Neural Attention Frameworks for Explainable Recommendation

Omer Tal, *Member, IEEE*, Yang Liu, *Member, IEEE*, Jimmy Huang, *Senior Member, IEEE*, Xiaohui Yu, *Member, IEEE*, and Bushra Aljbawi, *Member, IEEE*

Abstract—Neural attention, an emerging technique used to identify important inputs within neural networks, have become increasingly popular in the area of recommender systems. Not only allowing to better identify what defines users and items, attention-based recommender systems are further able to provide accompanying explanations. However, these representations usually capture only part of users' preferences and items' attributes, resulting in limited reasoning and accuracy. We therefore propose Dual Attention Recommender with Items and Attributes (DARIA), a novel approach able to combine two dependable neural attention mechanisms to better justify its suggestions. Utilizing the personalized history of users, DARIA identifies the most relevant past activities while considering the real-world features that contributed to the similarity. In addition, we adopt the novel approach of self-attention and introduce Self-Attention Recommender based on Attributes and History (SARAH). As a variation to DARIA, SARAH utilizes two self-attention components to describe users by their most characteristic past activities and items by their best depicting attributes. Various experiments establish the significant improvement of SARAH and DARIA over seven key baselines in diverse recommendation settings. By comparing our two proposed frameworks, we demonstrate the potential benefit of applying self-attention in different scenarios.

Index Terms—Recommender systems, collaborative filtering, deep learning, neural attention

1 Introduction

In today's age of information, it has become a prerequisite to have reliable data prior to making any decision. Preferably, we expect to obtain opinions from like-minded users before investing our time and money in any product or service. Personalized *recommender systems* (RS) are an increasingly important component in many on-line businesses and websites [1], [2], [3]. They consist of providing users with tailored lists of relevant items, such as movies, songs and products, thus improving the purchase and consumption experiences. Collaborative Filtering (CF) models based on latent features are probably the most popular implementations in recent years [4], [5], [6], where user preferences and item attributes are represented by concise vectors, learned by the system.

A recent development in recommender system research is the use of neural attention [7], [8], a technique that allows a model implemented using neural networks to focus on its most important input features. By applying neural attention, recommender systems are able to identify user features that are closely related to those of the recommended item. For example, when recommending a horror movie to a user who usually prefers drama movies, an attention-based recommendation model may give emphasis on the few horror movies she previously watched, rather on the less relevant numerous drama movies she viewed. Furthermore, while recommender systems based on deep neural networks are usually considered uninterpretable due to the use of multiple nonlinearities, attention allows such models to accompany reasoning to generated recommendations, while retaining the ability to learn accurate user and item latent representations [9], [10].

While previous work applied neural attention to generate explanations in the form of relevant past items [11], [12], such similarity is derived by latent weights defining each item within the system. Two items reported similar may not be deemed so by end users, contradicting the initial motivation for providing an explanation. To this end, we propose *Dual Attention Recommender with Items and Attributes* (DARIA), a novel approach to attention-based recommender systems. DARIA combines two neural attention mechanisms to identify the user's relevant past items and the real world features that can better explain reported similarities. By doing so, DARIA identifies which features of past items can better represent different users and provide clear explanations.

To identify relevant past items, DARIA follows the findings of previous work [13], [14], [15] and considers the context of each past activity. More specifically, we enrich past items' latent representations with those of items the same user recently interacted with. By integrating a recurrent neural network in our model, we are able to provide a more personalized user representation within DARIA, rather than relying only on the context-free items the user interacted with. DARIA further applies a co-attention mechanism [16] to compare interpretable features (e.g., textual reviews, categories, spatial location)

- *O. Tal, Y. Liu, and B. Aljbawi are with the Department of Physics and Computer Science, Wilfrid Laurier University, Waterloo, ON N2L 3C5, Canada. E-mail: {talx6630, aljb1640}@mylaurier.ca, yangliu@wlu.ca.*
- *J. Huang and X. Yu are with the School of Information Technology, York University, Toronto, ON M3J 1P3, Canada.. E-mail: {jhuang, xhyu}@yorku.ca.*

Manuscript received 15 May 2019; revised 11 Sept. 2019; accepted 29 Oct. 2019. Date of publication 12 Nov. 2019; date of current version 1 Apr. 2021. (Corresponding author: Yang Liu.)
Recommended for acceptance by D. Cai.
Digital Object Identifier no. 10.1109/TKDE.2019.2953157

of past items and the target item, and determine why some past items are deemed more relevant than others. By being able to seamlessly compare various features, DARIA performs feature selection for each given item, allowing it to mitigate the item cold-start problem, one of the most common challenges in modern recommender system research.

Although DARIA is able to identify the important components of its inputs by considering two perspectives of the same user-item interaction, i.e., historical data and item features, it may fail to fully capture the attributes that best define users and items within the system. Extending our previous examples, a model representing a user based only on the few horror movies she watched might result loss of relevant information (i.e., favorite actors, interest in book adaptations, etc). Moreover, DARIA is not able to explain which item attributes best describe the recommended item. We therefore build upon recent work [17], [18] and suggest an innovative approach to attention-based recommender systems. *Self-Attention Recommender based on Attributes and History* (SARAH) is a framework based on the rising self-attention paradigm. Unlike the standard neural attention which identifies the importance of input features with regard to a specific user-item interaction, self-attention attempts to compare the internal components of its input and determine which features are the most important.

Similar to DARIA, SARAH utilizes users' historical activities and items' features to generate product recommendations along with relevant explanations. While DARIA targets past feedbacks that are most applicable to the current experience, SARAH represents users by identifying the past items that best capture the general user preferences, as determined by a self-attention component. Furthermore, SARAH assumes that not all item features are equally important. For example, the quality of special effects has a major impact on science fiction movies, but less so for documentaries. We therefore represent items by applying a second self-attention mechanism to consider each item's most important real-world features. Due to the interpretable nature of its resulting attention scores, SARAH is able to provide reasoning to the end user in the form of most important features describing the recommended item and what past interactions best represent the user within the system.

In this work we empirically evaluate the two proposed frameworks in multiple settings. Both DARIA and SARAH are shown to consecutively outperform seven diverse baselines that include classic recommendation methods [5], [19], deep learning [6], [20] and neural attention [14], [21], [22]. Our experiments are performed over four datasets that combine restaurant recommendation in *Yelp* and product purchases in *Amazon Electronics*, *Movies* and *Home*. We further exhibit our suggested methods' ability to provide interpretable reasoning in the form of heat maps and examine the influence of multiple hyperparameters on our models.

The main contributions of our work are summarized as follows:

1) We propose a novel idea of stacking two attention layers over items and their features, in order to best represent the user-item interaction in DARIA.
2) To the best of our knowledge, SARAH is the first recommendation framework utilizing self-attention to model user history and item attributes. The use of self-attention presents a broad range of future possibilities to improve item recommendations and explain-ability.
3) We empirically evaluate DARIA and SARAH over varied datasets, demonstrating their ability to significantly provide more relevant recommendations compared to seven key baselines. We further provide multiple case studies demonstrating SARAH's interpretability, as well as the impact of self-attention on its output.

The following section provides relevant background and related methods to our work. In Sections 4 and 3 we present our two proposed frameworks, which are evaluated in Sections 5 and 6. Finally, we summarize our work in Section 7.

2 Related Work

2.1 Collaborative Filtering Recommenders

CF-based recommender systems have been gaining wide popularity in recent years. They are based on the assumption that users who shared similar preferences in the past will continue to do so in the future. These sets of past preferences are usually described as latent representations, best demonstrated in the classic matrix factorization (MF) model and its variations [4], [23], where vectors holding concise sets of weights are used to represent the user or item features.

However, real-world settings have proved the basic CF models to be insufficient in modeling users' complex factors leading to the liking of one product over another. A recent advance is the use of deep learning to better represent users and items. Probably the most resembling to the classic MF are deep models based on multi-layer perceptrons (MLP) [1], [2], [6], [24], where user and item latent vectors are denoted as embeddings and instead of dot product, their interaction is learned by multiple nonlinear fully connected layers composed of many neurons.

Utilizing the explosion of available data, deep learning based systems are able to provide better representations and use a wide variety of inputs. Convolutional neural networks (CNN) is a popular choice for learning latent representation from natural language input, such as item description or user reviews [20], [25]. Another common approach for textual modeling in RS is the use of recurrent neural networks (RNN) [26], [27], [28], a technique that treats its inputs as a timely sequence, and results in a representation for each component with regard to its predecessors and successors. Not limited to text only, RNNs have found great use in contextual recommender systems [29]. Instead of modeling each item a user has previously interacted with separately, the item will be represented differently depending on its context. For example, based on her history, one user will be recommended a comedy after watching a documentary, while the other a drama.

2.2 Attention-Based Neural Networks

Defined in the field of neural science [30], attention describes the ability to focus on what is perceived as the important part of an input. By integrating attention in neural networks, this concept has seen increasing success in computational tasks

such as image processing [31], textual translation [32], [33], summarization [34], and others [35], [36].

Recent works have been introducing attention-based neural networks to RS. [7] computes attention weights to identify the most important words in a textual input. [37] adopts attention to learn the user-item interaction function, while [14] generates a user embedding by weighting the user's previously interacted items saved in memory matrices with respect to the candidate item. Likewise, [38], [39] constructs the latent vector by comparing a target item to all items the user has read in the past. Resembling DARIA, [16] presents a method for co-attention where two attention layers attempt to identify relevant reviews and the most important words within them. However, our method applies co-attention in dedicated recommendation scenarios over users' context and items' features rather on the more general textual input.

