

Talk the Walk: Synthetic Data Generation for Conversational Music Recommendation

Megan Leszczynski^{*†}
Stanford University
mleszczy@cs.stanford.edu

Ravi Ganti^{*}
Google Research
gmravi@google.com

Shu Zhang^{*}
Google Research
shzhang@google.com

Krisztian Balog
Google Research
krisztianb@google.com

Filip Radlinski
Google Research
filiprad@google.com

Fernando Pereira
Google Research
pereira@google.com

Arun Tejasvi Chaganty^{*}
Google Research
arunchaganty@google.com

ABSTRACT

Recommendation systems are ubiquitous yet often difficult for users to control and adjust when recommendation quality is poor. This has motivated the development of conversational recommendation systems (CRSs), with control over recommendations provided through natural language feedback. However, building conversational recommendation systems requires conversational training data involving user utterances paired with items that cover a diverse range of preferences. Such data has proved challenging to collect scalably using conventional methods like crowdsourcing. We address it in the context of item-set recommendation, noting the increasing attention to this task motivated by use cases like music, news and recipe recommendation. We present a new technique, TalkTheWalk, that synthesizes realistic high-quality conversational data by leveraging domain expertise encoded in widely available curated item collections, showing how these can be transformed into corresponding item set curation conversations. Specifically, TalkTheWalk generates a sequence of hypothetical yet plausible item sets returned by a system, then uses a language model to produce corresponding user utterances. Applying TalkTheWalk to music recommendation, we generate over one million diverse playlist curation conversations. A human evaluation shows that the conversations contain *consistent* utterances with *relevant* item sets, nearly matching the quality of small human-collected conversational data for this task. At the same time, when the synthetic corpus is used to train a CRS, it improves Hits@100 by 10.5 points on a benchmark dataset over standard baselines and is preferred over the top-performing baseline in an online evaluation.

1 INTRODUCTION

Recommendation systems (RSs) help users choose from an overwhelming number of options, while having a significant impact on many businesses [30]. Traditionally, these RSs tend to rely on historical interaction data, such as viewing logs and user ratings, to personalize recommendations. Yet it is well recognized that relying only on such data makes it challenging for users to control recommendations based on their current context or when preferences change, and also hampers recommendation quality when limited historical data is available as in a cold-start setting [38]. Recent advances in natural language processing have enabled the development of *conversational recommendation systems* (CRSs) as a step towards addressing many of these challenges, allowing users

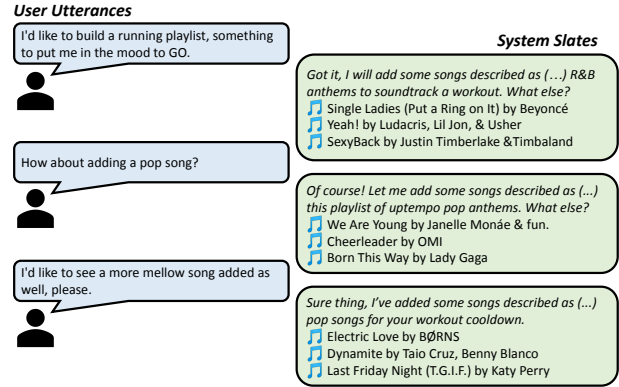


Figure 1: An example of a synthetic playlist curation conversation generated using our approach (shortened for presentation). Our domain-agnostic approach generates user utterances and system slates of recommended items.

to obtain improved recommendations following a back-and-forth with the system [24, 38]. Specifically, the natural language feedback allows users to steer the system (e.g., “How about some more upbeat music?”), reducing the need for historical data and providing recommendations that better fit the user’s current preferences (Figure 1). While significant progress has been made (see [87] for an overview, in addition to more recent advances in large language modeling [26, 28]), practical adoption is still limited. One of the key challenges is a lack of training data with natural, diverse and coherent multi-turn conversations paired with relevant item recommendations in each turn—meaning existing CRSs are unable to deliver the full control they desire in principle support. Our work addresses this challenge.

In previous work, data has been derived from actual user-system interactions [14], or from human-human interactions in a mocked-up system (“Wizard of Oz” experiments) [33, 54, 58, 62]. In the first case, data is biased by the limited capabilities of the existing system; in the second, by the difficulty in instructing participants to produce diverse yet on-topic conversation. These factors make it prohibitively expensive to collect large high-quality datasets in any domain, even through crowdsourcing [10, 45, 54, 83].

Zamani et al. [87] point out that “a fundamental characteristic of conversational recommender systems is that they support specific tasks and goals.” The primary objective of most existing

^{*}Core contributor.

[†]Work done while interning at Google Research.

recommender systems, albeit often not explicitly stated, is to assist the user in finding a specific item to purchase or consume: while the system may facilitate the exploration of alternative options, the goal is identifying a single item that meets the user’s needs and preferences. In this paper, we address a different and so far largely unexplored recommendation task: aiding the user in the compilation of a *set* of items satisfying a certain set of criteria or preferences, e.g., creating a music playlist for a long drive [12].

Given the challenges in collecting real recommendation conversations, for a novel task that has not been extensively studied, we instead show how to generate *high quality* synthetic conversations using existing non-conversational artifacts already frequently curated by day to day users. In particular, we observe that curated user generated content in the form of item collections such as playlists, movie watchlists, or recipe books, are widely available and capture the diverse domain expertise of their creators. These collections include coherent sets of items with titles, descriptions, and other metadata that reveal soft attributes [3] which users often express in preferences (e.g., “upbeat music,” “feel-good movies,” “healthy recipes”).

However, item collections lack two essential ingredients for conversational recommendation: (1) multiple turns with user utterances describing their preferences, and (2) corresponding items to recommend to the user. Our method, TalkTheWalk (TtW), tackles these problems in reverse order: generate slates (a set of recommended items [42]), then generate corresponding utterances (Figure 1). Concretely, we represent items and item collections in a shared embedding space using a standard dual encoder architecture [29, 51]. We then perform a biased random walk in this embedding space to pick a consistent sequence of slates from the item collections. Finally, we use the metadata from the item collections to prompt a dialog inpainting language model [16] to generate conversational user utterances that express preferences for each slate. Our approach is scalable (similar to historical interaction data), privacy-preserving (similar to crowdsourced datasets), and generates representative conversations that allow a CRS to be trained from scratch.

We use TtW to create TtWMusic, a dataset of over one million synthetic playlist curation conversations covering a breadth of domain expertise, from Japanese pop to electro-swing music. Music recommendation is an excellent example where conversational recommendation offers tremendous potential, as it provides a mechanism for a long playlist to be incrementally refined by a user reacting to what they are (or are not) hearing. In a crowdsourced evaluation, we find that over 70% of the generated conversations are realistic, and the consistency of the utterances and relevancy of the slates is comparable to that of a human-collected dataset. Moreover, we estimate that creating TtWMusic costs less than \$1,000, making it a cost-effective alternative to crowdsourcing.¹ We also evaluate TtWMusic by measuring its utility for bootstrapping a conversational music recommendation system through offline and online evaluation. Offline, we see up to a 2.9 point and 10.5 point improvement in Hits@10 and Hits@100, respectively, on a benchmark dataset compared to standard sparse and dense retrieval baselines in a zero-shot setting. Online evaluation shows that raters

more frequently prefer recommendations from our system to the top-performing baseline. In summary:

- We introduce TalkTheWalk (TtW), a novel method to generate training data for CRSs that leverages curated item collections and demonstrate it produces realistic item set curation conversations.
- We apply TtW on expert-curated music playlists, generating TtWMusic, a synthetic conversational music recommendation dataset with over one million conversations. We validate data quality in a quantitative comparison to a human-collected dataset.
- We demonstrate that TtWMusic can be used to bootstrap a conversational music recommendation system, outperforming standard retrieval baselines in offline and online evaluations that were trained without using such synthetic data.

