*DIL*) for modeling attributes that affect behavior across multiple domains. By exchanging hidden states between these layers, the RNNs implement the process of domain-independent attributes affecting domain-specific behavior and makes the DIL learn URs that capture domain-independence. Our evaluation results confirmed that the URs learned by our approach have greater utility in predicting behavior in the other domains from which these URs were not learned thereby demonstrating adaptability to various domains.

INDEX TERMS Big Five, e-commerce, personality, RNN, user modeling, user representation learning.

I. INTRODUCTION

For those who provide online services, learning user attributes is an essential part of their business. It serves as a basis for personalizing the service for individual users and thus is a key to building a good relationship with users. Among such services are e-commerce portals, e.g., Amazon and Alibaba, which deal in variety of goods and services ranging from clothes to foods to home electronics and from music and movie streaming to photo storage, covering a diversity of domains important in daily life. The number of service users surpassed a billion in 2014 [1] and the pace of growth has been further accelerated due to the global spread of COVID-19 [1].

Among various user attributes, it is especially beneficial for the e-commerce portals to learn attributes such as age, gender, and personality that have the following two characteristics. The first is *domain-independence*, meaning that the attributes with this characteristic affect people's behavior across various domains of daily life. For example, in general, young and old and male and female lead different lifestyles and thus have different needs and preferences for food, clothes, music, and movies, which affect their purchasing behavior.

The associate editor coordinating the review of this manuscript and approving it for publication was Ángel F. García-Fernández.

[1]https://www.statista.com/statistics/251666/number-of-digital-buyers-worldwide/

Similarly, existing studies have confirmed that people's personalities also affect their preferences for food [2], brands [3], [4], music [5] and movies [6]. Therefore, these domain-independent attributes can be used for personalizing a service in the diverse domains that the portals cover. Once they are learned, they can be used for recommending items in a domain even if a user has not browsed or purchased items in this domain previously. Such domain adaptability is a key difference from domain-specific attributes which affect purchasing behavior only in a specific domain. The second is *stability* meaning that the attributes do not change markedly and are long-term and thus can be used for personalizing the service over an extended period of time.

Thus far, researchers have studied many approaches for learning a representation of user attributes (user representation; UR) [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18]. These approaches learn user attributes from user-item interactions (e.g., browsing, purchasing, or reviewing items by users) without using ground truth and embed learned attributes into high-dimensional vector representations. Compared to users' manual registration (e.g., asking users to register their attributes when they sign up for the service), in which only a limited amount of information is collected so as not to overburden users and some users intentionally/unintentionally register false information, the approaches can enable service providers including the e-commerce portals to learn richer and more reliable information about user attributes.

Of the aforementioned two key characteristics, domain-independence and stability, learning URs with the latter characteristic has attracted considerable research attention. Approaches that leverage sequential interaction data (e.g., Recurrent Neural Network (RNN)-based approaches) [10], [12], [13], [14], [15], [16], [17], [18] can distinguish short- and long-term user attributes and thus can learn URs for long-term attributes. On the other hand, domain independence has received little attention despite its importance in offering personalized services in various domains. The URs either reflect only user attributes specific to domains from which the URs are learned, or reflect jointly the domain-specific and domain-independent attributes but plausibly not without confounding each other, which could erode the utility of the URs to personalize services in the other domains from which the URs are not learned.

In light of the above, we propose an approach to learn URs from user-item interaction with both domain-independent and long-term attributes. As in the existing UR learning approaches, our approach learns the URs from sequential interaction data in an unsupervised manner. It distinguishes user attributes along two axes: long- or short-term and domain-specific or domain-independent, and learns four types of user attributes that are distinct from each other. By adopting an RNN, our approach separately models long- and short-term attributes. What is novel in our approach is that to model domain-specific and domain-independent attributes, we introduce a multi-layered RNN that consists of two types of layers: 1) multiple *Domain Specific Layers* (*DSLs*) that model user behavior in individual domains; and 2) a single *Domain Independent Layer* (*DIL*) that learns domain-independent attributes. The RNN takes sequences of items that the user interacted with in multiple domains as input (e.g., purchased clothes, foods and home electronics). Each DSL takes items in its corresponding domain (e.g., DSL1 takes clothes, DSL2 takes foods, DSL3 takes home electronics) and reflects the user's intention for the next item in its hidden state. When updating the hidden state, the DSL uses not only its own hidden state but also the DIL's hidden state. Since this is done in all the DSLs, it makes the DIL's hidden state affect user intention in all the domains that the DSLs correspond to. In effect, therefore, this enables the DIL to learn attributes that affect user behavior across multiple domains, i.e., domain-independent attributes.

Using publicly available datasets, Amazon [19] and Retail-Rocket [20], that contain real-world data collected from e-commerce portals, we learned URs and evaluated their degree of domain-independence. We posit that if the URs are domain-independent, then: 1) URs of the same user are similar regardless of domains where the URs are learned, and 2) the URs have utility for predicting behavior in the other domains from which these URs are not learned. Based on 1) and 2), we conducted evaluations to answer the following questions:

Q1 How similar are the URs of the same user when they are learned in different domains?

Q2 How much utility do the URs have for predicting behavior in the other domains from which they were not learned?

In addition to the questions for the UR's degree of domain-independence, we set another question which focuses on one of concrete domain-independent attributes and examined the extent to which the URs reflect this attribute.

Q3 To what degree do the URs have relevance to user personalities?

Although the datasets did not contain information that directly indicate any domain-independent attributes, one of the datasets contained item review texts written by users. Existing research confirmed that features extracted from user written texts have significant correlations with the user's personality [21], [22], [23]. Therefore, we considered that we would be able to examine the URs' degree of relevance to personalities by evaluating how accurately the URs can predict the text features that are significantly correlated with personalities.

Our contributions are as follows. 1) We propose a multi-layer RNN that separates the RNN layers to learn domain-independent user attributes from the other layers designated to model domain-specific behavior. This enables our approach to learn URs that exclusively reflect domain-independent attributes, which existing research has not focused on. 2) Using real user-item interaction data, we demonstrate our approach can learn URs that reflect

domain-independent attributes and are adaptable to various domains from which the URs are not learned. We also confirmed the possibility that our URs reflect user personalities, one of the domain-independent attributes, to a greater extent than existing approaches do.

II. RELATED WORK

Existing approaches for UR learning can be categorized into non-sequential and sequential approaches [7]. Matrix Factorization (MF) [8], as one of the representative non-sequential approaches [7], learns URs from a user-item interaction matrix by factorizing it into user and item matrices. While MF can use only ID information of users and items, Factorization Machine (FM) [9] can additionally use side information about users and items (e.g., user's device type, movie genre). Several approaches have been proposed to extend FM that models only pairwise interactions between user and item features so that higher order interactions can be incorporated [11], [24], [25], [26], [27]. Among them, xDeepFM [11] differs from others in that it introduces Compressed Interaction Network (CIN) to explicitly model such interactions, which had been modeled implicitly by just inputting the features into a vanilla deep neural network (DNN) in [24], [25], [26], and [27]. While these approaches have been widely deployed in many commercial systems for their simplicity and effectiveness, they are unable to distinguish long- and short-term attributes.

On the other hand, sequential approaches can learn URs that explicitly reflect either long- or short-term attributes from a sequence of items a user has interacted with. Several approaches in this category use an RNN [10], [12], [13], [18]. Our approach also falls into this category, specifically, the Sequential User-based RNN (SURNN) [10] in this category is the one our approach is based on. In contrast to our multi-layered structure, SURNN consists only of a single layer RNN in which URs are stored in a user matrix that an RNN cell has. When an item data (e.g., watched movie ID, purchased clothes ID) is input to the RNN cell, it retrieves the UR from the user matrix and uses it together with the input item data to update the hidden state. While the hidden state is updated sequentially with the input item, the user matrix (i.e., the UR) stays the same. This makes it possible for the long-term attributes to be reflected in the URs and short-term attributes in the hidden states.

Another thread of sequential approaches that has attracted research attention recently is universal UR learning [14], [16], [17], [28], [29], [30]. They apply the ''pretrain-finetune'' concept to user representation learning. Using a large amount of sequential interaction data, they pretrain URs for various purposes (e.g., item recommendation, user profiling) and finetune them for downstream tasks, which contrasts to the existing approaches (e.g., SURNN) that learn URs for a specific task. Self-supervised User Modeling Network (SUMN) [17] uses interaction data represented by text (e.g., names of items a user purchased, search logs of users), from which it extracts a UR by an attention mechanism. It compares a UR extracted from past actions and pattern of actions in the future and trains the model to minimize the loss between them so that it can obtain the representations of long-term attributes. For the same purpose, U-BERT [16] uses review texts and AutoEncoder-coupled Transformer Network (AETN) [14] uses application (un)installation logs of mobile phones.

While the sequential approaches can distinguish long-/short-term attributes, to the best of our knowledge, none of them can distinguish attributes that are domain-independent and domain-specific. Although all the above approaches can take the interaction data across multiple domains as input, the URs learned from the data are highly likely to reflect not only domain-independent but also domain-specific attributes (i.e., they are jointly represented by the same URs). The domain-specific attributes have little utility for predicting behavior in the other domains where the URs are not learned and thus lessens the effectiveness in performing the task if the URs are used in such domains.

III. PROPOSED APPROACH

Fig. 1 outlines our proposed approach. The left figure exemplifies our multi-layer RNN and the right figure shows how the four types of user attributes are mapped to the variables of our RNN. In our RNN, the DSLs correspond to individual domains. For example, if the input data contain items in two domains, then there will be two corresponding DSLs as shown in the left figure. Each DSL takes a sequence of user-item interactions in the corresponding domain, each of which is represented by a pair of one-hot vectors of user ID and item ID. Using the input, the DSL's RNN cell (DSL cell) retrieves a UR and an item representation from the user and item matrices, respectively, and uses these retrieved representations to update the hidden state. Following SURNN, while the hidden state is updated sequentially with input items, the user matrix (i.e., set of URs) stays the same. This enables the DSL to represent the short-term attribute in its hidden state and the long-term attribute in the UR. In addition, since it learns these attributes only from user actions in the corresponding domain, the attributes are domain-specific as shown in quadrants I and II of the right figure.

While the above flow follows SURNN, the DSL differs from SURNN in that it uses not only its own hidden state but also the DIL hidden state to update its hidden state as shown in link (A) in the left figure. The updated hidden state is used to predict the next item in the corresponding domain, thus can be regarded as representing user intention for the next action while it also represents action history in the domain. Updating such DSL hidden states using the DIL hidden states means that the DIL hidden states affect user actions in individual domains. Because all the DSLs update their hidden states in this way, the DIL learns user attributes that affect user actions across multiple domains, i.e., domain-independent attributes (quadrants III and IV). As in the DSL, the DIL also represents short-term attributes in its hidden state and long-term attributes in the UR, but the difference being the DIL would contain domain-independent attributes.

