Recommendation Dialog Systems with User Side Information Assisted Collaborative Filtering

Abstract

Through its interactive nature, dialog systems have the opportunity to gather critical information and employ it in real time to adapt to individual users. Therefore, dialogs are suitable for recommendations. Previously, recommendation dialog systems mostly treat every user's interaction independently, no users' preferences were utilized to help another. We propose to leverage other users' preferences through user side information assisted matrix factorization to improve recommendation quality. Through actively acquiring user information in dialogs, we also addressed the cold-start problem for the recommendation. To test the effectiveness of the proposed dialog framework, we developed a movie recommendation system that updates recommendation though incorporating user preference obtained on the fly. Experiments with human users suggest that such system achieves better recommendation quality.

1 Introduction

Dialog-based recommendation systems can get information incrementally from users to build their preference models by actively asking questions. Recommendation dialog systems are instrumental in handling high-risk recommendation tasks, in which users usually need more information to make decisions (Chen and Pu, 2012). They are also valuable for recommending products with specific features that are easy for users to describe (Chen and Pu, 2012). Traditional dialog-based recommendation 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 users' preferences across conversations (Thompson et al., 2004; Ramachandran et al., 2015), to our knowledge, no dialog-based recommendation systems have incorporated other users' information when making recommendations to 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 uses matrix factorization to incorporate collaborative filtering. Specifically, we collect user preferences through dialog and use that as side information to assist matrix factorization. Based on the framework, we built a movie recommendation system. Through user study with human users, 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 the source code of our recommendation dialog framework, i.e., 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 can provide movie recommendation service to the general public for an extended period for AI education. The published conversation data set not only consists of the natural language interaction between human and the system, but also user self-reported recommendation satisfaction, interaction engagement and system utterance appropriateness ratings. Also, the data set also records users qualitative comments about their likes and dislikes about the system, which is extremely valuable for understanding user needs and expectations.

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

2 Related Work

Recommendation systems can be classified into two categories: ratings-based recommendation and critiquing-based recommendation (Ricci et al., 2011). Ratings-based systems make recommendation 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). Critiquing-based systems would obtain information through user interaction before making recommendations. Recommendation dialog systems are one type of critiquing-based systems. Rating-based systems may suffer from inadequate, non-descriptive, and highly noisy ratings when generating recommendations. Latent factor models, such as matrix factorization, have been successfully used in ratings-based systems because rather than basing recommendations on neighboring items of the same user, the ratings can be characterized by factors of items and users variety. E-commerce leaders, like Amazon and Netflix, have included latent factor models in their services (Koren et al., 2009). However, such recommendation systems suffer from the cold start problem with new users and items when no ratings are available. Previous research tackles the cold start problem by assuming specific constraints, such as users may like movies with similar actors (Schein et al., 2002). In this paper, instead of using these assumptions which may lead to overgeneralization problems, we propose to use dialog driven recommendation systems to obtain user side information to handle the cold start problems in matrix factorization.

There are two types of recommendation dialog systems: similarity-based and filtering-based. Similarity-based systems ask users to provide a preferred example item, such as their favorite movie, such as Chill [2]. These systems only utilize item similarity to produce recommendations, which have low usability for low-frequency recommendation tasks. This is because users do not have a previously preferred example in these tasks. Filtering-based systems search their databases to find items that match the exact product features as specified (Bridge, 2002; Mori et al., 2017). Though some systems integrate previous interactions from the same user to create a sense of personalization (Thompson et al., 2004; Ramachandran et al., 2015), they do not include other users' preferences to assist the current user's interaction. Also, filtering-based systems will run into trouble if no item fits all the criteria specified by the user.

We propose a new recommendation dialog system that integrates collaborative filtering to tackle the cold start problem in rating-based systems. We first obtain user preferences through dialogs and then apply it to assist matrix factorization in producing the recommendation. In addition, matrix factorization learns a latent factor space to capture the user preference instead of constraining the recommendation with explicit user requirements like the filtering-based methods. The learned latent space makes the recommendation generalizable.

[2] http://andchill.io/

3 Framework Description

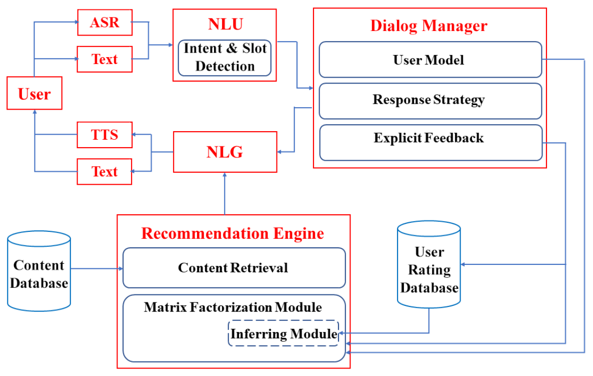


Figure 1: Proposed recommendation dialog system framework. The natural language understanding module (NLU) will first process the user response and use the result to create a user model. The recommendation engine leverages this model as side information to assist matrix completion to generate a list of recommendation. The user's explicit feedback about the recommendation updates the user model. Then the recommendation will be updated based on the user model on the fly.

