Self-Supervised Bot Play for Transcript-Free Conversational Recommendation with Justifications

Anonymous ACL submission

Abstract

Conversational recommender systems offer a way for users to engage in multi-turn conversations to find items they enjoy. Dialog agents for conversational recommendation rely on expensive human dialog transcripts, limiting their usage to domains where such data exists. We develop an alternative, two-part framework for training multi-turn conversational recommenders that accommodate a common paradigm of conversation: experts provide and justify suggestions, while users can critique and respond. We can thus adapt conversational recommendation to a wider range of domains where crowd-sourced ground truth dialogs are not available. First, we train a recommender system to jointly suggest items and justify its reasoning via subjective aspects. We then fine-tune this model to incorporate iterative user feedback via self-supervised bot-play. Experiments on three real-world datasets demonstrate that our system can be applied to different recommendation models across diverse domains to achieve state-of-the-art performance in multi-turn recommendation. Human studies show that systems trained with our framework provide more useful, helpful, and knowledgeable suggestions in warm- and cold-start settings.

1 Introduction

Traditional recommender systems often give static suggestions, affording users no way to meaningfully express their preferences and feedback. Conversational recommendation allows users to interact with agents and suggestions, increasing their willingness to trust and accept recommendations (Qiu and Benbasat, 2009). Techniques for conversational recommendation are based on the *paradigm* of conversation: how an agent can explain their suggestions and how users can give feedback.

Recent work has explored conversational recommendation through dialog agents trained to suggest items and ask the user questions in free-form dialog (Wärnestål, 2005). While such models can generate natural-sounding text, they require large training corpora comprising transcripts from crowd-sourced recommendation games (Kang et al., 2019). To create high-quality training data, crowd-workers must be knowledgable about many items in the target domain—this expertise requirement limits data collection to a few common domains like movies. It is thus difficult to scale dialog-based recommenders to domains where users have specific preferences about subjective aspects but no dialog transcripts exist (e.g. food and literature).

	Justification	Multi-Turn	Transcript-Free
LLC (2020a)	✗	✗	✓
CE-VAE (2020b)	✓	✗	✓
M&M VAE (2021)	✓	✗	✓
Li et al. (2018)	✗	✓	✗
Kang et al. (2019)	✓	✓	✗
Zhou et al. (2020)	✓	✓	✗
Ours	✓	✓	✓

Table 1: Critiquing systems (top) are not equipped for multi-turn interactions. Dialog agents (bottom) learn multi-turn behavior via large corpora of domain-specific transcripts. Our framework allows us to train conversational recommenders without costly transcript data.

We address this challenge of data scarcity by proposing a framework for training conversational recommender systems based on conversational critiquing and self-supervised bot-play. Rather than use free-form dialog, many conversational critiquing systems present users with items and natural text aspects that justify their suggestions (Zhou et al., 2020). Users can then critique individual aspects to guide the next turn's recommendations. Our approach reflects this *realistic interactive paradigm* where the agent suggests items and explains their suggestions, while the user specifies their preferences via specific feedback. Our framework does not rely on supervised dialog examples and can be applied to *any* setting where product reviews or opinionated text can be harvested.

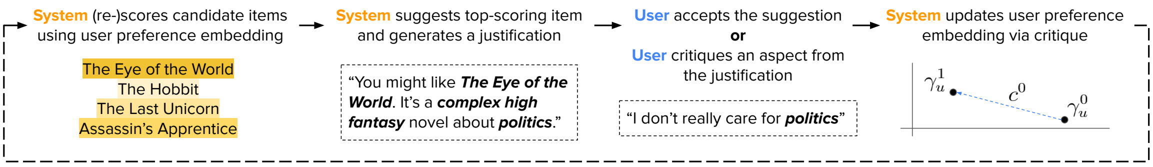

Figure 1: In our conversational recommendation workflow, the system scores candidates and generates a justification for the top item. If the user critiques an aspect, the system uses the critique to update the latent user representation.

We propose a framework comprising two parts: First, we learn to jointly recommend items and generate justifications based on subjective aspects, leveraging ideas from conversational critiquing systems (Wu et al., 2019) trained via next-item recommendation. We then fine-tune our model for multi-turn recommendation via multiple turns of bot-play in a recommendation game based on natural-text product reviews and simulated critiques.

Our framework is model-agnostic—we apply our method to two different underlying recommendation architectures (Sedhain et al., 2016; Rendle et al., 2009) and evaluate our models on three large real-world recommendation datasets with user reviews but no dialog transcripts. Our method reaches goal items faster and with greater success than state-of-the-art (SOTA) methods. We conduct a study with real users, showing that it can effectively help users find desired items in real time, even in a cold-start setting.

We summarize our main contributions as follows: 1) We present a framework for training conversational recommender systems using bot-play on historical user reviews, without the need for large collections of human dialogs; 2) We apply our framework to two popular recommendation models (**BPR-Bot** and **PLRec-Bot**), with each showing superior or competitive performance in comparison to SOTA recommendation and critiquing methods; 3) We demonstrate that our framework can be effectively combined with query refinement techniques to quickly suggest desired items.

2 Related Work

Justifying Recommendations Users prefer recommendations that they perceive to be transparent or justified (Sinha and Swearingen, 2002). Some early recommender systems presented the same attributes of suggested items to all users (Vig et al., 2009). Another line of work attempts to generate natural language explanations of recommendations. McAuley et al. (2012) mine key aspects from textual user reviews via topic extraction. These aspects of interest can be expanded into full sentences, constructed via template-filling (Zhang et al., 2014) or recurrent language models (Ni et al., 2019). Due to their unstructured nature, however, sentence-level justifications have not been used for iteratively refining recommendations. In this work, we allow the user to provide feedback about specific aspects mentioned across natural language product reviews in large recommendation datasets.

Conversational Critiquing Critiquing systems allow users to incrementally construct preferences, mimicking how humans refine their preferences based on conversation context (Tversky and Simonson, 1993). Early critiquing methods treated user feedback as hard constraints to shrink the search space (Burke et al., 1996). Wu et al. (2019) introduced a critiquing model with justifications comprising natural language aspects mined from user reviews—with which users can then interact. Antognini et al. (2020) provide a single-sentence explanation alongside a set of aspects, but require users to interact only with the aspect set. Luo et al. (2020b) use a variational auto-encoder (VAE) (Kingma and Welling, 2014) for joint recommendation and justification, learning a bi-directional mapping function between latent user and aspect representations. Current critiquing techniques are either trained only for next-item recommendation, or to handle a single turn of critiquing (Antognini and Faltings, 2021), and struggle to incorporate feedback in multi-turn settings. We adopt techniques for encoding user feedback from critiquing systems (Luo et al., 2020a), but we introduce a multi-step, model-agnostic bot-play method to explicitly train our models for multi-turn conversational recommendation.