2 RELATED WORK

This paper addresses the task of item-set recommendation in a conversational setting using synthetically generated data. Below, we review related work in each of these areas.

2.1 Item Set Recommendation

Traditionally, recommender systems focus on helping the user find a specific item that they want to buy or consume. Recently, several real-life scenarios have been identified where recommendations consist of a set of complementary items that need to be considered together, e.g., when recommending outfit [13] or shopping baskets [81], highlighting the importance of methods that can suggest item sets rather than just individual items. The generation of music playlists has also been identified as a specific recommendation problem, where the task is to “create a sequence of tracks fulfilling the target characteristics in the best possible way” [8]. In this paper, we do not focus on the sequence of tracks, but rather on aiding the user in identifying items that could be included in a playlist. Viewing the playlist as a set of items also relates to the problem of entity list completion (also referred to as example-augmented search) that has been extensively studied in information retrieval [1]. There, the input consists of a textual query and a small set of example entities. Benchmarks addressing this problem include the INEX 2007–2009 Entity Ranking track [17, 19, 20] and the TREC 2010–2011 Entity track [4, 5]. Approaches primarily concentrate on the representation of entities for the purpose of measuring the similarity between them. For example, Bron et al. [9] combine text-based and structure-based entity representations (in terms of RDF triples), while Zhang and Balog [88] consider entity-focused tables and identify additional entities based on a table’s caption and entities already present. Analogous to these works, we consider user utterances (textual queries) along with previous items (example entities) in the conversation history as input. Finally, we also follow recent work in dense entity retrieval, adopting use of a dual encoder to embed queries and items [29, 51, 53]. Novel to our work is that item sets are constructed with explicit user feedback at each turn, in contrast to the more implicit signals of music streaming services through actions like saving or skipping songs [27].

Also related to item sets is the task of *slate recommendation*, which involves grouping together and presenting items as ordered sets of collections, called *slates*. Approaches for slate generation

¹See Section 4.4 for cost estimate details.

include List Conditional Variational Auto-Encoders [41] and Slate-MDPs [77]. Xian et al. [84] extend slate generation to include corresponding explanations based on important item attributes. Unlike those, we generate slates based on natural language conversational interactions.

2.2 Conversational Recommender Systems

Conversational recommender systems help users find items of interest through a sequence of interactions (i.e., conversation) [38]. Strongly related to conversational information seeking, it often encompasses recommendation, conversational search, and conversational question answering [87]. Most closely related to our work, CRSs allow users to provide direct feedback to improve recommendations and rely less on historical interaction data compared to collaborative filtering-based RSs [46, 47]. However limited conversational training data means CRSs are often trained using reinforcement learning [15, 52, 92, 96], or supervised learning on scripted dialogue flows [32]. Recent work has often collected conversational data through crowdworkers [6, 12, 33, 43, 54, 57, 58, 62, 94], yet existing datasets are usually limited to specific domains such as movie recommendation, and relatively small sizes² also promote overfitting [82]. While for individual item recommendations, a CRS can ask the user about their preferences on specific attributes [76, 90], item sets like playlists cannot be easily characterized using a pre-defined set of attributes. Thus, the item set is gradually formed by utilizing turn-level user feedback, steering the recommender system towards the desired target collection. Closely related to CRSs is example critiquing, where users can provide feedback on slates, using a fixed set of attributes or tags (e.g., “more upbeat”) [31]. Göpfert et al. [31] build a RS by modeling critiques as updates to a user embedding with learned concept activation vectors (CAVs), which represent the tags (e.g., “upbeat”) in an embedding space. We apply similar techniques to synthetically generate training data, but use item collection embeddings, rather than a fixed set of CAVs, to approximate user preferences. Our work also builds on existing work in conversational search and conversational question answering for retrieval modeling [48, 61, 86].

2.3 Synthetic Data Generation

Synthetic data generation is frequently used to help address both *privacy* and *data scarcity* challenges for RSs [23, 67]. As traditional RSs rely on historical interaction data (i.e., a user-item matrix), researchers have noted the need to consider user privacy when developing reproducible approaches to system development [11, 39, 71]. Several solutions have been proposed to mitigate privacy risks, including transforming the historical data by partial replacement [56, 75] or using randomized perturbation [66]. Our approach does not use historical data, alleviating those risks. At the same time, real user data is often scarce in *new* applications because existing systems are limited in their capability to respond to the full gamut of likely user requests. Most commonly, previous approaches generate synthetic data by sampling an explicitly defined probabilistic model; the model can be manually configured or made to match aggregate statistics of historical data [18, 65]. However, these

approaches only generate attributes and item ratings to express user preferences, rather than natural language utterances as we do.

Closest to our work, several studies focus on synthetic data generation for CRSs using item rating data [21], textual review data [90], and user logs (e.g., watch history) [94]. Dodge et al. [21] and Zhang et al. [90] use natural language templates to generate utterances, while Zhou et al. [94] generate utterances by retrieving candidate utterances and then using human annotators to rewrite them to be more conversational. In contrast, our work uses a language model [16] to generate user utterances. Lara and Tiwari [50] highlight the challenges in evaluating synthetic data generated by large language models; we evaluate the synthetic data both directly through a crowdsourced evaluation and indirectly through the performance of models trained on the data. Finally, recent work on user simulation [2, 89] studies how CRS evaluation can be automated, by mimicking how a user would respond at each turn. In contrast, we focus on automating the generation of entire conversations, including queries and target slates, for training CRSs.

3 TOWARDS A CONVERSATIONAL RECOMMENDATION SYSTEM

We now sketch the structure of the CRS that we present in this paper. Assume a CRS where the user has a target set of items (or *slate*) s^* in mind³, and in each turn the user interacts with the CRS by providing a natural language query. In turn the CRS responds by returning a slate of results s_t that satisfy the user’s query in light of the conversation history. Formally, given a user utterance u_t and the history of previous utterances and slates, $H_t = (u_1, s_1, \dots, u_{t-1}, s_{t-1})$, a CRS predicts a new slate of items that ranks the user’s target s^* highest.