Due to its novelty, few works have implemented self-attention in recommender systems. [40] generates a user embedding vector by employing self-attention in an encoder framework over the sequence of items the user has interacted with. However, the model does not output an item recommendation, but only the user latent vector. [22] utilizes self-attention to identify the most important past items in the user's history by generating three transformations of the input embedding and performing a weighted average. [41] first exploits self-attention to capture interactions among features from users and items, followed by constructing another layer that integrates auxiliary information with different weights. [42] propose a different approach to employ users' history by applying self-attention over sequences of recent items. While current recommendation models utilize self-attention solely over users' past items, our proposed method further identifies what features best define the recommended item and provides interpretable outputs.

2.3 Explainable Recommender Systems

The heavy use of latent factors in CF has led such methods to being perceived as a black box only able to predict a recommendation. However, as found by previous researches [43], the ability to justify a recommendation is vital and improves the trust users have in the system. Explainable recommender systems are therefore gaining popularity, dedicated to generating recommendations along with human-interpretable reasoning. Facing with the increasing popularity of deep learning, the challenge of explainable deep recommender systems is to utilize the strengths of neural networks in modeling users and items using multiple layers, while retaining enough data to explain the system's output [44], [45], [46].

The success of attention networks in recommender systems has introduced a new opportunity for explainable RS, by enabling a deep model to be composed of vectors representing the importance of various inputs. In [9], the authors embed user and item properties, in order to find the most relevant attributes to the given interaction. Similarly, [10] compares item features along with the target user to find the traits that are most important to her. [8] adopts attention to determine which item reviews are more valuable with respect to the current interaction, in order to provide them to the user. A number of notable works [47], [48] adopted knowledge graphs to define the system's inputs and their relation, however unlike other models based on neural attention, the use of knowledge graphs is restricted to specific applications (i.e., news, movies) and can not be equally applied to other fields.

Unlike previously described models, in this work we introduce explainable recommender systems composed of multiple attention layers over the user's context and the item's attributes. By implementing neural attention, DARIA and SARAH are able to learn an interpretable portrait of their inputs. Moreover, user-item interactions are represented using their most important components, making them more accurate. To the best of our knowledge, this work is the first to exploit self-attention for users and items for the task of explain-ability. A resembling work is [49], where the model's reasoning is based on a movie's content and user's past activity, however this is done by clustering rather than collaborative filtering.

3 DARIA

In this section we present our proposed framework, Dual Attention Recommender with Items and Attributes (DARIA). Due to the extensive use of recommender systems in e-commerce websites and the privacy requirements towards users' data, practitioners usually have access to only limited information about the user, but are often afforded a vast amount of information on the attributes of items. Therefore, in constructing DARIA we choose to focus on historical data available in the system as each user's input, while the widely available and task-specific attributes will be used as the item's input. Moreover, the use of attributes allows DARIA to alleviate the item cold-start problem.

DARIA attempts to describe a given user-item instance by focusing on the most correlating components of its two inputs. This method assumes that although a user has previously interacted with many items, not all past feedbacks are relevant in predicting her interest towards a given item. For example, to decide whether to recommend a horror movie to a user, DARIA will compare the importance of the user's past movies to the recommended horror movie and will likely emphasize similar horror movies she watched. We therefore follow previous work [14], [38] and employ neural attention to model this behavior, while disregarding irrelevant past items when determining whether to recommend a given item.

However, we claim that for the task of explain-ability it is insufficient to only provide what past actions contributed to the recommendation. It is more intuitive and general to further explain what features in these relevant past interactions have had the highest impact on this reported contribution. We will therefore extract only the most relevant past items, as determined by the previously described attention network, and apply a second attention layer over their attributes, compared to those of the recommend item. This layer's output will then be the importance distribution of each relevant past item's features compared to these of the recommended item. It will be further used to construct the concise representation of the user-item interaction and to generate implicit predictions in turn. As a result, DARIA yields the importance distributions over

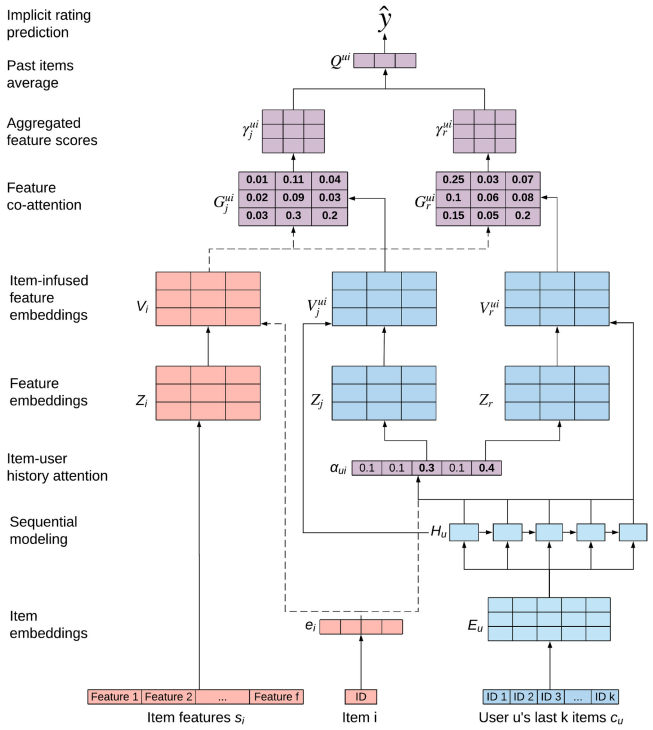

Fig. 1. DARIA framework.

items and features, learned by the attention layers, as explanatory factors to the end user.

By implementing DARIA we hope to shed light on the impact of applying multiple neural attention layers in deep recommender systems. The proposed method is illustrated in Fig. 1, where each interaction is fed as a triplet of item i, the set of its f features, $\boldsymbol{s}_i$, and a set of the last k items a user u has interacted with, $\boldsymbol{c}_u$. Throughout this section, notations in bold style denote vectors and matrices, in lower-case and upper-case, respectively, while normal style denotes scalars.

3.1 Past Items Importance

DARIA's first goal is to identify the most relevant past actions of user u with regard to the candidate item i while ignoring the least relevant past feedbacks. We define hyperparameter k as the number of last items used to represent each user, and provide the flexibility to explore different time spans. The list of k input items is denoted as $\boldsymbol{c}_u$ and demonstrated in the right side of Fig. 1. We therefore describe each user using the latent representations of past items $\boldsymbol{E}_u = \{\boldsymbol{e}_{u0}, \ldots, \boldsymbol{e}_{ux}, \ldots, \boldsymbol{e}_{uk}\}$, where $\boldsymbol{e}_{ux}$ is the embedding of user u's xth item.

Since users' actions are frequently effected by previous activities (e.g., interest in a sequel after watching a movie), we attempt to enrich user u's item embeddings with the contextual information found within the sequence of interactions. More specifically, we employ a RNN over the k embeddings, so the representation of each item will include that of its predecessors to some degree. Following the findings of previous works [26], we chose to implement our recurrent network using gated recurrent units (GRU), an architecture that achieves competitive performance compared to long short-term memory units (LSTM), but with less parameters, making it more efficient:

$$\begin{aligned}
\boldsymbol{f}_x &= \sigma(\boldsymbol{W}_f \boldsymbol{e}_x + \boldsymbol{R}_f \boldsymbol{h}_{x-1} + \boldsymbol{b}_f) \\
\boldsymbol{o}_x &= \sigma(\boldsymbol{W}_o \boldsymbol{e}_x + \boldsymbol{R}_o \boldsymbol{h}_{x-1} + \boldsymbol{b}_o) \\
\tilde{\boldsymbol{c}}_t &= tanh(\boldsymbol{W}_{\tilde{c}} \boldsymbol{e}_x + \boldsymbol{f}_x \odot \boldsymbol{R}_{\tilde{c}} \boldsymbol{h}_{x-1} + \boldsymbol{b}_{\tilde{c}}) \\
\boldsymbol{h}_x &= (1 - \boldsymbol{o}_x) \odot \boldsymbol{h}_{x-1} + \boldsymbol{o}_x \odot \tilde{\boldsymbol{c}}_x \ ,
\end{aligned} \tag{1}$$

where $\boldsymbol{f}_x$ is the forget gate for input item x, $\boldsymbol{o}_x$ is the output gate, $\tilde{\boldsymbol{c}}_x$ is the new candidate state combining the new input with the previous hidden state and $\boldsymbol{h}_x$ is current state for item x modeled by the output gate. $\odot$ denotes the element-wise product, $\boldsymbol{W}_f$, $\boldsymbol{W}_o$, $\boldsymbol{W}_{\tilde{c}}$, $\boldsymbol{R}_f$, $\boldsymbol{R}_o$ and $\boldsymbol{R}_{\tilde{c}}$ are the GRU weight matrices while $\boldsymbol{b}_f$, $\boldsymbol{b}_o$ and $\boldsymbol{b}_{\tilde{c}}$ are the bias vectors.

A neural attention layer is then applied to identify the most relevant $r < k$ items in user u's history with regard to item i. We first generate a score for each of the k items using dot-product between its embedding's latent features and those of e_i:

$$\begin{aligned}
a_{uix} &= \boldsymbol{h}_{ux} \cdot \boldsymbol{e}_i \ , \\
\boldsymbol{a}_{ui} &= \{a_{ui0}, a_{ui1}, \ldots, a_{uik}\} \ , \\
\boldsymbol{\alpha}_{ui} &= softmax(\boldsymbol{a}_{ui}) \ ,
\end{aligned} \tag{2}$$

where x is one of the k items in user u history, $\boldsymbol{a_{ui}}$ is the set of u's attention scores between i and each past item, and $\boldsymbol{\alpha_{ui}}$ is an importance distribution of the k items with regard to item i, and will likely be different for another recommended item i'.