Dialog Agents for Recommendation We view recommenders as domain experts who can elicit preferences from human customers and suggest appropriate items over the course of a session (Burke et al., 1997). A recent line of work formulates conversational recommendation as goal-oriented

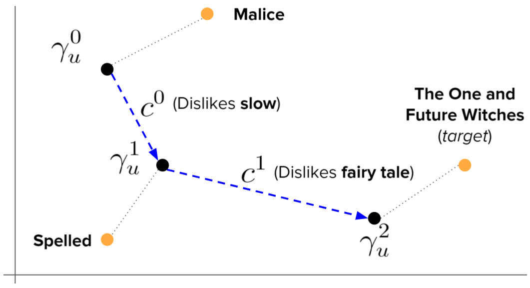

Figure 2: User feedback about aspects (c^0, c^1) modifies our prior latent user preference vector γ_u^0 to bring it closer to the target item embedding.

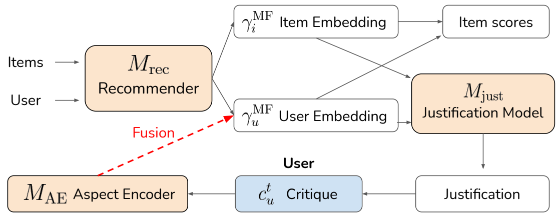

Figure 3: Given a user, items, and aspect critique vector, our model encodes the critique $M_{\text{AE}}(c_u^t)$ and fuses it with the user embedding γ_u^{MF}. The fused user representation γ_u and item representation γ_i are then used to predict the justification and score items.

dialog: at each turn, the user is either a) asked if they prefer a specified aspect; or b) recommended an item (Christakopoulou et al., 2016; Zhang et al., 2018). Bot-play has been explored as a way to train such dialog agents (Li et al., 2018; Kang et al., 2019), which requires models to be trained and fine-tuned using existing dialog transcripts. Such approaches are expensive and limited to domains where crowd-sourced workers can reliably and accurately play the roles of expert and seeker in Wizard-of-Oz style data collection (Dahlbäck et al., 1993). By allowing users to critique natural text aspects of a suggested item, our framework for conversational recommendation allows for multi-turn recommenders that can be trained using only product review texts, widening the scope of domains in which we can train conversational agents. In Table 1 we compare our approach to recent frameworks for critiquing and dialog agents for conversational recommendation.

3 Model

Our model comprises (Figure 3): 1) A matrix factorization recommender model M_{rec} that embeds users and items in an h-dimensional latent space; 2) A justification head M_{just} that predicts the natural language aspects of an item toward which the user holds preferences; and 3) A critiquing function f_{crit} that modifies a user's preference embedding based on aspect-level feedback. We support multi-step critiquing (Figure 2): at each turn a user may indicate which aspects they dislike about the current suggestions via a critique c^t. The critiquing function then modifies the latent user representation γ_u via the critique to bring it closer to the target item.

3.1 Base Recommender System

Our method can be applied to any recommender that learns user and item representations. We show its effectiveness with two popular methods:

Bayesian Personalized Ranking (BPR) (Rendle et al., 2009) is a matrix factorization recommender system that aim to decompose the interaction matrix $\mathbf{R} \in \mathbb{R}^{|U|\times|I|}$ into user and item representations (Koren et al., 2009). BPR optimizes a ranked list of items given implicit feedback (binary interactions between users and items). Scores are computed via inner product of h-dimensional user and item embeddings: $\hat{x}_{u,i} = \langle \gamma_u^{\text{MF}}, \gamma_i^{\text{MF}} \rangle$. At training time, the model is given a user u, observed item i and unobserved item j. We maximize the likelihood that the user prefers the observed item: $\mathcal{L}_R = P(i >_u j|\Theta) = \sigma(\hat{x}_{u,i} - \hat{x}_{u,j})$, where σ represents the sigmoid function $\frac{1}{1+e^{-x}}$.

Projected Linear Recommendation (PLRec) is an SVD-based method to learn low-rank user/item representations via linear regression (Sedhain et al., 2016). The PLRec objective minimizes:

$$\arg\min_{W} \sum_{u} \| r_u - r_u V W^T \|_2^2 + \Omega(W) \quad (1)$$

where V is a fixed matrix obtained by taking a low-rank SVD approximation of $\mathbf{R}$ such that $\mathbf{R} = U\Sigma V^T$, and W is a learned embedding. We obtain an h-dimensional embeddings for users ($\gamma_u^{\text{MF}} = r_u V$) and items ($\gamma_i^{\text{MF}} = W_i$).

3.2 Generating Justifications

Our justification model (aspect prediction head) consists of a fully connected network with two h-dimensional hidden layers predicting a score $s_{u,i,a}$ for each natural language aspect a. This model takes the sum of user and item embeddings as input. At training time, we incorporate an aspect prediction loss $\mathcal{L}_A$ by computing the binary cross entropy (BCE) for each aspect given the likelihood the user cares about the aspect. At inference time, we again compute the likelihood for each aspect

$p_{u,i,a} = \sigma(s_{u,i,a})$ and sample from the Bernoulli distribution with $p_{u,i,a}$ to determine which aspects a appear in the justification.

3.3 Encoding Aspects

We posit that the user's latent representation are partially explained by their written reviews. Thus, we jointly learn an aspect encoder M_{AE} alongside our recommendation model. This takes the form of a linear projection from the aspect space to the user preference space: $M_{\text{AE}}(c_u^t) = W^T c_u^t + b$, where $c_u^t \in \mathbb{Z}^{|K|}$ is the critique vector representing the strength of a user's preference for each aspect. We fuse this aspect encoding with the latent user embedding from M_{rec} to form the final user preference vector: $\gamma_u = f(\gamma_u^{\text{MF}}, M_{\text{AE}}(c_u^t))$. For the BPR-based model, we fuse via summation; for PLRec, we take the mean. In training, the aspect encoder takes in the user's aspect history: $c_u^t = \mathbf{k}_u^U$.

