Information Retrieval Journal (2022) 25:306–328
https://doi.org/10.1007/s10791-022-09410-1

Learning user preferences through online conversations via personalized memory transfer

Nagaarchana Godavarthy[1] · Yuan Wang[1] · Travis Ebesu[2] · Un Suthee[3] · Min Xie[4] · Yi Fang[1]

Received: 10 March 2021 / Accepted: 14 May 2022 / Published online: 26 June 2022

Abstract
Dialogue systems are becoming an increasingly common part of many users' daily routines. Natural language serves as a convenient interface to express our preferences with the underlying systems. In this paper, we aim to learn user preferences through online conversations. Compared to the traditional collaborative filtering setting where feedback is provided quantitatively, conversational users may only indicate their preferences at a high level with inexact item mentions. To tackle the ambiguities in natural language conversations, we propose Personalized Memory Transfer which learns a personalized model in an online manner by leveraging a key-value memory structure to distill user feedback directly from conversations. This memory structure enables the integration of prior knowledge to transfer existing item representations/preferences and natural language representations. The experiments were conducted on two public datasets and the results demonstrate the effectiveness of the proposed approach.

Un Suthee:The work was done while he was at Walmart Labs.

✉ Yi Fang
yfang@scu.edu

Nagaarchana Godavarthy
ngodavarthy@scu.edu

Yuan Wang
ywang4@scu.edu

Travis Ebesu
tebesu@gmail.com

Un Suthee
sutheec@amazon.com

Min Xie
min.xie@instacart.com

[1] Santa Clara University, Santa Clara, CA, USA

[2] Pinterest, Inc., San Francisco, CA, USA

[3] Amazon Web Services, San Francisco, CA, USA

[4] Instacart, San Francisco, CA, USA

Keywords Conversational recommendation · Memory networks · Transfer learning

1 Introduction

Virtual assistants such as Amazon Alexa, Apple Siri, Google Assistant, and Microsoft Cortana are becoming an increasingly common part of many users' daily routines. Conversational agents are being quickly adopted in industry to handle customer service requests at banks, set up travel accommodations, and make product recommendations at online retailers. Natural language serves as a convenient interface to computing systems and is a natural form to express our preferences with others. Consequently, conversational recommendation systems have emerged to elicit the dynamic preferences of users through multi-turn interactions in natural language.

In the setting of conversational recommendations, two parties are interacting with one another centered around discussing items, e.g., movies (Li et al., 2018). The first party (seeker) expresses his/her preferences and asks for relevant movie suggestions from the second party who acts as the recommender. This recommender's goal is to understand the seeker and provide personalized movie recommendations based on the conversation. An example dialogue is shown in Fig. 1. The conversational recommendations setting poses ambiguity where the user expresses feedback in the form of natural language, in contrast to the traditional collaborative filtering setting where feedback is typically provided in explicit (ratings) or implicit (clicks/views) form. The seeker may not specify which item(s) he or she likes but rather describes it at a high level such as "I would like to watch a suspenseful, but clean family friendly movie" in Fig. 1. Even when the user makes specific references, items or movies mentioned in the dialogue may not be exactly accurate. For example, *A Space Odyssey* and *Star Wars: The Last Jedi* are not the full names of the movies (which should be *2001: A Space Odyssey (1968)* and *Star Wars: Episode VIII - The Last Jedi (2017)* respectively based on the IMDB database). Also, the *Star Wars* that the seeker mentioned is a movie franchise and cannot be mapped to a specific movie. Some prior work on conversational recommendations used

Seeker:	Hi
Rec:	Hello
Seeker:	How are you? I would like to watch a suspenseful, but clean family friendly movie.
Rec:	Oh! I love suspenseful movies. Have you seen **The Illusionist**, it is pg-13 however
Seeker:	We have not seen that. We enjoyed **A Space Odyssey**
Rec:	If you like **A Space Odyssey**, you can also try **Star Wars: The Last Jedi** for something G Rated
Seeker:	Honestly I'm not a big fan of **Star Wars**. I mean they are great movies but for some reason i just don't like it
Rec:	You can try **Planet of the Apes** the older one is quite suspenseful and family friendly.
Seeker:	Those sound good! I'm going to look into those movies.
Rec:	I hope you enjoy, have a nice one.
Seeker:	Thank you for your help! Have a great night! Good bye

Fig. 1 Example dialogue between the Seeker asking for recommendations and the Recommender providing suggestions. Movie mentions are in bold

synthetic data (Dodge et al., 2016; Suglia et al., 2017; Lei et al., 2020) or assumed an entity tagger was present mapping movie mentions in the dialogue to unique identifiers (Li et al., 2018) which may not be available in practice.

An ideal conversational recommendation system would learn to adapt to user's preferences as the conversation progresses. Inspired by key-value memory networks (Miller et al., 2016) originally proposed for question answering and the successful application of memory networks to the recommender domain (Ebesu et al., 2018; Hu et al., 2018), we propose Personalized Memory Transfer (PMT) to learn user preferences in a natural chit-chat conversational setting. To tackle the ambiguity in natural language, we treat each conversation as a virtual item by combining the known item representations. As the conversation progresses, PMT updates the user representation based on the observed virtual items. Since collecting large amounts of conversational data with labels may be cost prohibitive or infeasible due to privacy concerns, we propose to leverage two forms of transfer learning to address data sparsity.

First, we learn an interaction function measuring the user's level of interest in a given item from a large-scale collaborative filtering dataset (e.g. MovieLens (Harper & Konstan, 2015)) and transfer the learned movie representations to the conversational recommendation domain. Next, each movie is represented by a memory slot with a corresponding pair of key and value memory in the key-value memory structure. This memory structure is tailored such that the model uses the keys to address relevant memories with respect to the current user's utterance (query) followed by a reading phase which returns the final output memory using the value memory which serves as a virtual item. A pre-trained natural language encoder maps each movie's plot to a low-dimensional representation and the learned item preferences act as the basis for the key and value memories, respectively. During the conversation the user's utterance is mapped to a low-dimensional semantic vector with the natural language encoder which addresses the key memories by identifying relevant movies for the user. For each conversation we learn a new user representation in an online manner by leveraging the virtual item as a positive instance along with considering user's satisfaction to evaluate the pairwise objective. As the conversation progresses PMT updates the user representation with standard backpropagation via stochastic gradient descent guided by the virtual item extracted and sentiment from the user's utterance permitting personalized movie recommendations.

It is worth noting that a full-fledged conversational recommendation system should include a response generation component, which is to generate human-understandable responses for communicating with users and making recommendations. There exist multiple strategies and active research on how to generate readable, fluent, and consistent natural language responses (Gao et al., 2021). In this paper, we only focus on the task of learning user preferences over conversations, which is a challenging problem in itself. The proposed approach can be used in combination with the existing response generation methods. Our contributions can be summarized as follows:

- To the best of our knowledge, this is the first work that learns user preferences in an online fashion from conversations where there may only exist high level user feedback or inexact item mentions.
- We propose a novel memory network named Personalized Memory Transfer (PMT) to distill user feedback directly from user's utterances by integrating prior knowledge of existing item representations/preferences and pre-trained language models, which allows the learning of a new user representation over conversations.

– Experimental results on two public conversational datasets demonstrate the effectiveness of PMT to learn a personalized user representations with two interaction functions and natural language encoders in an online setting.

The remainder of the paper is organized as follows: Section 2 introduces the research status and related work. Section 3 presents the proposed approach. Section 4 demonstrates and analyzes our experimental results. Finally, Sect. 5 concludes the paper with a discussion on future work.

2 Related work

2.1 Conversational recommendation systems

Recommendation systems are vital to keeping users engaged and satisfied with personalized recommendations in the age of information explosion. Modern E-commerce, entertainment and social media platforms provide personalized content by analyzing user preferences based on explicit and/or implicit feedback and infer their potentially preferred items. Based on the type of input data, the recommender systems can be summarized into collaborative filtering systems, content-based recommender systems, and hybrid recommender systems (Zhang et al., 2019). Early researches in collaborative filtering formulated the recommendation task as predicting the user rating score on the candidate items (Sarwar et al., 2001). The rating-based recommendation models did not perform well in top-n recommendation, which motivated the ranking-based recommendation by learning a model based on the relative preference of a user over pairs of items (Rendle et al., 2009b). Content-based recommendation utilizes item or user features (such as item description, user profiles, and attributes) to find similar items or users for recommendations (Lops et al., 2011).