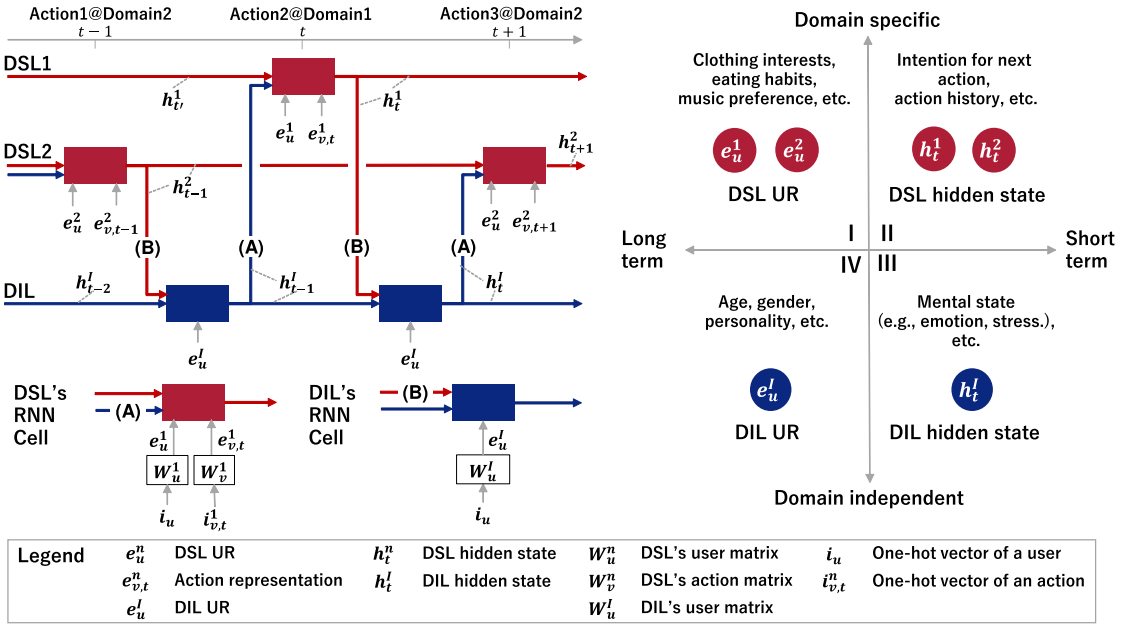


FIGURE 1. Example of our multi-layered RNN and relationship between the four types of user attributes and the RNN variables.

TABLE 1. List of notations.

Notation	Meaning	Subscript	Superscript
N_u, N_v^n	# of users/items	u: # of users, v: # of items	n: domain n
$\boldsymbol{i_u} \in \mathbb{R}^{N_u}, \boldsymbol{i}_{v,t}^n \in \mathbb{R}^{N_v^n}$	One hot vector of …	u: user ID, v, t: item ID of t-th action	
$d_{\text{ue}}^l, d_{\text{ue}}^n, d_{\text{ve}}^n, d_h^l, d_h^n$ $(d_{\text{ve}}^n = d_h^n)$	# of dimensions of …	ue: UR, ve: item representation, h: hidden state	l: DIL n: DSL n
$\boldsymbol{W}_u^l \in \mathbb{R}^{d_{\text{ue}}^l \times N_u}$, $\boldsymbol{W}_u^n \in \mathbb{R}^{d_{\text{ue}}^n \times N_u}$, $\boldsymbol{W}_v^n \in \mathbb{R}^{d_{\text{ve}}^n \times N_v^n}$	User/item matrix	u: user matrix, v: item matrix	
$\boldsymbol{e}_u^l \in \mathbb{R}^{d_{\text{ue}}^l}$, $\boldsymbol{e}_u^n \in \mathbb{R}^{d_{\text{ue}}^n}$, $\boldsymbol{e}_{v,t}^n \in \mathbb{R}^{d_{\text{ve}}^n}$	UR/item representation	u: UR, v, t: item representation of t-th action	
$\boldsymbol{h}_t^l \in \mathbb{R}^{d_h^l}, \boldsymbol{h}_t^n \in \mathbb{R}^{d_h^n}$	Hidden state	t: hidden state updated by t-th action	

In addition to link (A), we also added link (B), i.e., the DIL also updates its hidden state using the DSL hidden state. Our motivation is to reflect in our RNN the process in which user actions affect domain-independent attributes, especially the short-term attributes (quadrant III), among which is a mental state (e.g., emotion and stress), as it changes dynamically and affects future actions across various domains. While a mental state affects what people will do next, which link (A) corresponds to, a mental state is also affected by what they did before, which is reflected by link (B).

Another motivation for link (B) is to abstract in our RNN the moderating effect of domain-independent attributes, especially that of personality, on the relationship between actions and a mental state. For example, watching a horror movie is likely to induce anxiety and fear for those high in *Neuroticism*, which is one of the Big Five personality traits [31] and is known to indicate the response level to negative stimuli (e.g., threat) [32]. In contrast, those low in *Neuroticism* are more likely to just enjoy the movie with little anxiety and fear. As such, the same action induces different mental states depending on the personality of the individual. We reflect such moderating effect in our RNN via the DIL updating its hidden states, using its UR that represents personality, and the DSL hidden states.

We detail next the hidden state updating and model training. Refer to Table 1 for the notations and descriptions.

A. UPDATING THE HIDDEN STATES

Input to our RNN is formatted as

$$data_a = [x_{a,1}, x_{a,2}, \ldots, x_{a,t}, \ldots, x_{a,T}], \tag{1}$$

where $x_{a,t} = (\boldsymbol{i_u}, \boldsymbol{i}_{v,t}^n)$ denotes user a's t-th action. Once $x_{a,t}$ is input to a corresponding DSL (i.e., DSL n), the DSL first retrieves a UR and item representation from the user and item matrices, i.e., $\boldsymbol{e}_u^n = \boldsymbol{W}_u^n \boldsymbol{i_u}$ and $\boldsymbol{e}_{v,t}^n = \boldsymbol{W}_v^n \boldsymbol{i}_{v,t}^n$, respectively. It then updates its hidden state $\boldsymbol{h}_t^n$ using $\boldsymbol{e}_u^n$, $\boldsymbol{e}_{v,t}^n$, its previous hidden state $\boldsymbol{h}_{t'}^n$ (t' is the timing when the previous action was taken in domain n) and the DIL hidden state $\boldsymbol{h}_{t-1}^I$, which is received via link (A). The update in the DSL is formulated as

follows:

$$\boldsymbol{h}_t^n = f_{\mathrm{RNN}}^n(\, \boldsymbol{h}_{t'}^n + \boldsymbol{W}_I^n \boldsymbol{h}_{t-1}^I,\ \boldsymbol{e}_{v,t}^n,\ \boldsymbol{e}_u^n\ ), \tag{2}$$

where function f_{RNN}^n is used to update the hidden states, which can be implemented using Gated Recurrent Unit (GRU) [33] or Long Short-Term Memory (LSTM) [34]. As formulated in (2), $\boldsymbol{h}_{t-1}^I$ is added to $\boldsymbol{h}_{t'}^n$ after the linear transformation via $\boldsymbol{W}_I^n \in \mathbb{R}^{d_h^n \times d_h^I}$. This is done for two reasons. One is to make the number of dimensions of $\boldsymbol{h}_{t-1}^I$ conform to that of $\boldsymbol{h}_{t'}^n$. The other is to adjust the effect of $\boldsymbol{h}_{t-1}^I$ on the individual domains. As described, we regard $\boldsymbol{h}_{t-1}^I$ as reflecting a mental state. A mental state affects a user's intentions for the next actions to varying extents depending on the individual domains in which the next action is to be taken.

After updating a hidden state, the DSL sends it to the DIL via link (B), which is done every time the DSL updates its hidden state. When the DIL receives $\boldsymbol{h}_t^n$, it retrieves a UR from its user matrix ($\boldsymbol{e}_u^I = \boldsymbol{W}_u^I \boldsymbol{i}_u$) and updates its hidden state $\boldsymbol{h}_t^I$ using $\boldsymbol{h}_t^n$, $\boldsymbol{e}_u^I$, and its previous hidden state ($\boldsymbol{h}_{t-1}^I$). The update in the DIL is formulated as follows:

$$\boldsymbol{h}_t^I = f_{\mathrm{RNN}}^I(\, \boldsymbol{h}_{t-1}^I + \boldsymbol{W}_n^I \boldsymbol{h}_t^n,\ \boldsymbol{e}_u^I\ ), \tag{3}$$

where f_{RNN}^I is a function to update hidden states and can be implemented using GRU/LSTM. As in the DSL, the DIL also applies linear transformation ($\boldsymbol{W}_n^I \in \mathbb{R}^{d_h^I \times d_h^n}$) to $\boldsymbol{h}_t^n$. Because how a previous action affects a mental state may differ depending on the domain where this action was taken, we constructed $\boldsymbol{W}_n^I$ for each DSL.

We describe in Appendix VII how we implemented f_{RNN}^n and f_{RNN}^I in the experiment.

B. MODEL TRAINING

To train our RNN, we first divide $data_a$ into overlapping sliding windows with window size w and slide size s, e.g., $win_1 = [x_{a,t}, x_{a,t+1}, \ldots, x_{a,t+w-1}]$, $win_2 = [x_{a,t+s}, x_{a,t+s+1}, \ldots, x_{a,t+s+w-1}]$,..., and feed them to the RNN. When the window is fed, the RNN predicts the next item for each sequence in the window, e.g., if the input is win_1, the output is $[\hat{x}_{a,t+1}, \hat{x}_{a,t+2}, \ldots, \hat{x}_{a,t+w}]$. The predicted items are compared with the actual items to calculate the loss that is used to learn the parameters of the DSL and DIL cells (e.g., gates' weights and biases), the user and item matrices ($\boldsymbol{W}_u^n$, $\boldsymbol{W}_u^I$, and $\boldsymbol{W}_v^n$), and the transformation matrices ($\boldsymbol{W}_I^n$ and $\boldsymbol{W}_n^I$).

As shown in Fig. 2, we have two options to calculate the loss, $Loss_1$ and $Loss_2$ (indicated by thick grey arrows). $Loss_1$ is based on our design notion that the DSL hidden state represents user intention for the next action in the corresponding domain, and the DSL predicts the next item in the domain using its hidden state. In Fig. 2, for example, DSL1 predicts the item of Action3@Domain1 using $\boldsymbol{h}_t^1$ from Action1@Domain1. However, if the user takes an action in another domain, Action2@Domain2, which is before the target action Action3@Domain1, such an action taken cannot be considered in the prediction at Action3@Domain1. Action2@Domain2, however, would actually affect the user's

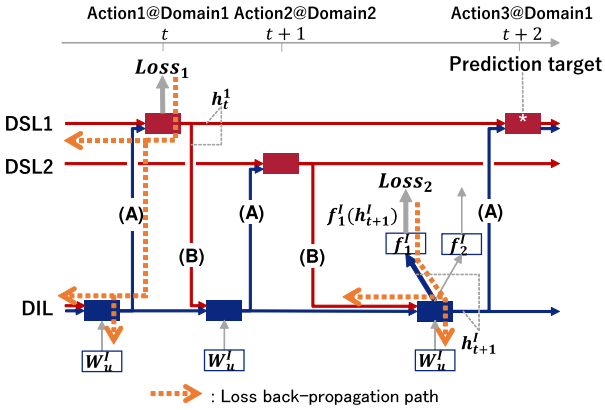


FIGURE 2. Two options for loss calculation, $Loss_1$ and $Loss_2$.

mental state, consequently, what he will do in Domain1, but is not considered in $Loss_1$. In addition, before reaching the DIL user matrix ($\boldsymbol{W}_u^I$), $Loss_1$ needs to be used through two RNN cells, inside each of which are functions that cause vanishing gradient (i.e., sigmoid, tanh).

Given the above issues, we designed $Loss_2$ that is calculated by the DIL. It uses its hidden state $\boldsymbol{h}_{t+1}^I$ that is updated immediately before the target action. The DIL hidden state is input to f_n^I, which is a non-linear function (e.g., multilayer perceptron with ReLu activation) constructed for each domain, and the DIL predicts the target action from the output $f_n^I(\boldsymbol{h}_{t+1}^I)$. $Loss_2$ is calculated taking all the past actions into account. In addition, $Loss_2$ goes through only one RNN cell to reach the DIL user matrix. These resolve all issues with $Loss_1$.

Similar to $Loss_1$, we constructed $Loss_2$ based on our notion that the RNN reflects the process in which a mental state affects actions in individual domains. For $Loss_1$, this is achieved with link(A). On the other hand, for $Loss_2$, it is $f_n^I(\boldsymbol{h}_t^I)$ that is used for the next item prediction, and thus, user intentions for the next actions are now represented by $f_n^I(\boldsymbol{h}_t^I)$. For $Loss_2$, therefore, it is f_n^I that reflects the process instead of link(A). Note that the other aspects of our original design concept is not modified, i.e., the role of link (B) and how our RNN reflects the moderating effect of personality remain the same. We also left link (A) in the modified design because it enables the DSL to reflect not only action history but also context of past actions ("in what mental state a user took past actions") to its hidden state.