3.4 Training

To train our BPR-based model, we jointly optimize each component. Each training example comprises a user and observed / unobserved items. We predict scores for each item: $\hat{x}_{u,i} = \langle \gamma_u^{\text{MF}} + M_{\text{AE}}(\mathbf{k}_u^U), \gamma_i \rangle$. We first compute the BPR loss (see Section 3.1) with the predicted observed / unobserved scores. We add the aspect prediction loss, scaled by a constant λ_{KP} to the ranking loss for our training objective: $\mathcal{L} = \lambda_{\text{KP}} \mathcal{L}_A - \mathcal{L}_R$. We find empirically that $\lambda_{\text{KP}} \in \{0.5, 1.0\}$ works well.

To train our PLRec-based model, we follow Luo et al. (2020a) and separately optimize M_{rec}, M_{just}, and M_{AE}. The user and item embeddings are learned via eq. (1). We solve the following linear regression problem to optimize M_{AE}:

$$\arg\min_{W,b} \sum_u \| \gamma_u^{\text{MF}} - M_{\text{AE}}(\mathbf{k}_u^U) \|_2^2 + \Omega(W) \quad (2)$$

Finally, we optimize the aspect prediction (justification) loss $\mathcal{L}_A$ to train the justification head.

3.5 Critiquing with Our Models

To perform conversational critiquing with a model trained using our framework, we adapt the latent critiquing formulation from Luo et al. (2020a), as shown in Figure 1. At each turn t of a session for user u, the system assigns scores $\hat{x}_{u,i}^t$ for all candidate items i, and presents the user with the highest scoring item $\hat{i}$. The system also justifies its prediction with a set of predicted aspects $\hat{k}_{u,i}^t$. The user may either accept the recommended item (ending the session) or critique an aspect from the justification: $a \in \{a | \hat{k}_{u,i,a} = 1\}$.

Algorithm 1: Bot play framework for fine-tuning conversational recommenders.

Recommender and Justifier $M_{\text{rec}}, M_{\text{just}}$;
Critique fusion function f_{crit};
Seeker model M_{seeker};
for each user u **do**
  for goal item $g \in I_u^+$ (Reviewed Items) **do**
    initialize loss $\mathcal{L}$;
    initialize γ_u^1 from M_{rec};
    for turn $t \in range(1, T)$ **do**
      compute scores $\hat{x}_{u,i}^t = M_{\text{rec}}(\gamma_u^t, i)\ \forall\, i \in I$;
      $\mathcal{L} \leftarrow \mathcal{L} + \delta^t \cdot \mathcal{L}_{\text{CE}}(g, \hat{x}_{u,i}^t)$;
      recommend item $\hat{i}^t = \arg\max_i \hat{x}_{u,i}^t$;
      if $\hat{i}^t = g$ **then** break with success;
      generate justification $\hat{k}_{u,\hat{i}^t} = M_{\text{just}}(\gamma_u^t, \gamma_{\hat{i}^t})$;
      M_{seeker} critiques justification: c_u^t;
      $\gamma_u^{t+1} \leftarrow f_{\text{crit}}(\gamma_u^t, c_u^t)$;
return fine-tuned agent

Given a user critique, the system modifies the predicted scores for each item and presents the user with a new item and justification:

$$\hat{x}_{u,i}^{t+1} = M_{\text{rec}}(\hat{\gamma}_u^{t+1}, i) \quad (3)$$

$$\hat{k}_{u,i}^{t+1} = M_{\text{just}}(\hat{\gamma}_u^{t+1}, i) \quad (4)$$

$$\hat{\gamma}_u^{t+1} \leftarrow f_{\text{crit}}(\hat{\gamma}_u^t, c_u^t) \quad (5)$$

Effectively, a user critique modifies our prior for the user's preferences; we then re-rank the items presented to the user.

At inference time, c_u^t is the cumulative critique vector, initialized with the user's aspect history:

$$c_u^t = c_u^{t-1} - \max(\mathbf{k}_u^U, 1) \odot m_u^t; \quad c_u^0 = \mathbf{k}_u^U \quad (6)$$

where $\odot$ is element-wise multiplication. Here the critique should match the strength of a user's previous opinion of the aspect $\mathbf{k}_u^U$. Even if a user has not mentioned an aspect in their previous reviews, the max ensures a non-zero effect from each critique.

3.6 Learning to Critique via Bot Play

We propose a framework for critiquing via bot play that simulates user sessions when provided just a set of user reviews. We first pre-train our expert model (recommender, justifier, and aspect encoder). A seeker model is pre-trained via a simple prior: provided a target item and justification, it selects the most popular aspect present in the justification but not the target's historical aspects $\mathbf{k}_i^I$ to critique. For each training example (user

	Users	Items	Reviews	A	A/I	A/U
Books	13,889	7,649	654,975	75	27.0	25.0
Beer	6,369	4,000	935,524	75	60.2	54.6
Music	5,635	4,352	119,081	80	20.0	16.5

Table 2: Dataset statistics, including avg. unique aspects mentioned in reviews per item (A/I) and user (A/U).

and a goal item they have reviewed), we allow the expert and seeker models to converse with the goal of recommending the goal item. We fine-tune the expert by maximizing its reward (minimizing loss) in the bot-play game (Algorithm 1). We end the session after the goal item is recommended or a maximum session length of $T = 10$ turns is reached. We define the expert's loss as the cross entropy loss of recommendation scores per turn: $\mathcal{L}^{\text{expert}} = \sum_t^T \delta^{t-1} \cdot \mathcal{L}_{\text{CE}}(g, \hat{x}^t_{u,i})$ where δ is a discount factor[1] to encourage successfully recommending the goal item at earlier turns, and $\mathcal{L}_{\text{CE}}(g, \hat{x}^t_{u,i})$ is the cross-entropy loss between predicted scores and the goal item.

4 Experimental Setting

We select hyperparameters for our initial models via AUC, and for bot-play fine-tuning via the success rate at 1 (SR@1) on the validation set. We train each model once, taking the median of three evaluation runs per experimental setting. For baseline models, we re-used the authors' code. We will release code upon publication.

Datasets We evaluate our models on three public real-world recommendation datasets with 100K+ reviews each: Goodreads Fantasy (Books) (Wan and McAuley, 2018), BeerAdvocate (Beer) (McAuley et al., 2012), and Amazon CDs & Vinyl (Music) (McAuley et al., 2015). We keep only reviews with positive ratings, setting thresholds of $t > 4.0$ for Beer and Music and $t > 3.5$ for Books. We partition each dataset into 50% training, 20% validation, and 30% test splits Table 2.