Most conventional recommender systems highly rely on users' historical trajectories to generate personalized recommendation. It lacks the capability of capturing users' dynamic intentions/demands. This intrinsic limitation motivates online recommendation with its goal to adapt the recommendation results with the user's online actions (Li & Karahanna, 2015). Much existing work models it as a multi-arm bandit problem (Wang et al., 2017; Wu et al., 2018). While achieving remarkable progress, the bandit-based solutions are still insufficient especially in the warm start scenarios (Lei et al., 2020). In the recent years, conversational recommender systems have attracted an increased attention in the research community as they enable a system to interact with users using natural language. The general idea of such systems is to support a task-oriented, multi-turn dialogue with their users.

Sun and Zhang (2018) represented a dialogue vector as a set of facet-value pairs and fuses the vector with user and item ratings via a factorization machine. Li et al. (2018) assumed that movies mentioned in the conversation are known in advance. The model encodes the incoming utterances and progressively constructs the input vector to the user-based autoencoder to predict ratings. Our work differs from the existing works as our model does not need any prior knowledge of users. Unlike (Li et al., 2018), our work does not need to tag a movie directly in the conversation. A similar but related approach is interactive recommendation which elicits specific questions to the user regarding their preferences but may diminish user experience (Christakopoulou et al., 2016, 2018; Zhu et al., 2019). For a comprehensive literature review on conversational recommendation systems, readers can refer to two recent survey articles (Jannach et al., 2021; Gao et al., 2021).

2.2 Task-oriented dialog system

A task-oriented dialog system's intent is to assist users with a given task through natural language such as hotel booking, travel or online shopping. Task-oriented dialogue systems are one important branch in dialogue system research. Though conversational recommendation is an emerging research topic, many of the basic concepts and model designs were originated from task-oriented dialogue systems. Pioneering work by (Yan et al., 2017) leveraged natural language processing and crowdsourcing to build a dialog system for E-commerce. Bordes et al. (2017) used a memory network to store user utterances and predicts the response for a restaurant reservation task. Wen et al. (2017) modeled a task-oriented conversation as a sequence-to-sequence mapping problem while augmenting the model with the dialogue history. Mo et al. (2018) learned a personalized dialogue agent by transferring relevant knowledge from conversations via reinforcement learning framework. For non-textual utterance such as images, Cui et al. (2019) proposed a multimodal encoder which is a combination of image and text encoders to jointly transform an utterance into a dialog vector. Zhang et al. (2018) proposed the "systems ask, users respond" paradigm for conversational recommendation in e-commerce. A system agent is designed to ask users different questions to obtain clarified demands continuously, and a multi-memory network is utilized to analyze the user utterances for recommendation.

Understanding user preferences and intentions from dialogues is the key requirement for conversational recommendation or dialog systems in general, since subsequent tasks such as response generation heavily rely on this information. Much existing work focused on the multi-turn strategy and core recommendation logic (Gao et al., 2021), while they circumvented the extraction of user preferences from raw natural language utterances and often required the preprocessed input such as rating scores (Zou et al., 2020b) and YES/NO answers (Zou et al., 2020a), which is unnatural in real-world human conversations. Some recent work extracted semantic information in users' raw utterance by utilizing deep learning, e.g., Li et al. (2018) based on recurrent neural network (RNN), Liu et al. (2020) based on convolutional neural network (CNN), and Penha and Hauff (2020) based on the bidirectional encoder representations from transformers (BERT) (Devlin et al., 2019).

2.3 Cross-domain recommendation

Cross-domain recommendation seeks to transfer knowledge from a data-rich source domain and utilize it in a new target domain (Ricci et al., 2015). This may help alleviate the cold-start problem or lack of sufficient data for personalized recommendations in the new target domain. Typical cross-domain recommendation models are extended from single-domain recommendation models (Li et al., 2020). Hu et al. (2018) introduced a memory component jointly with a transfer network to selectively transfer source content information to the target domain for a given user. These approaches assume that different patterns characterize the way that users interact with items of a certain domain. However, our approach differs that it does not require users to be overlapped in the two domains and can address the user cold-start problem. Gao et al. (2019) proposed two levels of attention mechanisms to transform the interaction history of a user in the source domain as an additional user latent factor in the target domain. Their approach exploits the transferable information from the set of overlapped items which may not be feasible to apply to the conversational domain where the source and target domains are often multimodal. Another recent approach by

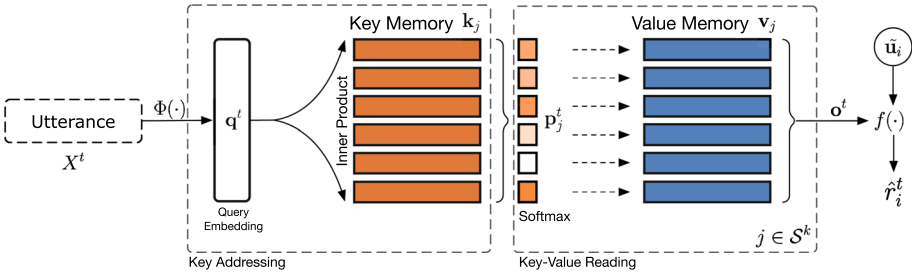

Fig. 2 Illustration of PMT encoding the user's utterance through the key-value memory structure producing the final output memory to estimate the interaction function for a new user

(Kanagawa et al., 2019) uses domain separation networks (Bousmalis et al., 2016) to train a multi-class classifier from the source domain to predict the preferred item in the target domain, but the idea may not be applicable to natural language conversations.

3 Methodology

In this section, we describe our proposed model Personalized Memory Transfer (PMT) to learn user preferences in an online conversational chit-chat setting. We use movie recommendations as a motivating example, but the proposed method is generic and can be applied to other domains as well. In this setting, there exist two types of ambiguities: (1) the user expresses preference in the form of natural language instead of explicit (ratings) or implicit (clicks/views) feedback; (2) movie mentions within the conversation may not be able to be mapped to unique item identifiers, e.g., users might simply just ask for animation movie recommendation. We tackle the aforementioned challenges by treating each conversation as a virtual item with a combination of the (pre-trained) known item representations based on their content semantic similarity obtained from a key-value structured memory (Miller et al., 2016). This virtual item can then be used as a positive instance in the pairwise training of the model.

More specifically, PMT performs online updates to a user representation throughout the conversations by consulting the stored semantic movie representation in the memory. The key-value memory structure is tailored such that the model uses the keys to address relevant memories with respect to the current user's utterance (query) followed by a reading phase which returns the output memory using the value memory. This memory structure enables the model to transfer prior knowledge of item representations/preferences and natural language to bridge two different forms of transfer learning. The output memory returned by the key-value memory structure acts as a mixture of item preferences extracted from user's natural language conversation, and it will guide the updates to the user representation with standard backpropagation via stochastic gradient descent. An overview of the process can be seen in Fig. 2. The subsections below describe the architecture in detail.

3.1 Key-value memory

One of the major components of PMT is the key-value memory. Each movie j is represented by a memory slot with a corresponding pair of key memory $\mathbf{k}_j$ and value memory

$\mathbf{v}_j$. Intuitively, the key memory represents each movie in a low-dimensional semantic space to identify similar movies according to the current user's utterance which in turn produces a relevance score weighting the appropriate value memories according to similarity. This weighted value memory or output memory acts as a weighted combination of item preferences represented in a low-dimensional latent space.

Formally, we define the key memory for the *j th* movie via embedding a sequence of words w_l corresponding to a movie's plot (or summary) $D_j = \{w_1, \ldots, w_{|D_j|}\}$ with an encoder:

$$\mathbf{k}_j = \boldsymbol{\Phi}(D_j) \tag{1}$$

where $\boldsymbol{\Phi}(\cdot)$ is a natural language encoder or feature extractor which maps a variable length textual document to a single fixed d_e dimensional semantic vector. The encoding function could be a pre-trained language model such as Universal Sentence Encoder (Cer et al., 2018), Transformer (Vaswani et al., 2017), ELMo (Peters et al., 2018), and BERT (Devlin et al., 2019). Note that this can also be used to encode the user's utterances as well. In Sect. 4.5, we investigate the effect of the natural language encoder on the recommendation performance by experimenting with different pre-trained sentence encoders. Once the encoder is pre-trained on a large corpus the parameters remain frozen during our training process. This is the first form of transfer learning in PMT. We would like to point out that the definition of the keys is not restricted to a language model representation, alternative representations such as knowledge bases (Miller et al., 2016) or additional meta information such as actors, genres, or directors may also be incorporated to achieve more objective recommendations (Kang et al., 2017).

