A Recommendation Dialog System Driven by Collaborative Filtering

Anonymous ACL submission

Abstract

Dialog systems are ubiquitous, taking on a variety of roles from personal assistants to language tutors. Through its interactive nature, dialog systems have the opportunity to gather critical information and employ that information in real time, which traditional systems cannot, making it useful for recommendation tasks. Previously, recommendation dialog systems mostly treated every user's interaction independently, no users' preferences were utilized to help another. We propose to leverage other users' preferences through matrix factorization to improve recommendation quality. On the other hand, using the user preference information captured by the dialog system, the recommendation system can overcome the cold-start problem. To test the effectiveness of the proposed dialog framework, we developed a movie recommendation system that is able to update recommendation results though incorporating user preference obtained on the fly. Experiments with human users suggest that such dialog systems that leverage other users information through matrix factorization achieves better recommendation quality.

1 Introduction

Dialog-based recommendation systems can help users incrementally build their preference models and refine them once they are provided with more information. They are extremely useful in handling high risk recommendation tasks as users usually need more information to make decisions (Chen and Pu, 2012). It is also useful for recommending products with specific features that are easy for users to describe (Chen and Pu, 2012). Traditional recommendation dialog systems use criteria filtering methods to recommend items and treat each user independently (Bridge, 2002; Mori et al., 2017). Though previous research has created personalized recommendation systems that track user's preferences across conversations (Thompson et al., 2004; Ramachandran et al., 2015), to our knowledge, no recommendation dialog systems has incorporated other users' information to help the current user. In this paper, we propose to utilize collaborative filtering to incorporate other users information to improve current user's experience.

We designed a recommendation dialog framework that incorporates matrix factorization to achieve collaborative filtering. Specifically, we collect user preference through dialog and use them as side information to assist matrix factorization. Based on the framework, we built a recommendation system to recommend movies. Through user studies, we found that the collaborative filtering-based dialog system performs better than the criteria filtering-based system in terms of recommendation satisfaction.

We not only published our source code for the recommendation dialog framework, the example movie recommendation system, but also the conversations collected[1]. The recommendation dialog framework is applicable in various domains besides movie recommendations, such as travel, real-estate and general products. We also maintain a working system that is able to provide movie recommendation service to the general public in an extended period time for the purpose of AI education. The published conversation data set not only consists of the natural language interaction between human and the system, but also user self-

[1]https://github.com/chatboxACL

reported recommendation satisfaction, interaction engagement and system utterance appropriateness ratings. In addition, the data set also records users qualitative comments about their likes and dislikes about the system, which is extremely valuable for the purpose of understanding the user needs and expectations.

2 Related Work

Generally speaking, recommendation systems can be classified into two categories: the ratings-based recommendation and the critiquing-based recommendation (Ricci et al., 2011). Ratings-based system completes the recommendation in one turn based on the product ratings (Srebro et al., 2005; Mnih and Salakhutdinov, 2008) or some implicit user feedback (Hu et al., 2008; Yi et al., 2013a). These systems produce a recommendation by eliciting an inadequate, non-descriptive rating that is highly noisy. Latent factor models, such as matrix factorization have been successfully used in ratings-based systems, because rather than basing recommendations by neighboring items of the same user, the ratings can be characterized by items and users based on a high variety of factors. Their lack of domain restrictions and ability to understand implicit relationships in the absence of explicit ratings (Koren et al., 2009; Ramachandran et al., 2015). E-commence leaders, like Amazon and Netflix, have made latent factor models salient parts of their services (Koren et al., 2009). However, such recommendation systems suffer from the cold start problem with new users and new items. Previous research tackles the cold start problem by assuming certain constrains, such as, users may like movies with similar actors (Schein et al., 2002). In this paper, instead of using these assumptions which foster errors, we use recommendation dialog systems, one type of critiquing-based systems to handle the code start problem.

There are two types of recommendation dialog systems, the similarity-based and filtering-based systems. Similarity-based dialog recommendation systems ask users to provide a preferred example item, such as user's favorite movie in movie recommendation. Chill [2] is one example similarity-based system. These systems only utilize item similarity to produce recommendation and have low usability for low frequency recommendation tasks. Because in these tasks, users do not have a previously preferred example.

Filtering-based system, will search the data base to find items that match the exact product features specified (Bridge, 2002; Mori et al., 2017). Though some systems integrate previous interactions from the same user to create a sense of penalization (Thompson et al., 2004; Ramachandran et al., 2015), they did not include other users' preference to assist current user's interaction. Also filtering-based systems will run into troubles if there is no item that fits all the criteria the user specifies in the data base. In this work, we propose a new type of recommendation dialog system that integrate collaborative filtering to tackle the cold start and strict constraint problems together. Specifically, we obtained the user preference through dialog, and use that information as the side information to assist matrix factorization, which is a latent factor method that has no fixed constraints.