Given a collection of items \mathcal{X} , we model a CRS as a *ranking function* $\rho : \mathcal{X} \rightarrow \mathbb{R}$ using a dual encoder architecture [29, 44, 63]. Dual encoders independently embed queries $q \in \mathcal{Q}$ and items $x \in \mathcal{X}$ into normalized dense vectors, and compute the ranking function using cosine similarity: $\rho(x; q) = f(q)^\top g(x)$, where $f : \mathcal{Q} \rightarrow \mathbb{R}^d$ and $g : \mathcal{X} \rightarrow \mathbb{R}^d$ are embedding functions for queries and items respectively. At turn t , the query is a function of the conversation history H_t and the latest user utterance u_t ; the predicted slate s_t consists of items in \mathcal{X} ranked according to $\rho(x; q_t)$. We model f and g using a shared language model. See Section 6.1 for the exact input representation used.

In terms of CRS functionality, we note that we do not model preference elicitation, where systems ask users questions to provide better recommendations [14, 52, 76, 91, 95]. Instead, we focus on modeling user interest revelation, set retrieval and memory [68].

4 FROM CURATED ITEM COLLECTIONS TO ITEM SET CURATION CONVERSATIONS

We describe TalkTheWalk, which uses curated item collections to generate item set curation conversations. Curated item collections contain substantial domain expertise: they not only group items into coherent collections, but also provide valuable metadata

²For example, the popular ReDial dataset [54] has only $\sim 10k$ conversations.

³To simplify evaluation, we assume that the user’s target is fixed throughout the conversation.

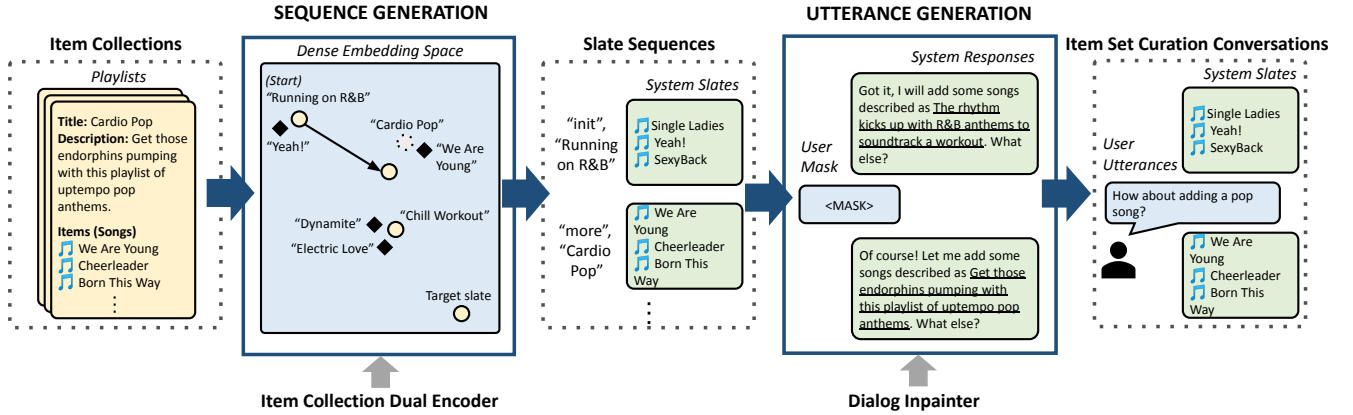


Figure 2: An overview of TalkTheWalk that transforms item collections into item set curation conversations. (1) First, during sequence generation, we generate slate sequences using a biased random walk in an embedding space. (2) Then, during utterance generation, we generate user utterances by prompting a dialog inpainter language model with templated system responses filled in with slate metadata. Combining outputs from the two steps, TalkTheWalk outputs item set curation conversations of user utterances paired with item slates. We do not use the templated system responses after utterance generation.

about each collection; for music, those are tracks, playlists, and playlist descriptions, respectively. Importantly, while we focus on music, similar collections exist in other domains where various user generated content is frequently created by passionate users with an interest in the topic at hand. However, such collections lack two key features present in item set curation conversations: a sequence of user utterances describing preferences (instead of the non-conversational metadata), and a corresponding sequence of item *slates* presented in response to the utterances (instead of a fixed collection of items). We address this lack by first using item collections to generate sequences of slates and then generating user utterances for the sequences. We first discuss the input and output of the data generation pipeline (Section 4.1) and then discuss the two main components: sequence generation (Section 4.2) and utterance generation (Section 4.3). Finally, we use TalkTheWalk to generate TtWMusic, a new dataset with over 1M synthetic playlist curation conversations (Section 4.4). See Figure 2 for an overview of our approach.

4.1 Input and Output

We summarize the input and output of the data generation pipeline and introduce notation used in the rest of the paper.

4.1.1 Input. We assume a corpus of items $x \in \mathcal{X}$ (e.g., songs in a music corpus). The main input to data generation is a dataset of item collections \mathcal{Z} . An item collection $z \in \mathcal{Z}$ consists of a set of items $x \in \mathcal{X}$ sharing coherent metadata $\phi(z)$, for instance a playlist and its description. Item collections can be categorized by type $\sigma(z) \in \mathcal{T}$ (e.g., mood or artist-based playlists). In addition to item collections, we assume two pretrained models as input: a pretrained dual encoder to represent item collections and items in a shared dense embedding space (hereafter referred to as the item collection DE); and a pretrained *dialog inpainter*, a language model which predicts masked utterances in a conversation [16]. Formally,

given a partial conversation $C_T = (u_1, \dots, u_{t-1}, \diamond, u_{t+1}, \dots, u_T)$ with T turns, where \diamond represents a masked utterance, a dialog inpainter specifies a probability distribution $p_\theta(u_t | C_T)$ parameterized by θ .

4.1.2 Output. The output of data generation is an item set curation conversation with T turns $C_T = (u_1, s_1, \dots, u_T, s_T)$, where each turn t has a user utterance u_t and a corresponding slate of items $s_t \subset \mathcal{X}$.

4.2 Sequence Generation

The goal of sequence generation is to generate a realistic sequence of slates that may occur in a conversation. We first discuss the desired sequence properties, then introduce our approach leveraging item collections to generate sequences (see Figure 2 left and Algorithm 1).

4.2.1 Desired Properties. We are guided by following desirable properties for slates in an ideal conversation:

- (P.1) Slate s_t should be *consistent* with, or closely related to, the previous turn’s slate s_{t-1} . If the user first asked for workout music, they are more likely to ask for pop than classical music.
- (P.2) The change between consecutive slates should be *coherent*, corresponding to user preferences expressed in natural language; item collection z_t should approximate this change.
- (P.3) Each turn should bring the user *closer to their target slate* s^* . If the user ultimately wants good workout music, we expect slates to include more high energy music in later turns.

4.2.2 Approach. To realize these properties, we start by representing items x , item collections z_t , and slates s_t as vectors $(\tilde{x}, \tilde{z}_t, \tilde{s}_t)$ in a shared embedding space \mathbb{R}^d using the item collection DE.