While $\boldsymbol{\alpha}_{ui}$ will be provided to the user as an explanatory factor, reporting how related each of the items she previously liked to the recommended item i, we wish to further investigate what features impact the items' significance levels. However, there is no need to further explore the effect different features have for irrelevant past items. To this end, we use max-pooling to keep only the r items with the highest attention weights in $\boldsymbol{\alpha}_{ui}$, denoted as $\boldsymbol{\beta}^{ui} = \{ui_0, ui_1, \ldots, ui_r\}$, where the remaining $k - r$ past items of user u are discarded, such that:

$$\boldsymbol{\alpha'}_{ui} = \underset{\boldsymbol{\beta}^{ui} \subset \boldsymbol{\alpha}_{ui}, |\boldsymbol{\beta}^{ui}| = r}{\arg\max} \ . \tag{3}$$

Each relevant past item $j \in \boldsymbol{\beta}^{ui}$ is then transformed into its features embedding, denoted as: $\boldsymbol{Z}_j = \{z_{j0}, \ldots, z_{jf}\}$, where $z_{jl} \in \mathbf{R}^{\hat{d}}$ is the latent vector representing item j's lth attribute and $\hat{d}$ is the number of latent weights.

3.2 Item Features Attention

To determine what features made each of the r past items more related to i than other items, we apply a co-attention layer and compare the features of each past item j with those of item i. While each item is composed of features, some can be shared between very different items. For example, two movies can be of the same genre and share participating actors, but one is more popular than the other due to reasons that are not captured by the set of given attributes. Therefore, in addition to features we also apply the item's historical information, described as its embedding, to construct the final representing vector. The two sets of inputs are fed to the network as plain item and feature ids, implemented as one-hot encoding vectors, where 1 indicates the current item or feature and 0 the rest. Two embedding functions are used to transform the inputs into representing

vectors, $\boldsymbol{e}_i \in \mathbb{R}^d$ for item i and $\boldsymbol{Z}_i$ for its features. These embeddings result in two descriptions for the same item, one follows CF paradigms using a latent vector, and the other with its attributes which are interpretable.

To combine the two representations in a way that allows the same attribute to have different effects on various items but still enables to isolate the different features, we let the model learn their combination by employing a fully connected nonlinear layer over the embeddings' concatenation:

$$\boldsymbol{v}_{il} = ReLU(\boldsymbol{w}_v[\boldsymbol{e}_i, \boldsymbol{z}_{il}] + \boldsymbol{b}_v) \ , \tag{4}$$

where $ReLU$ is a nonlinear activation function, $\boldsymbol{w}_v \in \mathbf{R}^{d'}$ is the weight vector, $\boldsymbol{b_v}$ the bias and d' denotes the number of latent weights used to represent newly modified features in $\boldsymbol{v}$. $\boldsymbol{V}_i = [\boldsymbol{v}_{i0}, \ldots, \boldsymbol{v}_{if}]$ is the layer's output denoting the feature-infused representation of item i, where f is the number of features. By combining the item's attributes with its past interactions, we allow the model to potentially represent items with little or no history, making DARIA relatively resilient to the item cold-start problem. For example, features such as a movie's budget, cast or director can allow DARIA to better define a movie, even with no previous ratings. Similarly, a relevant past item j with respect to user u and target item i can be described following the same structure:

$$\begin{aligned} \boldsymbol{v_{jl}^{ui}} &= ReLU(\boldsymbol{w}_v[\boldsymbol{e}_j^{ui}, \boldsymbol{z}_{jl}] + b_v) \ , \\ \boldsymbol{V_j^{ui}} &= [\boldsymbol{v}_{j0}^{ui}, \boldsymbol{v}_{j1}^{ui}, \ldots, \boldsymbol{v}_{jf}^{ui}] \ . \end{aligned} \tag{5}$$

Unlike the attention technique previously used in Eq. (2) to compare items, the design of our model allows us to compare all feature combinations between every past item and the target item, resulting the co-attention component as demonstrated in Fig. 1. We can therefore evaluate the relevance of each feature for j with all other features of i:

$$\boldsymbol{G}_j^{ui} = sofmtax(\boldsymbol{V}_i^T \times \boldsymbol{V}_j^{ui}) \ , \tag{6}$$

where $\times$ denotes matrix multiplication and $\boldsymbol{G}_j^{ui} \in \mathbf{R}^{f \times f}$ is a two-dimensional matrix where the f features of past item j can be depicted as columns while those of the recommended item i as rows.

To describe the overall user-item interaction by past items' feature importance compared to i, DARIA then transforms the scores of an item j lth feature, j_l, by employing a simple summation over each of the matrix $\boldsymbol{G}_j^{ui}$ columns:

$$\boldsymbol{\gamma}_{jl}^{ui} = \sum_{t=0}^{f} \boldsymbol{G}_{jlt}^{ui} \ , \tag{7}$$

where t is a feature of candidate item i as well as a row in $\boldsymbol{G}_j^{ui}$, while l is a feature of u's past item j and a column in the attention matrix. By summing all rows of a feature l, we are able to retrieve its aggregated attention score with regard to all features of item i.

In addition, $\boldsymbol{\gamma}_{jl}^{ui}$ will act as an explanatory factor given to the user. It provides an insight towards what features makes a past item j relevant in estimating the preference towards item i. $\boldsymbol{\gamma}_{jl}^{ui}$ will be further utilized as a weighting vector to generate a representation of user u's past item j according to its attention scores with the candidate item i:

$$\boldsymbol{q}_j^{ui} = \sum_{l=0}^{f} \boldsymbol{\gamma}_{jl}^{ui} \cdot \boldsymbol{v}_{jl}^{ui} \ , \tag{8}$$

where $\boldsymbol{v}_{jl}^{ui} \in \mathbf{R}^{d'}$ is the feature l's embedding of past item j with regard to user u and item i, $\boldsymbol{q}_j^{ui} \in \mathbf{R}^{d'}$ is j's feature-based latent vector, f the number of item features and d' is each feature's latent dimensionality as described in Eq. (4).

Although DARIA previously compared each past item of user u to the target item i in order to identify the r most relevant items, it considered all r items equal. However, some items can be extremely more relevant than others. For example, a movie from the same series as the target movie will be more relevant than movies from the same genre, which in turn are more relevant than unrelated movies. To support this behavior, we utilize the importance scores of the r items as derived in Eq. (3) to perform a weighted average on the respective interpretable vectors $\boldsymbol{q}_*^{ui}$:

$$\boldsymbol{Q}^{ui} = \sum_{j=0}^{r} \boldsymbol{\alpha'}_{uij} \cdot \boldsymbol{q}_j^{ui} \ . \tag{9}$$

3.3 Rating Prediction

Finally, DARIA transforms the concise representation of the user-item interaction, as derived in $\boldsymbol{Q}^{ui}$, to a prediction score in the range of $[0-1]$. Since there is a strong connection between the latent factors of past items, we employ a single-layered neural network, activated by the sigmoid function:

$$\hat{y} = \sigma(\boldsymbol{w_y} \times \boldsymbol{Q^{ui}} + b_y) \ , \tag{10}$$

where $\boldsymbol{w_y} \in \mathbf{R}^{d'}$ and b_y are the weight vector and bias term, respectively. We apply the sigmoid function as the network's non-linearity to produce an implicit prediction, used as DARIA's output. To allow DARIA to be applied in a wide variety of recommendation scenarios, we adapt every sample to the implicit rating problem, where 1 denotes an interacted item, indicating actions such as a page click or a purchase, and 0 items the user has no interest in. We use negative sampling to generate instances of 0-class ratings from the dataset, defined as Y^- to differ from the positive set Y.

Due to the use of implicit ratings and the sigmoid function in prediction, the learning problem can be now viewed as a task of binary classification. The model's output probability can be then defined as:

$$p(Y, Y^- \mid \boldsymbol{E^i}, \boldsymbol{E^s}, \Theta_f) = \prod_{(u,i) \in Y} \hat{y}_{ui} \prod_{(u,j) \in Y^-} (1 - \hat{y}_{uj}) \ , \tag{11}$$

where (u, i) is a positive interaction, (u, j) a negative sampled instance, $\boldsymbol{E^i} \in \mathbb{R}^{m \times d}$ and $\boldsymbol{E^s} \in \mathbb{R}^{f \times \tilde{d}}$ are the embeddings of all m items and f features,respectively, and Θ_f denotes the model's parameters.

Taking the negative log-likelihood of p results in the binary cross-entropy loss function for the prediction portion of the model:

$$L = -\sum_{(u,i) \in Y \cup Y^-} y_{ui} log(\hat{y}_{ui}) + (1 - y_{ui}) log(1 - \hat{y}_{ui}). \tag{12}$$

To optimize Eq. (12), we apply a gradient descent method, and more specifically the Adaptive Moment Estimation (Adam) [50]. This optimizer automatically adjusts the learning rate and yields faster convergence than the standard gradient descent in addition to making the learning rate optimization process more efficient.

4 SARAH

In this section, we introduce a variation to DARIA, denoted as Self-Attention Recommender based on Attributes and History (SARAH). Unlike standard neural attention layers that are based on comparing tuples of inputs one at a time to determine importance, self-attention is composed of measuring the internal components of the input to identify it's most valuable elements [17], [18]. Implemented using paradigms of deep neural networks, SARAH utilizes each item's input features to generate an accurate representation of the item, where a novel self-attention layer determines what features should have higher impact. Likewise, a second self-attention layer specifies the most characterizing items each user has previously interacted with, used to construct the user latent vector. Our proposed method combines the two representations to result in an implicit prediction to whether a given user will be interested in the candidate item or not, while sharing the self-attention importance weights as explanatory factors.

By applying independent self-attention layers to represent users and items, SARAH assumes that the importance of previous user activities is not effected by the recommended item in question. For example, a user that usually watches action movies will be likely represented by features relating to such movies (e.g., special effects, crime, etc) while a recommended horror movie will be represented by the features that best define the genre.

To develop our framework, we will first describe the item self-attention network, resulting in a latent vector using an item and its features as input. Then, we will introduce the user self-attention component that transforms the user's last interacted items to a latent representation, along with her contextual data. The two resulting representations describe the candidate item and target user, respectively, while giving relative weights to their components based on their importance. Finally, we provide a brief comparison between our two proposed models, DARIA and SARAH. The complete architecture of our framework is shown in Fig. 2.