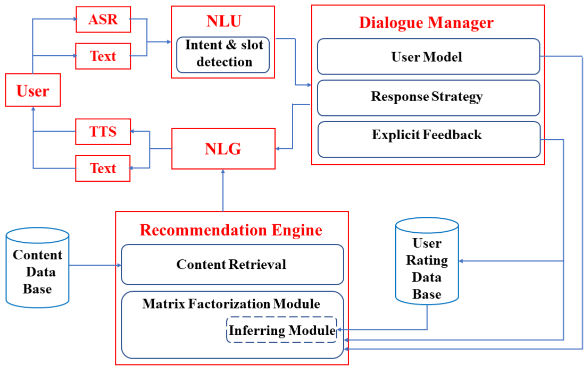


Figure 1: Proposed Recommendation Dialog System Framework A user utterance is sent to a natural language understanding module (NLU) and is used to create a user model. The recommendation engine uses this model as side information for matrix completion in order to generate recommendation. The user's explicit feedback would also update the user model and provide on-the-fly recommendation updates.

3 Framework Description

We propose a recommendation dialog framework that integrates matrix factorization in the recommendation engine. The framework is general and adaptable to various recommendation tasks. Figure 1 shows the information flow of the system.

ResponseVoice API[3] is used for automatic speech recognition (ASR) and Annyang API[4] for text-to-speech (TTS) generation. We selected these APIs because they are open source and have

[2] http://andchill.io/

[3] https://responsivevoice.org/

[4] https://www.talater.com/annyang/

good performance. If a user interacts with the system through a typing interfaces, these two components will be disabled. After the system receives the user utterance, the "natural language understanding module" (NLU) detects user intents and pre-defined slots (entities) from it. In movie recommendation setting, the pre-defined slots are entities, such as movie genre. These entities will be used as user side information to assist matrix factorization later. We pre-trained these intent classifier and slot detectors with human annotated utterances using LUIS[5]. Then the detected intent and slot information will be fed into the dialog manager. One thing worth mention is that we also implemented spell correction for pre-processing the user input and stemming for post-processing of LUIS output. These two steps are essential as typos of actor and director names are prevalent in user utterances and sometimes users say "romantic comedy" instead of "romance and comedy".

There are three components in the dialog manager, the "user model", the "response strategy" and the "explicit feedback module". The "user model" keeps track of all the user information, which are the slots values, throughout the conversation. When the "user model" has collected sufficient user information to make a recommendation, the dialog manager will request a recommendation from the recommendation engine. Otherwise, the dialog manager will select another missing user model feature to inquire using the response strategy module. After the dialog manager decides which system action to perform, the "natural language generation module" (NLG) takes the action request from the "response strategy module" or the "recommendation engine" to generate a natural language response. This component can be implemented using either neural network-based methods or template based methods. A template generation method is used in the movie recommendation task, because the system response is relatively simple.

After the system has all the user side information, the "recommendation engine" is called to provide a recommendation. The recommendation engine's main component is a "matrix factorization model". We will explain the details of the model in the next section. The matrix factorization module takes information from a "user rating data base" (e.g. Netflix Prize corpus (Bennett et al., 2007)) and utilizes the side information and produces a list of ranked movies. The system then picks the top ranked movie and returns the details of that movie to the user by pulling information from the "content data base". In the movie recommendation task, the IMDB database which has movie information, such as length and plot is used as the "content data base". After we provided the recommendation to the user, the "explicit feedback" module will collect the user feedback with respect to the recommendation, such as whether the user likes the recommendation or not. The collected feedback will update the "user rating data base" in order to track users' preferences. If the user feedback towards the recommendation is negative, the recommendation engine will produce another recommendation until user is satisfied or left the conversation.

The framework supports multiple users interacting with the system simultaneously. We have tested exhaustively for scalability as well for our example movie recommendation system. The users could interact with the system through speaking or typing in a web-browser on a variety of devices, such as laptops, tables and smart phones.

4 Method: Matrix Factorization with Noisy Side Information

Since we need to build a model based on both user ratings and side information, classical low-rank matrix factorization does not work. We adopt the dirty statistical model for matrix completion with general side information (Chiang et al., 2015), as described below.