Each value memory $\mathbf{v}_j$ is a low-dimensional latent representation of each movie j. We use the learned item representations from a pre-trained interaction function as the value memories. The pre-training can be done on a traditional large-scale collaborative filtering dataset such as MovieLens. This is the second form of transfer learning in PMT. Section 3.4 includes the details about the interaction function. Similar to the key memory, we are not restricted to setting the value memory to a specific representation from the interaction function. In addition, the flexibility of the key-value memory structure allows the key memory and value memory to be of different dimensionality.

We would like to point out that the entire memory structure is fixed and not learnable throughout the conversation. Although this work focuses on the movie domain, the representation of the keys can be easily extended to other application domains such as using product descriptions for product recommendation or paper abstracts for paper recommendation.

3.2 Key addressing

To identify movies the user may be interested in, we query the key memory by encoding the user's utterance to a low-dimensional representation producing a relevance score for each movie. Specifically, an utterance consisting of a variable length sequence of words $X^t = \{w_1, \ldots, w_{|X^t|}\}$ at a given turn t is encoded as:

$$\mathbf{q}^t = \boldsymbol{\Phi}(X^t) \tag{2}$$

where $\mathbf{q}^t$ represents the low-dimensional query embedding encoding the current user's utterance. Using the same natural language encoder as the key memory in the previous

subsection allows the movie's plot and each utterance to be mapped into the same space permitting semantic comparison in vector space (Cer et al., 2017). The model uses this query vector to identify relevant movies in the key memory the user may be interested in based on past utterances. Each memory slot is addressed by computing the similarity of the given utterance and each movie with:

$$s_j^t = (\mathbf{q}^t)^T \mathbf{k}_j \quad \forall\, j \in \mathcal{I} \tag{3}$$

where s_j^t expresses the user's level of interest in the j^{th} movie based on the encoded movie's summary and the t^{th} utterance $\mathbf{q}^t$ as measured by the inner product. $\mathcal{I}$ is the set of all movies. For example, if a user expresses interest in animated children movies, the encoded query should produce higher relevance scores to semantically related children movies in the key memory.

3.3 Key-value reading

Reading the key-value memory structure bridges the two forms of transfer learning (natural language encoder and item representations/preferences) in separate semantic spaces. At a high level, the similarity computed between the query and key memories weighs the relevant value memories. In other words, the user's utterance determines the amount of weight allocated to each movie's value memory yielding a mixture of item representations.

Including the entire set of all movies can be computationally expensive and may introduce noise to the final memory output representation. Similar to the key-hashing performed in (Miller et al., 2016), we select a subset of top-k movies with the highest similarity score according to Eq. (3) which we denote as $\mathcal{S}^k$. To obtain the final output memory we first normalize the similarity with the softmax function:

$$p_j^t = \frac{\exp(s_j^t)}{\sum_{l \in \mathcal{S}^k} \exp(s_l^t)} \quad \forall\, j \in \mathcal{S}^k \tag{4}$$

obtaining a distribution of the current user's preference over each movie $j \in \mathcal{S}^k$ from the utterance at turn t. Alternatively, we can also interpret this as an attention mechanism where the attention places higher weights on movies similar to the utterance.

Finally, the output memory representation is weighted by the probability each movie is relevant to the current user's utterance in the conversation via:

$$\mathbf{o}^t = \sum_{j \in \mathcal{S}^k} p_j^t \mathbf{v}_j \tag{5}$$

where $\mathbf{v}_j$ is the latent representation of movie j. This weighted value memory increases the impact of corresponding movies whose query and keys have high similarity in the semantic space while decreasing the influence of movies which may be irrelevant. Each value memory may represent concrete variations such as the genre of a movie or more subjective aspects such as visual aesthetics or a completely hidden and uninterpretable latent structure. By weighting each of the value memory according to its relevance we construct a weighted combination of preferences representing multiple degrees of each variation.

We now have the final memory output representation extracted from the user's utterance X^t at turn t. As discussed before, we assume that it may not be possible to map user's natural language conversation to particular movies, the memory output representation $\mathbf{o}^t$ which

is a combination from the known item representations based on their content semantic similarity obtained by the key-value memory structure serves as a good virtual "positive" item representation for the user.

3.4 Interaction function

In this section, we first present the interaction function in the classic collaborative filtering setting and then show how we can utilize it in our proposed model via pre-training.

In collaborative filtering, the interaction function measures a user's level of interest in a given item. This interaction function can estimate the scores of unobserved entries with M users and N items in the $M \times N$ ratings matrix $\mathbf{R}$, which are used to rank each item. The set of all users and items are denoted as $\mathcal{U}$ and $\mathcal{I}$, respectively. Formally, we assume the interaction function takes the following form which produces a ranking score for a given user $i \in \mathcal{U}$ and item $j \in \mathcal{I}$ as:

$$\hat{r}_{ij} = f(\mathbf{u}_i, \mathbf{v}_j) \tag{6}$$

where $\mathbf{u}_i$ and $\mathbf{v}_j$ are generally learnable parameters in a shared d dimensional space.

We can define multiple forms of the interaction function $f(\cdot)$. The parameters of all interaction functions can be learned by optimizing the pairwise ranking BPR loss (Rendle et al., 2009a) via stochastic gradient descent (SGD). The most basic interaction function is just a linear version via the inner product:

$$f(\mathbf{u}_i, \mathbf{v}_j) = \mathbf{u}_i^{\mathsf{T}} \mathbf{v}_j \tag{7}$$

which can yield one of the early influential algorithms in collaborative filtering: Singular Value Decomposition (SVD) (Salakhutdinov & Mnih, 2008), and other related matrix factorization (MF) based methods.

A linear function may lack the flexibility to disentangle complex item preferences and generalize to another dataset. Hence we can use a nonlinear variant generalized matrix factorization (GMF) (He et al., 2017):

$$f(\mathbf{u}_i, \mathbf{v}_j) = \mathbf{h}^{\mathsf{T}} \phi(\mathbf{u}_i \odot \mathbf{v}_j) \tag{8}$$

where $\odot$ is the elementwise product; $\mathbf{h} \in \mathbb{R}^d$ is an additional parameter to be learned and $\phi(\cdot)$ is a nonlinear activation function. We can adopt the rectified linear unit (ReLU) function $\phi(x) = \max(0, x)$ as the nonlinear function due to its nonsaturating behavior (Nair & Hinton, 2010). GMF degenerates to matrix factorization if we set the nonlinear activation function $\phi(\cdot)$ to the identity function and constrain the vector $\mathbf{h}$ to the $\mathbb{1}$ vector with all values as 1.

In our work, we use the interaction function introduced above (either MF or GMF) as the last layer of the proposed architecture, as illustrated in Fig. 2. We can estimate the ranking score of the output $\mathbf{o}^t$ in Eq.(5) for the user i using the interaction function as below

$$\hat{r}_i^t = f(\tilde{\mathbf{u}}_i, \mathbf{o}^t) \tag{9}$$

where $\tilde{\mathbf{u}}_i$ is the user representation for the utterance at turn t. As discussed in Sect. 3.3, $\mathbf{o}^t$ is a combination from the known movie representations $\mathbf{v}_j$ and can be viewed as a virtual movie representation of the utterance X^t at turn t.

Inspired by the recent success of pre-trained models in many domains, we noticed that both latent representation of movie $\mathbf{v}_j$ and interaction function $f(\cdot)$ can be pre-trained on the existing large-scale movie rating datasets such as MovieLens. Thus, we do not have to estimate $\mathbf{v}_j$ and $\mathbf{o}^t$, if there is no fine tuning. The only parameter that needs to learn is the user representation $\tilde{\mathbf{u}}_i$. During the conversation, user representation $\tilde{\mathbf{u}}_i$ is dynamically updated based on each new utterance represented by a combination of the known movies. The next section explains the loss function and the optimization procedure in model training.