To maintain consistency across turns (P.1), we represent the user’s preferences at turn t as a vector \tilde{r}_t and ensure that \tilde{r}_{t+1} is close to \tilde{r}_t and \tilde{s}_t . We also assume that each user has a target vector \tilde{r}^* representing their target slate s^* . We sample the initial user

vector $\tilde{\mathbf{r}}_1$ and the target user vector $\tilde{\mathbf{r}}^*$ from \mathcal{Z} to ensure they are coherent (i.e., represent real item collections).⁴

To make the changes between consecutive slates coherent (P.2), we model each update as a linear combination of the user vector $\tilde{\mathbf{r}}_t$ and a nearby item collection vector $\tilde{\mathbf{z}}_t$:

$$\tilde{\mathbf{r}}_{t+1} = \alpha \tilde{\mathbf{r}}_t + \beta \tilde{\mathbf{z}}_t \quad (1)$$

where α and β are t -dependent parameters that we will solve for below. To select $\tilde{\mathbf{z}}_t$, we first randomly sample an item collection type $\sigma_t \in \mathcal{T}$, and then sample an item collection of that type from the neighborhood of $\tilde{\mathbf{r}}_t$.⁵ Intuitively, $\beta > 0$ represents a positive preference for \mathbf{z}_t and moves the user vector in the direction of $\tilde{\mathbf{z}}_t$, while $\beta < 0$ represents a negative preference for \mathbf{z}_t and moves the user vector in the opposite direction of $\tilde{\mathbf{z}}_t$. The item collection type corresponds to different types of user preferences, such as a mood preference or an artist preference when looking for music.

Finally, to ensure each turn brings the user closer to their target slate (P.3), we choose the weights α and β that minimize the distance to $\tilde{\mathbf{r}}^*$ while keeping $\tilde{\mathbf{r}}_{t+1}$ unit norm:

$$\alpha^*, \beta^* = \operatorname{argmax}_{\alpha, \beta} \langle \alpha \tilde{\mathbf{r}}_t + \beta \tilde{\mathbf{z}}_t, \tilde{\mathbf{r}}^* \rangle \text{ s.t. } \|\alpha \tilde{\mathbf{r}}_t + \beta \tilde{\mathbf{z}}_t\|_2 = 1. \quad (2)$$

Optimal values of α and β can be found in closed form as one of:

$$\alpha = \frac{\pm(w - qv)}{\sqrt{(q^2 - 1)v^2 - (w - qv)^2(q^2 - 1)}} \quad (3)$$

$$\beta = -\alpha q \pm \sqrt{(\alpha q)^2 - \alpha^2 + 1},$$

where $q = \tilde{\mathbf{r}}_t^T \tilde{\mathbf{z}}_t$, $v = \tilde{\mathbf{z}}_t^T \tilde{\mathbf{r}}^*$, and $w = \tilde{\mathbf{r}}_t^T \tilde{\mathbf{r}}^*$. Given α^* and β^* , we can use (1) to update the user vector to $\tilde{\mathbf{r}}_{t+1}$.

Recall our goal is to output a sequence of slates of recommended items. If the preference for \mathbf{z}_t is positive, we simply use the items in \mathbf{z}_t to form the slate \mathbf{s}_t . If the preference for \mathbf{z}_t is negative, we do not want to recommend the items in \mathbf{z}_t ; instead, we use the item neighbors of the updated user vector $\mathcal{N}_x(\tilde{\mathbf{r}}_{t+1})$ to form \mathbf{s}_t . Formally, the slate \mathbf{s}_t and preference type p_t are defined as:

$$\mathbf{s}_t = \begin{cases} \mathbf{z}_t & \beta > 0 \\ \mathcal{N}_x(\tilde{\mathbf{r}}_{t+1}) & \beta \leq 0 \end{cases} \quad p_t = \begin{cases} \text{more} & \beta > 0 \\ \text{less} & \beta \leq 0. \end{cases} \quad (4)$$

Along with each \mathbf{s}_t , we also store p_t and item collection \mathbf{z}_t to express the preference for the slate.⁶ We repeat for T turns, building the sequence (Algorithm 1).

4.3 Utterance Generation

Given a sequence of states, the next step generates corresponding user utterances. Suppose that \mathbf{s}_{t-1} had “Running on R&B” songs and \mathbf{s}_t “Cardio Pop” songs; the utterance u_t should express new preferences (viz., pop music), while assuming prior context (viz., workout music): e.g., “How about adding a pop song?”. Ideal utterances are (1) realistic, using natural language rather than keywords, (2) diverse, covering a wide variety of ways users can express themselves, and (3) contextual, building on previous conversation turns.

⁴Specifically, we first sample $\tilde{\mathbf{r}}^*$ from \mathcal{Z} and then sample $\tilde{\mathbf{r}}_1$ from the item collection neighbors of $\tilde{\mathbf{r}}^*$ with a neighbor index in [64, 128] to encourage $\tilde{\mathbf{r}}_1, \tilde{\mathbf{r}}^*$ to be related.

⁵Based on preliminary experiments, we use $\operatorname{softmax}(\mathcal{N}(\tilde{\mathbf{r}}_t) \tilde{\mathbf{r}}^* / \tau)$ as the sampling distribution (neighborhood size 64 and softmax temperature $\tau = 0.1$).

⁶We use an *init* preference type for the first turn of all sequences.

Algorithm 1 Sequence Generation Procedure

Input: item collections \mathcal{Z} , item collection types \mathcal{T} , number of turns T

Output: sequence of slates $[(\mathbf{s}_1, p_1, \mathbf{z}_1), \dots, (\mathbf{s}_T, p_T, \mathbf{z}_T)]$

```

1: seq = []
2:  $\tilde{\mathbf{r}}^* = \text{sample}(\mathcal{Z})$  ▷ Sample target user vector.
3:  $\tilde{\mathbf{r}}_1 = \text{sample}(\mathcal{N}_z(\tilde{\mathbf{r}}^*))$  ▷ Sample initial user vector.
4: for  $t = 1, 2, \dots, T$  do
5:    $\sigma_t = \text{sample}(\mathcal{T})$  ▷ Sample collection type.
6:    $\tilde{\mathbf{z}}_t = \text{sample}(\{\tilde{\mathbf{n}} \in \mathcal{N}_z(\tilde{\mathbf{r}}_t) \mid \sigma(\tilde{\mathbf{n}}) = \sigma_t\})$  ▷ Get collection.
7:    $\alpha, \beta = \text{findOptimalWeights}(\tilde{\mathbf{r}}_t, \tilde{\mathbf{z}}_t, \tilde{\mathbf{r}}^*)$  ▷ See (2) and (3).
8:    $\tilde{\mathbf{r}}_{t+1} = \alpha \tilde{\mathbf{r}}_t + \beta \tilde{\mathbf{z}}_t$  ▷ Update user vector.
9:    $\mathbf{s}_t = \text{getSlate}(\tilde{\mathbf{r}}_{t+1}, \mathbf{z}_t, \beta)$  ▷ See (4).
10:   $p_t = \text{getPreferenceType}(\beta, t)$  ▷ See (4).
11:  seq.append( $(\mathbf{s}_t, p_t, \mathbf{z}_t)$ ) ▷ Update slate sequence.
12: end for
13: return seq

```

To achieve this, we use the pretrained dialog inpainter. First, we write system response templates for each preference type and item collection type (e.g., “Of course! Let me add some songs described as <description>. What else?” and “Got it! Let me remove some songs described as <description>. What else?”). We then set up a partial conversation with system responses that describe each slate \mathbf{s}_t by instantiating the above templates with the corresponding preference type p_t , item collection type $\sigma(\mathbf{z}_t)$, and item collection metadata $\phi(\mathbf{z}_t)$. For example, to express *more* “Cardio Pop”, the system response could be “Of course! Let me add some songs described as Get those endorphins pumping with this playlist of uptempo pop anthems. What else?”. Finally, we use a dialog inpainter to generate the “missing” user utterances u_t (see Figure 2 right), completing the desired output. Note that in this work we *only* use the templated systems responses to prompt the dialog inpainter; we defer other uses, e.g., generating non-templated system responses to future work.