4.1 Inductive Matrix Completion

Let $R \in \mathbb{R}^{n_1 \times n_2}$ be the underlying rank-k matrix that aims to be recovered where $k \ll min(n_1, n_2)$. Moreover, let

$$X \in \mathbb{R}^{n_1 \times d_1}, Y \in \mathbb{R}^{n_1 \times d_2}$$

be the feature set where each row X_i or Y_i denotes the feature of the i-th row entity of R. Let Ω be the entries sampled from R with cardinality $|\Omega| = m$. Note that usually $d_1, d_2 \leq min(n_1, n_2)$ but can exceed k.

Traditional matrix factorization (Koren et al., 2009) learns the low-rank underlying matrix by

[5] https://www.luis.ai/

solving

$$\arg\min_N \sum_{i,j\in\Omega}(R_{ij}-N_{ij})^2+\lambda\|N\|_*,$$

where $\|\cdot\|_*$ is the trace norm regularization to enforce low-rankness of N. However, this model is not able to incorporate the side information X, Y collected by our system.

Inductive matrix completion is a popular model to incorporate side information (Jain and Dhillon, 2013; Xu et al., 2013; Yi et al., 2013b). They assume the features are noise-free,

$$col(R) \subseteq col(X) \text{ and } row(R) \subseteq col(Y)$$

This feature set is *perfect* because it fully describes the true latent feature space of R. While we could recover the low matrix R directly, we can use the formulation of inductive matrix completion to recover a smaller matrix $M \in \mathbb{R}^{d_1\times d_2}$ such that

$$R = XMY^T \tag{1}$$

(Jain and Dhillon, 2013). Inductive matrix completion is shown to be theoretically preferred (Xu et al., 2013) and useful in applications like predicting gene-disease associations (Natarajan and Dhillon, 2014), although in practice most given features X and Y will not be perfect. Our framework exploits this loose feature criteria to map utterance entities to user preferences. Practically, given features X and Y are not perfect and can introduce noise and reduce performance. (Chiang et al., 2015) defines a dirty statistical model that is robust to handling these noisy features.

4.2 Statistical Model with Noisy Side Information

(Chiang et al., 2015) introduces a dirty statistical model for matrix completion by balancing feature information and observations. The underlying matrix R is recovered jointly by the low rank estimate from the feature space $R = XMY^T$ and with a n_1 by n_2 low-rank matrix N. Thus we have

$$\min_{M,N} \sum_{(i,j)\in\Omega} \ell((XMY^T+N)_{ij}, R_{ij}) + \lambda_M \|M\|_* + \lambda_N \|N\|_* \tag{2}$$

where M and N are regularized with trace norm. We estimate the underlying low rank matrix R by

$$XM^*Y^T + N^*. \tag{3}$$

Note that λ_M and λ_N control the importance between features and noise. When $\lambda_M = \infty$, (2) becomes a standard matrix completion. Similarly, when $\lambda_N = \infty$, (2) becomes inductive matrix completion. λ_M and λ_N are evaluated by the average performance in 10-fold cross validation. With optimal M discovered, a new user rating ranking can be computed by XMY^T. Note that MY^T is constant until retraining and can be used for any new user. Thus only a single vector multiplication operation needs to be performed. XMY^T provides rating values on movies for the new user.

In this work, instead of using MSE, which cares about the absolute rating scores, we used normalized discounted cumulative gain (NDCG) as the evaluation metric (details shown below). Because for recommendation tasks, the ranking is all that matters and NDCG assigns higher weights to the top ranked items.

$$NDCG = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)} \tag{4}$$

where rel_i is the relevance of the movie and i is the position in the ranking and p is the number of items considered.

4.3 Experimental Settings

We used a subset of the Netflix prize data set, which has 5000 users and 1188 movies. We picked the users that have the largest number of ratings. We included genre, actors/actresses, directors and MPAA ratings as user side information. For existing users, we inferred their side information by using their movie ratings. For example, we construct the actor preference of a user by computing the statistics of different actors appeared in her top rated movies. For movie information, we simply used the movie details recorded in the Netflix database. There are in total 22 genres, such as drama and comedy and six types of MPAA ratings, such as PG-13 and R. There are many actors/actress and directors in total. Due to the long tail issue, we only selected the most frequently mentioned actors for computational effectiveness; 575 actors/actresses and directors. We inferred user side information from movie ratings, so the information is not accurate and may introduce noise. Therefore noisy matrix factorization algorithm is used to handle the noise in the inferred side information. We combined actors and directors in inferring side information, as they have a huge amount of overlap.

4.4 Results

We compared the results of the matrix factorization algorithm by incorporating different side information. The details are shown in Table 1.

System	NDCG
genre	0.9006
genre + MPAA *	0.9010
genre + actor/director + MPAA *	0.9011

Table 1: The NDCG score performs the best with including all the side information compared with only genre information with statistical significance.