Loss Calculation: In both $Loss_1$ and $Loss_2$, the next item is predicted as follows:

$$\hat{\boldsymbol{y}}_t = \boldsymbol{W}_v^{n\top} \boldsymbol{h} + \boldsymbol{b}_v. \tag{4}$$

$\hat{\boldsymbol{y}}_t \in \mathbb{R}^{N_v^n}$ is a vector which has the same number of dimensions as the number of items in Domain n (N_v^n) and contains a prediction score of each item in a corresponding dimension. $\boldsymbol{h}$ takes different forms in $Loss_1$ and $Loss_2$: $\boldsymbol{h} = \boldsymbol{h}_{t'}^n$ in $Loss_1$ and $\boldsymbol{h} = f_n^I(\boldsymbol{h}_{t-1}^I)$ in $Loss_2$. $\boldsymbol{b}_v$ denotes a bias vector.

Using $\hat{\boldsymbol{y}}_t$, a DIL/DSL cell calculates WARP (Weighted Approximate Rank Pairwise) loss [35]. To calculate the loss, it randomly samples items in $\hat{\boldsymbol{y}}_t$ until it finds a negative sample with higher score than the positive sample (e.g., a movie ID that a user actually watched in t-th action). WARP loss is formulated as follows:

$$Loss_{\mathrm{WARP}} = \ln(\frac{N_v^n - 1}{N})(r^- - r^+), \tag{5}$$

where N is the number of items the RNN cell randomly sampled until it finds the first negative sample with higher score than the positive sample, i.e., the more it draws random samples, the less the loss is, which means the prediction is more accurate. r^- and r^+ denote the scores of the first negative sample and that of the positive sample, respectively.

IV. EXPERIMENT

We first learned the URs using our approach ($Loss_1$ and $Loss_2$) and the different baselines we selected, and then evaluated them to answer Q1∼Q3. The scripts used in the experiments will be made available at https://osf.io/tyv78/.

A. DATASET

We used two open datasets in our experiments: "Amazon Review Dataset (2018)" (Am) [19] and "RetailRocket" (RR) [20], both contain user-item interaction logs collected in real e-commerce portals. These datasets cover various kinds of items, e.g., items in Am range from clothing to grocery, electronics to outdoor goods. There are also various datasets publicly available today that contain user behavior including movie rating [36], music listening [37], and news reading [38]. However, they are "domain-specific" datasets, covering behavior only in limited domains of daily life and only domain-specific attributes can be learned from them (e.g., movie/music preference). In contrast, Am and RR provide hints for how users behave in various domains of daily life. For example, logs for clothing items indicate what kind of clothes they usually wear; grocery item logs indicate their eating habits; and electronics item logs reflect how they use IT. We posit that this would offer us more opportunity to learn domain-independent attributes than the domain-specific datasets.

Table 2 summarizes data used in the experiment. Am contains sequences of item review actions. Each log consists of user ID, item name, category, and genre, and review text. Out of 17 product genres in the original dataset, we selected six genres shown in the table based on the number of logs and regarded each genre as a domain. We then selected users who have more than 14 logs for each of the following four domains: 'C', 'E', 'H', and 'T' (we did not include 'G' and 'S' because doing so drastically decreased the number of users). In each domain, there are hundreds to thousands of item categories, which are represented by concatenation of several category labels (e.g., 'Men'+'Shoes'+'Athletic', 'Girls'+'Clothing'+'Dresses'). We assigned IDs to item categories (not to category labels) and used these category IDs to represent users' review actions rather than item IDs to suppress data sparsity.

RR contains sequences of item browsing and purchasing actions. Each log consists of user ID, item ID, category and genre, and action type (browse/purchase). There are 258 item genres, each of which has several dozen categories. Each genre has only a small number of logs, hence we merged the genres into four groups and regarded them as domains (i.e., G1∼G4). In the original dataset, the genres are represented by random numbers (e.g., 213, 169) so we could not merge them based on their relations but could only merge them so that the number of logs is balanced between the domains. Therefore, in RR, irrelevant item genres might have been merged into the same domain. Nevertheless, we used RR to examine how our approach performs in such a case assuming situations where logs are not labelled with domains properly. We selected users who had more than nine browse logs in each of the six combinations of two domains (C_2^4). As in Am, we used item category IDs to represent user actions in RR.

B. BASELINE

As baselines, we selected the approaches that satisfy the following two conditions because of their diversity in terms of domains for UR learning: (1) can learn URs from user actions represented by ID and (2) can learn URs without conducting special tasks that are only doable in limited domains. Based on these conditions, we excluded approaches for universal UR learning because they need user actions to be represented by texts (purchased/reviewed item names [17], [28], [29], item review texts [16]) or multiple special tasks such as "shop/price preference prediction" need to conduct [30].

From the sequential approaches, we selected SURNN [10] to validate the effectiveness of our multi-layer structure to learn domain-independent attributes. From the non-sequential approaches, we selected MF [8] and FM [9] for their simplicity and popularity and xDeepFM [11] for its superior performance in the category.

C. USER REPRESENTATION LEARNING

We first made combinations of multiple domains for learning URs. We expected that URs learned by our approach (DIL URs) would explicitly reflect domain-independent attributes that affect behavior in all the domains in a combination, whereas the baselines would jointly represent domain-independent and domain-specific attributes in the same URs. In Am, combinations of two, three, four, and five domains were made, i.e., C_2^6, C_3^6, C_4^6, and C_5^6 combinations (e.g., 'CE' for two domains, 'CEH' for three domains,…; 55 combinations in total[2]). In RR, we made C_2^4 combinations (e.g., 'G1G2'; six combinations in total). Then we learned the URs in each of all these domain combinations. The motivation was to 1) examine how our URs' performance is dependent on the domains in which they are learned, and 2) how their

[2]We did not make 'GS' for UR learning because their logs were insufficient, so we had $C_2^6 - 1 = 14$ combinations for two domains.

TABLE 2. Summary of the data used in the experiments.

Amazon (Am)			
# of users (N_u)	Domains	# of categories (N_v^n)	# of logs
691	Clothing, shoes, and jewelry (C)	4,625	26,978
	Electronics (E)	719	47,990
	Home and kitchen(H)	1,195	50,686
	Tools and home improvement (T)	736	21,133
	Grocery and gourmet food (G)	587	13,566
	Sports and outdoors (S)	1,038	15,581

RetailRocket (RR)			
# of users (N_u)	Domains	# of categories (N_v^n)	# of logs*
632	Group1 (G1)	238	43,515
	Group2 (G2)	248	30,685
	Group3 (G3)	222	34,076
	Group4 (G4)	200	36,752

* # of browse logs

performance changes as we increase the number of domains. For each domain combination, 80% of the logs were used for training and 20% for validation. Note that we represented item genres and categories by their IDs and did not use their text information for learning URs. In RR, only browse logs were used (purchase logs were not used for UR learning).

For our approach and SURNN (sequential approaches), we implemented their RNN cells via LSTM and trained the models by predicting item category ID to be reviewed (in Am)/browsed (in RR). For our approach, we used DIL URs for the subsequent evaluations. On the other hand, for MF, FM and xDeepFM (non-sequential approaches), we let them conduct prediction for each pair of a user ID and item category ID, i.e., predict whether the user reviews products in the category in Am and predict how many times the user browses the products in the category in RR. In the experiment, all the approaches learned URs whose number of dimensions was 32, 16, and 8. For other details of UR learning, refer to Appendix A.

D. EVALUATION OF USER REPRESENTATIONS

1) USER REPRESENTATION SIMILARITY (**Q1**)

We made pairs of the domain combinations and, for each pair, evaluated the similarity between URs of the same user. For example, when the pair is ['CE','HT'], we compared UR^i_{CE} and UR^i_{HT}, which denote user i's URs learned in 'CE' and 'HT', respectively. Because the URs learned in different domain combinations are in different latent spaces (e.g., the first dimension of UR_{CE} and UR_{HT} have different characteristics), we did not compare them directly but compared them after projection between the spaces. That is, we evaluated the similarity between $\boldsymbol{W}^{\mathrm{prj}}_{\mathrm{CE}\to\mathrm{HT}}UR^i_{\mathrm{CE}}$ and UR^i_{HT} and between $\boldsymbol{W}^{\mathrm{prj}}_{\mathrm{HT}\to\mathrm{CE}}UR^i_{\mathrm{HT}}$ and UR^i_{CE}, where $\boldsymbol{W}^{\mathrm{prj}}{}_{C_S\to C_T} \in \mathbb{R}^{d_{\mathrm{ue}}\times d_{\mathrm{ue}}}$ is the projection matrix from a source domain combination (C_S) to a target domain combination (C_T). This enabled us to compare the URs in the same space.

In the evaluation, we excluded the pairs that have domain(s) in common (e.g., ['CE', 'CH']) because user attributes specific to the common domain would make the URs similar regardless of their domain independence. After excluding such pairs, we examined all the possible pairs exhaustively including those in which the number of domains of C_S and C_T are different (e.g., ['CE','HTS'], ['CE','HTGS']).

We first randomly divided the users into five groups and used four of them to train $W^{\mathrm{prj}}_{C_S\to C_T}$ and the remaining group for testing. We repeated this five times by changing the test group (i.e., five-fold cross validation).

As the similarity metric, we used the mean reciprocal rank(MRR) based on Euclidean distance (MRR_{EUC}):

$$MRR_{\mathrm{EUC}}(C_S, C_T) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{rank_i}, \tag{6}$$

where $MRR_{\mathrm{EUC}}(C_S, C_T)$ denotes MRR_{EUC} when projecting UR_{C_S} to UR_{C_T}. We examined distances between $W^{\mathrm{prj}}_{C_S\to C_T}UR^i_{C_S}$ and all the URs learned in C_T and arranged the distances in ascending order. $rank_i$ denotes the rank of the distance between $W^{\mathrm{prj}}_{C_S\to C_T}UR^i_{C_S}$ and $UR^i_{C_T}$ (i.e., distance between user i's URs) in the test group. N denotes the number of users in the test group. The closer the same user's URs are, the higher $rank_i$ is, which makes MRR_{EUC} higher (better).

We used this metric instead of raw Euclidean distance so that we can compare the results between different pairs. Suppose we have a pair of two domains, pair A (e.g., ['CE','HT']), and that of three domains, pair B (e.g., ['CEG','HTS']). Distances between the URs in A and those in B are calculated in different spaces, and thus it is impossible to compare their distances. MRR_{EUC} enables us to make a comparison between A and B. If B's MRR_{EUC} is better than A, it means the URs in B have a higher degree of domain independence than A.

2) BEHAVIOR PREDICTION (**Q2**)

Typically, to predict user action, user features are extracted from logs in the same domain where the prediction is to be conducted (e.g., predict a song the user will listen to by using his music preference that is extracted from his listening history). In such a scenario, we posit that, if the URs learned in different domains reflect domain-independent attributes, they will improve prediction accuracy when used in addition to user features extracted from the target domain (we term such user features as domain-specific features, DFs). To examine this, we predicted user actions in a target domain using DFs and the URs learned in different domains and evaluated the prediction accuracy. Specifically, we predicted whether a user reviewed items with a specific category label in Am and whether a user purchased items of a specific genres in RR.