We follow the pipeline of Wu et al. (2019) to extract subjective aspects from user reviews: 1) Extract high-frequency unigram and bigram noun- and adjective phrases; 2) Prune bigram keyphrases using a Pointwise Mutual Information (PMI) threshold, ensuring aspects are statistically unlikely to have randomly co-occurred; and 3) Represent reviews as sparse binary vectors indicating whether each aspect was expressed in the review. Aspects describe qualities ranging from taste for beers (e.g. citrus) and emotions for music (e.g. soulful) to perceived character qualities in books (e.g. strong female).

Multi-Step Critiquing Following prior work on critiquing (Luo et al., 2020a; Li et al., 2020), we simulate multi-step recommendation sessions to assess model performance. We simulate user sessions following Algorithm 1, with two main differences: (1) We randomly sample user u and their goal item g from the *test* set, and (2) We do not compute loss or update our model during a session. We set a maximum session limit of $T = 10$ turns.

To evaluate how our models behave with different user behaviors, we simulate each observation with three different critique selection strategies (Li et al., 2020): 1) **Random**: We assume the user randomly chooses an aspect—this assumes no prior knowledge on the part of the user; 3) **Pop**: We assume the user selects the most popular aspect used across all training reviews; and 3) **Diff**: We assume the user selects the aspect that deviates most from the goal item reviews—the aspect with the largest frequency differential between the goal item and current item: $\arg\max_a(\mathbf{k}^I_{\hat{i}^t,a} - \mathbf{k}^I_{g,a})$. In all settings, a user may only see any single item once and may only critique each aspect once per session.

Candidate Algorithms Our method can apply to any base recommender system; here we train bot-play models based on BPR and PLRec—**BPR-Bot** and **PLRec-Bot** respectively. We assess linear critiquing baselines that co-embed critique and user representations (Luo et al., 2020a), where f_{crit} is a weighted sum of the user preference vector γ_u and embeddings for each critiqued aspect. **UAC** uniformly averages γ_u and all critiqued aspect embeddings. **BAC** averages γ_u with the *average* of critiqued aspect embeddings. **LLC-Score** learns weights by maximizing the rating margin between items containing critiqued aspects and those without. Instead of directly optimizing the scoring margin, **LLC-Rank** (Li et al., 2020) minimizes the number of ranking violations. These models cannot generate justifications; we binarize the historical aspect frequency vector for the item ($\mathbf{k}^I_{u,\hat{i}^t}$) as a justification at each turn. We also compare against a SOTA interactive recommender, **CE-VAE** (Luo et al., 2020b), which learns a VAE with a bidirectional mapping between critique vectors and the

[1]We use a discount factor of $\delta = 0.9$

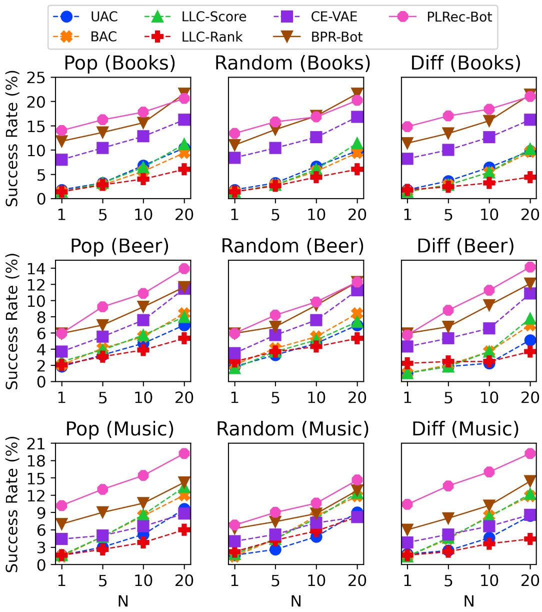

Figure 4: Success Rate @ N (% sessions where target item rank $\leq$ N) across datasets and user models. BPR-Bot (brown triangle) and PLRec-Bot (pink circle) outperform baselines (dashed) in all settings.

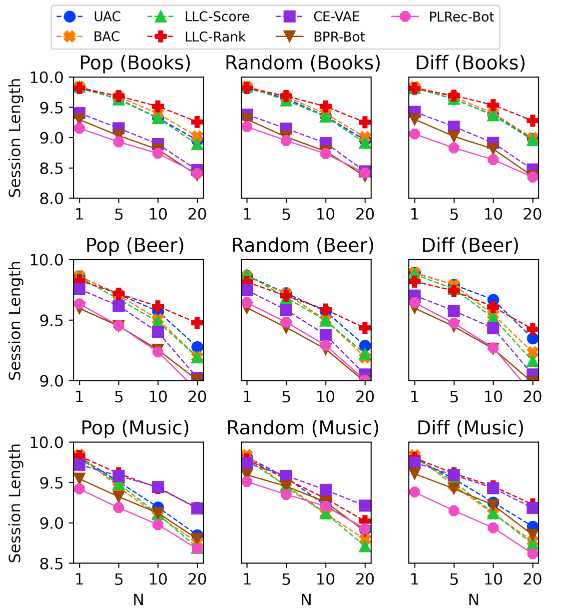

Figure 5: Avg. # of turns for target item to reach rank N, across datasets and user models. BPR-Bot (brown triangle) and PLRec-Bot (pink circle) promote targets faster than baselines (dashed), especially for low N.

user latent preference space.

5 Experiments

RQ1: Can our framework enable multi-step critiquing? We measure multi-step critiquing performance via average success rate (Figure 4)–the percentage of sessions where the target item reaches rank N—and session length (Figure 5). As our bot-play fine-tuning seeker model picks critiques by popularity, we expect our models to perform best in the Pop setting. However, BPR-Bot and PLRec-Bot succeed faster and at a higher rate than baselines in *all* user settings, including random critiquing with no prior on user behavior. Linear critiquing models (UAC, BAC, LLC-Score/Rank) perform poorly on multi-step critiquing compared to models that can generate justifications. This suggests that personalized justifications help users choose more effective aspects to critique.

Our models can also generate personalized justifications that are more helpful for narrowing down a user's preferences compared to CE-VAE: BPR-Bot and PLRec-Bot out-perform the baseline in all settings. We have thus shown that our bot-play framework enables the training of multi-turn conversational recommenders *without the need for costly supervised dialog transcripts*.

In general, the large item space makes it difficult for critiquing models to reach the goal item within the turn limit, with the best model reaching the goal item in only 6-15% of sessions. This suggests that practical conversational recommenders may benefit from constraint-based filtering as well as an initial set of user requirements—while users often start a session with a seed set of reqirements—e.g. in car buying, whether they want an SUV or coupe (Pu and Faltings, 2000). We demonstrate in RQ3 that our model can be combined with constraint-based query refinement to quickly achieve significantly higher success rates.