We evaluated the results on NDCG score over all movies in the Netflix corpus. We observed that by incrementally adding more side information, the NDCG score increased. Both models, genre + MPAA and genre +actor/director+ MPAA outperformed the model with only genre information with statistical significance ($p < 0.05$). This indicates that though there is noise in the side information, adding more user side information can still improve the overall performance due to the power of the dirty matrix factorization model. However, adding actor/director information to the genre+MPAA model only improved the result slightly. This may due to the fact that the actor/director information is sparse and has noise. Because during the side information inference process, we limited the number of actor/director to 575.

5 User Experiment Design

We tested two movie recommendation dialog systems with users to test their usability.

- ***Criteria Filtering (CF)***: Recommendations are generated by filtering movies based on the user's preferences. The movie with the highest average rating (according to the Netflix Prize data) that fits all the user specified preference is chosen to be recommended. If there is no movie that matches all of the user preferences, we retract the preference constraints in the order of MPAA, directors, actors and genres until there is one movie that fits the criteria. The order of the relaxing constraints was designed according to expert knowledge. We use this system as our baseline model in experiments.
- ***Matrix Factorization (MF)***: We designed this system following the proposed recommendation framework described in Section 3. The recommendation was generated by matrix factorization incorporating side information: preferred genres, actors/actresses, directors and MPAA ratings.

1 **SYS**: *Hi, I'm Chatbot. I'm trying to give better movie recommendations than Netflix can. We will start with some questions to figure out your interests. Then I will give you a movie available on Netflix to watch!*
2 **SYS**: *What's your favorite movie genre?*
3 **USR**: Comedy
4 **SYS**: *Can you tell me some actors or actresses you like?*
5 **USR**: Jim Carrey, John Candy, Rowan Atkinson
6 **SYS**: *Who is your favorite director?*
7 **USR**: Mel Brooks
8 **SYS**: *What MPAA rating should this movie be?*
9 **USR**: PG-13
10 **SYS**: *I'm thinking about what you might like.*
11 **SYS**: *How about Ace Ventura: Pet Detective (1994)?*
12 **SYS**: *Ace Ventura: Pet Detective stars Courteney Cox, Sean Young, Tone Loc, Tom Shadyac, Jack Bernstein, James G. Robinson, Ira Newborn, Julio Macat, Don Zimmerman, Jim Carrey and is directed by Jack Bernstein, Tom Shadyac, Jim Carrey.*
13 **SYS**: *This film is 86 minutes long. It is a comedy movie, and is rated PG-13.*
14 **SYS**: *Have you watched this movie before?*
15 **USR**: Yes
16 **SYS**: *Would you want another recommendation then?*
17 **USR**: Yes
18 **SYS**: *How about Liar Liar (1997)?*
19 **SYS**: *Liar Liar stars Brian Grazer, Justin Cooper, Cary Elwes, Paul Guay, Jim Carrey, Maura Tierney, Stephen Mazur, John Debney, Russell Boyd, Tom Shadyac and is directed by Paul Guay, Stephen Mazur.*
20 **SYS**: *This film is 86 minutes long. It is a comedy and fantasy movie, and is rated PG-13.*
21 **SYS**: *Have you watched this movie before?*
22 **USR**: No
23 **SYS**: *Do you like this recommendation?*
24 **USR**: Yes
25 **SYS**: *Bye! Please provide feedback so I can do better next time!*

Table 2: An example conversation generated by the matrix factorization system.The system keeps suggesting movies until users find a movie they like and have not seen before.

We recruited human users on Amazon Mechanical Turk to interact with our systems. We only recruited users that are located in the U.S., and with an approval rating for previous tasks above 98%. Each users may only interact with either chatbot once, as to prevent contamination of the results. The chatbot UI human users interact with can be seen in Appendix A. An example of the dialog generated by the criteria filtering model is shown in Table 2. The chatbot will ask specific questions to collect user's movie preferences, such as genre, actor, director and MPAA rating. When enough information is collected, the chatbot will present a movie recommendation to the user and ask if they have seen it before. If they have seen it, the system will ask if the user want another recommendation. If the user wants another recommendation, the chatbot will present the next best recommendation until finding a movie user has never seen before and liked. An example conversation of the matrix factorization dialog system is shown in Table 2. After the users accepts the recommendation, they will be directed to a survey. A snapshot of the survey is shown in Appendix B.