4.4 Case study: Generating a Conversational Music Recommendation Dataset

We demonstrate our approach by generating TtWMusic, with over one million playlist curation conversations.

Data generation inputs. For the item collections \mathcal{Z} , we use a proprietary set of expert-curated playlists (hereafter referred to as ExpertPlaylists). ExpertPlaylists contains two types of playlists: (1) theme playlists, where all songs on the playlist share a common theme (e.g., genre, mood, activity) (19,129 playlists⁷) and (2) artist playlists, where all songs on the playlist share the same artist (121,704 playlists). $\phi(\mathbf{z})$ provides the title and description of playlists \mathbf{z} , and $\psi(\mathbf{x})$ provides the title, artist names, album, and 128-dimensional audio vector [35] of songs \mathbf{x} .

For the item collection DE, we train a dual encoder over ExpertPlaylists using a standard contrastive loss [80]. Both the queries and items use multi-modal inputs composed of text (WordPiece [49, 72])

⁷We divide the theme playlists into train/dev/test splits (15,276/1,895/1,958) to train the item collection DE.

Table 1: A summary of TtWMusic and the Conversational Playlist Creation Dataset (CPCD); TtWMusic is several orders of magnitude larger than CPCD and other manually collected conversational datasets. While it has a similar conversation length distribution, it features longer queries and more items per slate. We also report the number of examples, average description length, and collection size for the (non-conversational) curated item collection used in this work, ExpertPlaylists.

Statistic	ExpertPlaylists	TtWMusic	CPCD	
	Train	Train	Dev	Test
# of examples	15,276	1,037,701	450	467
# of tracks	332,594	332,594	106,736	
Avg. # of turns	-	5.6	5.8	5.6
Avg. query len.	106.5	80.3	53.8	55
Avg. # of items	53.6	47.2	19	18.3

and audio (MuLan [35]) tokens. The query q is a representation of a playlist z , generated by concatenating $\phi(z)$ with $\psi(x)$ for a sample of "seed" songs $x \in z$.⁸ The item x is a representation of a randomly sampled song in z , generated using $\psi(x)$. We initialize the dual encoder from a pretrained T5 1.1-Base [70] and continue training on the ExpertPlaylists theme playlists.

For the dialog inpainter, we use a T5-XXL model pretrained on conversational question-answering datasets and social media discussion threads (similar to InpaintSTOQ [16])—as large-scale conversational recommendation data is not available—and find this works well.

Generation and post-processing. We generate over 1M playlist curation conversations using our approach. We select among artist and theme playlist types with equal probability during sequence generation, resulting in user utterances about artists (e.g., "Can I have some Lady Gaga?") and broad attributes (e.g., "How about something more upbeat?"). Using the aggregate statistics of a crowd-sourced conversational music recommendation dataset (CPCD) [12] as guidance (Table 1), we generate 6 turns for each conversation. We then filter out any utterances in the generated data that: (1) do not mention the corresponding artists in artist-type queries, (2) include offensive language, (3) are longer than 450 characters, or (4) have substantial overlap (more than 50 characters) with the following system response. We refer to the final synthetic dataset as TtWMusic and present summary statistics in Table 1.

Cost estimates. Querying the item collection DE during sequence generation and the dialog inpainter during utterance generation are the most resource-intensive steps of TalkTheWalk. It takes about 30s to generate each sequence on commodity hardware, and about 1s to inpaint each conversation using a TPU. We used Apache Beam⁹ to parallelize computation. In total, TtWMusic required about 6,000

vCPU-hours ($\approx \$200$) for sequence generation and about 200 TPUv3-hours ($\approx \$500$) for utterance generation to generate 1 million examples. Using spot instances on Google Cloud, this costs about \$700, several orders of magnitude cheaper than crowdsourcing.

5 EVALUATING THE SYNTHETIC DATA QUALITY

We used crowdsourcing to evaluate the quality of our synthetic dataset, TtWMusic. As the primary use-case for TtWMusic is training CRSs, it does not need to be indistinguishable from human-collected conversations, but should represent *consistent* preferences with corresponding *relevant* slates.¹⁰ We confirmed that TtWMusic generally satisfies these criteria by asking the questions shown in Table 2 to crowdworkers specializing in music labeling tasks.¹¹

As a point of reference, we also assessed human-collected conversations from the Conversational Playlist Creation Dataset [12] (CPCD) using the same criteria. CPCD consists of music playlist-seeking conversations between two people in a Wizard-of-Oz setting, with one acting as the user, and other acting as a recommendation system. Users are asked to come up with a music-listening scenario (e.g., "a long commute") and create a playlist for it by conversing with the system; wizards are asked to recommend songs for the user, and can also elicit preferences from the users. As TtWMusic does not include system responses, annotators are shown a uniform system response, "What else?", for both datasets.

Ten annotators reviewed 330 conversations from TtWMusic and CPCD each, rating 1668 and 1308 turns respectively.¹² Each conversation was reviewed by three annotators each. The results are summarized in Table 2. Overall, we find that TtWMusic contains consistent preferences and relevant slates, though less so than the human-collected conversations. We first discuss the two turn-level questions and then discuss the conversation-level question.

How consistent are the user’s preferences given the conversation so far? Consistent preferences are believable and likely given the previous turns (e.g., asking for pop—instead of classical music—when looking for workout music). Raters found nearly 80% of turns to be very consistent and fewer than 4% to be not at all consistent, validating our sequence generation approach.

How relevant are the results for the user’s preferences given the conversation so far? Whereas the previous question evaluates the user utterances, this next question evaluates how relevant the generated slate of results are to the stated preferences, e.g., if the user utterance asks for romantic songs by Lady Gaga, does the result slate reflect the same? Raters find that 95.1% of slates in TtWMusic to be at least somewhat relevant, which is comparable to the expert-selected slates in CPCD.

How natural do you think this conversation is? Finally, we are also interested in evaluating the naturalness of conversations as a whole and ask raters: *Can you imagine yourself or someone you know having this conversation?* We expect conversation-level ratings to

¹⁰We show that these criteria are sufficient to train a strong CRS in Section 7.

¹¹We measured reasonable pairwise agreement for consistency (88.5%), relevance (81.90%) and naturalness (72.20%).

¹²Turns in CPCD that did not include system responses (e.g., because the system elicited preferences from the user instead) were omitted in this study.

⁸We use five seed songs in our experiments.