RQ2: Does *bot-play* specifically improve multi-step critiquing ability? We next demonstrate that our bot-play fine-tuning is responsible for gains in multi-step critiquing performance by comparing BPR-Bot (left) and PLRec-Bot (right) in Figure 6 against ablated versions that were trained using the first step of our framework but *not* fine-tuned via bot-play. For clarity, we display only results using the Pop user behavioral model, as we observe the same trends with the Random and Diff user models. In domains with relatively high aspect occurrence across reviews (Books, Beer), bot-play confers a 3-6% improvement in success rate for various N. This demonstrates that we can effectively train conversational recommender systems using our bot-play framework using domains with rich user reviews

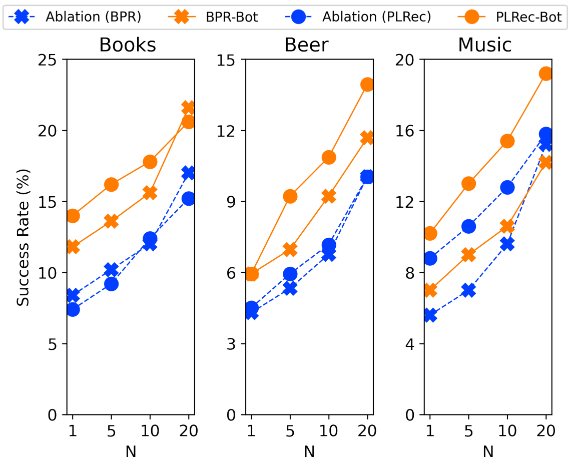

Figure 6: Success Rate @ N (% sessions where target item rank $\leq$ N), comparing bot-play (orange) against non-bot-play ablations (blue). Bot-play improves target item ranking across datasets compared to the ablation, for both BPR-Bot (crosses) and PLRec-Bot (circles).

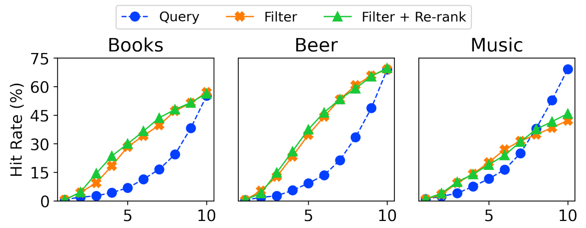

Figure 7: Hit rate by turn for query refinement models on each dataset with multi-step critiquing up to 10 turns.

in lieu of crowd-sourced dialog transcripts. In domains with more sparse coverage of subjective aspects (i.e. Music), we observe lower improvement when using bot-play—our model may encounter insufficient cases of rare aspects being critiqued. In future work, we will explore adding noise to our user model to ensure that the bot-play process encounters more rare aspects.

We confirm that our method is model-agnostic, as it improves recommendation success rates for both the matrix factorization-based (BPR) and linear (PLRec) recommender systems. Models with a higher latent dimensionality ($h \in [50, 400]$ for PLRec-Bot vs. h=10 for BPR-Bot) benefit more from bot-play, suggesting that our method learns to effectively navigate complex preference spaces.

RQ3: Can our models be effectively combined with query refinement? So far, we have assumed that users provide *soft* feedback: even if a user has critiqued aspect a during a session, future suggested items may still contain aspect a. This assumption holds for some aspects: for example, even if previous users mentioned that a song was dispassionate, a user may find it emotional and enjoyable. However, the user may reject the suggestion right after reading reviews. We thus try treating critiques as hard constraints: users should not receive items whose reviews mention critiqued aspects. We compare three models with turn-0 ranked lists of candidate items initialized from BPR-Bot. The **Query** baseline model suggests an item each turn and asks the user if they like aspect a—the aspect that most evenly divides the remaining candidate items: $\arg\min_a ||I_a^+| - |I_a^-||$. The **Filter** model generates a justification for each suggested item that the user can critique. The hybrid **Filter+Re-rank** model incorporates our learned critiquing function to modify the user preference vector and re-rank the remaining candidate items. We conduct user simulations with the Pop user model and plot the hit rate by turn—rate of achieving the goal item g at or before turn t—in Figure 7.

While binary queries guarantee targets will eventually be found, the queried aspect may be unrelated to suggested items. Models that allow users to critique justifications reach high success rates much faster than binary querying in the first 6-10 turns. Re-ranking after filtering improves performance across domains, suggesting that we have learned how user critiques relate to their latent preferences for other aspects.

For the Beer and Books domains, the filtering approach reaches higher success rates compared to binary querying within the session turn limit (70.7% vs. 69.7% and 57.0% vs. 55.2%, respectively). We see less of a benefit in the Music domain. Aspect sparsity may play a role: per Table 2, only 25% of possible aspects are expressed for the average item. Music also contains a longer tail of rare (expressed only for a few items) aspects compared to Books and Beer—as such, user critiques prune fewer items on average.

Our bot-play framework can be easily adapted to train models incorporating hard critiquing constraints by pruning candidate items. One possible extension involves masking the fine-tuning loss to only adjust the scores of non-pruned items, setting pruned item scores to a large negative value: $\hat{x}_{u,i} = -1e15 \; \forall \; i \in I_a^+$. We also wish to explore fine-tuning with a ranking loss during bot-play, to encourage the model to rank items containing a critiqued aspect $i \in I_a^+$ below those without.

BPR-Bot vs	Useful W	Useful L	Informative W	Informative L	Knowledgeable W	Knowledgeable L	Adaptive W	Adaptive L
Ablation	**78**	10	**73**	11	**68**	15	**85**	5
CE-VAE	**83**	9	**74**	10	**63**	16	**81**	8
PLRec-Bot vs	Useful W	Useful L	Informative W	Informative L	Knowledgeable W	Knowledgeable L	Adaptive W	Adaptive L
Ablation	**86**	5	**78**	7	**74**	8	**81**	9
CE-VAE	**87**	7	**79**	11	**77**	12	**83**	10

Table 3: Session-level human evaluation via ACUTE-EVAL. W/L percentages are reported while ties are not. All results statistically significant with $p < 0.05$.

	Useful	Informative	Adaptive	Like
No Bot	0.67±0.24	0.75±0.21	0.64±0.27	41%
Ours	**0.79±0.24**	**0.88±0.18**	**0.78±0.23**	**69%**

Table 4: Turn- and session-level feedback from cold-start user study. Statistically significant results in **bold**.