We elicited three metrics in the survey: recommendation satisfaction (Walker et al., 1997), user engagement (Yu et al., 2015) and system utterance appropriateness (Banchs and Li, 2012) in a 1-5 Liker scale. The recommendation satisfaction metric assesses the recommendation performance, while the user engagement metric assesses the entire interaction experience. Users were also asked to rate the appropriateness of every system utterance with respect to theirs. At the end of the survey, we also ask users to leave a open-ended comment on the overall interaction experience. At the end of the three-week user experiments, we collected in total 90 conversations, 45 each using the two systems described above.

6 Results and Analysis

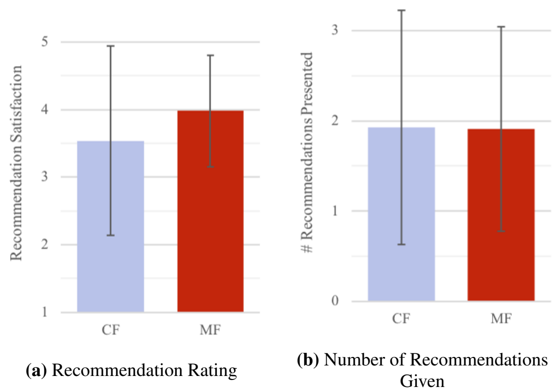
From our experiments, we found that matrix factorization with four types of use information, performs better in terms of recommendation satisfaction compared to criteria filtering. Two systems performed similarly in terms of the number of recommendation tries, user engagement and system utterance appropriateness. We will describe the results with respect to each metric as follows:

6.1 Recommendation Satisfaction

As shown in Figure 2a, we found that matrix factorization system was better than criteria filtering system according to user ratings with statistical significance (t-test, $p<0.04$). The criteria filtering system received an average rating of 3.54 (S.D. = 1.40) where the matrix factorization system received an average rating of 3.98 (S.D. = 0.83).

When the user was presented with a recommendation, they have the chance to reject it, and then the system will recommend the next best movie, with one or more follow questions, until the user is satisfied or no recommendation is given. Across all trials, users were satisfied within one or two recommendations as shown in Figure 2b. The criteria filtering system averaged 1.92 (S.D. = 1.30) recommendations, where matrix factorization system averaged 1.91 (S.D. = 1.13). Being able to satisfy the user with less suggestions means the system is providing more accurate recommendations more effectively. Thus, we compare criteria filtering and matrix factorization on the average tries each system needs before a recommendation is accepted. Figure 2b demonstrates that two systems perform similarly on this metric (t-test, $p>0.47$).

We also asked the user if they have seen the recommended movies before, since one of the goals of our system is to recommended movies that users may not have seen. We found that while the average number of users who have seen the first recommended movies were lower for matrix factorization system, there was no statistical significance to the difference (t-test, $p>0.17$). The average number of user who have seen the first recommended movie for criteria filtering system was 0.63 (S.D. = 0.48), where that for matrix factorization system was 0.53 (S.D. = 0.50).



(a) Recommendation Rating

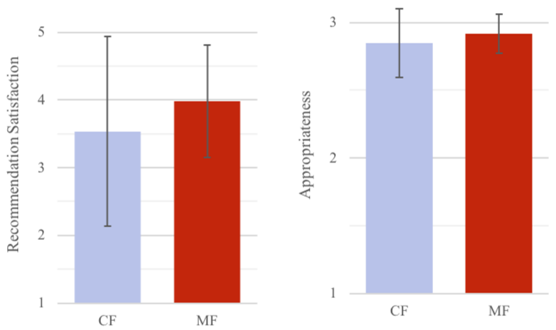
(b) Number of Recommendations Given

Figure 2: Matrix factorization dialog systems receive better recommendation satisfaction. CF stands for criteria filtering and MF stands for matrix factorization

6.2 User Engagement

We define engagement as the interest to continue the conversation (Yu et al., 2015), which captions an important user interaction experience aspect. A user with low engagement may quit the chatbot early, or attempt to quit conversations by entering low-effort answers.

As shown in Figure 3a, the user-reported engagement score was similar between the criteria filtering system and the matrix factorization (t-test, $p>0.11$). The criteria filtering system received an average engagement rating of 3.88 (S.D. = 1.02), where the matrix factorization received an average engagement rating of 4.13 (S.D. = 0.88). Since both chatbot systems share identical natural language understanding and generation modules, all the systems are overall engaging.