3.5 Learning the user preference vector

Since the nature of the data resembles the implicit feedback setting, we take the pairwise assumption that a given user i prefers the observed item j^+ over the unobserved item j^-. However, without previous user interactions we cannot use standard techniques to perform the sampling. Instead, the positive item is inferred from the conversation via reading the key-value memory structure which produces a virtual item representation as $\mathbf{o}^t$ from the utterance X^t. Since users may express positive or negative preferences towards particular recommendations throughout the conversation, we integrate a sentiment classifier $y = g(X^t) \in [0, 1]$ which handles the uncertainty associated with the negative sampling procedure. For each turn in the conversation, our training data consists of a tuple for each user utterance $X^t \in \mathcal{X}$ and sampled negative item j^- which are used to minimize the cross-entropy objective:

$$\mathcal{L}(X^t, j^-) = -y \log(\sigma(\hat{y})) - (1 - y) \log(1 - \sigma(\hat{y})) \tag{10}$$

where $\sigma(x) = 1/(1 + \exp(-x))$ is the logistic sigmoid function and $\hat{y} = \hat{r}_i^t - \hat{r}_{ij^-}$.

The intuition is as follows, we want to maximize the relevance score $\hat{r}_i^t$ of the weighted combination of movie preferences extracted by the current utterance X^t over the relevance score $\hat{r}_{ij^-}$ from the sampled negative movie j^- assuming the sentiment is positive i.e. $y = 1$. Conversely, if the sentiment is negative the update to the user preference vector is performed in the opposite direction. In reality, utterances have varying levels of sentiment with some recommendations being favored over others, allowing y to act as a gate determining the level of satisfaction the user has with the recommendation. The only parameter learned is the new user's latent factor $\tilde{\mathbf{u}}_i$ while all other parameters are held fixed. The loss function is optimized in an online fashion using stochastic gradient descent (SGD) which allows updating the user representation as the conversation progresses. Parameter updates are performed after each seeker's utterance. Note only the utterances require the encoder function $\boldsymbol{\Phi}(\cdot)$ during the online setting and the movie summaries can be encoded and cached ahead of time.

3.6 Recommendation

Since the final output memory representation acts as a virtual item extracted from the natural language conversation, it does not directly correspond to a single item and therefore cannot be used to recommend items. The final output memory is only used to evaluate the loss function with respect to the interaction function. To perform recommendations, the newly learned user representation is plugged into the interaction function. The ranking score for the new user i and item j is:

$$\hat{r}_{ij} = f(\tilde{\mathbf{u}}_i, \mathbf{v}_j) \tag{11}$$

Note that here we use the newly learned user representation $\tilde{\mathbf{u}}_i$ and the true item representation $\mathbf{v}_j$ (not the estimated output memory). The top-n movies with the highest ranking scores can be presented to the user. Algorithm 1 shows the procedure on performing the recommendations in PMT. In the experiments, we evaluate these top-n recommendations against the item that the Seeker actually liked at turn t. As pointed out in Sect. 1, the focus of this paper is on learning user preferences from conversations. In future work, we will explore to generate natural language responses for communicating with Seeker and making recommendations.

Input: Conversation $\mathcal{X}$, Encoder $\Phi(\cdot)$, interaction function $f(\cdot)$, learning rate α

Randomly initialize new user latent factor $\tilde{\mathbf{u}}_i$
for $X^t \in \mathcal{X}$ **do**
  if *Speaker* == *Seeker* **then**
    Encode utterance $\mathbf{q}^t \leftarrow \Phi(X^t)$
    Compute output memory $\mathbf{o}^t$ (Eq. (5))
    Compute ranking score $\hat{r}_i^t$ (Eq. (9))
    Sample negative item j^-
    Update $\tilde{\mathbf{u}}_i \leftarrow \tilde{\mathbf{u}}_i - \alpha \nabla_{\tilde{\mathbf{u}}_i} \mathcal{L}(X^t, j^-)$
  else *Speaker*==*Recommender*
    Recommend Movies (Eq. (11))
  end
end

Algorithm 1: Procedure for performing recommendations in PMT

4 Experiments

4.1 Datasets

We validate our proposed methodology on two public conversational datasets. The first one is called Recommendations through Dialogue (ReDial)[1] dataset (Li et al., 2018) which consists of 206,102 utterances in 11,348 dialogues. The dialogues were crowd sourced and collected from Amazon Mechanical Turk where two users are paired up to converse around the topic of movies. Each user plays a specific role. The first user known as the seeker tries to explain their movie preferences and asks for appropriate movie suggestions. The second user known as the recommender tries to understand the seeker and provide appropriate movie recommendations. There are a total of 956 users and 51,699 movie mentions. The second dataset is called Goal-driven Recommendation Dialogue (GoRecDial) which includes 9,125 dialogues and 170,904 utterances between pairs of human workers recommending movies to each other. It was collected through the task specifically designed as a

[1] https://redialdata.github.io/website/

cooperative game between two players working toward a quantifiable common goal. We refer the readers to (Kang et al., 2019) for full details.

Movie plots are collected from IMDB[2] and the interaction function is pre-trained using the latest version of MovieLens (Harper & Konstan, 2015) consisting of 27M ratings from 283K users over 53K items. We split 80% of the ratings for training, 10% for cross-validation and the remaining 10% for hold-out purposes. Following (Li et al., 2018) we match up the movies between the MovieLens dataset and each testbed with string matching using the authors' provided implementation[3]. After preprocessing and removing invalid entries we obtain 5,045 movies on the ReDial dataset and 2,065 movies on GoRecDial. We only keep movie suggestions by the recommender which was not marked as dislike by the seeker. We perform 5-fold cross-validation and report the mean. Hyperparameters are tuned on a single fold then held fixed for the remaining folds.

4.2 Evaluation metrics

We use standard recommendation system evaluation metrics for top-n ranking, Normalized discounted cumulative gain (NDCG), Precision (P), Recall (R) and Mean reciprocal rank (MRR) (Manning et al., 2008). Users are generally interested in only a few top-ranked movies, Precision@n, NDCG@n and MRR@n are used to compare the top-n recommendation performance. As the data resembles an implicit feedback setting where the level of user feedback is extracted from the conversation, rank aware metrics such as NDCG and MRR alone may be insufficient since a negative entry could adversely impact the metric but in reality the user may not be aware of the item's existence. We have Recall as one metric for this consideration. Specifically, we treat the movies mentioned by the recommender in utterance X^t at turn t as the ground truth where the seeker did not give the recommendation 'dislike'. Note the recommendation is performed prior to observing the utterance with the ground truth and the model has only seen the utterances prior to turn t.

In the experiments, all the movies are treated as the candidates for ranking including those the user had previously talked about. In the real conversations, a user may only refer to a movie by an inexact or vague mention and thus it would be difficult to automatically identify and exclude the mentioned movies. While both ReDial and GoRecDial datasets have annotated movie mentions, the real-world conversations may not have such information. It is worth noting that all the baseline methods also used the same set of candidate movies as our methods for a fair comparison.

4.3 Baselines and settings

We validate the effectiveness of our model against a few baseline methods.

Random: The top-n recommendations are randomly selected from all the candidate movies in a uniform distribution.

Popularity: The movies with the highest popularity that occur in the MovieLens training set are presented to the user. While simple, this baseline could sometimes yield competitive performance in the benchmark recommendation tasks (Dacrema et al., 2019).

[2] https://www.imdb.com

[3] https://github.com/RaymondLi0/conversational-recommendations

TF-IDF (Term frequency inverse document frequency): The movies are ranked according to the TF-IDF vector space similarity between the user's utterance and each movie's plot via the inner product.

Semantic Retrieval: This method uses Eq. (3) to rank the movies according to the inner product similarity where the encoder is the Universal Sentence Encoder. Movies with the highest similarity to the utterance are presented to the user.

SVD (Singular Value Decomposition) (Salakhutdinov & Mnih, 2008) and **NMF** (Non-negative Matrix Factorization) (Févotte & Idier, 2011): These two baselines are classic collaborative filtering methods based on matrix factorization. We used movie preferences from the conversations to form user rating matrices (binary: like vs dislike) and then learn latent user and item factors for rating predictions. These methods require movie mentions identifiers annotated in each utterance.