4.1 Item Self-Attention

The item self-attention component's goal is to produce an interpretable representation for a candidate item, where each is composed of a given set of attributes. These can be derived from various inputs, such as textual reviews, description, categories, geo-spatial location, etc. While all items are comprised of the same attributes, the importance of each attribute differs. Using self-attention we represent each item as the weighted distribution of its attributes. This allows SARAH to learn an item representation regardless of the target user, where the weights are interpretable and point towards the most relevant features.

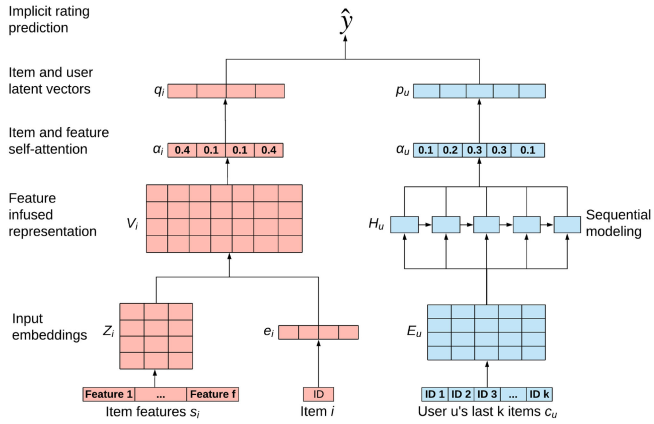

Fig. 2. SARAH framework with sample weights.

As demonstrated in the left section of Fig. 2, we adopt the modified feature representations, defined in Eq. (4), and employ a self-attention network to transform $\boldsymbol{V}_i$ into an interpretable vector, comprised of each feature's importance. By feeding it to a two-layer perceptron, $\boldsymbol{V}_i$ is reduced to a concise representation of a single attention score for each input feature:

$$\boldsymbol{a}_i = \boldsymbol{w_2}(tanh(\boldsymbol{W_1}\boldsymbol{V}_i) + \boldsymbol{b_1}) + b_2 \ , \tag{13}$$

where $\boldsymbol{W_1} \in \mathbb{R}^{f \times d}$, $\boldsymbol{w_2} \in \mathbb{R}^f$ are the respective weight matrix and vector, $\boldsymbol{b_1} \in \mathbb{R}^d$, $b_2 \in \mathbb{R}$ are the layer's bias terms and $tanh$ is the hyperbolic tangent function. To increase the self-attention model's ability to generalize and reduce exaggerated importance towards any single feature, we introduce a dropout function over $\boldsymbol{V}_i$. The resulting vector $\boldsymbol{a}_i$ can be seen as the importance weights of each feature over the item i; however, for it to be utilized as a weight distribution we feed it to a softmax function:

$$\boldsymbol{\alpha}_i = softmax(\boldsymbol{a}_i) \ . \tag{14}$$

$\boldsymbol{\alpha}_i$ will then be employed as a weighting vector, responsible for constructing a one-dimensional representation for the given item i. This is achieved by performing a weighted average over the item features' latent factors according to their relative importance:

$$\boldsymbol{q}_i = \sum_{l=0}^{f} \boldsymbol{\alpha}_{il} \boldsymbol{v}_{il} \ , \tag{15}$$

where $\boldsymbol{q}_i$ is the item i's latent vector. In addition, $\boldsymbol{\alpha}_i$ will act as an explaining factor to the end user, where the features of a recommended item i are ranked based on their attention scores. Examples of the vector's values are presented in the left part of Fig. 2.

4.2 User Self-Attention

As in DARIA, we utilize the embeddings of user u's past items, $\boldsymbol{E_u}$, to represent the recommended target user. Moreover, we apply the same GRU layer as in Eq. (1) to add sequential context to $\boldsymbol{E_u}$, resulting $\boldsymbol{h_{uj}} \in \boldsymbol{H_u}$ as the contextual representation of user u's past feedback for item j. Although a user can be defined as the set of items she previously interacted with, some items are more important in

this sequence than others. To this end, we introduce a self-attention network over the user's k items, similarly to the layers presented in Section 4.1:

$$\begin{aligned} \boldsymbol{a}_u &= \boldsymbol{w_4}(tanh(\boldsymbol{W_3}\boldsymbol{H}_u) + \boldsymbol{b_3}) + b_4 \\ \boldsymbol{\alpha}_u &= softmax(\boldsymbol{a}_u) \ , \end{aligned} \tag{16}$$

where the input $\boldsymbol{H}_u \in \mathbb{R}^{k \times d}$ is the set of k items' sequential embeddings $\{\boldsymbol{h}_{u0}, \boldsymbol{h}_{u1}, \boldsymbol{h}_{u2}, \ldots, \boldsymbol{h}_{uk}\}$. The user self-attention network is learned using the weights $\boldsymbol{W_3}$, $\boldsymbol{w}_4$ and the bias terms $\boldsymbol{b_3}$, b_4. The resulting vector $\boldsymbol{\alpha}_u \in \mathbb{R}^k$ is the importance distribution of the k items for the user u, used to provide recommendation reasoning to the end user, by presenting her with the information towards how she is perceived by the system. Potential values of $\boldsymbol{\alpha}_u$ are given as example in the right side of Fig. 2.

To retrieve a meaningful representation to be modeled along with the candidate item i, we weight each of user u's k sequential item embeddings using the attention importance vector, $\boldsymbol{p}_u$:

$$\boldsymbol{p}_u = \sum_{j=0}^{k} \boldsymbol{\alpha}_{uj} \boldsymbol{H}_{uj} \ , \tag{17}$$

resulting in a vector that emphasizes the latent characteristics of the most relevant historical interaction of the user.

To combine the user and item weighted representations and to result in a prediction, it is possible to apply numerous collaborative filtering techniques. More specifically, we adopt the classic dot product operation:

$$\hat{y} = \sigma(\boldsymbol{q_i}^T \cdot \boldsymbol{p_u}) \ , \tag{18}$$

where $\boldsymbol{q_i}$ and $\boldsymbol{p_u}$ are two latent vectors describing an item i and user u, respectively and $\cdot$ is the dot product operator. By doing so, we avoid any excessive alteration to the user and item interpretable vectors and keep the weighted distributions relevant to explain the given prediction. SARAH's training follows the same steps as DARIA, described in detail in Section 3.3.

4.3 Comparison to DARIA

Although there are numerous similarities between DARIA and SARAH, including their expected inputs, structure of generated outputs and the use of recurrent neural network to model user context, the two methods are fundamentally different in their implementations. The main distinction lies in the concept behind each method. While DARIA attempts to learn the combined interaction of each user and item tuple, SARAH assumes that users and items should have similar latent representations regardless of the current recommendation. The user representation in DARIA is therefore heavily effected by the items that are closely related to the recommended item, as determined by the neural attention layer, and will be significantly different for multiple recommended items. While in SARAH a user is represented by all her k past items in a different degree of importance, in DARIA we only include the $r < k$ past items that are most similar to the candidate item. For example, a certain movie that is deemed as the most representative of the user by SARAH, might be filtered out and not have any impact on the same user representation in DARIA.

In addition, even though the two methods generate an importance distribution over item features as an explanatory factor, the same data has different meaning across our frameworks. While in SARAH features are applied to define each item in an interpretable way, DARIA adopts features to explain why a past item is considered similar to the recommended item. This is then aggregated to describe the features' importance of the user past items with regard to the recommended item. Finally, while the recommendation generated by SARAH is achieved by measuring the similarity between user and item representations using dot product, DARIA applies a neural network over the aggregated feature importance.

By Comparing the two methods, as done in Section 6, we attempt to examine two opposing techniques based on neural attention and determine whether we should describe users and items independently to produce accurate predictions, or alternately generate recommendations by learning the user and item interactions.

5 Experimental Settings

5.1 Datasets

To evaluate our proposed frameworks in diverse recommendation settings, we adopt datasets from different fields.

Amazon. We chose to employ the Electronics, Movies and Home datasets provided by [51]. The three datasets feature over 16 million reviews combined, written about products purchased in the popular e-commerce website. Each of the given datasets is pre-filtered to include only users and items with more than 5 reviews.

Yelp[1]. The public dataset provided by Yelp presents more than 5.9 million reviews by users of locations in the successful location-based social network. While Amazon's datasets are dedicated each to a single area, the one by Yelp ranges over various location types, such as hotels, restaurants, grocery stores, gas stations and more. To allow items to differ by relevant features (i.e., preventing the existence of many disjoint attributes), we limit the data to focus on food-related locations (restaurants, bars, fast food, e.g.,), using dataset-provided categories.

In each scenario, we divide ratings into training, validation and test sets using a time-based split, where each user's last interaction constructs the test set and the one before is used for validation. We further filter out users with less than 15 reviews, to allow sufficient historical records to represent each user. For example, if a user input consists of her last 10 interactions, no less than 3 samples are left to construct the training set while the remaining 2 define the validation and test sets. To provide negative instances we sample 3 zero-rated samples for each positive rating over the training and validation sets. Statistics regarding the final datasets excluding negative samples are presented in Table 1.

Due to the availability of textual inputs in many recommendation scenarios and to allow a similar ground for comparison to other methods, we chose topics derived from

1. https://www.yelp.com/dataset/challenge

TABLE 1
Datasets' Statistics

Dataset	Yelp	Electronics	Movies	Home
Users	43,702	14,346	19,566	3,725
Items	51,068	34,389	30,761	14,008
Positive Ratings	1,681,773	364,899	852,091	90,848
Rating Sparsity	99.925%	99.926%	99.858%	99.826%
Samples/User	38.48	25.43	43.55	24.39
Samples/Item	32.96	10.74	27.72	6.72

user reviews as the item features in the model's and baselines' experiments. To learn the topics, we first feed the reviews as bag-of-words to a latent dirichlet allocation (LDA) model [52]. Then we learn word embeddings using a simple CNN network over our training data. Each topic is finally represented by the product of its distribution over the semantic word embeddings. Although we chose to focus on topics, the proposed framework can easily be fed with various other features, such as categories, location and more, given their latent vectors. Furthermore, the model is not limited to topics derived from LDA, but can equally apply other textual embedding methods [17], [53], [54].