Note that we did not predict only from the URs because such prediction would not determine the URs' degree of domain-independence. For example, if we learn URs in "*Video games*(V)" and "*Software*(W)" and predict actions in "*Electronics*(E)", UR_{VW} might have high utility for predicting actions in 'E' even if UR_{VW} only reflects domain-specific attributes because of the relation between the three domains. UR_{VW} would reflect how users use IT, which would contribute to prediction in 'E'. Using DFs learned in 'E' in addition to UR_{VW} makes such domain-specific attributes (i.e., how users use IT) in UR_{VW} redundant because of the intersection between UR_{VW} and the DFs in 'E.' This prevents predictions using the URs that are not domain-independent from resulting in high accuracy.

At first, using SURNN, we learned the URs in the target domain and used them as DFs (number of DF dimensions was 16). Then, for each category label (Am) / item genre (RR) in the target domain that satisfies the criterion for the number of positive samples,[3] we built a prediction model by logistic regression (σ). For example, when the target domain is 'E' in Am, we used $UR_{\mathbb{C}\setminus\mathrm{E}}$, where $\mathbb{C} \setminus \mathrm{E}$ denotes any one of domain combinations that do not include 'E,' e.g., 'CH', 'CHT'. We performed the prediction as follows:

$$\hat{y}_{label=1} = \sigma(b_0 + \boldsymbol{b_1} DF_{\mathrm{E}} + \boldsymbol{b_2} UR_{\mathbb{C}\setminus\mathrm{E}}), \tag{7}$$

where $\hat{y}_{label=1}$ denotes the probability that a user reviews items with category label 1. b_0 is an intercept and $\boldsymbol{b_1}$ and $\boldsymbol{b_2}$ are vectors of partial coefficients. We conducted evaluation for all the domain combinations of $\mathbb{C}\setminus$E. We trained and tested this logistic regression model by five-fold cross validation and evaluated the models by ROC-AUC. We also evaluated the model that used only the DFs, $\hat{y}_{label=1} = \sigma(b_0 + \boldsymbol{b_1} DF_{\mathrm{E}})$, and compared it with the above models to determine how the URs improved prediction.

3) TEXT FEATURES PREDICTION (**Q3**)

Lastly, we evaluated the URs' degree of relevance to personalities. From item review texts, we first extracted features that are correlated with personality scores determined by the Big Five personality trait model [31], which is currently the most widely accepted personality model in scientific community. We then evaluated the accuracy of predicting the text features from the URs. This evaluation was conducted only in Am since item review texts are available only in Am.

Many researchers have reported personality affects the way people write texts (e.g., word usage in essays and tweets). Among such studies, we referred to literature that reported correlations between the specific text features and the Big Five scores [21], [22], [23], and extracted all the features that are significantly correlated with the Big Five (40 features in total) from all the review texts across 17 domains. To extract the features, we used LIWC [39], the text feature extraction tool that was used in [21], [22], and [23]. The features include general information about the text (e.g., word count) as well as frequency of word categories used in the text such as categories about psychological constructs (e.g., negative/positive affect), personal concern categories (e.g., work, home), and word class categories (e.g., articles, auxiliary verbs). Then we averaged the features per user and vectorized them, $\boldsymbol{P} \in \mathbb{R}^{40}$. Before predicting this, we reduced its dimensionality by principal component analysis because it was relatively large for the dataset size. Specifically, we made vectors consisting of the first $\sim$ k-th primary component scores of $\boldsymbol{P}$ ($\boldsymbol{P'} \in \mathbb{R}^k$). We regulated k by changing the threshold (Th) in the following equation: $\arg\min_k \sum_{i=1}^{k} loading_i \geq Th$, where $loading_i$ denotes i-th primary component's loading (we used $k = 21$ and 15 by setting $Th = 0.9$ and 0.8, respectively). We then predicted $\boldsymbol{P'}$ from the URs by conducting linear transformation, i.e., $\hat{\boldsymbol{P}}' = \boldsymbol{W^P UR}$ ($\boldsymbol{W^P} \in \mathbb{R}^{k \times d_{\mathrm{ue}}}$). We trained $\boldsymbol{W^P}$ and evaluated prediction accuracy by five-fold cross validation. We used Euclidean distance between $\hat{\boldsymbol{P}}'$ and $\boldsymbol{P'}$ as the accuracy metric.

V. RESULTS

In this section, we describe the results for URs with 16 dimensions, in which our URs performed best. For the results of the URs with 8 and 32 dimensions that we also tested, refer to Appendix B. Appendix C details how we tested statistical significance for the results in this section.

A. USER REPRESENTATION SIMILARITY (Q1)

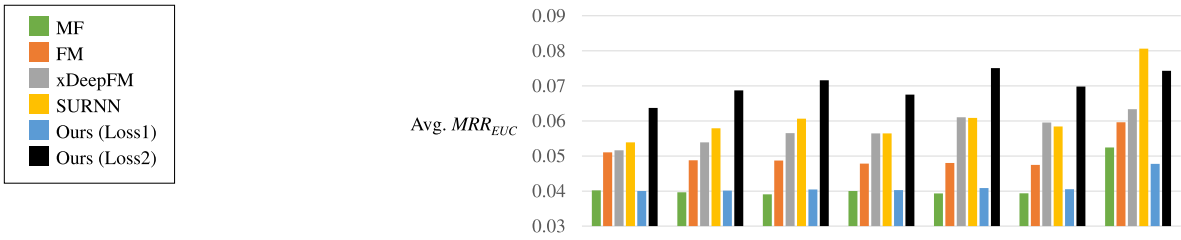
Fig. 3 shows the results.[4] In Am, URs learned by our approach ($Loss_2$) resulted in the highest MRR_{EUC} average in all the conditions. As shown in 3) in the figure, the difference from the baselines are statistically significant in all the conditions in Am. These results confirm that URs of the same user are most similar when learned by our approach and suggest that our URs would have the highest degree of domain-independence among all the evaluated approaches.

The results also indicate our approach learns URs with the highest degree of domain independence in most of the domain combinations. As shown in 1) in the figure, out of 238 pairs in total, our URs ($Loss_2$) performed the best in 212 pairs, out of which the differences from the baselines are significant in 193 pairs as shown in 2). Comparing $Loss_1$ and $Loss_2$, the latter, which we designed to resolve the issues of the former, outperformed the former as we expected.

Another notable result in Am is the relations between MRR_{EUC} average and the number of domains in source and target domain combinations (C_S and C_T). In conditions 1$\sim$3 in the graph (two domains were used in C_S), MRR_{EUC} of ours ($Loss_2$) increases as the number of domains in C_T increases. Results of conditions 1, 4, and 6 (two domains were used in C_T) also show that it increases as the number of domains in C_S increases. These results suggest that the more we

[3] $0.05N_u \leq n \leq 0.5N_u$; n is the number of positive samples (the number of users who actually reviewed/purchased the items).

[4] We show in the graph the averaged results for readability. For the full set of raw results, refer to https://osf.io/tyv78/



Condition no.		1	2	3	4	5	6	7
Dataset		Am	Am	Am	Am	Am	Am	RR
# of domains in a source domain combination (C_S)		2	2	2	3	3	4	2
# of domains in a target domain combinations (C_T)		2	3	4	2	3	2	2
# of source and target pairs		78	56	14	56	20	14	6
1)	# of "best pairs"	67	54	13	48	19	11	1
2)	# of significant "best pairs"	56	54	13	43	18	9	0
3)	Overall significance	***	***	***	***	***	***	--

FIGURE 3. Q1 - UR similarity test results (the higher, the better). Each group of bars shows MRR_{EUC} average in the same condition (e.g., bars in condition 1 shows the average of 78MRR_{EUC} values). In the table, 1) shows # of pairs for which MRR_{EUC} of ours ($Loss_2$) was the highest of all the approaches; 2) shows # of such pairs for which MRR_{EUC} of ours ($Loss_2$) was significantly higher than all the baselines ($p < .05$). 3) shows whether MRR_{EUC} average across all the pairs is significantly better in ours ($Loss_2$) compared to all the baselines (* $p < .01$).**

add domains, the more our approach learns about domain-independent attributes.

In contrast to the results in Am, our approach did not outperform SURNN in RR. We consider this is because we merged multiple item genres into a group without considering their relations and put them together into the same DSL. In such cases, the DSL cannot learn domain-specific user attributes, which makes it impossible for our approach to distinguish between domain-specific and domain-independent user attributes.

B. BEHAVIOR PREDICTION (Q2)

Fig. 4 shows the results[4]. In Am, ROC-AUC of our approach ($Loss_2$) improves as the number of domains increases. As the graphs shows, when using more than two domains to learn URs, our URs ($Loss_2$) resulted to the highest average ROC-AUC among all the approaches including prediction solely from the domain features (DFs) without using the URs (shown by the dashed lines). As shown in 1) in the figure, our URs ($Loss_2$) performed best for all the domain combinations whose number of domains are more than two except for two combinations in condition 6. Furthermore, as shown in 2) in the table, when the number of domains is more than three, their superiority is statistically significant for all the domain combinations except for one combination in condition 7. These results are in line with the observations that we made in the results for Q1. That is, our URs have a higher degree of domain-independence than the baseline URs and increasing the domains for UR learning enables our approach to learn more about domain-independent attributes; and our approach learned the best URs from most of the domain combinations whose number of domains is more than two. $Loss_2$ also outperformed $Loss_1$ as in the results for Q1.

However, it should also be noted that while ours ($Loss_2$) improves as the number of domains increases, the degree of improvement decreases. We discuss the implications of this finding in the next section.

Looking at the baseline URs, their ROC-AUC are significantly lower than the predictions by DFs or almost the same as the predictions by DFs (except for SURNN's URs in conditions 3, 4, and 12). This supports our speculation described in Section II that domain-specific attributes in their URs have little utility for predicting behavior in the other domains where the URs are not learned and thus undermine the task performance in such domains.

In RR, our approach (both $Loss_1$ and $Loss_2$) did not outperform the baselines as in Q1.

C. TEXT FEATURES PREDICTION (Q3)

Fig. 5 shows the result[4]. For both $k = 21$ and 15, when the number of domains was less than five, the URs learned by our approach ($Loss_2$) achieved significantly shorter average distance, i.e., higher accuracy of predicting the text features that are significantly correlated with the Big Five, than all the baselines as shown in 3) in the figure. Out of 49 domain combinations whose number of domains is less than five, our URs ($Loss_2$) performed best in 43 and 39 combinations when $k = 21$ and 15, respectively, as shown in 1). The results indicate it is highly likely that our URs have a higher degree of relevance to the Big Five personality traits than the baseline URs.

However, while the prediction accuracy improves as the number of domains increases, the degree of improvement decreases as we observed in the results for Q2. When learning the URs in five domains, the superiority of our approach ($Loss_2$) to SURNN diminishes. As in the results for Q1 and Q2, $Loss_2$ outperformed $Loss_1$ in this evaluation as well.

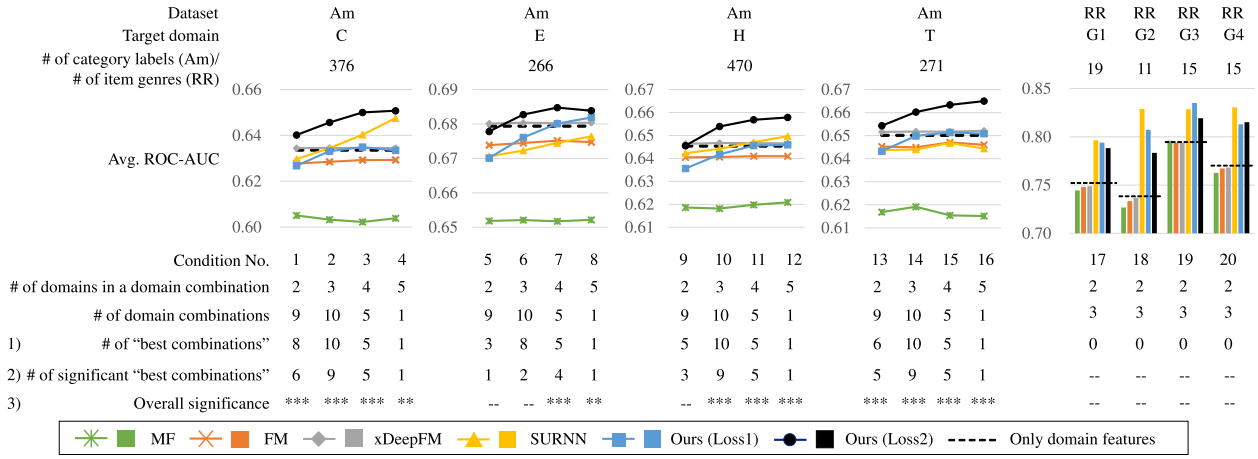


FIGURE 4. **Q2 - Behavior prediction results** (the higher, the better). Each point/bar shows the average of # *of category labels* (Am) / # *of item genres* (RR) × # *of domain combinations* results (e.g., each point in condition 1 shows the average of 376 × 9 = 3, 384 results of ROC-AUC). Dashed lines show the results of predictions only by the domain features (DFs). In the table, 1) shows # of domain combinations in which ours ($Loss_2$) learned the best URs in terms of average ROC-AUC; 2) shows # of such combinations for which we confirmed statistical significance ($p < .05$); and 3) shows whether overall ROC-AUC average across all the domain combinations is significantly higher in ours ($Loss_2$) compared to all the baselines (** $p < .05$, *** $p < .01$).