⁹<https://beam.apache.org>

Table 2: Percentage of results rated as Not at all (0), Somewhat (0.5) and Very (1) in a qualitative human evaluation of synthetic conversations generated using our approach (TtWMusic) and human-collected conversations (CPCD). We also report a weighted average of the responses for each question.

Question	TtWMusic				CPCD			
	Not at all	Somewhat	Very	Avg.	Not at all	Somewhat	Very	Avg.
<i>How consistent are the preferences?</i>	3.7%	16.5%	79.7%	88.0%	1.0%	9.1%	89.8%	94.4%
<i>How relevant is the slate?</i>	4.8%	26.2%	68.9%	82.1%	1.9%	24.1%	74.0%	86.0%
<i>How natural is the conversation?</i>	3.0%	46.2%	50.8%	73.9%	0.3%	35.3%	64.4%	82.0%

be lower than the turn-level ratings as a single unnatural turn can render the whole conversation unnatural. Raters find that 97% of conversations in TtWMusic are at least somewhat natural, and 50.8% of conversations are very natural. Interestingly, raters only rate 64.4% of conversations in CPCD to be very natural despite being human-collected, highlighting how hard it is to guide crowdworkers to create natural conversations.

6 BUILDING A CONVERSATIONAL MUSIC RECOMMENDATION SYSTEM

We now show how our synthetic dataset, TtWMusic, can be used to train a conversational music recommender.

6.1 Model

Recall that we model a CRS using a dual encoder architecture. Given a conversation history $\mathbf{H}_t = (u_1, s_1, \dots, u_{t-1}, s_{t-1})$ and user utterance u_t , we use \mathbf{H}_t and u_t to construct a query q_t , and then predict a slate s_t whose items x maximize the ranking function $\rho(x; q_t) = f(q_t)^\top g(x)$, i.e., are closest to q_t in embedding space. Similar to the item collection DE (Section 4.4), we use multi-modal inputs consisting of text (e.g., WordPiece [49, 72]) and audio (e.g., MuLan [35]) tokens. We construct q_t by concatenating the utterances with audio and text representations of the top- k songs in each slate in reverse chronological order.¹³ For example, the query at turn t is:

$$u_t \text{ [SEP]} d(s_{t-1}) \text{ [SEP]} u_{t-1} \dots d(s_1) \text{ [SEP]} u_1 \text{ [SEP]} a(s_{t-1}) \dots a(s_1),$$

where $d(s_t)$ is a textual representation of s_t , $a(s_t)$ is its audio representation, and [SEP] represents a separator token. We use the same representation for items as in the item collection DE (Section 4.4).

6.2 Training and Inference

For training, we use a standard contrastive loss with in-batch negatives: given an example i with query q_i and target slate s_i , we randomly sample an item $x_i \in s_i$ to be a positive target, and take items from other slates in the batch, x_j where $j \neq i$, to be negative targets. To improve robustness, we augment our training data as follows: (1) we generate conversations of varying lengths by randomly truncating conversations to its first t turns, and (2) we generate slates of varying lengths by randomly truncating slates to their first k items.

¹³In our experiments, we use up to three songs from each slate.

For inference, we build an index of pre-computed item embeddings. We embed queries as in training, and use nearest neighbor search to return a slate s_t with the top- k items for q_t .

7 EVALUATING RECOMMENDATION PERFORMANCE

We now turn to answering the key motivating question for TalkTheWalk: Is it possible to bootstrap a conversational recommendation system entirely from scratch using only non-conversational artifacts? To answer this question, we focus on a “zero-shot” setting where models are not fine-tuned on a target dataset. Using the music domain as an example, we show that our synthetic dataset, TtWMusic, indeed suffices to build a strong conversational recommendation system that significantly outperforms traditional baselines in both offline and online evaluations.

7.1 Experimental Setup

We describe the benchmark dataset, evaluation metrics, model implementation, baselines, and online human evaluation.

7.1.1 Benchmark dataset. We use the Conversational Playlist Creation Dataset (CPCD) [12] introduced in Section 5 to evaluate models on recommendation performance. Each conversation consists of multiple turns, and each turn consists of a user query and the result slate returned by the system with binary ratings (like or dislike) for each item. Because we do not model explicit user feedback in TtWMusic, we remove disliked songs from slates in the conversation history. The target slate s^* includes liked songs across all turns of the conversation. Systems are evaluated on their ability to retrieve songs in the target slate from a corpus of 106k songs given the user query and conversation history in each turn. See Table 1 for a summary of key statistics.

7.1.2 Evaluation metrics. Evaluating RSs is challenging because there are often missing ratings [21, 79]: we do not know the relevance of songs that are neither liked nor disliked by a user. Standard ranking-based metrics treat items without ratings as not relevant—even though they may actually be relevant—and thus provide a lower bound on recommendation performance.¹⁴ Following prior work [21, 43, 93], we compare systems using a standard ranking metric, Hits@ k , which is 1 if and only if any of the top- k retrieved songs are in the target slate. Unless otherwise stated, we report macro-averaged Hits@ k , averaging Hits@ k across turns within a conversation and then across conversations.

¹⁴Indeed, we observe large performance differences between offline and online tests.

An important question is what cut-off k should we use for evaluating the song ranking? While smaller values of k represent realistic recommendation scenarios where users are shown a small slate of items, Valcarce et al. [79] found that larger values of k have more discriminative power in offline experiments and can help mitigate the missing rating problem. Consequently, we report performance at several values of k (10, 20, 100).

Finally, to support evaluation over multiple turns, we must take into account several considerations. First, the target slate at each turn only consists of songs that have not been seen in the history up to that point. *As the metrics depend on the target slate—which changes across turns—we cannot directly make comparisons across turns.* To compare across models within a turn, we use the same history and target slate across all models, assuming a “gold” history, as opposed to building the history using model predictions from previous turns. Second, at each turn, songs may be liked by users, but not added to the history (e.g., when there is a limit to the number of songs in the history). We refer to these songs as leftovers (per [64]) and keep them in the target slate, resulting in a larger target slate compared to removing all previously liked songs.

7.1.3 Model implementation. We initialize our dual encoder models from a pretrained T5 1.1-Base [70] and train on TtWMusic for 100k steps using Adafactor [74] with a batch size of 512 and a constant learning rate of $1e-3$. Training completes in about 8 hours using 32 TPUv4 chips.¹⁵ We select the checkpoint with highest Hits@10 on the CPCD development set from steps 25k, 50k, 75k, and 100k. We denote this system as DE>TtWMusic to emphasize that it is a standard dual encoder (DE) trained on TtWMusic.

7.1.4 Baselines. We compare DE>TtWMusic to four baselines: BM25, a sparse retrieval, bag-of-words baseline; Contriever [36], an unsupervised dense retrieval baseline; DE>ExpertPlaylists, a dense retrieval baseline trained over playlist description-song pairs from the ExpertPlaylists dataset (i.e., the item collection DE in Section 4.4); and two variants which use a pretrained Query Rewriter (QR) to rewrite the history, BM25+QR and DE>ExpertPlaylists+QR. Query rewriting has been widely used to achieve state of the art conversational search [55, 78, 85]; following Lin et al. [55], we fine-tune a T5-based query rewriter on the CANARD dataset [22]. Our BM25 system includes conversation history by concatenating user queries from previous turns; we find omitting conversation history leads to worse performance. BM25+QR does not use concatenated user queries, because the QR incorporates the conversation history in its output. Contriever is also implemented using a T5-Base architecture, but trained on span pairs from the C4 dataset following Izacard et al. [36]. DE>ExpertPlaylists is implemented and trained identically to DE>TtWMusic, but using a different training dataset (ExpertPlaylists); at test time, we use the same conversation history including previous queries and results.