5.2 Evaluation Metrics

To evaluate the recommendation performance of our proposed models and baselines, we focus on their ability to provide relevant top-10 suggestion to the user. For every item the user had reviewed in the test set, we sample 99 negative items she has not encountered with and provide these 100 interactions to each model. Since the test set is comprised of one positive instance for each user, the final set will hold N positive items and $99 * N$ negative samples. Towards the task of model evaluation, we follow previous work [6], [9], [14], [21] and adopt the popular HR@10 and NDCG@10 metrics to measure the performance in predicting top-10 items:

- *Hit-Ratio (HR):* Returns the percentage of users that had a relevant item in their top-10 suggestions. Defined as:

$$HR@10 = \frac{1}{N}\sum_{u} \mathbf{I}(|\boldsymbol{R^u} \cap \boldsymbol{T^u}|) \ , \tag{19}$$

  where $\boldsymbol{R^u}$ and $\boldsymbol{T^u}$ are user u's top 10 predicted and ground truth items, respectively, N is the number of users and $\mathbf{I}$ is a function that returns 1 when its input is positive, and 0 otherwise.
- *Normalized Discounted Cumulative Gain (NDCG):* A measure that assigns weights based on an item's position within the top 10 suggestions:

$$NDCG@10 = \frac{1}{Z}\sum_{j=1}^{10} \frac{2^{rel_j} - 1}{log_2(j+1)} \ , \tag{20}$$

  where rel_j denotes the relevancy of an item in position j and Z is a normalizing factor.

We report the average scores of the aforementioned metrics over all users in the test set. Parameter tuning in the validation set is done by measuring the ROC score.

5.3 Baselines

To evaluate the performance of DARIA and SARAH over the four datasets, we compare their results to seven key baselines, ranging from classic MF and probabilistic methods to deep learning-based models:

- *SASRec* [22]: Self-attention based Sequential Recommendation. A novel model combining attention mechanisms with long-term semantics.
- *CMN* [21]: Collaborative Memory Networks. A method that adopts attention network and memory matrices to provide a neighborhood-based component in an hybrid RS.
- *RUM* [14]: Recommender system with external User Memory networks. A trending deep model that employs attention to identify the most relevant items and features in the target user history compared to the candidate item.
- *NeuMF* [6]: Neural Matrix Factorization. A state-of-the-art framework for recommendation using only past feedbacks. NeuMF combines a generalization of MF with MLP over tuples of users and items.
- *ConvMF* [20]: Convolutional Matrix Factorization. A strong baseline for recommendations using textual input. This probabilistic model utilizes CNN to learn an item latent vector.
- *NMF* [19]: Non-negative Matrix Factorization, a CF method that takes only the rating matrix as input.
- *CTR* [5]: Collaborative Topic Regression. A probabilistic model that estimates an item latent vector using its topics, derived from LDA.

By comparing our proposed frameworks to each of the seven baselines, we measure their performance along with models that best represent different recommendation approaches. CTR [5] and NMF [19] are two well-known methods for classic collaborative filtering, while ConvMF [20] and NeuMF [6] are found to be highly effective in different scenarios. Furthermore, both CTR and ConvMF employ textual data to represent items, similar to DARIA and SARAH in our selected setting. By evaluating RUM [14] and CMN [21], two emerging approaches that utilize neural attention, we are able to analyze the impact of our suggested approaches over the standard attention paradigm.

5.4 Parameters Settings

Our proposed models are implemented using Tensorflow[2], and released as open source[3]. As part of our application, all parameters are initialized to follow uniform distribution, and in order to avoid over-fitting when training the model an early stopping criteria is integrated. Items' latent dimensionality and the size of each user's input are determined by grid search in the ranges of {4,8,16,32,64,128} and {4,6,8,10,12}, respectively. We further set the number of item features as well as the RNN dimensionality to be in {10,30,50,70,90}. Consequentiality, we applied the best reported values from Section 6.2 to evaluate DARIA and SARAH. Furthermore, the latent vector of every feature was generated using the word embeddings of GloVe [55] with

2. https://www.tensorflow.org
3. https://github.com/omer-tal/SARAH

TABLE 2
Performance Comparison Between SARAH, DARIA, and the Seven Baselines

Dataset	Yelp		Electronics		Movies		Home	
Metric	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10
SASRec	0.892 (3.65%)	0.626 (7.8%)	0.574 (19.44%)	0.371 (21.54%)	0.669 (11.54%)	0.458 (11.56%)	0.452 (25.14%)	0.307 (12.66%)
CMN	0.875 (6.47%)	0.592 (14.56%)	0.428 (83.45%)	0.267 (93.08%)	0.534 (39.71%)	0.32 (59.61%)	0.098 (479.23%)	0.048 (606.96%)
RUM	0.750 (21.51%)	0.459 (44.71%)	0.438 (57.35%)	0.270 (75.98%)	0.518 (44.06%)	0.319 (60.06%)	0.323 (75.21%)	0.168 (103.59%)
NeuMF	0.808 (14.43%)	0.489 (37.70%)	0.338 (106.66%)	0.169 (173.71%)	0.501 (48.92%)	0.281 (81.68%)	0.295 (91.70%)	0.147 (131.92%)
ConvMF	0.468 (96.57%)	0.269 (148.93%)	0.341 (102.27%)	0.201 (126.23%)	0.329 (126.93%)	0.185 (175.81%)	0.261 (116.25%)	0.141 (142.80%)
NMF	0.644 (39.49%)	0.399 (62.85%)	0.436 (57.24%)	0.267 (69.19%)	0.444 (68.14%)	0.254 (100.98%)	0.333 (69.76%)	0.175 (95.47%)
CTR	0.362 (129.74%)	0.202 (215.89%)	0.462 (49.60%)	0.236 (93.01%)	0.43 (73.70%)	0.242 (111.33%)	0.348 (62.29%)	0.159 (115.40%)
DARIA	$^{+}_{*}\mathbf{0.892}^{\triangle\diamond}_{\star\circ}$ (3.7%)	$^{+}_{*}\mathbf{0.614}^{\triangle\diamond}_{\star\circ}$ (9.9%)	$^{+}_{*}\mathbf{0.637}^{\triangle\diamond\ddagger}_{\star\circ}$ (7.59%)	$^{+}_{*}\mathbf{0.401}^{\triangle\diamond\ddagger}_{\star\circ}$ (12.34%)	$^{+}_{*}\mathbf{0.678}^{\triangle\diamond\ddagger}_{\star\circ}$ (10.12%)	$^{+}_{*}\mathbf{0.432}^{\triangle\diamond}_{\star\circ}$ (18.08%)	$^{+}_{*}\mathbf{0.548}^{\triangle\diamond\ddagger}_{\star\circ}$ (3.23%)	$^{+}_{*}\mathbf{0.318}^{\triangle\diamond\ddagger}_{\star\circ}$ (7.62%)
SARAH	$^{+}_{*}\mathbf{0.925}^{\triangle\diamond\ddagger}_{\star\circ\bullet}$	$^{+}_{*}\mathbf{0.675}^{\triangle\diamond\ddagger}_{\star\circ\bullet}$	$^{+}_{*}\mathbf{0.686}^{\triangle\diamond\ddagger}_{\star\circ\bullet}$	$^{+}_{*}\mathbf{0.451}^{\triangle\diamond\ddagger}_{\star\circ\bullet}$	$^{+}_{*}\mathbf{0.746}^{\triangle\diamond\ddagger}_{\star\circ\bullet}$	$^{+}_{*}\mathbf{0.51}^{\triangle\diamond\ddagger}_{\star\circ\bullet}$	$^{+}_{*}\mathbf{0.565}^{\triangle\diamond\ddagger}_{\star\circ\bullet}$	$^{+}_{*}\mathbf{0.342}^{\triangle\diamond\ddagger}_{\star\circ\bullet}$

‡, $*$, $+$, *, $\circ$, $\triangle$, $\diamond$, *and* $\bullet$ *indicate the significant improvement compared to SASRec, CMN, RUM, NeuMF, ConvMF, NMF, CTR, and DARIA, respectively, based on a 5-sample unpaired t-test at the 0.01 level. Relative improvement of SARAH compared to each model is in brackets.*

dimensionality of 50. Our frameworks were optimized by applying a learning rate of 0.05 over no more than 100 epochs with a batch size of 8,192 instances.

6 Experiments and Evaluation

In this section we evaluate DARIA and SARAH by conducting a series of analyses to answer the following research questions: *RQ1*: Are the proposed frameworks able to achieve competitive recommendation performance compared with the baselines? *RQ2*: How different hyperparameter values effect the models' ability to learn and present explanations? *RQ3*: Does our models produce relevant and understandable reasoning along with their provided suggestions?

6.1 Overall Performance (RQ1)

First, we compare the prediction performance of the seven baselines and our suggested models. Table 2 displays the comparisons *w.r.t.* HR@10 and NDCG@10 over the different datasets. Several observations can be derived: Analyzing the results over the more balanced datasets, Yelp and Movies, demonstrates the strengths of deep learning based methods, as NeuMF, RUM, CMN, SYSRec DARIA and SARAH are able to provide significantly better recommendations compared to CTR and NMF. The two methods based on textual input alone are found to be insufficient in modeling the user-item interaction. Both the probabilistic method CTR and CNN-based ConvMF achieve the lowest scores over the Yelp and Movies data. It may be due to the use of only 50 units to represent words' embeddings, but the results hints that while textual data is a valuable input, it shouldn't be solely relied upon to learn users' and items' representations.