We propose a recommendation dialog framework that integrates matrix factorization for providing recommendations. The framework is generalizable to all recommendation tasks. Figure 1 shows the information flow of the system. We discuss each component of the framework as follows:

3.1 ASR and TTS

ResponseVoice API[3] is used for automatic speech recognition (ASR) and Annyang API[4] for text-to-speech (TTS) generation. We selected these

[3] https://responsivevoice.org/

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

APIs because they are open source and have good performance. If a user interacts with the system through a typing interface, these two components are disabled.

3.2 Natural Language Understanding

Our language understanding component is a mixture of a third-party and a customized model. We take movie recommendation as an example. We first define specific slots for the task, such as movie genre, and pre-train the slot detector using human annotated utterances on LUIS [5]. We also spell corrected user input. We also mapped some common synonyms and phrasing variances of the user input, such as from funny movies to comedy, to our defined slots to increase the system's language understanding. The detected slot information is then passed on to the dialog manager.

3.3 Dialog Manager

The dialog manager is responsible for selecting actions based on the system state, and tracking user information and feedback. There are three main components: "response strategy", "explicit feedback module", and "user model."

3.3.1 Response strategy

Our dialog manager is rule-based. While the dialog flow is restricted, this fixed structure can benefit our movie recommendation. The dialog manager will ask the user for their genre, actor, director, and MPAA rating preferences in this fixed order. Each question is drawn from a set of predefined templates with slight surface form variations to provide variety. The result of the NLU is used as user side information. Once all four types side information is collected, the recommendation framework requests the recommendation engine for a movie, which is then presented to the user using predefined slots.

3.3.2 Explicit feedback module

Once the user has received a movie recommendation, the dialog manager first asks if the user has watched the movie before. This is because, in early trials, some users commented on wanting movies that they have not watched. If the user has watched the movie, the dialog manager asks if they want another recommendation; if the user does not, the dialog manager asks if they like the recommendation. If the user chooses to have another recommendation, the recommendation engine is called and another recommendation is given. The conversation ends until either the user has accepted a recommendation or no recommendations are left. For all utterances, if the user responds with irrelevant text or the system cannot parse the response, the dialog manager will say it does not understand and prompt the user to repeat until a meaningful response is obtained.

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

3.3.3 User model

The user model keeps track of the user responses and recommended movies throughout the conversation. This information is stored in the user rating database during the conversation session and written to a log file once the session completes; these log files are used for user trial analysis. The user model also keeps track of a unique session ID that is used to identify the user throughout the conversation and in the log files. Should the user be disconnected due to unexpected circumstances, they can resume the conversation by matching their session ID with the user model.

3.4 Recommendation Engine

The recommendation engine generates the movie recommendations using the "matrix factorization model", which is explained in detail in Section 4. The matrix factorization model takes information from the "user rating database" (e.g., Netflix Prize corpus (Bennett et al., 2007)) and utilizes the side information, i.e., the user movie preferences response, to produce a list of ranked movies. The movie descriptions are requested from the IMDB database and are mostly prefetched to reduce recommendation latency. The movie recommendation is sent to the "explicit feedback" module of the dialog manager for user feedback. When the user indicates whether they like or dislike the recommended movie, the recommendation engine uses online learning to adopt this rating to its existing rankings. This on-the-fly learning is ideal for real-time conversational systems.

4 Matrix Factorization

The core of our recommendation engine is inductive matrix factorization. Information from the chat is correlated to features of the matrix factorization. The recommendation engine then computes the matrix factorization with this side information and provides a rank of movies. This final ranking can be optimized by online learning from implicit user feedback.

4.1 Side Information

Information generated from the chatbot is referred to as side information. They are represented as genre, actor, director, and MPAA preferences in our model. We train our matrix factorization on side information inferred from the Netflix Prize corpus' user ratings.

4.2 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 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. User and movie side information is captured by X and Y respectively.

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 these loose feature criteria to map utterance entities to user preferences.

While MSE is a useful indicator of overall performance, it cares about the absolute rating scores. This work uses normalized discounted cumulative gain (NDCG) as the evaluation metric (details are shown below). For recommendation tasks, the ranking is all that matters and DCG assigns higher weights to the top-ranked items.

$$DCG = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)} \tag{2}$$

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. The DCG score is normalized when it is divided by the optimal solution DCG. This normalized DCG provides a closeness to the optimal ranking.