6 Human Study

Human Evaluation Following Li et al. (2019), we conduct a comparative evaluation of 100 simulated user sessions on four criteria: which agent seems more useful, informative, knowledgeable and adaptive. We compare each bot-play model (**BPR-Bot** and **PLRec-Bot**) against an ablative version (with no bot-play) and the best baseline (CE-VAE). Each sample is evaluated by three annotators. We observe substantial inter-annotator agreement, with Fleiss κ (Fleiss and Cohen, 1973) of 0.67, 0.79, 0.73, and 0.60 for the usefulness, informativeness, knowledgeable, and adaptiveness criteria, respectively. Scores are shown in Table 3.

BPR-Bot and PLRec-Bot are judged to be significantly more informative and knowledgeable than ablative models and CE-VAE, showing that our justification module accurately presents important aspects of each suggestion. The usefulness and adaptiveness criteria capture how models help the user achieve their end goal (i.e. finding the most relevant item in as few turns as possible). Bot-play models are judged to be more useful than alternatives and follow critiques more consistently when adapting their recommendations. Our framework allows us to train conversational agents that are useful and engaging for human users: evaluators overwhelmingly judged the models trained via bot-play to be more useful, informative, knowledgeable, and adaptive compared to CE-VAE and ablated variants.

Cold-Start User Study We conduct a user study using the Books dataset to evaluate if our model is a useful real-time conversational recommender. We recruited 64 human users—half interacting with **BPR-Bot** and half with the ablation (no bot-play). We initialize each session with the mean of all learned user embeddings. At each turn, the user sees the three top-ranked items with justifications (aspects) and can critique multiple aspects. On average, users critiqued two aspects per turn.

At each turn, we again ask users if the generated justifications are *informative*, *useful* in helping to make a decision, and whether our system *adapted* its suggestions in response to the user's feedback. We provide four options for each question: no/weak-no/weak-yes/yes, mapping these values to a score between 0 and 1 (Kayser et al., 2021), with normalized aggregated scores for each question in Table 4. **BPR-Bot** significantly out-scores the ablation in all three metrics ($p < 0.01$), showing that fine-tuning via our bot-play framework instills a stronger ability to respond to critiques and provide meaningful justifications—even for unseen users. At the end of a session, we additionally ask the user how frequently (if at all) they would choose to engage with our interactive agent in their daily life. Users preferred BPR-Bot by significant margins—69% indicated they would "often" or "always" use BPR-Bot to find books compared to 41% for the ablation.

7 Conclusion

In this work we develop conversational recommenders that can engage with users over multiple turns, justifying suggestions and incorporating feedback about item aspects. We present a model-agnostic framework for training conversational recommenders in this modality via self-supervised bot-play in any domain with only review data. We use two popular underlying recommender systems to train the **BPR-Bot** and **PLRec-Bot** agents using our framework, showing quantitatively on three datasets that our models 1) offer superior multi-turn recommendation performance compared to current SOTA methods; 2) can be effectively combined with query refinement to quickly converge on suitable items; and 3) can effectively refine suggestions in real-time, as shown in user studies. In future work, we aim to adapt our framework to natural language critiques (i.e. utterances), allowing users to more flexibly express feedback.

References

Diego Antognini and Boi Faltings. 2021. Fast multi-step critiquing for vae-based recommender systems. *CoRR*, abs/2105.00774.

Diego Antognini, Claudiu Musat, and Boi Faltings. 2020. Interacting with explanations through critiquing. *CoRR*, abs/2005.11067.

Robin D. Burke, Kristian J. Hammond, and Benjamin C. Young. 1996. Knowledge-based navigation of complex information spaces. In *AAAI*.

Robin D. Burke, Kristian J. Hammond, and Benjamin C. Young. 1997. The findme approach to assisted browsing. *IEEE Expert*, 12(4):32–40.

Konstantina Christakopoulou, Filip Radlinski, and Katja Hofmann. 2016. Towards conversational recommender systems. In *KDD*.

Nils Dahlbäck, Arne Jönsson, and Lars Ahrenberg. 1993. Wizard of oz studies: why and how. In *IUI*.

Joseph L Fleiss and Jacob Cohen. 1973. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and psychological measurement*, 33(3):613–619.

Dongyeop Kang, Anusha Balakrishnan, Pararth Shah, Paul A. Crook, Y-Lan Boureau, and Jason Weston. 2019. Recommendation as a communication game: Self-supervised bot-play for goal-oriented dialogue. In *EMNLP-IJCNLP*.

Maxime Kayser, Oana-Maria Camburu, Leonard Salewski, Cornelius Emde, Virginie Do, Zeynep Akata, and Thomas Lukasiewicz. 2021. e-vil: A dataset and benchmark for natural language explanations in vision-language tasks. *CoRR*, abs/2105.03761.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *ICLR*.

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

Hanze Li, Scott Sanner, Kai Luo, and Ga Wu. 2020. A ranking optimization approach to latent linear critiquing for conversational recommender systems. In *RecSys*.

Margaret Li, Jason Weston, and Stephen Roller. 2019. Acute-eval: Improved dialogue evaluation with optimized questions and multi-turn comparisons.

Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. 2018. Towards deep conversational recommendations. In *NeurIPS*.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2020. On the variance of the adaptive learning rate and beyond. In *ICLR*.

Kai Luo, Scott Sanner, Ga Wu, Hanze Li, and Hojin Yang. 2020a. Latent linear critiquing for conversational recommender systems. In *WWW*.

Kai Luo, Hojin Yang, Ga Wu, and Scott Sanner. 2020b. Deep critiquing for vae-based recommender systems. In *SIGIR*.

Julian J. McAuley, Jure Leskovec, and Dan Jurafsky. 2012. Learning attitudes and attributes from multi-aspect reviews. In *ICDM*.

Julian J. McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-based recommendations on styles and substitutes. In *SIGIR*.

Jianmo Ni, Jiacheng Li, and Julian J. McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *EMNLP*.

Pearl Pu and Boi Faltings. 2000. Enriching buyers' experiences: the smartclient approach. In *CHI*.

Lingyun Qiu and Izak Benbasat. 2009. Evaluating anthropomorphic product recommendation agents: A social relationship perspective to designing information systems. *J. Manag. Inf. Syst.*, 25(4):145–182.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: bayesian personalized ranking from implicit feedback. In *UAI*.