7.1.5 Online human evaluation. It is well understood that users interact with online systems differently based on the results they are shown and may rank systems differently than an offline evaluation [37, 87]. We conducted an online evaluation to compare the

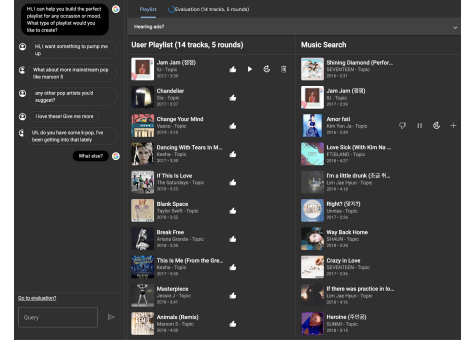


Figure 3: Screenshot of the conversational recommendation system. Users are presented with their query history (left), the songs they liked so far (“User Playlist”) and the results slate from the music retrieval systems (“Music Search”). Users can listen to songs, rate them and respond with a follow-on query.

two best performing systems in our offline evaluation, BM25 and DE>TtWMusic.¹⁶

We recruited 11 fluent English speakers, based in the United States, who regularly listen to music from an online crowdsourcing platform. Users interacted with a web-based interface (Figure 3) that allows them to express their preferences conversationally and rate results from the two systems. Users were provided instructions that provided an overview of the interface and its features. Similar to Chaganty et al. [12], they were asked to start each conversation with a context for their listening session, e.g., “I want to create a playlist to pump me up when I’m feeling tired,” and encouraged to express broad and general preferences. After each turn, users were required to rate all the songs in the result slate and to complete at least 5 rounds of conversation before proceeding to evaluation.

Finally, to ensure the same user does not rate songs differently based on their conversation history or their previous experience with the system, we present raters a single, combined, slate of results from the two systems using *team-draft interleaving* [69], a standard A/B testing method in the literature [34].

7.1.6 Limitations. We recognize the following limitations in our evaluation setup: Our offline evaluation relies on a benchmark dataset, CPCD, that was collected by emulating the system with a human wizard. Users may interact differently with a real CRS. Likewise, during the online evaluation, users are primed by their previous experiences with a traditional RS, and the queries they ask may change as they become more familiar with a CRS.

7.2 Main Results

Our key result is that simply training a standard dual encoder on TtWMusic suffices to build a strong conversational music recommendation system that significantly outperforms traditional baselines in both offline and online evaluations.

7.2.1 Offline evaluation. Table 3a compares models on the CPCD test set. We observe that our model, DE>TtWMusic, consistently

¹⁵We estimate a cost of about \$900 using spot instances on Google Cloud.

¹⁶We used DE>TtWMusic without audio features; in offline evaluations, this system is comparable to DE>TtWMusic and also significantly outperforms BM25.

outperforms baselines. Furthermore, the gap increases for larger values of k : our model improves over baselines by 2.9 points for Hits@10, 4.5 points for Hits@20, and 10.5 points for Hits@100. The next best baseline is BM25. We attribute its stronger performance over DE>ExpertPlaylists and Contriever to the prevalence of artist and song-specific queries in CPCD. On these queries, matching keywords suffices, and bag-of-words-based models can perform well. Finally, we observe that the query rewriter does not significantly improve, and can even hurt, baseline performance. We hypothesize that it cannot generalize beyond the factual question answering setting it was trained on to playlist curation as in the CPCD dataset.

7.2.2 Online evaluation. We collected 227 conversations and 2454 item ratings after filtering out conversations with an average utterance length less than four, and any turns with irrelevant utterances like “hello” or “thank you.” We then computed per-turn hit rates for the two systems, finding that DE>TtWMusic had significantly higher Hits@10 (69.8%) than BM25 (62.0%) with $p < 0.01$. Qualitatively, DE>TtWMusic was also better able to understand broad queries and refinements and returned non-literal results to the user while BM25 tended to overemphasize lexical overlap (Table 4).

7.3 Analysis Results

We perform additional experiments on CPCD to better understand what factors contribute to our model’s performance.

7.3.1 How do slates produced by DE>TtWMusic and BM25 differ? We manually inspected the result slates returned by DE>TtWMusic and BM25 on CPCD queries, and observed the following key differences: (1) DE>TtWMusic excels at retrieving relevant results for *broad queries* (e.g., “Let us get some modern Christmas songs”), while BM25 tends to overemphasize lexical matches. This indicates that DE>TtWMusic is able to leverage the domain knowledge embedded in curated playlists like “Today’s Pop Christmas.” (2) DE>TtWMusic is better able to understand attributes (e.g., “I’d prefer female vocals” or “can you make a playlist of Nigerian songs?”); these attributes are often described in curated playlists like “Women Who Ruled 2013” or “Discover Nigeria.” (3) DE>TtWMusic underperforms on explicit artist or album queries (e.g., “I’d also like some songs of Olivia Rodrigo”), where it often retrieves music from similar artists too.

These observations suggest that BM25 and DE>TtWMusic may have complementary advantages, and a sparse-dense hybrid model could improve performance [25, 44, 59, 60, 73]. We study if this is the case by evaluating a simple hybrid model that interleaves the recommendations from the two systems by simply alternating the recommendations from each model. Thus, the top-10 recommendations have five results from DE>TtWMusic and five results from BM25. We find that this improves performance by 0.5-2 points for Hits@ k (Table 3a), validating the claim. We leave more sophisticated methods for combining recommendations (e.g., [59]) to future work.

7.3.2 How much performance is attributed to audio embeddings? The dual encoders in our experiments use multimodal inputs, leveraging both text and audio embeddings, while BM25 relies only on textual inputs. To understand the impact of the audio embeddings, we train a dual encoder over TtWMusic without including the audio

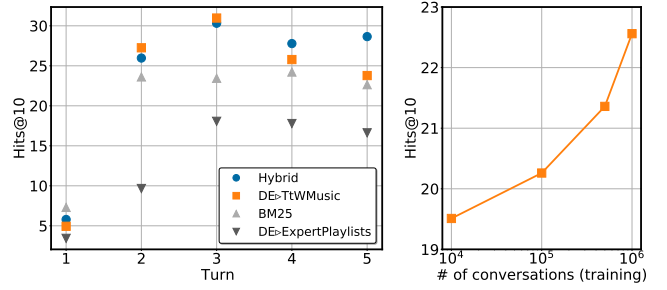


Figure 4: Performance analysis (Hits@10) on CPCD test. (Left) Results (averaged) per conversation turn. Because target slate sizes vary, comparisons cannot be made across turns (see Section 7.1.2). TalkTheWalk consistently outperforms BM25, the best baseline. (Right) Results (DE>TtWMusic) using varying amount of training data. The plot suggests that additional synthetic data will continue to improve model performance.

embeddings in the input. We find that removing audio embeddings leads to a 0.8–3.9 point drop in Hits@ k metrics, but DE>TtWMusic (No audio) still outperforms BM25. This indicates that, while audio embeddings do contribute to model performance, the song metadata (title, artists, etc.) alone is sufficient to outperform baselines.