There is a large discrepancy when evaluating the models across the different datasets. First, the results obtained over the Yelp data are significantly higher compared to the Electronics and Home datasets for all methods. In addition, the trends from the Yelp and Movies results are not kept. In terms of HR@10 and NDCG@10, CTR and NMF achieve similar results to RUM and CMN over the Electronics data, while all four baselines outperform ConvMF and NeuMF. Furthermore, the second best performing baseline in the Yelp dataset, CMN, results in the lowest scores over the Home data. This change can be explained by the analysis of the datasets' statistics, shown in Table 1. While the Home dataset is the most dense, and the rating matrix sparsity rate is similar between the Yelp and Electronics data, the datasets mainly differ by the average samples per user and especially per item. In the Electronics, and most notably the Home data, the insufficient number of samples for each item prevents from deep models, such as ConvMF and NeuMF, to learn relevant representations. However, methods based on attention networks (e.g., RUM and SYSRec) are seem to be more resilient to this problem. This, and the fact that each item is represented by its features as well as its embedding, allows both DARIA and SARAH to surpass all other methods in this scenario.

Comparing our two proposed frameworks, it is clear that SARAH consecutively achieves more accurate performance over DARIA. Since in both cases users and items are identified by identical inputs, it is clear the difference lies in the model itself. While it is more simple, the use of self-attention allows SARAH to learn separate representations with no regard to the given instance and still yield better item recommendations. Overall, SARAH outperforms the evaluated models across all datasets, as the use of features and self-attention allows it to be less prone to the item cold start problem, while utilizing RNN and pre-learned embeddings results in competitive performance in a more dense scenario. Table 2 further demonstrates the improvement is statistically significant using an unpaired t-test over five independent evaluations where $p < 0.01$.

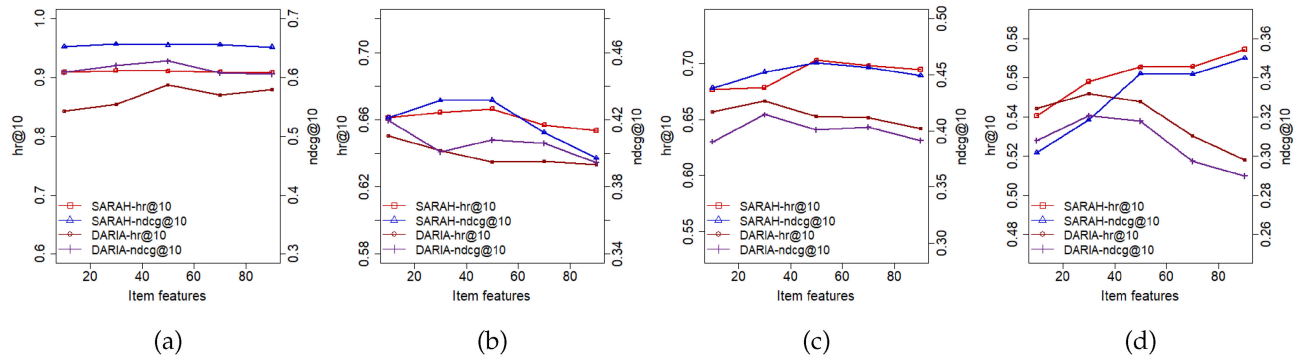

Fig. 3. HR@10 and NDCG@10 over number of features for (a) the Yelp, (b) electronics, (c) movies, and (d) home datasets.

6.2 Hyperparameter Analysis (RQ2)

In this section we study the effects of various factors over the performance of DARIA and SARAH.

6.2.1 Number of Item Features

As the main component for describing items within the systems, the number of features is used to construct both our models' input and one of the two explainable factors derived from them. Fig. 3 illustrates the influence of different sizes of input, ranging from 10 to 90 features, for DARIA and SARAH over all four datasets. The remaining hyperparameters are set to 10 and 50 for k and the item's latent dimensionality, respectively, while no RNN is applied. To provide a fair comparison all features are from the same type, topics derived by LDA. Each item is therefore described by its positive or negative correspondence to each dataset-specific topic (e.g., price, genre, location or service).

Unlike the Yelp dataset which is relatively unaffected by this parameter, the three Amazon datasets provide contrasting trends. This phenomena could be explained by the relative heterogeneity of items sold on Amazon compared to the diversity in restaurants and bars featured on Yelp, which results in noisy features when representing items by too many attributes, but requires enough features to successfully distinguish the different items.

Analyzing the Electronics and Movies dataset results, demonstrated in Figs. 3b and 3c, shows that SARAH requires at least 30 topics to perform well, while a decline in performance is derived from additional features. DARIA displays a similar trend that is slightly skewed, achieving its best performance with only 10 topics over the Electronics data and 30 when applied in the Movies scenario. Comparing DARIA and SARAH over the Home dataset, however, exhibits opposite results. SARAH performs best as more features are added, while applying too many features in DARIA leads to a sharp decline. This contradiction may be due to the relatively small number of samples per item in the Home dataset. Given the lack of sufficient historical items' data, SARAH requires additional features to better represent an item. In contrast, features in DARIA are combined with the user input, allowing more data to be used when only few features are available.

6.2.2 User k Items

Similar to the number of item features, different values of k effect each user's input, as this hyperparameter represents the number of items used to describe her in every interaction. Since the items defining a user cannot be used to optimize the model, this parameter further impact the training set size. As the minimal number of training features for every user is 13 items overall, we tested k values up to 12 items, resulting in users with one sample in the training, validation and test sets. We followed the results of previous experiment and applied 50 features and 50 latent weights for each item, with no RNN layer. The potential importance of this parameter can be seen in Fig. 4, as more items improve the performance over the Electronics and Movies Amazon datasets, until reaching a specific number of items where the training size turns to be insufficient. For the Home dataset however, the reported effect of this hyperparameter is similar to the impact of the number of features hyperparameter. As higher k values improve SARAH's predictions over the Home data, having more than 6 items to describe each user results in sub-optimal performance for DARIA, possibly due to the higher volume of training instances required to optimize the

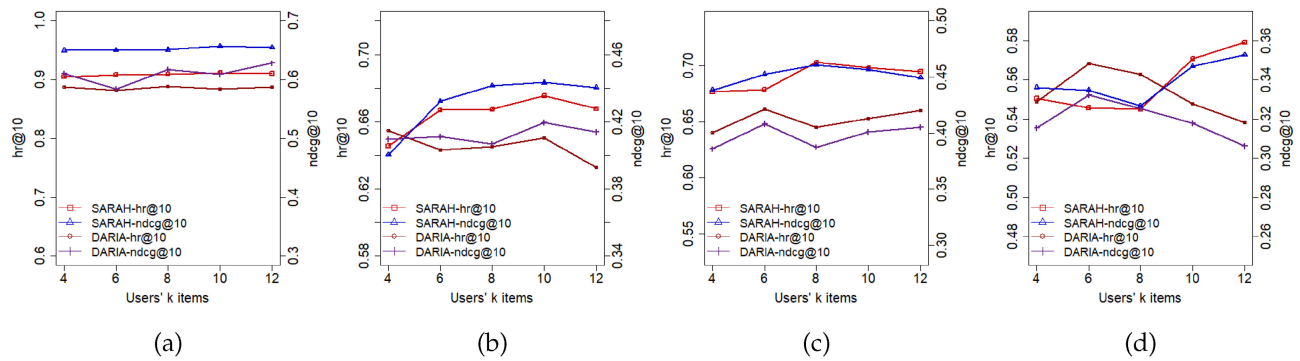

Fig. 4. HR@10 and NDCG@10 over k for (a) the Yelp, (b) electronics, (c) movies, and (d) home datasets.

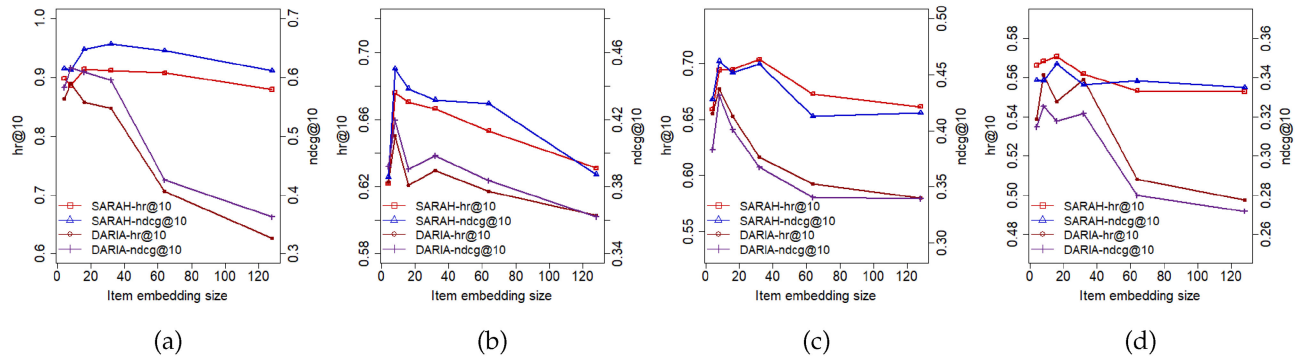

Fig. 5. HR@10 and NDCG@10 over item embedding size for (a) the Yelp, (b) electronics, (c) movies, and (d) home datasets.

model. On the Yelp dataset, where more samples per user are available on average, the two frameworks are more resilient to changes in this value.