ConvAE: This Conversational Autoencoder (ConvAE) approach was proposed in (Li et al., 2018) (together with the ReDial dataset). It consists of a dialogue generation model based on hierarchical recurrent encoder-decoder, a sentiment prediction model, and an auto-encoder recommender. Similar to SVD and NMF, this approach requires explicit movie annotations in the conversations.

DropoutNet (Volkovs et al., 2017): A deep learning based approach which adapts a deep neural network to handle the user cold-start problem on top of weighted matrix factorization (WMF), which demonstrated the state-of-the-art performance on public benchmarks.

For PMT, we first pre-train the two interaction functions Matrix Factorization (MF) and Generalized Matrix Factorization (GMF) with movie representations on the MovieLens dataset. All hyperparameters are obtained from a grid search by performance on the held out cross-validation set. We searched for the best latent dimension or embedding size from $\{10, 20, 30, 40, 50\}$, L_2 between 0.1 to $1e-7$, and the model was optimized with the SGD variant Adam (Kingma & Ba, 2015) using a standard learning rate of 0.001. We applied the pre-trained Universal Sentence Encoder with Transformer (USE-Transformer) (Cer et al., 2018) to encoder natural language texts in all the experiments (unless specified otherwise). Section 4.5 studies the effect of the sentence encoder on the recommendation performance.

In the experiments, the sentiment classifier was a logistic regression model with the USE-Transformer encoding of a given utterance as input. To train a sentiment classifier for each dataset, we used the labeled sentiment data from each dataset together with the labeled data from the Movie Review dataset[4], which resulted in 218,809 training instances for Redial and 75,005 for Goredcdial. The movie review dataset is in the same domain of movie discussions and shares some similar characteristics with the two testbeds. It has limited but high-quality binary sentiment labels (25,000 training examples). The majority of training data for sentiment classifiers still came from ReDial and GoRecDial themselves.

It is worth noting that several parts of the proposed model is pre-trained including sentence encoder/embeddings and sentiment classifier (pre-trained on sentiment labels), as well as the interaction function $f(\cdot)$ in Eq.(9) and the latent representation of movie $\mathbf{v}_j$ (pre-trained on MovieLens). Once they are pre-trained, the parameters are held fixed and not fined tuned to the downstream recommendation task as we considered the fact that the labeled domain data is limited. As demonstrated Sect. 4.4, even without fine-tuning of the pre-trained models, the proposed approach demonstrated good improvement over

[4] https://ai.stanford.edu/~amaas/data/sentiment/

Table 1 Experimental results on the ReDial dataset for different methods reporting Precision (P), Recall (R), normalized discounted cumulative gain (NDCG) and mean reciprocal rank (MRR) for different cut offs

	P@1	P@3	R@10	R@25	MRR@3
Random	0.0001	0.0002	0.0019	0.0048	0.0003
Popularity	0.0035	0.0020	0.0112	0.0220	0.0048
TF-IDF	0.0000	0.0000	0.0000	0.0080	0.0000
Semantic	0.0048	0.0036	0.0214	0.0411	0.0068
SVD	0.0000	0.0022	0.0116	0.0317	0.0006
NMF	0.0000	0.0002	0.0095	0.0274	0.0002
ConvAE	0.0014	0.0024	0.0408	**0.1013**	0.0034
DropoutNet	0.0039	0.0056	0.0346	0.0577	0.0074
PMT-MF	0.0017	0.0019	0.0153	0.0336	0.0034
PMT-GMF	**0.0060**†	**0.0061**	**0.0414**†	0.0798	**0.0091**†
	NDCG@3	NDCG@10	NDCG@25	MRR@10	MRR@25
Random	0.0003	0.0009	0.0016	0.0006	0.0009
Popularity	0.0042	0.0065	0.0092	0.0058	0.0067
TF-IDF	0.0000	0.0000	0.0018	0.0000	0.0003
Semantic	0.0069	0.0117	0.0166	0.0098	0.0113
SVD	0.0014	0.0046	0.0094	0.0024	0.0036
NMF	0.0008	0.0036	0.0079	0.0018	0.0029
ConvAE	0.0043	0.0157	0.0302	0.0084	0.0118
DropoutNet	0.0096	0.0173	0.0232	0.0131	0.0151
PMT-MF	0.0042	0.0077	0.0123	0.0058	0.0072
PMT-GMF	**0.0108**†	**0.0208**†	**0.0306**	**0.0160**†	**0.0192**†

The best results are highlighted in bold. †denotes the improvement over the best result of the baselines is statistically significant based on the paired t-test (p-value < 0.05)

the baseline methods. As shown in Algorithm 1, each new user latent factor $\tilde{\mathbf{u}}_i$ is the only learnable parameter during training. It is randomly initialized from a uniform distribution with the range computed from the variance of the pre-trained user representations maintaining the relative scale with respect to other parameters. During the conversation we use SGD with a learning rate of 0.1 to optimize the new user representation. At each turn t we update the parameters $\tilde{\mathbf{u}}_i$ and randomly sample negative items from the k least similar movies. All the experiments were conducted on a server with 2 Intel E5-2630 CPUs and 4 GeForce GTX TITAN X GPUs. The proposed models were implemented in PyTorch with the source code publicly available at GitHub[5].

4.4 Baseline comparison

In this section, we present the results of our proposed Personalized Memory Transfer (PMT) approach against the baseline methods on the two testbeds. Table 1 shows the

[5] https://github.com/agodavarthy/PMT

results on the ReDial dataset. As we can see, the Random baseline yielded very small values in all the metrics, indicating the difficulty of this recommendation task. The Popularity baseline performs much better than Random but generally worse than the other more competitive methods due to its lack of any form of personalization. The Semantic Retrieval method yielded much better results than TF-IDF, which is likely due to the fact that the user conversations often do not contain the keywords in the movie plots. There often exist some vocabulary gap between utterances and movie plots. For example, in the example dialogue shown in Fig. 1, the seeker was looking for a “suspenseful" movie. The plot of a relevant movie *A Space Odyssey* does not includes “suspenseful" but it contains a closely related word “mysterious". The much improved results of the Semantic baseline over TF-IDF demonstrated the advantages of the semantic sentence encoder in relating user utterances with movie plots.

The matrix factorization methods SVD and NMF performed better than Random, Popularity, and TF-IDF but worse than Semantic Retrieval, which indicates the importance of utilizing natural language information for conversational recommendations as SVD and NMF only looked at annotated movie preferences. ConvAE did not perform well on the very top ranked results as evidenced by the metrics of P@1, P@3, and NDCG@3, while obtaining the best result in R@25. It is worth noting that ConvAE replies on annotated movie mentions in the utterances, which may not be available in the real world scenarios. Nonetheless, the proposed PMT-GMF model yielded the best results in all the metrics except R@25, without utilizing the information of annotated movies. The majority of the improvements of PMT-GMF over the best baseline results were statistically significant based on the paired t-test (p-value < 0.05). DropoutNet generated the second best results in most metrics, which indicated the effectiveness of nonlinearity modeling on ReDial. This nonlinearity effect can also be seen via the comparison between the results of PMT-MF and PMT-GMF. PMT-GMF obtains better performance than the linear counterpart PMT-MF which suggests the presence of more complex nonlinear interactions may be required to disentangle user preferences and transfer the knowledge to another dataset.

On the GoRecDial dataset as shown in Table 2, the results show a similar pattern with the ones in Table 1 on ReDial, but with larger improvements observed on PMT-MF and PMT-GMF over the baseline methods. PMT-MF yielded the second best in the majority of the metrics after PMT-GMF. The PMT-GMF model obtained better results than all the baselines across all the metrics except R@25, which reaffirms the effectiveness of the proposed model. Similar to the results on ReDial, PMT-GMF outperforms PMT-MF, which demonstrates the advantage of modeling nonlinearity via the interaction function.