(a) User Engagement Rating

(b) Average Dialogue Appropriateness Rating

Figure 3: Criteria filtering and matrix factorization dialog systems perform similarly in user engagement and utterance appropriateness. CF stands for criteria filtering and MF stands for matrix factorization

6.3 Dialog Appropriateness

We define system utterance appropriateness as the coherence of the system utterance with respect to the user utterance (Banchs and Li, 2012). As shown in Figure 3b, both of the systems are rated similarly on system utterance appropriateness (t-test, $p>0.44$). The criteria filtering system received an average appropriateness rating of 2.85 (S.D. = 0.25), where the matrix factorization system received an average engagement rating of 2.92 (S.D. = 0.14). Since all the systems share identical natural language understanding and generation modules, the results match our expectation.

6.4 Open-Ended User Comments

In addition to numeric ratings, we asked users to openly talk about their interaction experience in the survey. This allows us to assess the system qualitatively. These open questions also enabled us to explore user needs and expectations for further improvement of the system.

Latency One of the common remarks across all systems is about the response time of the system. The matrix factorization-based system pretrain the computation model, so during the interaction, the test time is short. So the interaction latency of all the systems are all optimized towards natural interaction. Many users leave comments for all the systems, saying "It was awesome! So fast, so life like. I really think it is perfect".

Usability Many users comment on the usability of the systems positively saying,"It was very straightforward and easy to use." However, one user commented that he wish he could go back to change his responses and our system did not provide such function. In the future, we will address such function request.

Chattiness Another aspect of the system many users comment is on how social the system is. We found that people have different expectations, some users like the directness of the system, commenting: "I liked that it was quick, and straight forward. No chit chat to worry about."; some users complained that "I thought the chatbot was really direct and somewhat unsocial." We plan to make the system adapt to different users needs for chitchat or lack of thereof in the future.

System pros and cons One user commented on the filtering-based baseline system as: "it provided me with a recommendation that seemingly ignored my preferences". This happens when there is no movie that fits all the criteria the user specified and the system relax one or two criteria to find a movie. Users seem to be very annoyed by getting movie with relaxed criteria. One issue of the criteria filtering system is that the strict constraints makes it unable to explore other movies with similar attributes. Therefore, one user complained "Instead of thinking of others I might like, it just gave me a movie with that specific actor in it.". The matrix factorization based system did not receive any similar comments because it does not try to find the exact match but utlize latent factor to cater each individual's need.

Another problem with the filtering-based system is that it will pick the most highly rated movies after finding the set of movies fits all the user specified criteria. These highly rated movies happen to be classical movies. Some users don't like them and comment, "I wanted newer movies and it tended to take older ones for my suggestions.". While no users had similar complaints with the matrix factorization based system. Because it is more adaptive to individual information instead of picking the most highly rated movie. Many people comment on the matrix completion chatbot system as: "It's great. I was surprised that it found a movie that met my exact specifications." Another user commented on that " It was very quick and did feel personalized. The algorithm definitely works", even we never told users it is utilizing latent model for personalizing.

7 Conversation Analysis

To validate the usability of the dialog recommendation framework in a movie recommendation task, we conducted a brief analysis on all the human-system conversations collected. We found several interesting suggestive phenomena that would be of interest to the recommendation system community and the movie industry, although the sample size is rather small (90 participants) and biased (crowd workers from the U.S.).

User experience variation We found that people with experience interacting with chatbots are less engaged (t-test, $p<0.02$), but rated the system similarly to those with no experience in terms of recommendation rating (t-test, $p>0.36$). The average recommendation rating for those with chatbot experience is 3.73 (S.D. = 1.09), where those without is 3.82 (S.D. = 1.25). The average user engagement for those with chatbot experience is 3.83 (S.D. = 0.96), where those without is 4.29 (S.D. = 0.89). This suggests that users have more experience with dialog systems are more critical towards the overall user experience.

Domain specific questions We found that questions that required sophisticated movie domain knowledge often had a lack of variety in user responses. When asked for a preferred director, variety is not nearly as high as other questions such as genre, actor, MPAA. Most users say I don't care or randomly mention a director that is popular, such as Steven Spielberg (refer to our published corpus for details). Therefore, questions that demand specific domain knowledge may introduce noise and jeopardize the task performance. Such noise is more detrimental to criteria filtering dialog systems than the matrix factorization systems. Because the impact of filtering a question by a response that doesn't reflect the user's true preference can rule out most successful recommendations for criteria filtering systems.

Preferred genres One movie tends to have multiple genre tags associated with it. Most movie's 1st tag is action, comedy, or drama. These three genres constitute 62% movies in the Netflix corpus. However, often the most distinguishing feature is the 2nd or 3rd tags of the movie, such as sci-fi and horror. Users actually expressed more interests in the 2nd and 3rd genre tags. Of all the genres requested, 51% were for the 2nd or 3rd genre tags.