6.2.3 Item Embedding Size

Employed both to construct the input for the item self-attention in SARAH and to represent items a user interacted with, the number of weights used to describe an item has the potential to impact SARAH's two self-attention components. This hyperparameter is equally significant for DARIA, as it determines which past items should be ignored. Moreover, the item embedding size also determines DARIA's RNN dimensionality, effecting the user representation as well as the item. We have tested this hyperparameter with values ranging between 4 and 128 units, while applying 50 features and 10 past items as input, with no temporal context. As illustrated in Fig. 5, increasing the number of variables representing items improves the two models' performance significantly for most datasets, until reaching an optimal value found in the range of 16-32 units, beyond which the prediction capability decreases. In fact, optimizing this hyperparameter has had the most apparent outcome over our frameworks' performance. While in SARAH its effect is relatively moderated due to the use of separate RNN dimensionality to represent users, describing items with too many latent weights significantly deteriorate DARIA's recommendation capabilities.

6.2.4 RNN Output Weights

The number of units used to construct SARAH's RNN output layer influences each of the k user items' embedding size after being infused by sequential data. Along with impacting the model's ability to represent each item and its context for the target user, this hyperparameter also determines the final user embedding size used to estimate the output rating. We tested values in the range of 10 to 90 units, as well as when not using RNN (number of weights is 0), to determine the potential importance of sequentiality in the model. The additional hyperparameters were set to 50, 10 and 16 for the number of features, past items and item embedding. As illustrated in Fig. 6, utilizing contextual data brings great improvement to the model and allows it to reach its optimal scores. While values above 80 units result in decreased performance in most reported scenarios, the positive impact of the RNN layer is clearly visible even when items are represented using only 10 hidden units. For DARIA, however, the RNN dimensionality cannot be modified as it has to be equal to the item embedding size. Since this analysis was previously discussed, we did not include DARIA's performance in this evaluation.

6.3 Case Studies (RQ3)

In this section we evaluate SARAH's capability in providing explainable representations for items and users. These can accompany the model's output either as weight distributions or as messages of the sort: *"We suggest you an item best described by the following features..."* and *"This item is recommended due to these items you previously liked..."*. The given explanations' relevance is visualized using heat maps over randomly sampled items and users from the Amazon Movies dataset.

6.3.1 Item Features Explanations

In adopting items' content for explanation, our proposed framework is able not only to report which items in the user

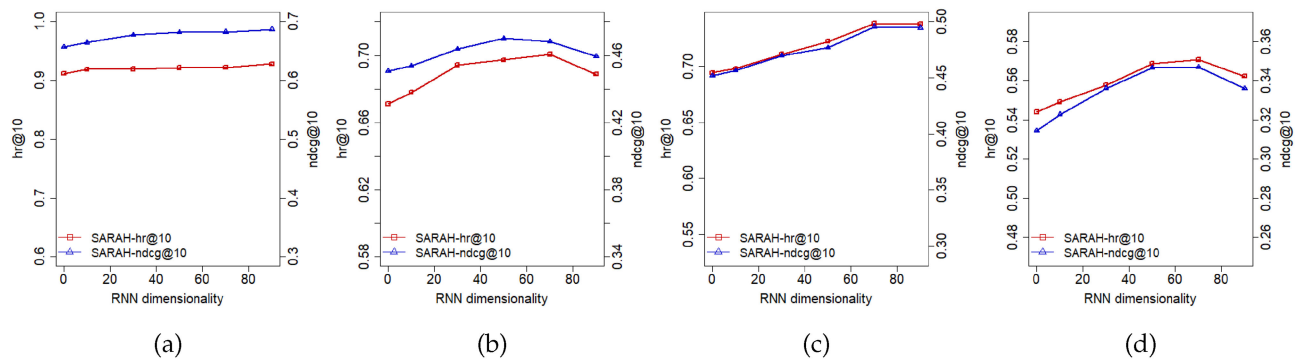

Fig. 6. HR@10 and NDCG@10 over RNN output layer size for (a) the Yelp, (b) electronics, (c) movies, and (d) home datasets.

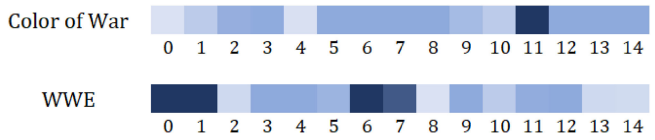

Fig. 7. Visualization of two items' attention scores based on 15 features. Darker colors denote higher weights.

history contributed to the recommendation, but also what features of the item define it best, allowing the user to make an informed decision given a list of top-10 suggestions. By utilizing attention scores, SARAH can weight features from different inputs and types seamlessly, and report these finding to the end user. Furthermore, by not having additional nonlinearities on top of the attention weights, these features have direct and full impact on the item representation and predicted output. To demonstrate the model's capability in providing meaningful feature distributions, heat maps over the attention scores generated in Eq. (14) for two randomly sampled items are presented in Fig. 7.

Employing attention weights can provide relevant information to the end user, allowing her to choose an item based on temporal preferences and not only by following historical patterns. For example, "Color of War" was recommended mainly due to the image quality, outlined by the 11th feature. WWE, on the other hand, is strongly represented by being appropriate to the whole family, being humoristic and its characters, denoted by the features in positions 0,1 and 6, respectively. Accompanying this type of data with the given suggestions allows users to select the items that answer their needs best, without further exploring the given recommendations.

6.3.2 User History Explanations

We further analyze the effect different past items have on a user representation, as illustrated in Fig. 8 using heat maps. First, we provide the importance of 10 past items for a randomly sampled user, u, in Fig. 8a. The presented map shows the attention values generated in Section 4.2, where darker colors denote higher impact of an item over the users' behavior. However, we wish to further delve into the effect these weights have in the process of item recommendation. To this end, we sample a positive item, i, and measure the similarity between each of the user k items' embeddings and that of the candidate item:

$$d(i) = [d(e_i, e_{u1}), d(e_i, e_{u2}), \ldots, d(e_i, e_{uk})] \ , \tag{21}$$

where d is the euclidean distance and e_* a past item embedding as presented in Section 4.1. The sample

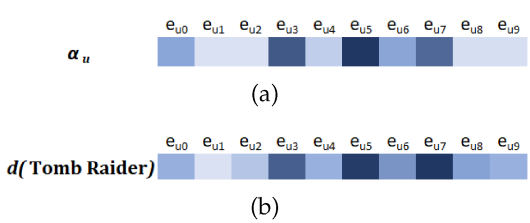

Fig. 8. Visualization of user attention scores and euclidean distance between past items and a positive item.

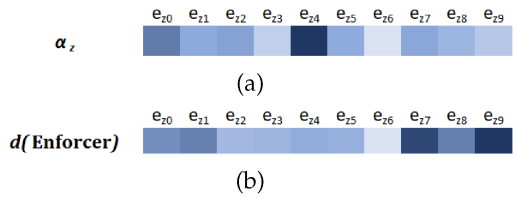

Fig. 9. Visualization of user attention scores and euclidean distance between past items and a negative item.

outcome of Eq. (21) for the movie "Tomb Raider" liked by user u is shown in Fig. 8b, where darker colors represent higher similarity. As can be witnessed by the two heat maps, there is an apparent correlation between corresponding columns. The items that best capture the user's behavior, denoted as e_{u3}, e_{u5} and e_{u7}, are also the most similar to the liked movie.

We conduct the same experiment over a negative example, provided in Fig. 9. The corresponding heat maps now represent the past items' importance for user z and the similarity of each past item to a movie the user did not like, "Enforcer". In the negative case, however, there is a clear discordance between the two heat maps. "Beauty and the Beast", the movie that best represent the user, denoted as e_{z4}, is relatively different from the candidate movie, which is a thriller. To conclude, the provided visualizations demonstrate that the past items best describing the user are closely related to items she likes, while there is no such relation to other items.

7 Conclusion and Future Work

In this paper we introduced DARIA and SARAH, two frameworks for interpretable recommendation. While DARIA applies two dependable neural attention components to select past items that are most similar to the recommended item and the features that contributed to this similarity, SARAH adopts the self-attention technique to identify the most relevant features representing items as well as determining the importance of past items in a user's history. The use of the two different attention paradigms allows our proposed methods to not only yield relevant item recommendations, but also to provide explanations in the form of features that best describe the recommended item and how they perceive the end user. Evaluated over four datasets, DARIA and SARAH were found to significantly outperform seven varied baselines in terms of hit ratio and ndcg. Furthermore, SARAH was able to improve over DARIA in all scenarios, demonstrating the potential of self-attention in recommender systems.

In future work, SARAH could be made interactive, allowing the user to adjust her personal attention weights and modify the way she is perceived by the system, making it more compatible with her interests.

Acknowledgments

This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada and the National Natural Science Foundation of China (NSFC 61572289).

REFERENCES

[1] H.-T. Cheng *et al.*, "Wide and deep learning for recommender systems," in *Proc. 1st Workshop Deep Learn. Recommender Syst.*, 2016, pp. 7–10.

[2] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for youtube recommendations," in *Proc. 10th ACM Conf. Recommender Syst.*, 2016, pp. 191–198.

[3] H. Yin, Y. Sun, B. Cui, Z. Hu, and L. Chen, "LCARS: A location-content-aware recommender system," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2013, pp. 221–229.

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

[5] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2011, pp. 448–456.

[6] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. 26th ACM Int. Conf. World Wide Web*, 2017, pp. 173–182.

[7] Y. Gong and Q. Zhang, "Hashtag recommendation using attention-based convolutional neural network," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 2782–2788.

[8] C. Chen, M. Zhang, Y. Liu, and S. Ma, "Neural attentional rating regression with review-level explanations," in *Proc. World Wide Web Conf.*, 2018, pp. 1583–1592.

[9] X. Wang, X. He, F. Feng, L. Nie, and T.-S. Chua, "TEM: Tree-enhanced embedding model for explainable recommendation," in *Proc. World Wide Web Conf.*, 2018, pp. 1543–1552.

[10] J. Chen, F. Zhuang, X. Hong, X. Ao, X. Xie, and Q. He, "Attention-driven factor model for explainable personalized recommendation," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2018, pp. 909–912.