We combined all the side information by first performing an outer product between each type of the side information and then linearizing the result. We found such method approximates the Netflix rating corpus better than a simple concatenation of all the side information.

4.3 Online Learning

Information generated from user conversations may not be representative of the users' true preferences. Thus, recommendations might not be well received initially. When the user rejects a recommendation, it is important to update the matrix completion results for that user on the fly. Rather than redoing the matrix factorization, we performed gradient descent to count in current movie. The algorithm updates with respect to the unitary matrices U and V such that $M = UV$ is,

$$U \Leftarrow U - \eta * 2X_i(X_i^T UV^T Y_j - R_{ij})Y_j^T V + \lambda U \tag{3}$$

$$V \Leftarrow V - \eta * 2U^T X_i(X_i^T UV^T Y_j - R_{ij})Y_j^T + \lambda V \tag{4}$$

With simple matrix multiplication, we can update the recommendations. This is ideal for real-time recommendation systems that require short response time.

4.4 Experimental Setting

We used a subset of the Netflix prize dataset, 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 infer their side information by using their movie ratings. For example, we construct the actors preferences of a user by computing the statistics of different actors appeared in top rated movies. For movie information, we simply used the movie details recorded in the Netflix database. There are 22 genres in total, such as drama and comedy, and six types of MPAA ratings, such as PG-13 and R. We selected the 532 most frequently mentioned actors/actresses and directors to mitigate the long tail issue for computational efficiency. We combined actors and directors in inferring side information, as they have a huge overlap.

4.5 Results

We compared the results of the matrix factorization with various side information.

Side Information	MF
MPAA	0.410
genre	0.435
genre + MPAA	0.470*
genre + MPAA + actor/director	0.620*

Table 1: Matrix factorization performs best with more side information.

Table 1 shows the DCG score of the matrix factorization. By including more types of side information, the model performs better. When we combine genre, MPAA and actor/director information the performance is better than only having a subset of this information with statistical significance ($p < 0.05$).

5 Movie Recommendation Dialog Systems

We evaluated two movie recommendation dialog systems performance with real users.

- ***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 preferences is recommended. If no movie matches all of the user preferences, we relax 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.

- ***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 above 98%. Each user may only interact with one chatbot once. The chatbot UI can be seen in Appendix B. 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 wants another recommendation. If the user wants another recommendation, the chatbot will present the next best recommendation until it finds a movie the user had never seen before and liked. An example conversation of the matrix factorization dialog system is shown in Table 2. After the user accepts the recommendation, they will be directed to a survey. A snapshot of the survey is shown in Appendix B.

We measured three metrics in the survey: recommendation satisfaction (Walker et al., 1997), user engagement (Yu et al., 2015) and dialog appropriateness (Banchs and Li, 2012) in a 1-5 Liker scale, the higher the better. 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 their responses. At the end of the survey, we also asked users to leave an 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.

5.1 Dialog Results

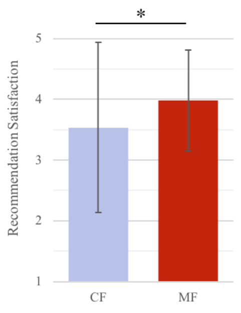
We found that matrix factorization outperforms criteria filtering with the same series of questions with respect to recommendation satisfaction. The two systems performed similarly in terms of the number of recommendation tries, user engagement, and system utterance appropriateness. We will describe the results with respect to each metric as follows:

5.1.1 Recommendation satisfaction

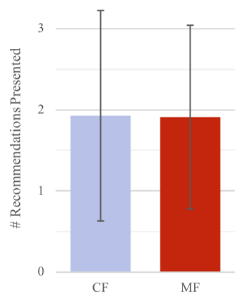
As shown in Figure 2a, we found that matrix factorization was better than criteria filtering according to user ratings (t-test, $p<0.04$). The criteria filtering system received an average rating of 3.54 (S.D. = 1.40) while the matrix factorization system received an average rating of 3.98 (S.D. = 0.83).

When the user was presented with a recommendation, they had the chance to reject it, so the system would recommend the next best movie until the user is satisfied or no recommendation item left. Being able to satisfy the user with fewer suggestions means the system is providing accurate recommendations more effectively. Users were satisfied within one or two recommendations for both systems across all trials (t-test, $p>0.47$), 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).

We also asked the user if they have seen the recommended movies before since one of the goals of our system is to recommend 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 in the matrix factorization setting. However, the difference was not statistically significant (t-test, $p>0.17$).