4.5 Effect of sentence encoder

In this section, we investigate the effect of the underlying sentence encoder on the recommendation performance of the proposed model. Specifically, We experiment with four popular deep contextualized pre-trained sentence encoders to transform the utterances in the conversations into sentence embeddings. The encoders are as follows:

Table 2 Experimental results on the GoRecDial dataset for different methods reporting Precision (P), Recall (R), normalized discounted cumulative gain (NDCG) and mean reciprocal rank (MRR) for different cut offs

	P@1	P@3	R@10	R@25	MRR@3
Random	0.0005	0.0003	0.0042	0.0110	0.0007
Popularity	0.0039	0.0014	0.0110	0.0368	0.0041
TF-IDF	0.0028	0.0021	0.0126	0.0217	0.0036
Semantic	0.0089	0.0068	0.0461	0.0792	0.0117
SVD	0.0000	0.0028	0.0488	0.1432	0.0020
NMF	0.0000	0.0004	0.0238	0.0680	0.0004
ConvAE	0.0068	0.0081	0.1220	**0.2434**	0.0210
DropoutNet	0.0064	0.0087	0.0650	0.1380	0.0118
PMT-MF	0.0169	0.0145	0.1007	0.1825	0.0248
PMT-GMF	**0.0202**†	**0.0176**†	**0.1231**	0.2169	**0.0286**†
	NDCG@3	NDCG@10	NDCG@25	MRR@10	MRR@25
Random	0.0009	0.0019	0.0036	0.0012	0.0017
Popularity	0.0042	0.0063	0.0126	0.0048	0.0066
TF-IDF	0.0047	0.0070	0.0092	0.0052	0.0058
Semantic	0.0154	0.0225	0.0325	0.0176	0.0198
SVD	0.0047	0.0186	0.0412	0.0093	0.0151
NMF	0.0017	0.0087	0.0194	0.0040	0.0069
ConvAE	0.0239	0.0552	0.0849	0.0354	0.0431
DropoutNet	0.0177	0.0314	0.0489	0.0206	0.0254
PMT-MF	0.0324	0.0523	0.0720	0.0370	0.0425
PMT-GMF	**0.0387**†	**0.0634**†	**0.0861**	**0.0445**†	**0.0509**†

The best results are highlighted in bold. †denotes the improvement over the best result of the baselines is statistically significant based on the paired t-test (p-value < 0.05)

Table 3 Experimental results of PMT-GMF on the ReDial dataset with different sentence encoders

	P@1	P@3	R@10	R@25	MRR@3
ELMo	0.0039	0.0040	0.0266	0.0528	0.0060
BERT	0.0056	0.0057	0.0368	0.0709	0.0087
USE-DAN	0.0035	0.0037	0.0254	0.0509	0.0056
USE-Transformer	**0.0060**	**0.0061**	**0.0414**	**0.0798**	**0.0091**
	NDCG@3	NDCG@10	NDCG@25	MRR@10	MRR@25
ELMo	0.0070	0.0134	0.0201	0.0104	0.0125
BERT	0.0103	0.0189	0.0275	0.0147	0.0174
USE-DAN	0.0066	0.0127	0.0192	0.0097	0.0117
USE-Transformer	**0.0108**	**0.0208**	**0.0307**	**0.0161**	**0.0192**

The best results are highlighted in bold

Table 4 Experimental results of PMT-GMF on the GoRecDial dataset with different sentence encoders

	P@1	P@3	R@10	R@25	MRR@3
ELMo	0.0146	0.0127	0.0900	0.1617	0.0206
BERT	0.0169	0.0145	0.1007	0.1825	0.0248
USE-DAN	0.0189	0.0167	0.1181	0.2093	0.0271
USE-Transformer	**0.0202**	**0.0176**	**0.1231**	**0.2169**	**0.0286**
	NDCG@3	NDCG@10	NDCG@25	MRR@10	MRR@25
ELMo	0.0279	0.0461	0.0634	0.0322	0.0371
BERT	0.0324	0.0523	0.0720	0.0370	0.0425
USE-DAN	0.0367	0.0605	0.0826	0.0424	0.0486
USE-Transformer	**0.0387**	**0.0634**	**0.0861**	**0.0445**	**0.0509**

The best results are highlighted in bold

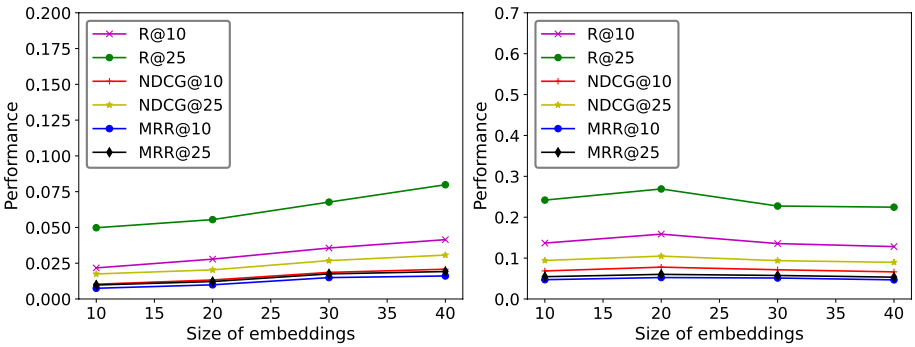

Fig. 3 Effect of embedding size on the proposed PMT-GMF model on the two datasets in different metrics

- ELMo (Peters et al., 2018). It uses a bi-directional LSTM to compute contextualized character-based word representations. We used TensorFlow Hub implementation of ELMo[6], trained on the 1 Billion Word Benchmark.
- BERT (Devlin et al., 2019). It is is a deep embedding model that learns vector representations of words and sentences by training a deep bidirectional Transformer network. We used the uncased BERT-Base model[7] trained on English Wikipedia and BooksCorpus.
- Universal Sentence Encoder (USE) (Cer et al., 2018). It has two variants: one is trained with a Deep Averaging Network (USE-DAN) and another with a Transformer network (USE-Transformer). We used TensorFlow Hub implementation of USE[8], trained on a variety of data sources including Wikipedia, web news, web question-answer pages, and supervised data from the Stanford Natural Language Inference (SNLI) corpus.

[6] https://tfhub.dev/google/elmo/2
[7] https://github.com/google-research/bert
[8] https://tfhub.dev/google/universal-sentence-encoder/4

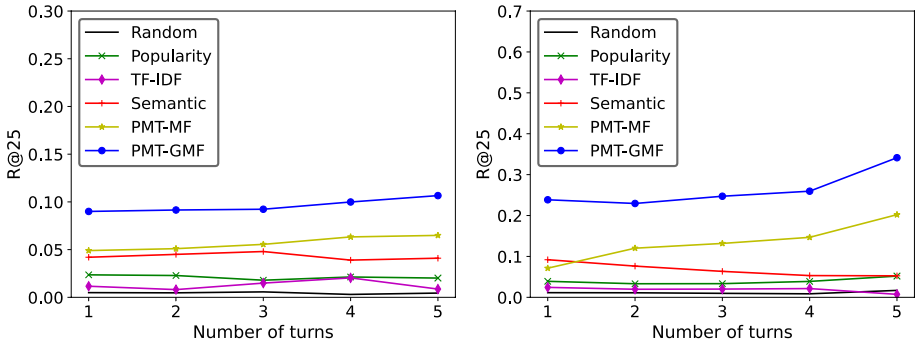

Fig. 4 Recall at 25 (R@25) of different methods over the number of turns in the conversations

Tables 3 and 4 contain the experimental results of the proposed PMT-GMF model with different sentence encoders on the two datasets respectively. We can see the USE-Transformer encoder achieved the best results in all the metrics on both datasets. The BERT encoder yielded very similar results with USE-Transformer on ReDial across all the metrics. On GoRecDial, both variants of USE obtained slightly better results than BERT. The minor improvement of USE-Transformer over BERT could come from the fact that USE-Transformer used more diverse sources of data (which include some informal texts) for pre-training than BERT did. The results were consistent with some existing work in other recommendation domains which showed USE could generate marginally better recommendation results than BERT as a sentence encoder (Hassan et al., 2019). Overall, the four state-of-the-art pre-trained text encoders yielded results in a comparable range.