Suvash Sedhain, Hung Bui, Jaya Kawale, Nikos Vlassis, Branislav Kveton, Aditya Krishna Menon, Trung Bui, and Scott Sanner. 2016. Practical linear models for large-scale one-class collaborative filtering. In *IJCAI*.

Rashmi R. Sinha and Kirsten Swearingen. 2002. The role of transparency in recommender systems. In *CHI*.

Amos Tversky and Itamar Simonson. 1993. Context-dependent preferences. *Management Science*, 39(10):1179–1189.

Jesse Vig, Shilad Sen, and John Riedl. 2009. Tagsplanations: explaining recommendations using tags. In *IUI*.

Mengting Wan and Julian J. McAuley. 2018. Item recommendation on monotonic behavior chains. In *RecSys*.

Pontus Wärnestål. 2005. Modeling a dialogue strategy for personalized movie recommendations. In *Beyond Personalization Workshop*, pages 77–82.

Ga Wu, Kai Luo, Scott Sanner, and Harold Soh. 2019. Deep language-based critiquing for recommender systems. In *RecSys*.

Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W. Bruce Croft. 2018. Towards conversational search and recommendation: System ask, user respond. In *CIKM*.

Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *SIGIR*.

Kun Zhou, Wayne Xin Zhao, Shuqing Bian, Yuanhang Zhou, Ji-Rong Wen, and Jingsong Yu. 2020. Improving conversational recommender systems via knowledge graph based semantic fusion. In *KDD*, pages 1006–1014. ACM.

A Additional Experimental Details

All experiments were conducted on a machine with a 2.2GHz 40-core CPU, 132GB memory and one RTX 2080Ti GPU. We use PyTorch version 1.4.0 and optimize our models using the Rectified Adam (Liu et al., 2020) optimizer. Best hyperparameters for each base recommender system model are shown in Table 6. We perform hyperparameter search over a coarse sweep of: $h \in [2, 500]$, $LR \in [1e-5, 1e-2]$, $\lambda \in [1e-5, 1e-2]$. Model parameter sizes are a function of the hidden dimensionality h and number of items $|I|$ and users $|U|$, and is dominated by $h \cdot (|I| + |U|)$.

As mentioned in Section 4, we re-use the authors' publicly code with relevant citations following intended usage (academic research). This includes usage of the NLTK package[2] to extract unigrams and bigrams from natural text reviews. We will release our code under the MIT license.[3]

All code and reviews in this dataset are in English. We hope to extend our work to identify related aspects in multi-lingual reviews in the future.

B Time Complexity

In Table 5, we report the mean and standard error of time taken per turn for LLC-Score, CE-VAE, BPR-Bot, and PLRec-Bot. As baseline code does not leverage the GPU, we also critique with PLRec-Bot and BPR-Bot on the CPU only. We observe LLC-Score and PLRec-Bot to be an order of magnitude slower per critiquing cycle compared to CE-VAE and BPR-Bot. BPR-Bot shows acceptable latency for real-world applications (sub-10 ms), and we observe empirically in our cold-start user study that we can host BPR-Bot as a real-time recommendation service. Time trials were conducted with batch size of 1; production throughput can be improved further with parallel processing. Each model executes using a different framework (`numpy` for LLC-Score, `Tensorflow` for CE-VAE, and `Pytorch` for PLRec-Bot/BPR-Bot), which may contribute to differences in inference speed.

C Human Evaluation

The datasets we used have been processed to remove offensive words and phrases before presenting them to human evaluators and users. We perform our human evaluation via the Amazon Me-

[2] https://www.nltk.org/

[3] https://opensource.org/licenses/MIT

chanical Turk (MTurk) platform, recruiting crowd-workers with a historical 99% acceptance rate on their work to ensure quality, and no other limitations. Crowd-workers were paid in excess of Federal minimum wage in the United States given the average time taken to complete an evaluation. Participants in the user study were recruited from Universities in the United States.

Users in both our user evaluation and user study were permitted to exit the task at any time and have their interactions wiped from the project. We do not collect biometrics or personally identifiable information (PII) from users in our user study, and users were informed that this study was part of an academic research project and may be published.

An image of the interface presented to crowd-workers in our human evaluation is shown in Figure 8. For the human evaluation, we presented two user simulation traces from different models (e.g. PLRec-Bot and CE-VAE) in a random order, then ask users to decided which of the two models is more useful, which is more informative, which is more knowledgeable, and which is more adaptive. Each user simulation trace is for the same user and target item, to be able to fairly compare models.

An image of the interface used for our cold-start user study is shown in Figure 9.

D Risks

As we aim to train conversational multi-turn recommendation agents, the primary risks of our approach lie in taking too long to present a user with good items or suggesting items they dislike. This risk is not unique to our approach and to some extent depends on the target domain (e.g. users may hold stronger opinions about food than they do computer hardware). One risk surface is the natural language aspects (and product names) that we surface to users as part of our recommend-and-justify approach. These could theoretically contain offensive or uncomfortable phrasing, but this risk can be minimized by a human-in-the-loop review of the aspect extraction process (e.g. blacklisting certain extracted aspects) or by applying toxic text detection to filter user reviews as a pre-processing step.