Gender imbalance Participants frequently requested actors over actresses and male directors over female directors (90.2% requested actors and 98.1% male directors). The percentage is highly skewed and the major reason behind that is that study (Lauzen, 2017) found that 33% of films employed 0 or 1 woman in the considered roles and there is only 9% female directors (statistics calculated in 2015). Our users are just given fewer actresses and female directors to choose from.

8 Conclusion and Future Work

We proposed a new recommendation dialog framework that utilizes other users information through matrix factorization. We implemented a movie recommendation dialog system based on the framework and conducted experiments with human users. We found that the matrix factorization-based system outperformed the traditional criteria filtering-based dialog system in terms of movie recommendation quality. Our recommendation dialog framework can also be applied in various domains, such as product and service recommendation. Through analyzing the human-system conversations, we also found some interesting observations, such as people who have chatbot interaction experience are more critical towards the system.

In the future, we plan to improve the natural language understanding module to handle the large variety in user utterances. As we have seen some users describe their preferred genre such as "movies that make me feel at edge" or " movies that have a happy ending", which our current understanding module cannot handle. Having a better understanding model will very likely to further improve the system's recommendation quality. We also plan to explore the possibility of using online learning and reinforcement learning to further optimize our system's performance. In addition, we would like to explore transferring the proposed recommendation framework from a module framework into a end-to-end trainable framework, in order to make the system easier to train and adapt to other domains.

We also plan to publish the movie recommendation system on cellphone application stores in order to get real users. So we can further improve the system's personalization ability by training on real world data.

References

Rafael E Banchs and Haizhou Li. 2012. Iris: a chat-oriented dialogue system based on the vector space model. In *Proceedings of the ACL*.

James Bennett, Stan Lanning, et al. 2007. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35. New York, NY, USA.

Derek G Bridge. 2002. Towards conversational recommender systems: A dialogue grammar approach. In *ECCBR Workshops*, pages 9–22.

Li Chen and Pearl Pu. 2012. Critiquing-based recommenders: survey and emerging trends. *User Modeling and User-Adapted Interaction*, 22(1-2):125–150.

Kai-Yang Chiang, Cho-Jui Hsieh, and Inderjit S Dhillon. 2015. Matrix completion with noisy side information. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3447–3455. Curran Associates, Inc.

Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 263–272. Ieee.

Prateek Jain and Inderjit S. Dhillon. 2013. Provable inductive matrix completion. *CoRR*, abs/1306.0626.

Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.

Martha M Lauzen. 2017. The celluloid ceiling: Behind-the-scenes employment of women on the top 100, 250, and 500 films of 2015.

Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*, pages 1257–1264.

Hayato Mori, Yuya Chiba, Takashi Nose, and Akinori Ito. 2017. Dialog-based interactive movie recommendation: Comparison of dialog strategies. In *International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pages 77–83. Springer.

Nagarajan Natarajan and Inderjit S. Dhillon. 2014. Inductive matrix completion for predicting gene–disease associations. *Bioinformatics*, 30(12):i60–i68.

Deepak Ramachandran, Mark Fanty, Ronald Provine, Peter Yeh, William Jarrold, Adwait Ratnaparkhi, and Benjamin Douglas. 2015. A tv program discovery dialog system using recommendations. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 435–437.

Francesco Ricci, Lior Rokach, and Bracha Shapira. 2011. Introduction to recommender systems handbook. In *Recommender systems handbook*, pages 1–35. Springer.

Andrew I. Schein, Alexandrin Popescul, Lyle H. Ungar, and David M. Pennock. 2002. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '02, pages 253–260, New York, NY, USA. ACM.

Nathan Srebro, Jason Rennie, and Tommi S Jaakkola. 2005. Maximum-margin matrix factorization. In *Advances in neural information processing systems*, pages 1329–1336.

Cynthia A. Thompson, Mehmet H. Göker, and Pat Langley. 2004. A personalized system for conversational recommendations. *J. Artif. Int. Res.*, 21(1):393–428.

Marilyn A Walker, Diane J Litman, Candace A Kamm, and Alicia Abella. 1997. Paradise: A framework for evaluating spoken dialogue agents. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, pages 271–280. Association for Computational Linguistics.

Miao Xu, Rong Jin, and Zhi Hua Zhou. 2013. *Speedup matrix completion with side information: Application to multi-label learning*. Neural information processing systems foundation.

Jinfeng Yi, Rong Jin, Shaili Jain, and Anil Jain. 2013a. Inferring users' preferences from crowdsourced pairwise comparisons: A matrix completion approach. In *First AAAI Conference on Human Computation and Crowdsourcing*.