4.6 Effect of embedding size

In this section, we analyze the effect of embedding size of key embedding $\mathbf{k}_j$ (Eq.(1)), query embedding $\mathbf{q}^t$ (Eq.(2)), movie embedding $\mathbf{v}_j$ (Eq.(5)), and user embedding $\mathbf{u}_i$ (Eq. (8)), on the performance of our model. Figure 3 presents the performance comparison with respect to the length of embedding varied from 10 to 40 on the two datasets in different metrics. As we can see, on the GoRecDial dataset the metrics in Recall and NDCG first increase and then decrease, while on the ReDial dataset the performance in all the metrics improves when the embedding size increases. This may be due to the fact that the GoRecDial dataset has less dialogues and utterances and thus the model is more likely to overfit when the embedding size (or model complexity) increases. In general, the results across different metrics have a consistent pattern over the embedding size.

4.7 Effect of conversation length

In this section, we examine the effect of the length of the conversation to determine whether updates to the user representation leads to improved performance. We bin the results of each recommendation based on the number of turns in a conversation. For example, if the

Table 5 Average response time (in seconds) of PMT-GMT with two different sentence encoders on the two datasets: BERT and USE-Transformer

PMT-GMF	Redial			GoRecDial		
	Encoding	Ranking	Total	Encoding	Ranking	Total
BERT	0.6490	0.1106	0.7596	0.6973	0.0960	0.7935
USE-Transformer	0.6040	0.1012	0.7163	0.6189	0.0898	0.7087

model has seen 3 turns of conversation before a recommendation, the result of that recommendation will be categorized into the 3-turn bin. Figure 4 plots recall at 25 (R@25) varying the number of turns from 1 to 5. We do not report the other metrics since they show similar trends.

As we can see, the Random and Popularity baselines seem insensitive to the number of turns in a conversation, which is expected since they do not update the recommendations based on the conversations. The performance of TF-IDF seems to peak around 4 turns on both datasets potentially due to the dependency on keywords and the increasing complexity of the user's utterances as the conversation progresses. The Semantic baseline shows comparable results with TF-IDF on ReDial and noticeably better performance on GoRecDial. On the other hand, PMT-GMF and PMT-MF demonstrate a steady increase and upward trend in performance as the number of turns increase on both datasets. The pattern is more visible on the GoRecDial dataset for the two proposed models. These results illustrate the effectiveness of the proposed models in updating the user representation over time and learning better user preferences. For all the different numbers of turns, PMT-GMF yields much better results than the other methods. In general, PMT-MF delivers the second best performance on the two testbeds and is followed by the Semantic baseline. The pattern in different number of turns is generally consistent with the overall results shown in Tables 1 and 2.

4.8 Efficiency analysis

In this section, we empirically study the efficiency of the proposed PMT-GMF model. Specifically, we tested how fast on average the model can make recommendations at run time given an utterance in the conversation. We recorded the response time for each instance in the test set. The response time can be further decomposed into two parts: time for encoding an utterance and time for generating a ranked list of recommended movies. The experiments were conducted on one GeForce GTX TITAN X GPU. Table 5 contains the average response time (in seconds) of PMT-GMT with two different sentence encoders on the two datasets. As we can see, the encoding process constitutes a large fraction of the response time on both datasets, since BERT and USE-Transformer are large language models with many parameters. The encoding took slightly longer time on GoRecDial than on Redial probably because the average length of utterances on Redial is a bit larger. On the other hand, the ranking time on GoRecDial was slightly longer than that on GoRecDial, which is likely due to the fact that Redial contains more candidates movies than GoRecDial does. Overall, the total response times on both datasets with different encoders were in a similar range and less than 0.8 second. It is worth noting that there is abundant room to further improve the efficiency by using Knowledge distillation, e.g., DistilBERT (Sanh et al.,

Seeker:	My favorite movies are computer animate comedies. I alos like some fantasy movies like Harry Potter.
Rec:	Do you like any family movies?
Seeker:	Sure, I like all walt disney movies such as Meet the Robinsons
Rec:	How about the family movies where there is a princess in it?
Seeker:	I love the Toy Story movies
Rec:	How about any of the princess movies?
Seeker:	I dont know if harry potter has a princess in it, but that would be the only one
Model:	*Forrest Gump* | ***Shrek*** | *Monsters, Inc.* | *Finding Nemo* | *Harry Potter and the Prisoner of Azkaban* | *Harry Potter and the Chamber of Secrets* | *American Beauty* | *Chicken Run* | *Up* | *Groundhog Day*
Rec:	RECOMMEND Shrek
Seeker:	Why did you recommend this movie?
Rec:	It was a computer animated comedy and also similar to the movies they said they liked.
Seeker:	ACCEPT Shrek
Rec:	Why did you accept the movie?
Seeker:	I like computer animated movies
Model:	*Shrek* | *Monsters, Inc.* | ***Finding Nemo*** | *Harry Potter and the Prisoner of Azkaban* | *Forrest Gump* | *Harry Potter and the Chamber of Secrets* | *American Beauty* | *Up* | *Ratatouille* | *Harry Potter and the Half-Blood Prince*
Rec:	RECOMMEND Finding Nemo
Rec:	have you seen finding nemo?
Seeker:	no, but i would love that too

Fig. 5 an example conversation from the GoRecDial dataset between recommendation seeker (Seeker) and recommender (Rec). Marked in blue in the figure are the top 10 movie recommendations by PMT-GMF. The positive recommendation is marked in bold based on the ground truth (Color figure online)

2019), to compress large sentence encoders. We will further investigate this in the future work.

4.9 Qualitative study

In this section, we conduct a qualitative study on a specific conversation instance and see how PMT-GMF learns user preferences as the dialogue progresses. Figure 5 shows an example conversation from the GoRecDial dataset between recommendation seeker (Seeker) and recommender (Rec). Marked in blue in the figure are the top 10 movie recommendations by PMT-GMF up to the conversation at that turn (based on the ranking score in Eq.(11)). The positive recommendation based on the ground truth is marked in bold.

At the beginning of the conversation, the Seeker expressed his/her interest in “computer animate comedies", “fantasy movies", and “Harry Potter". As we can see from the first top 10 recommendations by the proposed model, two movies belong to Harry Potter series. The “computer animate comedies" type of movies include *Shrek*, *Monsters, Inc*, *Finding Nemo*, *American Beauty*, *Chicken Run*, and *Up*. These results demonstrate

that the proposed approach can learn user preferences even when they are expressed at a high level (e.g., “computer animate comedies") with inexact movie mentions (e.g., “Harry Potter"). Only 2 out of the 10 recommendations seemed to be what the Seeker was looking for. *Forrest Gump* showed up as the top result probably because it is a comedy and a very popular movie. *Groundhog Day* is a fantasy comedy film but not computer animated.

The recommender (Rec) in the dataset recommended *Shrek* which was accepted by Seeker. It is worth noting that only *Shrek* is considered as a positive instance for this recommendation in the evaluation while some other top ranked movies by our model could also be relevant in the real world if they are presented to the Seeker. This offline evaluation methodology may explain why all the experimental results have relatively low values.

After the Seeker accepted the first recommendation, he or she further indicated “I like computer animated movies". As we can see, all the top three movies ranked by our model belong to “computer animated movies". The positive instance *Finding Nemo* appeared at the 3rd position. *Forrest Gump* was pushed down and *Groundhog Day* did not appear in the top 10 results any more. *Up* was moved up from the previous position. A new movie *Ratatouille* emerged in the top recommendations, which is a computer-animated comedy and released by Walt Disney as what was wanted by the Seeker. These results demonstrate that our model could effectively update preferences and recommendations based on the new utterances.

5 Conclusion and future work

In this work, we propose PMT to learn representation for a new user in an online fashion in a conversation setting. On the one hand, our work does not depend on historical interaction log from users so that it can handle cold start user. On the other hand, our work does not depend on mapping item mentions in user utterances to unique item identifiers which means we are able to extract user preferences directly from natural language conversation. The key idea of PMT is to leverage a key-value memory structure to transfer prior knowledge of natural language and item representations/preferences to the conversational domain. The experimental results on two public conversational testbeds reveal the advantages of PMT to learn an effective user representation in an online setting as the conversation progresses leading to better recommendations.

This work is just an initial step towards a promising new direction. As pointed out in Introduction, in this paper we have focused on learning user preferences from dialogues. As the proposed approach demonstrates its effectiveness, we plan to combine the proposed PMT model with a response generation component to produce human-understandable responses for communicating with users. Based on the predicted user preferences, we can use an existing response generation method such as retrieval based or generation based techniques (Gao et al., 2021). Moreover, we will explore an end-to-end learning framework to understand user sentiment and preferences from raw natural language, and to generate meaningful responses in a joint fashion. In addition, as shown in the experiments, the interaction function has a significant impact on the model performance. We plan to design more sophisticated interaction functions which explicitly integrates item preferences, content information and the natural language encoder.