(a) Recommendation Satisfaction

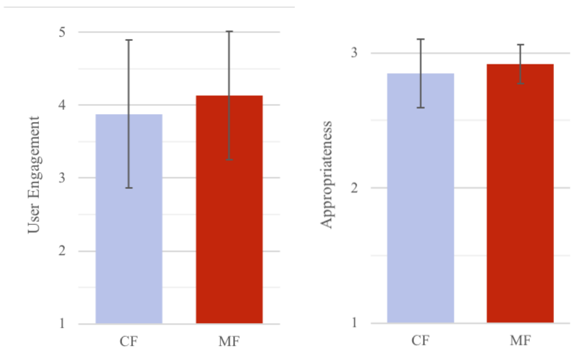


(b) Number of Recommendations

Figure 2: Matrix factorization dialog systems (MF) received better recommendation satisfaction than criteria filtering (CF). Both provided a satisfactory recommendation within similar number of tries.

5.1.2 User engagement

We define engagement as the interest to continue the conversation (Yu et al., 2015), and used it to measure user interaction experience. As shown in Figure 3a, the user-reported engagement score was similar between the criteria filtering system and the matrix factorization system (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 4.13 (S.D. = 0.88). All systems are overall engaging. Since both chatbot systems share identical natural language understanding and generation modules, their engagement scores are similar is expected.



(a) User Engagement **(b)** Dialog Appropriateness

Figure 3: Criteria filtering (CF) and matrix factorization dialog systems (MF) perform similarly in user engagement and dialog appropriateness.

5.1.3 Dialog appropriateness

We define dialog appropriateness as the coherence of the system utterance with respect to the user utterance (Banchs and Li, 2012). As shown in Figure 3b, both 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.

5.1.4 Open-ended user comments

In addition to numeric ratings, we asked users to write comments about their interaction experience for qualitative system assessment. These comments enabled us to explore user needs and expectations for potential system improvements.

One user criticized the filtering-based system for: "it provided me with a recommendation that seemingly ignored my preferences". This happens when no movie fits all the criteria and the system relaxes one or two criteria to find a movie. Users seem to be very annoyed by getting a movie with relaxed criteria. One issue of the criteria filtering system is that the strict constraints make 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 utilize latent factor to cater each individual's need.

Another problem with the filtering-based system is that it always pick the highest rated movies after finding the set of movies that fit all the user-specified criteria. These highly rated movies are often classic movies. Some users did not like them and commented, "I wanted newer movies and it tended to take older ones for my suggestions." However, 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 commented 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 the latent model for personalizing.

Latency Many users commented across all systems about the short system response time. All the systems received many comments such as "It was awesome! So fast, so life-like. I really think it is perfect". Because the matrix factorization-based system pre-trains its recommendation model, the interaction latency is small.

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

Chattiness Another aspect of the system many users commented on is how social the systems are. We found that people have different expectations. Some users like the directness of the system, commenting: "I liked that it was quick and straightforward. No chit chat to worry about.", while 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 thereof in the future.

6 Conversation Analysis

We analyzed the collected human-system conversations and found several interesting phenomena that would be of interest to the recommendation system community and the movie industry despite the small sample size (90 participants) and biased population(crowd workers from the U.S.).

6.1 User experience variation

We found that people who have experience interacting with chatbots reported being less engaged (t-test, $p<0.02$, 3.83 (S.D. = 0.96) with and 4.29 (S.D. = 0.89) without), but rated recommendation satisfaction (t-test, $p>0.36$) similarly with those that have no experience (3.73 (S.D. = 1.09) with experience and 3.82 (S.D. = 1.25) without). This suggests that users who have more experience with dialog systems may have a higher expectation and therefore are more critical towards the overall user experience.

6.2 Domain specific questions

We found questions that require sophisticated movie domain knowledge often lack variety among user responses. When asked for a preferred director, most users said they did not care or randomly mentioned a popular director, such as Steven Spielberg. Therefore, we suspect that questions that demand specific domain knowledge may introduce noise and jeopardize the task performance. However, noise is more detrimental to criteria filtering systems than matrix factorization systems because the filtering method looks for an exact match, while matrix factorization can generalize.

6.3 Preferred genres

One movie tends to have multiple genre tags associated with it. Most movie's first tag is action, comedy, or drama. These three genres cover 62% movies in the Netflix corpus. However, the most distinguishing feature is often the second or third tags of the movie, such as sci-fi and horror.

6.4 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.

7 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 a large variety of user utterances. 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. A better language understanding module will likely improve the system's dialog coherence, understanding, and recommendation quality. Similarly, incorporating features that capture the movie's sentiment can lead to improved recommendation satisfaction.

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.

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*.