Jinfeng Yi, Lijun Zhang, Rong Jin, Qi Qian, and Anil Jain. 2013b. Semi-supervised clustering by input pattern assisted pairwise similarity matrix completion. In *International Conference on Machine Learning*, pages 1400–1408.

Zhou Yu, Alexandros Papangelis, and Alexander Rudnicky. 2015. Ticktock: A non-goal-oriented multimodal dialog system with engagement awareness. In *Proceedings of the AAAI Spring Symposium*.

Appendix A: User Survey

The following is a sample of the survey presented to the users after the dialog interaction.

Survey

Your Turk ID:

Part 1

Rate how appropriate you feel the system's response with respect to your input. Try to make the decision for each round independently, try not to take context into consideration.

"Not appropriate" means the system response is not coherent at all,
e.g. Participant: How old are you? Chatbot: Apple

"Interpretable" means the system response is related and can be interpreted in a way.
e.g. Participant: How old are you? Chatbot: That's too big a question for me to answer.

"Appropriate" means the system response, is very coherent with the user's previous utterance.
e.g.Participant: How is the weather today? Chatbot: Very good.

Turn 1

Computer:

Hi, I'm Chatbot. I'm trying to give better movie recommendations than Netflix can. We will start with some questions to figure out your interests. Then I will give you a movie available on Netflix to watch!

What genre of films do you like to see?

You:

action

Your rating:

○ Not Appropriate ○ Interpretable ○ Appropriate

•••

Turn 5

Computer:

I'm thinking about what you might like.

How about The Matrix (1999)?

The Matrix stars Lana Wachowski, Keanu Reeves, Zach Staenberg, Laurence Fishburne, Bill Pope, Don Davis, Carrie-Anne Moss, Joel Silver, Lilly Wachowski, Hugo Weaving and is directed by Lilly Wachowski, Lana Wachowski.

This film is 136 minutes long. It is an action and sci-fi movie, and is rated R.

Have you watched this movie before?

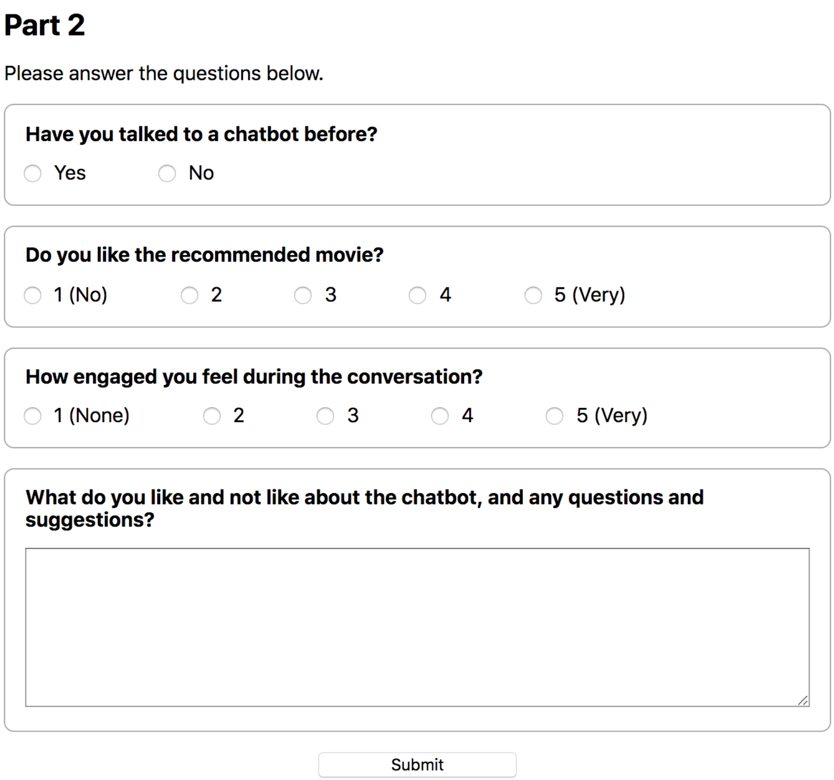
You:

yes

Your rating:

○ Not Appropriate ○ Interpretable ○ Appropriate

•••



Part 2

Please answer the questions below.

Have you talked to a chatbot before?

- Yes
- No

Do you like the recommended movie?

- 1 (No)
- 2
- 3
- 4
- 5 (Very)

How engaged you feel during the conversation?

- 1 (None)
- 2
- 3
- 4
- 5 (Very)

What do you like and not like about the chatbot, and any questions and suggestions?

Submit

Appendix B: The Dialog Interface

The following is the UI and a sample interaction a user has with the movie recommendation dialog system.