References

Bordes, A., Boureau, Y. L., & Weston, J. (2017). Learning end-to-end goal-oriented dialog. In: ICLR.

Bousmalis, K., Trigeorgis, G., Silberman, N., Krishnan, D., & Erhan, D. (2016). Domain separation networks. In: NIPS.

Cer, D., Yang, Y., Kong, Sy., Hua, N., Limtiaco, N., John, R.S., Constant, N., Guajardo-Cespedes, M., Yuan, S., & Tar, C. et al. (2018). Universal sentence encoder. Preprint arXiv:1803.11175.

Cer, D. M., Diab, M. T., Agirre, E., Lopez-Gazpio, I., & Specia, L. (2017). Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In: SemEval@ACL.

Christakopoulou, K., Radlinski, F., & Hofmann, K. (2016). Towards conversational recommender systems. In: SIGKDD, pp. 815–824.

Christakopoulou, K., Beutel, A., Li, R., Jain, S., & hsin Chi, E. H. (2018). Q & r: A two-stage approach toward interactive recommendation. In: SIGKDD.

Cui, C., Wang, W., Song, X., Huang, M., Xu, X. S., & Nie, L. (2019). User attention-guided multimodal dialog systems. In: SIGIR.

Dacrema, M. F., Cremonesi, P., & Jannach, D. (2019). Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In: RecSys.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In: NAACL.

Dodge, J., Gane, A., Zhang, X., Bordes, A., Chopra, S., Miller, A., Szlam, A., & Weston, J. (2016). Evaluating prerequisite qualities for learning end-to-end dialog systems. In: ICLR.

Ebesu, T., Shen, B., & Fang, Y. (2018). Collaborative memory network for recommendation systems. In: SIGIR.

Févotte, C., & Idier, J. (2011). Algorithms for nonnegative matrix factorization with the β-divergence. *Neural computation, 23*(9), 2421–2456.

Gao, C., Chen, X., Feng, F., Zhao, K., He, X., Li, Y., & Jin, D. (2019). Cross-domain recommendation without sharing user-relevant data. In: WWW.

Gao, C., Lei, W., He, X., de Rijke, M., & Chua, T. S. (2021). Advances and challenges in conversational recommender systems: A survey. *AI Open, 2*, 100–126.

Harper, F. M., & Konstan, J. A. (2015). The movielens datasets: History and context. *TIIS, 5*(4), 1–19.

Hassan, H. A. M., Sansonetti, G., Gasparetti, F., Micarelli, A., & Beel, J. (2019). Bert, elmo, use and infersent sentence encoders: The panacea for research-paper recommendation? In: RecSys.

He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T.S. (2017). Neural collaborative filtering. In: WWW.

Hu, G., Zhang, Y., & Yang, Q. (2018). Mtnet: A neural approach for cross-domain recommendation with unstructured text. In: SIGKDD.

Jannach, D., Manzoor, A., Cai, W., & Chen, L. (2021). A survey on conversational recommender systems. *ACM Computing Surveys (CSUR), 54*(5), 1–36.

Kanagawa, H., Kobayashi, H., Shimizu, N., Tagami, Y., & Suzuki, T. (2019). Cross-domain recommendation via deep domain adaptation. In *ECIR*. Springer.

Kang, D., Balakrishnan, A., Shah, P., Crook, P., Boureau, Y. L., & Weston, J. (2019). Recommendation as a communication game: Self-supervised bot-play for goal-oriented dialogue. In: EMNLP.

Kang, J., Condiff, K., Chang, S., Konstan, J. A., Terveen, L. G., & Harper, F. M. (2017). Understanding how people use natural language to ask for recommendations. In: RecSys.

Kingma, D., & Ba, J. (2015). Adam: A method for stochastic optimization. In: ICLR.

Lei, W., He, X., Miao, Y., Wu, Q., Hong, R., Kan, M. Y., & Chua, T. S. (2020). Estimation-action-reflection: Towards deep interaction between conversational and recommender systems. In: WSDM, pp. 304–312.

Li, P., & Tuzhilin, A. (2020). Ddtcdr: Deep dual transfer cross domain recommendation. In: WSDM, pp. 331–339.

Li, R., Kahou, S. E., Schulz, H., Michalski, V., Charlin, L., & Pal, C. (2018). Towards deep conversational recommendations. In: NIPS.

Li, S. S., & Karahanna, E. (2015). Online recommendation systems in a b2c e-commerce context: a review and future directions. *Journal of the Association for Information Systems, 16*(2), 2.

Liu, Z., Wang, H., Niu, Z. Y., Wu, H., Che, W., & Liu, T. (2020). Towards conversational recommendation over multi-type dialogs. In: ACL.

Lops, P., De Gemmis, M., & Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In *Recommender systems handbook* (pp. 73–105).

Manning, C. D., Raghavan, P., Schütze, H., et al. (2008). *Introduction to information retrieval*. Cambridge University Press.

Miller, A. H., Fisch, A., Dodge, J., Karimi, A., Bordes, A., & Weston, J. (2016). Key-value memory networks for directly reading documents. In: EMNLP.

Mo, K., Zhang, Y., Li, S., Li, J., & Yang, Q. (2018). Personalizing a dialogue system with transfer reinforcement learning. In: AAAI.
Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In: ICML.
Penha, G., & Hauff, C. (2020). What does bert know about books, movies and music? Probing bert for conversational recommendation. In: RecSys, pp. 388–397.
Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. S. (2018). Deep contextualized word representations. In: NAACL-HLT.
Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-thieme, L. (2009a). BPR : Bayesian Personalized Ranking from Implicit Feedback. UAI.
Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L. (2009b). Bpr: Bayesian personalized ranking from implicit feedback. In: UAI.
Ricci, F., Rokach, L., & Shapira, B. (2015). Recommender systems: Introduction and challenges. In *Recommender systems handbook* (pp. 1–34). Springer.
Salakhutdinov, R., & Mnih, A. (2008). Probabilistic matrix factorization. In: NIPS, Vol. 20.
Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. Preprint http://arxiv.org/abs/1910.01108
Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In: WWW, pp. 285–295.
Suglia, A., Greco, C., Basile, P., Semeraro, G., & Caputo, A. (2017). An automatic procedure for generating datasets for conversational recommender systems. In: CLEF.
Sun, Y., & Zhang, Y. (2018). Conversational recommender system. In: SIGIR.
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In: NIPS.
Volkovs, M., Yu, G. W., & Poutanen, T. (2017). Dropoutnet: Addressing cold start in recommender systems. In: NIPS.
Wang, H., Wu, Q., & Wang, H. (2017). Factorization bandits for interactive recommendation. In: AAAI.
Wen, T.H., Vandyke, D., Mrksic, N., Gasic, M., Rojas-Barahona, L. M., Su, P. H., Ultes, S., & Young, S. (2017). A network-based end-to-end trainable task-oriented dialogue system. In: ACL.
Wu, Q., Iyer, N., & Wang, H. (2018). Learning contextual bandits in a non-stationary environment. In: SIGIR, pp. 495–504.
Yan, Z., Duan, N., Chen, P., Zhou, M., Zhou, J., & Li, Z. (2017). Building task-oriented dialogue systems for online shopping. In: AAAI.
Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR), 52*(1), 1–38.
Zhang, Y., Chen, X., Ai, Q., Yang, L., & Croft, W. B. (2018). Towards conversational search and recommendation: System ask, user respond. In: CIKM.
Zhu, Y., Gong, Y., Liu, Q., Ma, Y., Ou, W., Zhu, J., Wang, B., Guan, Z., & Cai, D. (2019). Query-based interactive recommendation by meta-path and adapted attention-gru. In: CIKM.
Zou, J., Chen, Y., & Kanoulas, E. (2020a). Towards question-based recommender systems. In: SIGIR, pp. 881–890.
Zou, L., Xia, L., Gu, Y., Zhao, X., Liu, W., Huang, J. X., & Yin, D. (2020b). Neural interactive collaborative filtering. In: SIGIR, pp. 749–758.