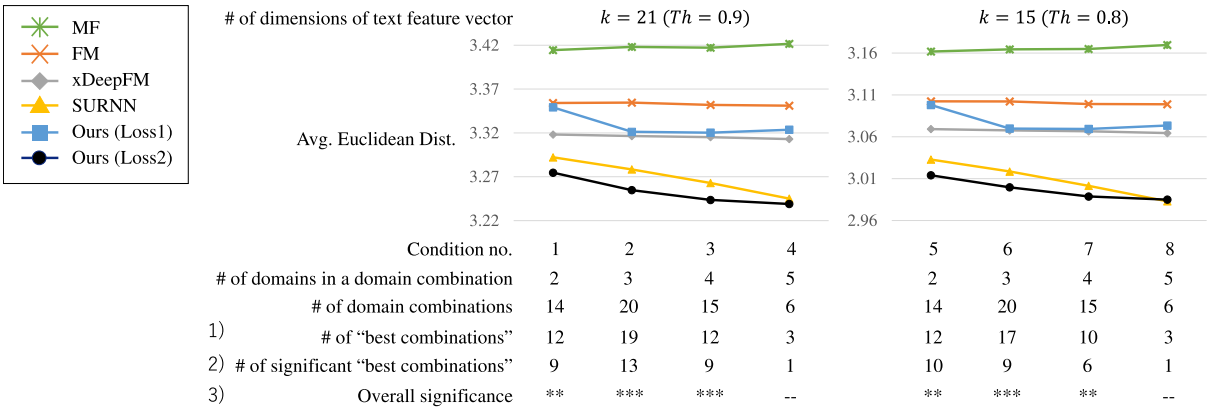


FIGURE 5. **Q3 - Text features prediction results** (the lower, the better). Each point shows the average Euclidean distance between the text feature vectors (P') and predicted vectors ($W^P UR$), e.g., each point of condition 1 shows the average of 14 × 691 (691 is # of users) Euclidean distances. In the bottom table, 1) shows # of domain combinations for which our URs ($Loss_2$) resulted in the shortest average distance; 2) shows the number of "best combinations" for which the average distance of ours ($Loss_2$) is significantly shorter than all the baselines ($p < .05$); and 3) shows whether overall average of Euclidean distances across all the domain combinations is significantly shorter in ours ($Loss_2$) compared to all the baselines (** $p < .05$, *** $p < .01$).

VI. DISCUSSION

The evaluation results for Q1 and Q2 in Am confirmed that compared to the baselines, our approach can learn URs with a higher degree of domain-independence. As shown in the evaluation for Q2, using our URs improves task performance in domains where the URs are not learned. This is especially beneficial for e-commerce portals, in which a user has not necessarily interacted with items of target domains before performing the tasks or only a limited number of logs are available for a user in the target domains since the portals deals in goods across a diverse range of domains. The portals can improve the task performance in such a domain by using our URs that are already learned in other domains

In the evaluation for Q3, the results indicate our URs also have a higher degree of relevance to personality, which is one of domain-independent attributes that has attracted much research attention as a basis for personalizing services including e-commerce services [40], [41], [42], [43]. If our URs actually contain personality information (though further study is necessary to confirm this as discussed later), it would potentially address two key issues of existing methods to determine personality: (1) the limited amount of information and (2) compromised reliability, which are described in more detail in the following.

It has been a common practice to determine human personality based on "trait theory" in psychology. It regards personality as consisting of several traits (e.g., *Extraversion*,

Neuroticism, *Agreeableness*, *Conscientiousness*, and *Openness* in the Big Five traits model [31]) and measures the score of each trait by responding to a questionnaire. While the measurement results are highly interpretable, they only provide a limited amount of information due to low dimensionality, i.e., personality is represented by a small number of traits, and coarse score granularity, i.e., scores are discrete rather than continuous because they are calculated by summing up answers to an X-point Likert or binary scale. In addition, people sometimes provide answers to questionnaires that are biased or not well thought out, which compromises the reliability of the measurements. While there have been many studies that automatically determine personality from daily behavior (e.g., [44], [45]), all of them employ supervised approaches, i.e., they still use questionnaire measurements as ground truths and therefore still subject themselves to (1) and (2).

In contrast to these methods, our approach can learn high dimensional representation of user attributes from sequences of user-item interactions without using questionnaires. Therefore, we deem it has potential to be a solution for (1) and (2) and the present study has made a significant step to achieving such a solution.

Limitation and Future Direction: There are several limitations in our approach that should be addressed by future research. One is about constructing the inputs to the DSLs. As indicated by the results in RR, where our approach underperformed the baselines, our approach cannot work as expected if the data are not properly labelled with domains. It is costly and not always feasible to label the data by human annotators. A functionality to automatically label data should be studied in the future.

Another limitation is that the degree of performance improvement of our URs decreases as the number of domains increases. We speculate this could have been caused by the overlap between the domain-independent attributes learned from a group of domains and those that can be learned by adding a new domain to the group. As the number of domains increases, domain-independent attributes are increasingly reflected in the URs, which we observed in the evaluations for Q1 and Q2. If much has already been learned, little could be learned further by adding a new domain. At the same time, adding a domain increases the number of parameters since it necessitates adding a DSL, which equates to learning more RNN cell weights and biases. In such manner, the benefit-cost balance deteriorates as the number of domains increases. One solution for this might be to put knowledge on multiple ''related'' domains into a single DSL. The challenge is how to automatically determine the relation between domains. This should be investigated in future.

Lastly, in future, our URs need to be compared with ground truths of long-term and domain-independent attributes to determine what specific attributes are reflected to what degree. This is important because our URs would reflect multiple attributes, some of which might be provided by the users themselves (e.g., when a user signs up for a service). In such cases, it is redundant to have such information in the URs and it is enough just to use the information provided by the users instead of the URs. Therefore, it is necessary to determine attributes reflected in the URs and then our approach needs to be extended so as to learn distinctive URs for each of the specific attributes. This would also improve the URs' interpretability and transparency in how they work for personalizing services.

VII. CONCLUSION

In this paper, we proposed an approach to learn URs that account for long-term and domain-independent attributes from sequences of user actions without using ground truths of the attributes. Using actual item review and browse logs in e-commerce portals, we confirmed that the URs learned by our approach have a higher degree of domain-independence than existing approaches, demonstrating adaptability to various domains. We also confirmed the possibility that our URs reflect the Big Five personality traits to a greater extent.

APPENDIX A IMPLEMENTATION OF USER REPRESENTATION LEARNING

In this section, we describe how we implemented the proposed and baseline approaches and learned the user representations (URs). We learned the URs whose number of dimensions (d^*_{ue}) was 32, 16, and 8.

A. SEQUENTIAL APPROACHES

For our approach and SURNN, we formatted the input action data as $(\boldsymbol{i_u}, \boldsymbol{i_{v,t}})$, where $\boldsymbol{i_{v,t}}$ was a one-hot vector of item category ID that a user reviewed (in Am)/browsed (in RR), and let their models predict the item category to be reviewed/browsed. In the description, we use the same notations in the main manuscript unless otherwise noted.

We set the size (w) and slide interval (s) of sliding windows to 15 and five, respectively. We optimized the loss function by the Adam optimizer with a learning rate of 0.005 and a batch size of 96 and stopped the training when the loss converges on the validation data. The dimension of the item representation and hidden state of the DSL, DIL and SRUNN was set to the same size as the UR (i.e., $d^*_{\mathrm{ve}}, d^*_h = d^*_{\mathrm{ue}}$). f^I_n of our approach ($Loss_2$) was implemented as a two-layer perceptron with ReLU as the activation function in which the first and second layers had $2 \times d^I_h$ and $2 \times d^n_{\mathrm{ve}}$ perceptrons, respectively.

We implemented RNN cells of DSL (f^n_{RNN}), DIL (f^I_{RNN}), and SURNN via LSTM, in which the hidden state ($\boldsymbol{h_t}$) is updated by the input, forget, and output gates ($\boldsymbol{I_t}, \boldsymbol{F_t}, \boldsymbol{O_t} \in \mathbb{R}^{d_h}$) and candidate and present memories ($\widetilde{\boldsymbol{C}}_t, \boldsymbol{C_t}, \in \mathbb{R}^{d_h}$). They are formulated as follows:

$$\boldsymbol{I_t} = \sigma(\boldsymbol{W_{ei}e} + \boldsymbol{W_{hi}h} + \boldsymbol{b_i}), \tag{8}$$

$$\boldsymbol{F_t} = \sigma(\boldsymbol{W_{ef}e} + \boldsymbol{W_{hf}h} + \boldsymbol{b_f}), \tag{9}$$

$$\boldsymbol{O_t} = \sigma(\boldsymbol{W_{eo}e} + \boldsymbol{W_{ho}h} + \boldsymbol{b_o}), \tag{10}$$

$$\widetilde{\boldsymbol{C}}_t = \tanh(\boldsymbol{W_{ec}e} + \boldsymbol{W_{hc}h} + \boldsymbol{b_c}), \tag{11}$$

TABLE 3. Input to an RNN cell of DSL, DIL, and SURNN (notations in parentheses denote the number of dimensions).

	$\boldsymbol{e}$ (d_e)	$\boldsymbol{h}$ (d_h)
DSL	$\boldsymbol{e}^n_{\boldsymbol{u}} \oplus \boldsymbol{e}^n_{\boldsymbol{v},t}$ $(d^n_{\text{ue}} + d^n_{\text{ve}})$	$\boldsymbol{h}^n_{t'} + \boldsymbol{W}^n_I \boldsymbol{h}^I_{t-1}$ (d^n_h)
DIL	$\boldsymbol{e}^I_{\boldsymbol{u}}$ (d^I_{ue})	$\boldsymbol{h}^I_{t-1} + \boldsymbol{W}^I_n \boldsymbol{h}^n_t$ (d^I_h)
SURNN	$\boldsymbol{e}^S_{\boldsymbol{u}} \oplus \boldsymbol{e}^S_{\boldsymbol{v},t}$ $(d^S_{\text{ue}} + d^S_{\text{ve}})$	$\boldsymbol{h}^S_{t-1}$ (d^S_h)

$$\boldsymbol{C_t} = \boldsymbol{F_t} \odot \boldsymbol{C_{t-1}} + \boldsymbol{I_t} \odot \widetilde{\boldsymbol{C}}_t, \tag{12}$$

$$\boldsymbol{h_t} = \boldsymbol{O_t} \odot \tanh(\boldsymbol{C_t}). \tag{13}$$

σ and $\odot$ denote sigmoid function and element-wise product, respectively. $\boldsymbol{e} \in \mathbb{R}^{d_e}$ and $\boldsymbol{h} \in \mathbb{R}^{d_h}$ are input to an RNN cell and $\boldsymbol{W_{e*}} \in \mathbb{R}^{d_h \times d_e}$, $\boldsymbol{W_{h*}} \in \mathbb{R}^{d_h \times d_h}$, and $\boldsymbol{b_*} \in \mathbb{R}^{d_h}$ are its learnable parameters. While the above formulae are common to the DSL, DIL, and SURNN, $\boldsymbol{e}$ and $\boldsymbol{h}$ take different forms between them as shown in Table 3.