[11] B. Xia, Y. Li, Q. Li, and T. Li, "Attention-based recurrent neural network for location recommendation," in *Proc. 12th Int. Conf. Intell. Syst. Knowl. Eng.*, 2017, pp. 1–6.

[12] W. Pei, J. Yang, Z. Sun, J. Zhang, A. Bozzon, and D. M. Tax, "Interacting attention-gated recurrent networks for recommendation," in *Proc. ACM Conf. Inf. Knowl. Manage.*, 2017, pp. 1459–1468.

[13] M. Fu, H. Qu, D. Moges, and L. Lu, "Attention based collaborative filtering," *Neurocomputing*, vol. 311, pp. 88–98, 2018.

[14] X. Chen *et al.*, "Sequential recommendation with user memory networks," in *Proc. 11th ACM Int. Conf. Web Search Data Mining*, 2018, pp. 108–116.

[15] H. Wang, N. Wang, and D.-Y. Yeung, "Collaborative deep learning for recommender systems," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2015, pp. 1235–1244.

[16] Y. Tay, L. A. Tuan, and S. C. Hui, "Multi-pointer co-attention networks for recommendation," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2018, pp. 2309–2318.

[17] Z. Lin *et al.*, "A structured self-attentive sentence embedding," in *Proc. 5th Int. Conf. Learn. Representations*, 2017.

[18] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Advances Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[19] X. Luo, M. Zhou, Y. Xia, and Q. Zhu, "An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems," *IEEE Trans. Ind. Inform.*, vol. 10, no. 2, pp. 1273–1284, May 2014.

[20] D. Kim, C. Park, J. Oh, S. Lee, and H. Yu, "Convolutional matrix factorization for document context-aware recommendation," in *Proc. 10th ACM Conf. Recommender Syst.*, 2016, pp. 233–240.

[21] T. Ebesu, B. Shen, and Y. Fang, "Collaborative memory network for recommendation systems," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2018, pp. 515–524.

[22] W.-C. Kang and J. J. McAuley, "Self-attentive sequential recommendation," in *Proc. IEEE Int. Conf. Data Mining*, 2018, pp. 197–206.

[23] Y. He, C. Wang, and C. Jiang, "Correlated matrix factorization for recommendation with implicit feedback," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 3, pp. 451–464, Mar. 2019.

[24] O. Tal and Y. Liu, "A joint deep recommendation framework for location-based social networks," *Complexity*, vol. 2019, pp. 2 926 749:1–2 926 749:11, 2019. [Online]. Available: https://doi.org/10.1155/2019/2926749

[25] L. Zheng, V. Noroozi, and P. S. Yu, "Joint deep modeling of users and items using reviews for recommendation," in *Proc. 10th ACM Int. Conf. Web Search Data Mining*, 2017, pp. 425–434.

[26] T. Bansal, D. Belanger, and A. McCallum, "Ask the GRU: Multi-task learning for deep text recommendations," in *Proc. 10th ACM Conf. Recommender Syst.*, 2016, pp. 107–114.

[27] C.-Y. Wu, A. Ahmed, A. Beutel, A. J. Smola, and H. Jing, "Recurrent recommender networks," in *Proc. 10th ACM Int. Conf. Web Search Data Mining*, 2017, pp. 495–503.

[28] A. Almahairi, K. Kastner, K. Cho, and A. Courville, "Learning distributed representations from reviews for collaborative filtering," in *Proc. 9th ACM Conf. Recommender Syst.*, 2015, pp. 147–154.

[29] Q. Liu, S. Wu, D. Wang, Z. Li, and L. Wang, "Context-aware sequential recommendation," in *Proc. IEEE 16th Int. Conf. Data Mining (ICDM)*, 2016, pp. 1053–1058.

[30] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 11, pp. 1254–1259, Nov. 1998.

[31] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu, "Recurrent models of visual attention," in *Proc. Advances Neural Inf. Process. Syst.*, 2014, pp. 2204–2212.

[32] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. 3rd Int. Conf. Learn. Representations*, 2015. [Online]. Available: http://arxiv.org/abs/1409.0473

[33] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," in *Proc. SSST@EMNLP 2014, Eighth Workshop Syntax, Semantics Struct. Statist. Transl.*, 2014, pp. 103–111, doi: 10.3115/v1/W14-4012.

[34] R. Paulus, C. Xiong, and R. Socher, "A deep reinforced model for abstractive summarization," *CoRR*, vol. abs/1705.04304. 2017. [Online]. Available: http://arxiv.org/abs/1705.04304

[35] T. Shen, T. Zhou, G. Long, J. Jiang, S. Pan, and C. Zhang, "DiSAN: Directional self-attention network for RNN/CNN-free language understanding," 2017, *arXiv:1709.04696*.

[36] J. Song *et al.*, "Hierarchical contextual attention recurrent neural network for map query suggestion," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 9, pp. 1888–1901, Sep. 2017.

[37] Y. Tay, L. Anh Tuan, and S. C. Hui, "Latent relational metric learning via memory-based attention for collaborative ranking," in *Proc. World Wide Web Conf.*, 2018, pp. 729–739.

[38] H. Wang, F. Zhang, X. Xie, and M. Guo, "DKN: Deep knowledge-aware network for news recommendation," in *Proc. World Wide Web Conf.*, 2018, pp. 1835–1844.

[39] X. He, Z. He, J. Song, Z. Liu, Y.-G. Jiang, and T.-S. Chua, "NAIS: Neural attentive item similarity model for recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 12, pp. 2354–2366, Dec. 2018.

[40] C. Zhou *et al.*, "ATRank: An attention-based user behavior modeling framework for recommendation," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018.

[41] S. Yun, R. Kim, M. Ko, and J. Kang, "SAIN: Self-attentive integration network for recommendation," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 1205–1208.

[42] S. Zhang, Y. Tay, L. Yao, and A. Sun, "Next item recommendation with self-attention," *CoRR*, vol. abs/1808.06414. 2018. [Online]. Available: http://arxiv.org/abs/1808.06414

[43] J. L. Herlocker, J. A. Konstan, and J. Riedl, "Explaining collaborative filtering recommendations," in *Proc. ACM Conf. Comput. Supported Cooperative Work*, 2000, pp. 241–250.

[44] V. Bellini, A. Schiavone, T. Di Noia, A. Ragone, and E. Di Sciascio, "Knowledge-aware autoencoders for explainable recommender sytems," in *Proc. ACM 3rd Workshop Deep Learn. Recommender Syst.*, 2018, pp. 24–31.

[45] C. Musto, F. Narducci, P. Lops, M. de Gemmis, and G. Semeraro, "Linked open data-based explanations for transparent recommender systems," *Int. J. Human-Comput. Stud.*, vol. 121, pp. 93–107, 2019.

[46] L. Shi, Z. Teng, L. Wang, Y. Zhang, and A. Binder, "DeepClue: Visual interpretation of text-based deep stock prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 6, pp. 1094–1108, Jun. 2019.

[47] J. Huang, W. X. Zhao, H. Dou, J.-R. Wen, and E. Y. Chang, "Improving sequential recommendation with knowledge-enhanced memory networks," in *Proc. 41st ACM Conf. Res. Develop. Inf. Retrieval*, 2018, pp. 505–514.

[48] Z. Sun, J. Yang, J. Zhang, A. Bozzon, L.-K. Huang, and C. Xu, "Recurrent knowledge graph embedding for effective recommendation," in *Proc. 12th ACM Conf. Recommender Syst.*, 2018, pp. 297–305.

[49] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos, "MoviExplain: A recommender system with explanations," in *Proc. 3rd ACM Conf. Recommender Syst.*, 2009, pp. 317–320.

[50] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Representations*, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980
[51] J. McAuley, C. Targett, Q. Shi, and A. Van Den Hengel, "Image-based recommendations on styles and substitutes," in *Proc. 38th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2015, pp. 43–52.
[52] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, 2003.
[53] C. Dos Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts," in *Proc. 25th Int. Conf. Comput. Linguistics: Tech. Papers*, 2014, pp. 69–78.
[54] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," in *Proc. Advances Neural Inf. Process. Syst.*, 2014, pp. 2177–2185.
[55] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2014, pp. 1532–1543.

Omer Tal is working toward the graduate degree in the Department of Physics and Computer Science, Wilfrid Laurier University. His research interests include the areas of neural networks and recommender systems. He is a member of the IEEE.

Yang Liu is an associate professor with the Department of Physics and Computer Science, Wilfrid Laurier University. Her main research interests include social network analysis and urban computing. She has published more than 50 papers in top conferences and journals, including SIGIR, WWW, the *IEEE Transactions on Knowledge Data Engineering*, TIST, JASIST, and IPM, and has served on the program committees of conferences such as WWW, CIKM, WI-IAT, and WAIM. She is a member of the IEEE.

Jimmy Huang received the PhD degree in information science from City, University in London, United Kingdom. He is now a york research chair professor and the director of Information Retrieval & Knowledge Management Research Lab, York University. Since 2003, he has published more than 230 refereed papers in journals and international conference proceedings, such as the *ACM Transactions on Information Systems*, *Journal of the Association for Information Science and Technology*, *Information Processing and Management*, *IEEE Transactions on Knowledge Data Engineering*, SIGIR, CIKM, KDD, ICDM, ACL, IJCAI, and AAAI. He is a senior member of the IEEE and ACM distinguished scientist.

Xiaohui Yu received the PhD degree in computer science from the University of Toronto, Canada, in 2006. His research interests include the areas of database systems and data mining. He is an associate professor with the School of Information Technology, York University, Toronto. He is a member of the IEEE.

Bushra Aljbawi is working toward the graduate degree in the Department of Physics and Computer Science, Wilfrid Laurier University. Her research interests include the areas of recommender systems, machine learning, and natural language processing. She is a member of the IEEE.