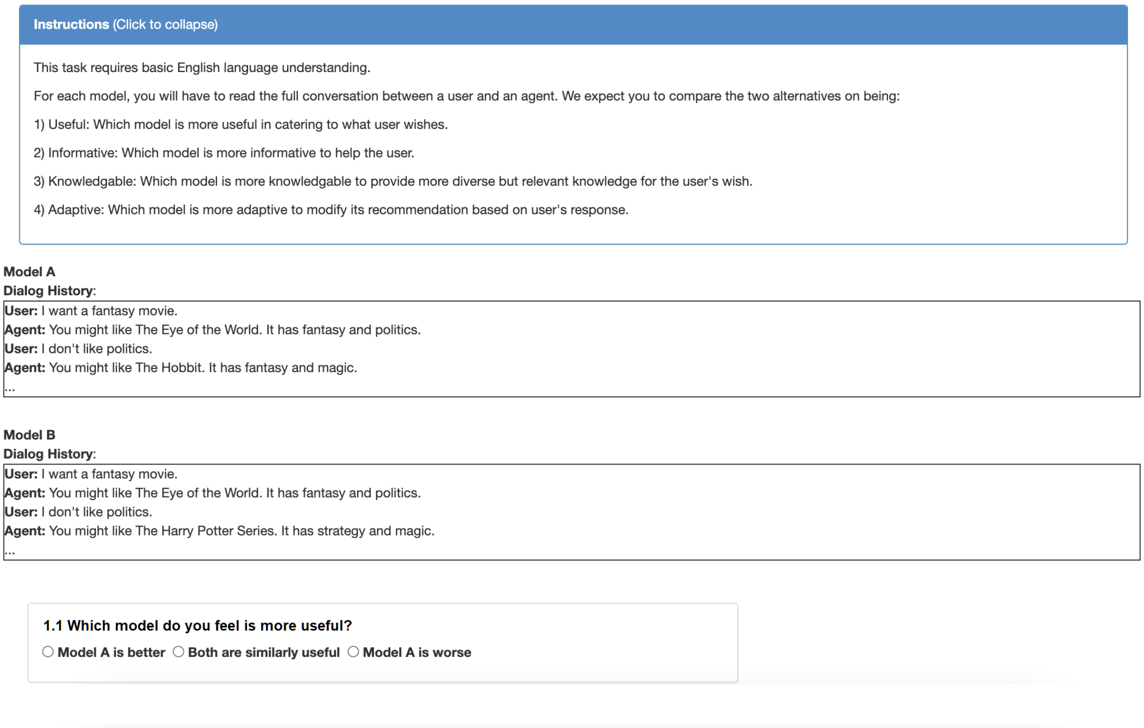

Figure 8: User interface for user evaluation, with two placeholder conversations. Users are asked which of the two models (presented in random order) is more useful, informative, knowledgeable, and adaptive.

	LLC-Score	CE-VAE	BPR-Bot	PLRec-Bot
Books	40.64 ± 20.46	4.61 ± 1.16	2.70 ± 3.95	48.84 ± 14.08
Beer	15.94 ± 14.52	3.26 ± 1.18	2.54 ± 2.36	49.43 ± 14.81
Music	42.21 ± 21.04	3.36 ± 1.37	2.25 ± 0.62	6.80 ± 7.53

Table 5: Mean and standard error of wall-clock time (ms) per turn of critiquing for linear (LLC-Score) and variational (CE-VAE) baselines vs. our models (BPR-Bot, BPR-PLRec)

Dataset	Model	h	LR	λ_{L2}	λ_{KP}	λ_c	β	Epoch	Dropout
Books	BPR	10	0.001	0.01	0.5	–	–	200	–
	PLRec	50	–	80	–	–	–	10	–
	CE-VAE	100	0.0001	0.0001	0.01	0.01	0.001	300	0.5
Beer	BPR	10	0.001	0.01	0.5	–	–	200	–
	PLRec	50	–	80	–	–	–	10	–
	CE-VAE	100	0.0001	0.0001	0.01	0.01	0.001	300	0.5
Music	BPR	10	0.01	0.1	1.0	–	–	200	–
	PLRec	400	–	1000	–	–	–	10	–
	CE-VAE	200	0.0001	0.0001	0.001	0.001	0.0001	600	0.5

Table 6: Best hyperparameter settings for each base recommendation model. UAC, BAC, LLC-Score, LLC-Rank models use PLRec as a base model. BPR-Bot uses BPR as a base model.

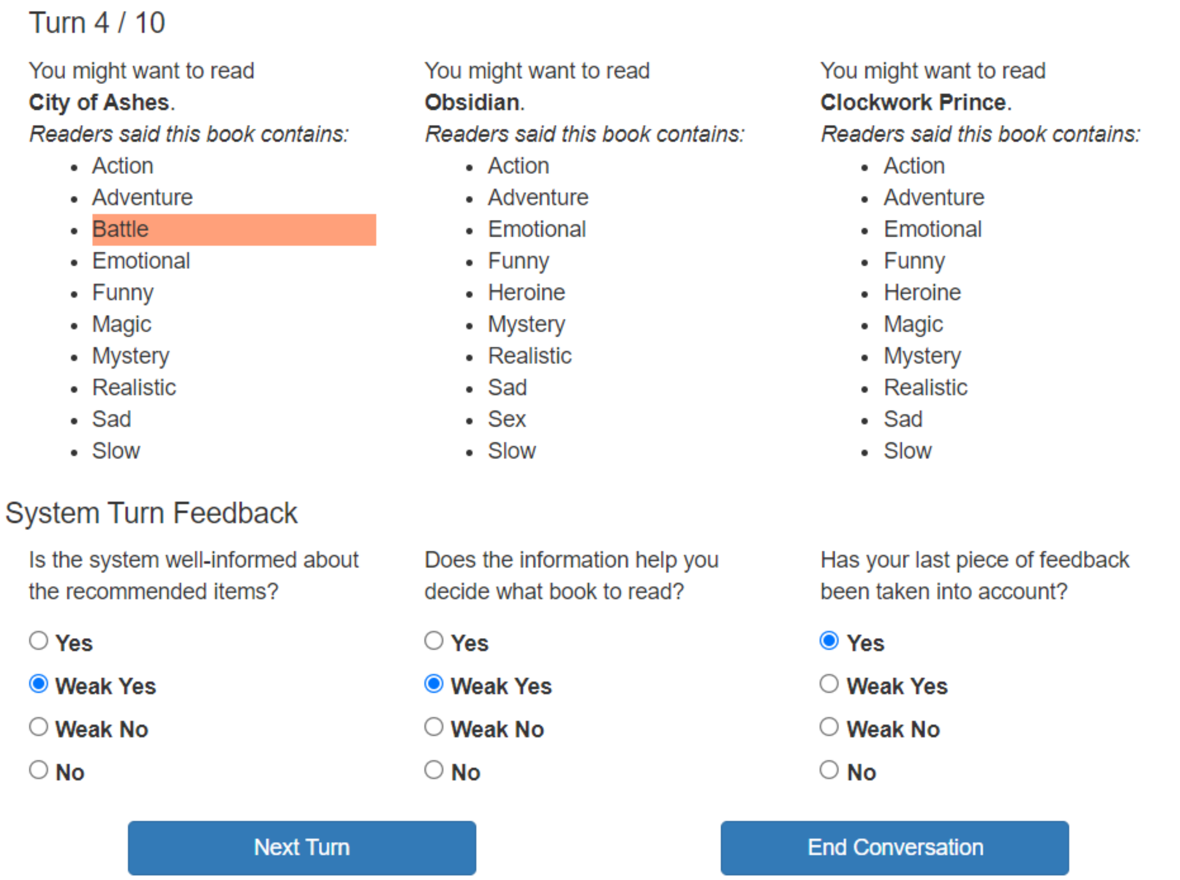

Figure 9: User interface for user study, with turn-level feedback prompts and an example of a critiqued aspect ("Battle")