As for SURNN, $\boldsymbol{e}^S_{\boldsymbol{u}} \in \mathbb{R}^{d^S_{\text{ue}}}$ and $\boldsymbol{e}^S_{\boldsymbol{v},t} \in \mathbb{R}^{d^S_{\text{ve}}}$ are a UR and item representation in SURNN and are retrieved from its user matrix $\boldsymbol{W}^S_{\boldsymbol{u}} \in \mathbb{R}^{d^S_{\text{ue}} \times N_u}$ and item matrix $\boldsymbol{W}^S_{\boldsymbol{v}} \in \mathbb{R}^{d^S_{\text{ve}} \times N_v}$ (N_v denotes the total number of item categories across all the domains), respectively, i.e., $\boldsymbol{e}^S_{\boldsymbol{u}} = \boldsymbol{W}^S_{\boldsymbol{u}} \boldsymbol{i_u}$ and $\boldsymbol{e}^S_{\boldsymbol{v},t} = \boldsymbol{W}^S_{\boldsymbol{v}} \boldsymbol{i_{v,t}}$. In the following, we describe how we trained the model in SURNN. For the proposed approach, refer to III-B in the main manuscript.

Model Training in SURNN: As in the proposed approach, the model predicted as follows:

$$\hat{\boldsymbol{y}}_t = \boldsymbol{W}^{S\top}_{\boldsymbol{v}} \boldsymbol{h}^S_{\boldsymbol{t-1}} + \boldsymbol{b_v}, \tag{14}$$

where $\hat{\boldsymbol{y}}_t \in \mathbb{R}^{N_v}$ contains a prediction score of each item category in a corresponding dimension. The loss was also calculated in the same way as in the proposed approach by WARP loss.

We retrieved $\boldsymbol{W}^S_{\boldsymbol{u}}$ and used it as the URs in the UR evaluations.

B. NON-SEQUENTIAL APPROACHES

For Matrix Factorization (MF), Factorization Machine (FM) and xDeepFM, we let them conduct prediction for each pair of a user ID and item category. In the following, we use $\hat{\boldsymbol{y}}_{a,b}$ to denote a prediction result for user a and category b. In Am, the models predicted whether the user reviews the item category (i.e., $\hat{\boldsymbol{y}_{a,b}} \in [0 \dots 1]$) because a user reviews the item category only once in almost all the cases in our data. On the other hand, in RR, they predicted how many times a user browses the category (i.e., $\hat{\boldsymbol{y}_{a,b}} \in \mathbb{R}$) because most users review the same category multiple times.

In all the approaches, we used the Binary Cross Entropy in Am and the Root Mean Square Error in RR as loss functions because they are widely used in binary classification and regression tasks, respectively.

1) MATRIX FACTORIZATION (MF)

In Am, we used Logistic MF [46] and predicted as follows:

$$\hat{y}_{a,b} = \sigma(\boldsymbol{e_u}^\top \boldsymbol{e_v} + b_u + b_v), \tag{15}$$

where $\boldsymbol{e_u} \in \mathbb{R}^{d_{\text{ue}}}$, $\boldsymbol{e_v} \in \mathbb{R}^{d_{\text{ve}}}$, b_u and b_v denote a UR, item representation, and user and item biases, respectively ($d_{\text{ue}} = d_{\text{ve}}$).

In RR, we used normal MF [8] and predicted as follows:

$$\hat{\boldsymbol{Y}} = \boldsymbol{W_u}^\top \boldsymbol{W_v}, \tag{16}$$

where $\hat{\boldsymbol{Y}} \in \mathbb{R}^{N_u \times N_v}$ contains $\hat{y}_{a,b}$ at a-th row and b-th column. $\boldsymbol{W_u} \in \mathbb{R}^{d_{\text{ue}} \times N_u}$ and $\boldsymbol{W_v} \in \mathbb{R}^{d_{\text{ve}} \times N_v}$ are matrices whose columns correspond to $\boldsymbol{e_u}$ and $\boldsymbol{e_v}$, respectively.

2) FACTORIZATION MACHINE (FM)

The model took $\boldsymbol{x} \in \mathbb{R}^d$ as input, which is a concatenation of one-hot vectors of a user ID, item category, and domain (i.e., $d = N_u + N_v + N_g$, where N_g denotes the number of domains). The prediction is formulated as follows:

$$\text{in Am, } \hat{y}_{a,b} = \sigma(f_{\text{FM}}(\boldsymbol{x})) \text{ and,} \tag{17}$$

$$\text{in RR, } \hat{y}_{a,b} = f_{\text{FM}}(\boldsymbol{x}), \text{ where} \tag{18}$$

$$f_{\text{FM}}(\boldsymbol{x}) = w_0 + \sum_{i=1}^{d} w_1 x_i + \sum_{i=1}^{d} \sum_{j=i+1}^{d} \boldsymbol{g_i}^\top \boldsymbol{g_j} x_i x_j. \tag{19}$$

x_i is the i-th element of $\boldsymbol{x}$. w_0, w_1 and $\boldsymbol{g_i} \in \mathbb{R}^{d_{\text{ue}}}$ are learnable parameters. We used $\boldsymbol{g_i}$ that corresponds to a one-hot vector of a user ID as a UR.

3) xDeepFM

As in FM, the model took $\boldsymbol{x} \in \mathbb{R}^d$ as input and predicted as follows:

$$\text{in Am, } \hat{y}_{a,b} = \sigma(f_{\text{xDFM}}(\boldsymbol{x})) \text{ and,} \tag{20}$$

$$\text{in RR, } \hat{y}_{a,b} = f_{\text{xDFM}}(\boldsymbol{x}), \text{ where} \tag{21}$$

$$f_{\text{xDFM}}(\boldsymbol{x}) = \boldsymbol{W}_{\text{linear}} \boldsymbol{x} + \boldsymbol{W}_{\text{DNN}} f_{\text{DNN}}(\boldsymbol{e_u}, \boldsymbol{e_v}, \boldsymbol{e_g}) + \boldsymbol{W}_{\text{CIN}} f_{\text{CIN}}(\boldsymbol{e_u}, \boldsymbol{e_v}, \boldsymbol{e_g}) + b. \tag{22}$$

$\boldsymbol{e_u}$, $\boldsymbol{e_v}$, and $\boldsymbol{e_g} \in \mathbb{R}^{d_{\text{ge}}}$ are a UR and item and domain representations, respectively ($d_{\text{ve}}, d_{\text{ge}} = d_{\text{ue}}$). They were extracted in the embedding layer of xDeepFM from their one hot vectors. We used $\boldsymbol{e_u}$ as a UR. $\boldsymbol{W_*}$ and b are learnable parameters and f_{DNN} and f_{CIN} represent calculation in the plain deep neural network (DNN) and compressed interaction network (CIN), respectively. For detail of f_{DNN} and f_{CIN}, we refer to [11]. We set the batch size, the number of DNN hidden units and CIN layer size to 128, (256, 128, 64), and (128, 128), respectively.

APPENDIX B RESULTS IN DIFFERENT d_{ue} SETTINGS

In this section, we describe the evaluation results of URs whose number of dimensions (d^*_{ue}) is 32 and 8. We conducted this evaluation only in Am because we confirmed our approach did not work as expected in RR as described in

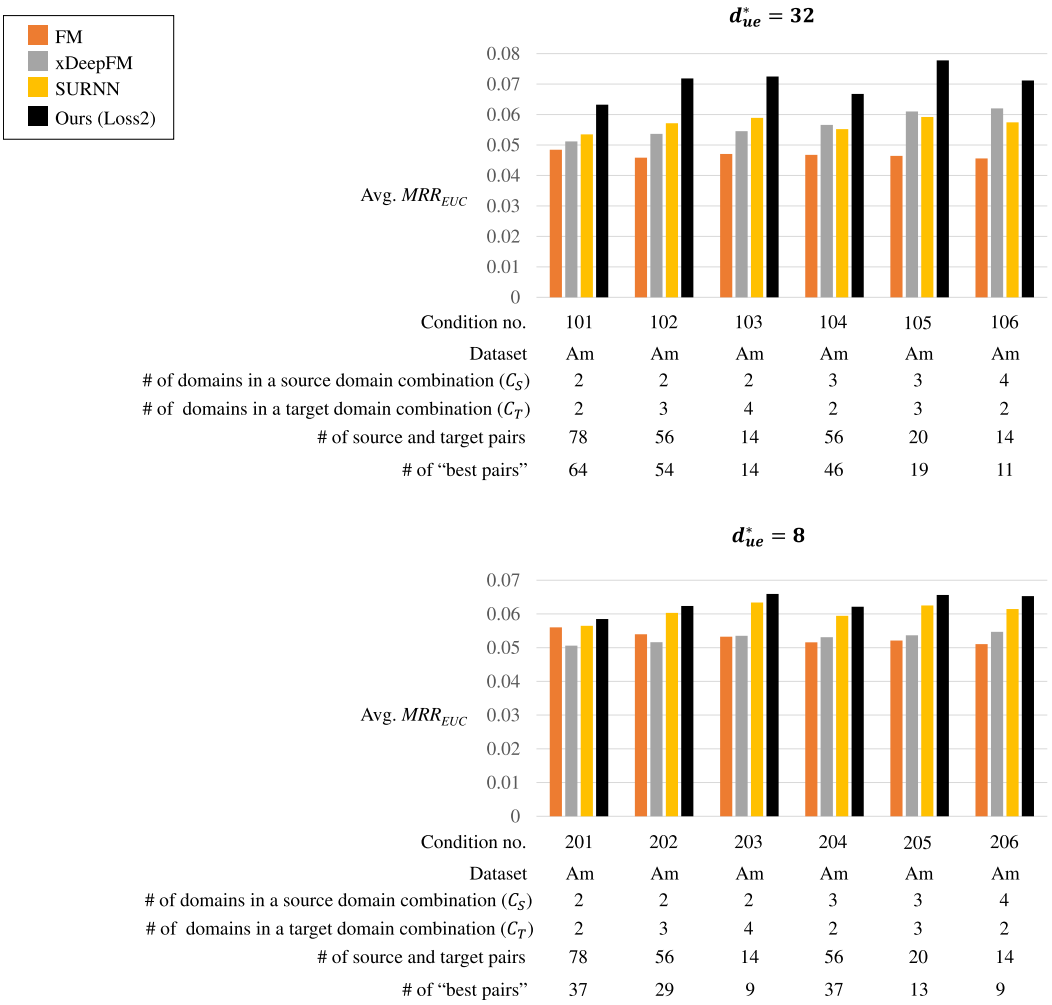


FIGURE 6. Q1 - UR similarity test results (the higher, the better). Each group of bars shows MRR_{EUC} average in the same condition (e.g., bars in condition 101 shows the average of 78MRR_{EUC} values). In the table, "best pair" means a pair for which MRR_{EUC} of ours ($Loss_2$) was the highest of all the approaches.

the main manuscript. Also note that we did not evaluate MF because its performance was significantly inferior to other approaches when $d^*_{ue} = 16$. We also omitted evaluation of our approach ($Loss_1$) because its performance was consistently inferior to $Loss_2$ when $d^*_{ue} = 16$. Evaluation results for Q1∼Q3 are shown in Figs. 6∼8, respectively.

APPENDIX C STATISTICAL TEST

In this section, we describe how we tested statistical significance of differences between our approach ($Loss_2$) and the baselines.

A. USER REPRESENTATION SIMILARITY TEST

We tested statistical significance from two perspectives: A) for each pair of the domain combinations and B) for each condition (we had conditions 1∼7 as shown in Fig. 3 in the main manuscript).

A) We compared $rank_i$ values between our approach and a baseline in the same pair of domain combinations (e.g., ['CE', 'HT']). For each of N_u users, we had a pair of $rank_i$ values; one is a result of our approach and the other is a result of the baseline. In total, there were N_u pairs. For these pairs, we conducted the Wilcoxon signed-rank (WSR) test, which is a test for paired-samples of nonparametric data. It examines whether distribution of two groups (in our case, $rank_i$ values of our approach and those of the baseline) are significantly different. We concluded that MRR_{EUC} is significantly better than the baseline if distribution of $rank_i$ in our approach is significantly higher than the baseline ($p < .05$). We conducted this comparison with all the baselines and, if the results were $p < .05$ for all of them, we concluded that MRR_{EUC} of our approach was significantly better than all the baselines for this pair of domain combinations. "2) # of significant best pairs" in Fig. 3 shows the number of such pairs of domain combinations.

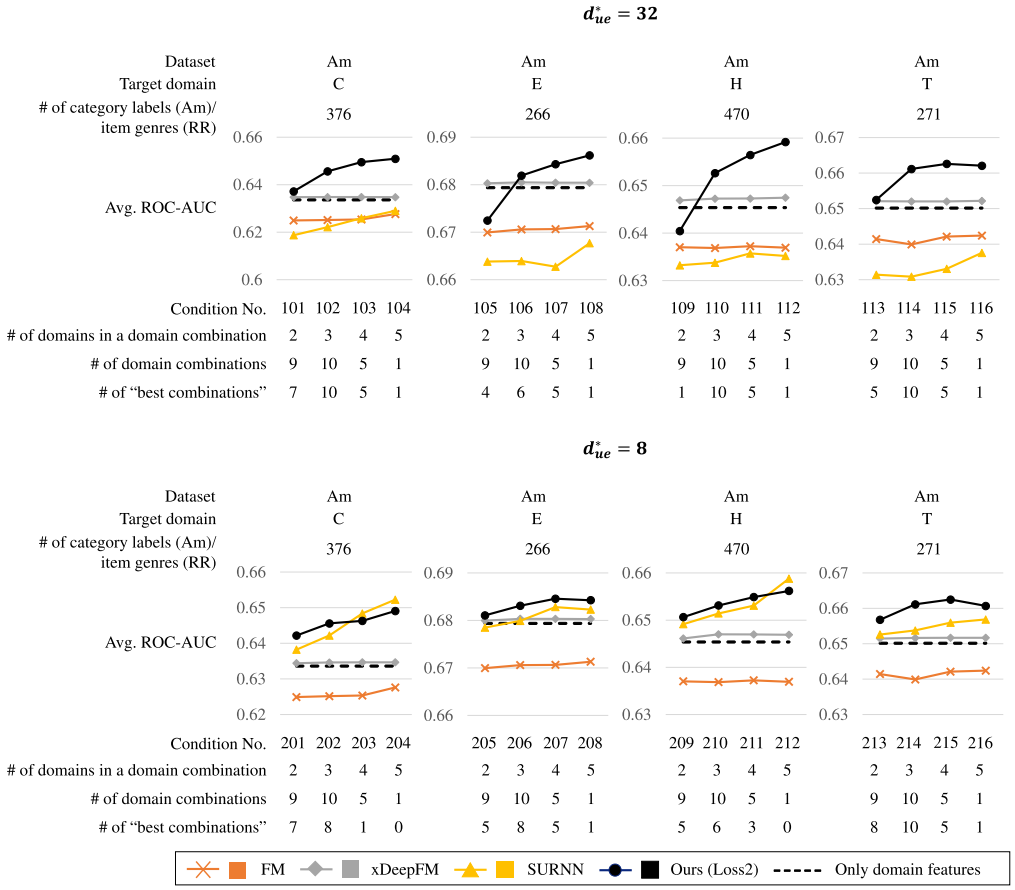


FIGURE 7. Q2 - Behavior prediction results (the higher, the better). Each point/bar shows the average of # *of category labels* (Am) / # *of product genres* (RR) × # *of domain combinations* results (e.g., each point in condition 1 shows the average of 376 × 9 = 3,384 results of ROC-AUC). Dashed lines show the results of predictions only by the domain features. In the table, "best combination" means a domain combination in which ours ($Loss_2$) learned the best URs in terms of average ROC-AUC.

B) We compared MRR_{EUC} values within the same condition. For example, in condition 1 in Fig. 3, we had 78 MRR_{EUC} values for each approach (i.e., 78 pairs when we compare our approach and a baseline). We conducted the WSR test for these pairs. As in A), this comparison was conducted with all the baselines and we examined whether the MRR_{EUC} values of our approach ($Loss_2$) were significantly higher than all the baselines. "3) Overall significance" in Fig. 3 shows the maximum p value of the WSR tests.

B. BEHAVIOR PREDICTION

We examined statistical significance for A) each domain combination and B) each condition.

A) For example, in condition 1 in Fig. 4, we had 376 results of ROC-AUC (376 is the number of category labels in 'C') per approach for each domain combinations. We conducted the WSR test for these results. If the distribution of ROC-AUC was significantly higher in our approach ($Loss_2$) than all the baselines ($p < .05$), we concluded that our approach learned the best URs in this domain combination. "2) # of significant best combinations" in Fig. 4 shows the number of such domain combinations.

B) We examined the significance for all the results in the condition. For example, in condition 1 in Fig. 4, we obtained 376 × 9 results of ROC-AUC for each approach. We conducted the WSR test for these results to see if the distribution of ROC-AUC of our approach is significantly higher than the baselines. "3) Overall significance" in Fig. 4 shows the maximum p value of the WSR test results. Note that, we used raw ROC-AUC values rather than the averages within the same domain combination.

C. TEXT FEATURE PREDICTION

As we did in VII-B, we examined statistical significance for A) each domain combination and B) each condition.

A) We had N_u Euclidean distances per approach for each domain combination. We conducted the WSR test for N_u pairs of Euclidean distances between our approach and each baseline. This comparison was

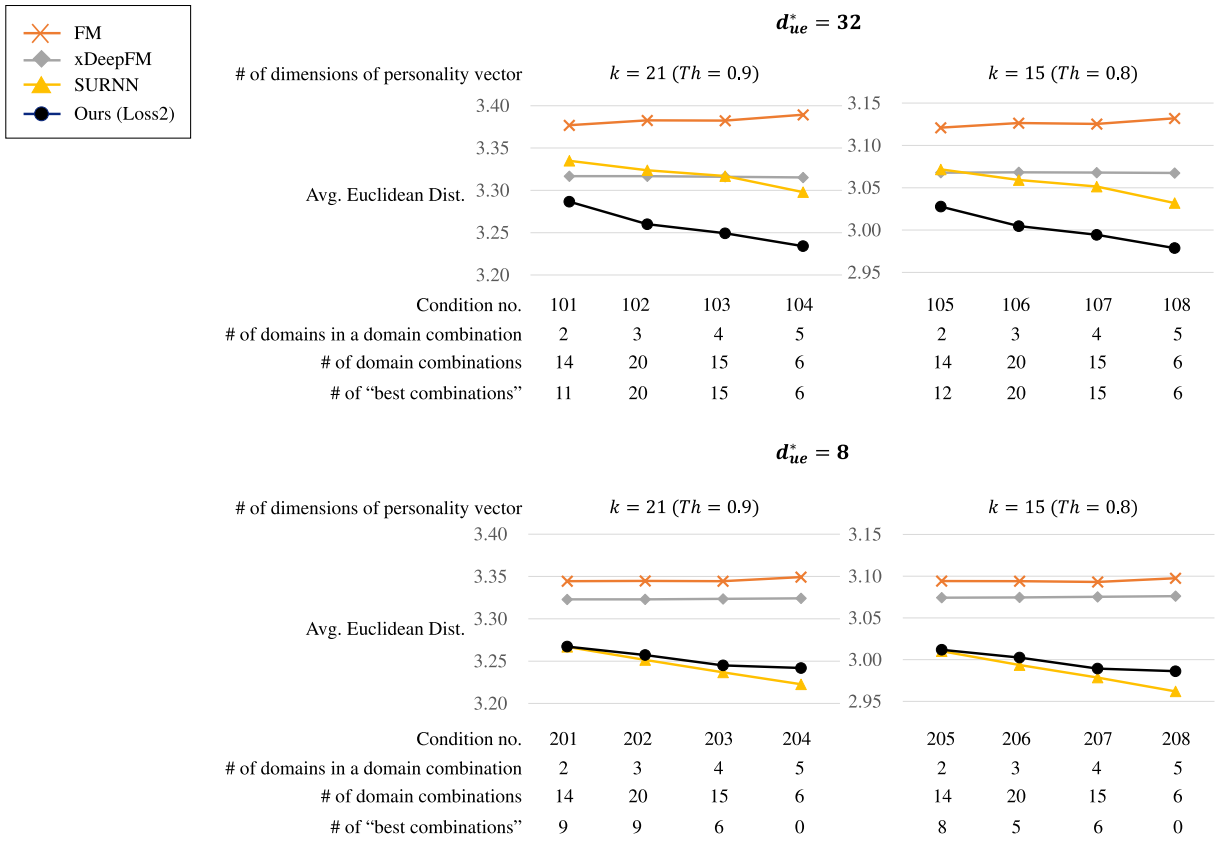


FIGURE 8. **Q3 - Text features prediction results (the lower, the better). Each point shows the average Euclidean distance between the personality vectors (P') and predicted vectors ($W^P UR$), e.g., each point of condition 1 shows the average of 14 × 691 (691 is # of users) Euclidean distances. In the bottom table, "best combinations" means a domain combination for which URs learned by ours ($Loss_2$) resulted in the shortest average distance.**

conducted by the WSR test for all the baselines, "2) # of best combinations" in Fig. 5 shows the domain combinations for which the Euclidean distances of our approach are significantly shorter than all the baselines ($p < .05$).

B) We first averaged Euclidean distances within the same domain combination for each approach. For example, in condition 1 in Fig. 5, there were 14 domain combinations. Hence, we had 14 averages of Euclidean distances for each approach. We conducted the WSR test for pairs of these averaged distances between our approach ($Loss_2$) and each baseline. "3) Overall significance" in Fig. 5 shows the maximum p value of the WSR test results.

REFERENCES

[1] (Apr. 2022). *COVID-19 Boost to e-Commerce Sustained into 2021, New Unctad Figures Show*. [Online]. Available: https://unctad.org/news/covid-19-boost-e-commerce-sustained-2021-new-un%ctad-figures-show

[2] R. Shepherd and P. Sparks, "Modelling food choice," in *Measurement of Food Preferences*, H. J. H. MacFie and D. M. H. Thomson, Eds. Boston, MA, USA: Springer, 1994, pp. 202–226, doi: 10.1007/978-1-4615-2171-6_8.

[3] R. C. Mulyanegara, Y. Tsarenko, and A. Anderson, "The big five and brand personality: Investigating the impact of consumer personality on preferences towards particular brand personality," *J. Brand Manage.*, vol. 16, no. 4, pp. 234–247, Jan. 2009.

[4] C. Yang, S. Pan, J. Mahmud, H. Yang, and P. Srinivasan, "Using personal traits for brand preference prediction," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 86–96.

[5] R. Hu and P. Pu, "Enhancing collaborative filtering systems with personality information," in *Proc. 5th ACM Conf. Recommender Syst. (RecSys)*, New York, NY, USA, 2011, pp. 197–204.

[6] J. Golbeck and E. Norris, "Personality, movie preferences, and recommendations," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining*, New York, NY, USA, Aug. 2013, pp. 1414–1415.

[7] S. Li and H. Zhao, "A survey on representation learning for user modeling," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, Jul. 2020, pp. 4997–5003.

[8] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[9] S. Rendle, "Factorization machines with libFM," *ACM Trans. Intell. Syst. Technol.*, vol. 3, no. 3, pp. 1–22, May 2012.

[10] T. Donkers, B. Loepp, and J. Ziegler, "Sequential user-based recurrent neural network recommendations," in *Proc. 11th ACM Conf. Recommender Syst.*, New York, NY, USA, Aug. 2017, pp. 152–160.

[11] J. Lian, X. Zhou, F. Zhang, Z. Chen, X. Xie, and G. Sun, "XDeepFM: Combining explicit and implicit feature interactions for recommender systems," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Jul. 2018, pp. 1754–1763.

[12] Z. Tao, S. Li, Z. Wang, C. Fang, L. Yang, H. Zhao, and Y. Fu, "Log2Intent: Towards interpretable user modeling via recurrent semantics memory unit," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Jul. 2019, pp. 1055–1063.

[13] D. Kim, S. Kim, H. Zhao, S. Li, R. A. Rossi, and E. Koh, "Domain switch-aware holistic recurrent neural network for modeling multi-domain user behavior," in *Proc. 12th ACM Int. Conf. Web Search Data Mining*, New York, NY, USA, Jan. 2019, pp. 663–671.

[14] J. Zhang, B. Bai, Y. Lin, J. Liang, K. Bai, and F. Wang, "General-purpose user embeddings based on mobile app usage," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 2831–2840.
[15] Z. Liu, H. Chen, F. Sun, X. Xie, J. Gao, B. Ding, and Y. Shen, "Intent preference decoupling for user representation on online recommender system," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, Jul. 2020, pp. 2575–2582.
[16] Z. Qiu, X. Wu, J. Gao, and W. Fan, "U-bert: Pre-training user representations for improved recommendation," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, no. 5, May 2021, pp. 4320–4327.
[17] J. Gu, F. Wang, Q. Sun, Z. Ye, X. Xu, J. Chen, and J. Zhang, "Exploiting behavioral consistence for universal user representation," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, no. 5, May 2021, pp. 4063–4071.
[18] M. Quadrana, A. Karatzoglou, B. Hidasi, and P. Cremonesi, "Personalizing session-based recommendations with hierarchical recurrent neural networks," in *Proc. 11th ACM Conf. Recommender Syst.*, New York, NY, USA, Aug. 2017, pp. 130–137.
[19] J. Ni, J. Li, and J. McAuley, "Justifying recommendations using distantly-labeled reviews and fine-grained aspects," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 188–197.
[20] *Retailrocket Recommender System Dataset | Kaggle*, Retailrocket, 2017. [Online]. Available: https://www.kaggle.com/datasets/retailrocket/ecommerce-dataset
[21] J. Golbeck, C. Robles, M. Edmondson, and K. Turner, "Predicting personality from Twitter," in *Proc. IEEE 3rd Int. Conf. Privacy, Secur., Risk Trust IEEE 3rd Int. Conf. Social Comput.*, Oct. 2011, pp. 149–156.
[22] C. Sumner, A. Byers, R. Boochever, and J. G. Park, "Predicting dark triad personality traits from Twitter usage and a linguistic analysis of tweets," in *Proc. 11th Int. Conf. Mach. Learn. Appl.*, vol. 2, 2012, pp. 386–393.
[23] J. B. Hirsh and J. B. Peterson, "Personality and language use in self-narratives," *J. Res. Personality*, vol. 43, no. 3, pp. 524–527, Jun. 2009.
[24] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, and R. Anil, "Wide & deep learning for recommender systems," in *Proc. 1st Workshop Deep Learn. Recommender Syst.*, 2016, pp. 7–10.
[25] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "DeepFM: A factorization-machine based neural network for CTR prediction," 2017, *arXiv:1703.04247*.
[26] Y. Qu, H. Cai, K. Ren, W. Zhang, Y. Yu, Y. Wen, and J. Wang, "Product-based neural networks for user response prediction," in *Proc. IEEE 16th Int. Conf. Data Mining (ICDM)*, Dec. 2016, pp. 1149–1154.
[27] R. Wang, B. Fu, G. Fu, and M. Wang, "Deep & cross network for ad click predictions," in *Proc. ADKDD*, Aug. 2017, pp. 1–7.
[28] K. Shin, H. Kwak, K.-M. Kim, M. Kim, Y.-J. Park, J. Jeong, and S. Jung, "One4all user representation for recommender systems in e-commerce," 2021, *arXiv:2106.00573*.
[29] Q. Sun, J. Gu, B. Yang, X. Xu, R. Xu, S. Gao, H. Liu, and H. Xu, "Interest-oriented universal user representation via contrastive learning," 2021, *arXiv:2109.08865*.
[30] Y. Ni, D. Ou, S. Liu, X. Li, W. Ou, A. Zeng, and L. Si, "Perceive your users in depth: Learning universal user representations from multiple E-commerce tasks," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Jul. 2018, pp. 596–605.
[31] L. R. Goldberg, "An alternative 'description of personality': The big-five factor structure," *J. Personality Social Psychol.*, vol. 59, no. 6, p. 1216, 1990.
[32] D. Nettle, *Personality: What Makes You the Way You Are*. Oxford, U.K.: Oxford Univ. Press, 2009.
[33] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," 2014, *arXiv:1409.1259*.
[34] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[35] J. Weston, S. Bengio, and N. Usunier, "WSABIE: Scaling up to large vocabulary image annotation," in *Proc. 22nd Int. Joint Conf. Artif. Intell.*, 2011, pp. 1–7.
[36] F. M. Harper and J. A. Konstan, "The MovieLens datasets: History and context," *ACM Trans. Interact. Intell. Syst.*, vol. 5, no. 4, pp. 1–19, Dec. 2015.
[37] B. Brost, R. Mehrotra, and T. Jehan, "The music streaming sessions dataset," in *Proc. World Wide Web Conf. (WWW)*, New York, NY, USA, 2019, pp. 2594–2600.
[38] F. Wu, Y. Qiao, J.-H. Chen, C. Wu, T. Qi, J. Lian, D. Liu, X. Xie, J. Gao, W. Wu, and M. Zhou, "MIND: A large-scale dataset for news recommendation," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, Jul. 2020, pp. 3597–3606.
[39] J. Pennebaker, R. Boyd, K. Jordan, and K. Blackburn, "The development and psychometric properties of LIWC2015," Tech. Rep., 2015. [Online]. Available: https://repositories.lib.utexas.edu/handle/2152/31333?show=full
[40] S. S. Li and E. Karahanna, "Online recommendation systems in a B2C E-commerce context: A review and future directions," *J. Assoc. Inf. Syst.*, vol. 16, no. 2, pp. 72–107, Feb. 2015.
[41] M. A. S. N. Nunes and R. Hu, "Personality-based recommender systems: An overview," in *Proc. 6th ACM Conf. Recommender Syst. (RecSys)*, New York, NY, USA, 2012, pp. 5–6.
[42] S. Dhelim, H. Ning, N. Aung, R. Huang, and J. Ma, "Personality-aware product recommendation system based on user interests mining and metapath discovery," *IEEE Trans. Computat. Social Syst.*, vol. 8, no. 1, pp. 86–98, Feb. 2021.
[43] C. Bologna, A. C. De Rosa, A. De Vivo, M. Gaeta, G. Sansonetti, and V. Viserta, "Personality-based recommendation in e-commerce," in *Proc. 21st Conf. User Modeling, Adaptation, Personalization*, Rome, Italy, S. Berkovsky, E. Herder, P. Lops, and O. C. Santos, Eds., vol. 997, Jun. 2013.
[44] A. Vinciarelli and G. Mohammadi, "A survey of personality computing," *IEEE Trans. Affect. Comput.*, vol. 5, no. 3, pp. 273–291, Jul. 2014.
[45] Y. Mehta, N. Majumder, A. Gelbukh, and E. Cambria, "Recent trends in deep learning based personality detection," *Artif. Intell. Rev.*, vol. 53, no. 4, pp. 2313–2339, Apr. 2020.
[46] C. C. Johnson, "Logistic matrix factorization for implicit feedback data," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 1–9.

YUICHI ISHIKAWA received the B.E. and M.E. degrees in information and communication engineering from The University of Tokyo, Japan, in 2000 and 2002, respectively. He is currently pursuing the Ph.D. degree in information science with Kyushu University, Fukuoka, Japan. He is a Senior Research Scientist with KDDI Research Inc., Japan, where he leads a research team in the area of computational psychology. His research interests include personality recognition from daily behavior and personalization in various services based on personality.

ROBERTO LEGASPI has been investigating for more than two decades how human behavior can be computed for human-centric AI interactions. His focus is to enable an AI to unobtrusively detect and understand human behavioral cues in order to adapt its responses to human needs. His current research interests include AI modeling and adapting its interactions to human sense of agency.

KEI YONEKAWA received the B.S. and M.S. degrees from the Department of Electrical Engineering, The University of Tokyo, in 2012 and 2014, respectively. In 2014, he joined KDDI Corporation, where he was engaged in the operation of the infrastructure of cloud service. In 2015, he joined KDDI Research, Inc., where he is currently a Researcher. His research interests include machine learning, data mining, transfer learning, and MLOps.

YUGO NAKAMURA (Member, IEEE) was born in 1992. He received the B.E. degree from the Advanced Course of Production System Engineering, National Institute of Technology, Hakodate College, Japan, in 2015, and the M.E. and Ph.D. degrees from the Graduate School of Information Science, Nara Institute of Science and Technology, Japan, in 2017 and 2020, respectively. He is currently an Assistant Professor with the Graduate School, Kyushu University, and the Faculty of Information Science and Electrical Engineering, Kyushu University. His current research interests include the Internet of Things, ubiquitous computing, and human–computer interaction. He is currently a member of ACM and IPSJ.

SHIGEMI ISHIDA (Member, IEEE) received the B.E. degree in electrical engineering from the Shibaura Institute of Technology, Tokyo, Japan, in 2006, and the M.S. and Ph.D. degrees in electrical engineering and information system from The University of Tokyo, Japan, in 2008 and 2012, respectively. He was a Visiting Scholar at the Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN, USA, in 2013. From 2013 to 2021, he was an Assistant Professor at the Faculty of Information Science and Electrical Engineering, Kyushu University, Fukuoka, Japan. He is currently an Associate Professor with the Department of Media Architecture, School of Systems Information Science, Future University Hakodate, Japan. His research interests include wireless sensor networks, low-power wireless communications, localization systems, cross technology communications, and intelligent transportation systems. He is currently focused on localization systems and sensing technologies for better understanding of the world around us.

TSUNENORI MINE received the B.E. degree in computer science and computer engineering and the M.E. and D.E. degrees in information systems from Kyushu University, in 1987, 1989, and 1993, respectively. He is currently an Associate Professor with the Department of Advanced Information Technology, Faculty of Information Science and Electrical Engineering, Kyushu University. His research interests include developing real services using artificial intelligence techniques, in particular, natural language processing, text mining, data mining, recommendation, and multiagent systems. He received a Best Paper Award from the *Journal of Information Processing* Society of Japan (IPSJ) for his work on a parallel parsing algorithm, in 1993, and an IPSJ Activity Contribution Award, in 2014. He is currently leading several joint research projects with several companies and academic institutions to develop technologies and theories that are both practical and academically novel.

YUTAKA ARAKAWA (Member, IEEE) received the B.E., M.E., and Ph.D. degrees from Keio University, Japan, in 2001, 2003, and 2006, respectively. He is currently a Professor with the Graduate School and the Faculty of Information Science and Electrical Engineering, Kyushu University. His current research interests include human activity recognition, behavior change support systems, and location-based information systems. He is a member of ACM, IPSJ, and IEICE.