7.3.3 How do models compare across turns? Figure 4 (left) compares models on turn-level performance.¹⁷ We show the first five turns, where we average Hits@10 across conversations for each turn. Note that as the conversations vary in length, the number of conversations for each turn vary (the minimum is 349 conversations at Turn 5). The target slate size also decreases across turns as songs are included in the conversation history; thus, we cannot compare across turns. DE>TtWMusic consistently outperforms BM25 and DE>ExpertPlaylists across turns, with the exception of the first turn where BM25 performs the best. This suggests that our model is better able to leverage conversation history than the baselines. We identify the relatively stable performance of BM25 in later turns as an artifact of including leftovers in our evaluation: BM25 predicts the same items in later turns, which match leftovers in early turns.

7.3.4 How does performance scale with the amount of training data? Figure 4 (right) shows Hits@10 on CPCD test as we vary the number of conversations in the synthetic training dataset. We find that the performance improves from 19.5% to 22.6% for 10k to 1M conversations and does not level off, suggesting that generating more than 1M conversations may further improve the performance. In contrast, most crowdsourced or real-world conversational recommendation datasets contain <30k conversations (e.g., [6, 40, 54, 57, 62, 94]).

7.3.5 How important are the synthetic data generation components? We ablate the two components of synthetic data generation: sequence generation and utterance generation (see Section 4). We generate new synthetic datasets in each ablation and train dual encoders on the datasets following the protocol in Section 7.1.3. To ablate sequence generation, we use a random sequence generator which randomly selects an item collection \mathbf{z}_t each turn to use as

¹⁷We exclude the worse-performing models with a query rewriter for readability.

Table 3: Results on the CPCD test set. (a) Main results (DE: dual encoder; QR: query rewriter). Underlined numbers denote statistical significance compared to our model, DE>TtWMusic, using a paired randomization test ($p < 0.05$). (b) Model and synthetic data ablations. For synthetic data ablations, models are trained over synthetic dataset variants with 100k conversations (instead of 1M).

(a) Main results.				(b) Model and dataset ablations.			
Model	Hits@10	Hits@20	Hits@100	Model	Hits@10	Hits@20	Hits@100
BM25	19.7	27.4	45.5	DE>TtWMusic	22.6	31.9	56.0
+QR	15.5	21.7	34.0	(No Audio)	20.8	31.1	52.1
Contriever	16.2	23.1	39.4	Training Dataset	Hits@10	Hits@20	Hits@100
DE>ExpertPlaylists	13.1	19.6	43.2	TtWMusic-100k	20.3	30.2	52.4
+QR	13.1	20.2	42.5	– Sequence Gen.	11.9	16.3	28.9
DE>TtWMusic	22.6	31.9	56.0	– Utterance Gen.	18.9	26.7	43.4
+ BM25 (hybrid)	23.1	33.9	57.1				

Table 4: Example conversations collected in online human evaluation. The top-ranked song from both systems is shown under the user query. ✓ and ✗ denote user ratings; ♪ symbols link to the song online. Unlike BM25, DE>TtWMusic retrieves songs that are relevant to the user query even when there is no literal match (“Happiness”) and is able to maintain context across turns (“Mother”).

User	Can you make me a playlist of sad songs, I broke up with girlfriend
▷TtWMusic	✓ Happiness by Hobo Johnson ♪
BM25	✗ I Can’t Make You Love Me by Teddy Swims ♪
User	can you add more of female voices
▷TtWMusic	✓ Mother by Courtney Love & The Turning ♪
BM25	✗ I Can’t Make You Love Me by Dave Thomas Junior ♪
User	Looking to create a playlist to help me sleep. Soft sounds with minimal lyrics.
▷TtWMusic	✓ Suburban Call For Concrete Dreams by Ave Air ♪
BM25	✓ Soul Blind With Lyrics by Shane Phipps ♪
User	Piano music please. Maybe add some classical music.
▷TtWMusic	✓ Rise by Arelius ♪
BM25	✗ Morning Mood - Grieg - Classical Piano - Classical Sleep Music and Ocean Sounds... ♪

the slate (i.e., $s_t = z_t$). We then use our standard method to generate utterances with dialog inpainting. As this method does not encourage consistency between turns, we expect the synthetic data to be less useful as training data than our full method. To ablate dialog inpainting, we first use our standard method for generating sequences of slates. Then, instead of using dialog inpainting to generate utterances, we use templated user utterances, similar to the templated system responses (e.g., “Add some songs described as Get those endorphins pumping with this playlist of uptempo pop anthems.”). As this method generates less diverse and realistic user utterances, we also expect the synthetic data to be less useful as training data than our full method. Table 3b shows the results on CPCD test when we train models on 100k conversations in each

dataset.¹⁸ We see that removing either component leads to drops in Hits@ k at all values of k , with a 23.5 point drop in Hits@100 when removing sequence generation and a 9 point drop in Hits@100 when removing dialog inpainting.

8 CONCLUSION

We introduced a general technique, TalkTheWalk, to convert curated item collections into synthetic item set curation conversations, and demonstrated the benefits of building a conversational recommendation system using such data in the music domain. TalkTheWalk can be easily adapted to other domains and other languages given corresponding item collections and an appropriate language model. In the future, we plan to explore how to incorporate traditional RS signals such as popularity and explicit user feedback (e.g., ratings). Finally, noting that language models are used to create the conversational utterances, further work is warranted assessing what biases may be present in the utterances produced [7], and how they can be reduced.

REFERENCES

- [1] Krisztian Balog. 2018. *Entity-Oriented Search*. The Information Retrieval Series, Vol. 39.
- [2] Krisztian Balog, David Maxwell, Paul Thomas, and Shuo Zhang. 2022. Report on the 1st Simulation for Information Retrieval Workshop (Sim4IR 2021) at SIGIR 2021. *SIGIR Forum* 55, 2 (2022).
- [3] Krisztian Balog, Filip Radlinski, and Alexandros Karatzoglou. 2021. On Interpretation and Measurement of Soft Attributes for Recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21)*. 890–899.
- [4] Krisztian Balog, Pavel Serdyukov, and Arjen P. de Vries. 2011. Overview of the TREC 2010 Entity Track. In *Proceedings of the Nineteenth Text REtrieval Conference (TREC '10)*.
- [5] Krisztian Balog, Pavel Serdyukov, and Arjen P. de Vries. 2012. Overview of the TREC 2011 Entity Track. In *Proceedings of the Twentieth Text REtrieval Conference (TREC '11)*.
- [6] Saravanan Ganesh Amit Dubey Andy Cedilnik Bill Byrne, Karthik Krishnamoorthi and KyuYoung Kim. 2020. Taskmaster-2. <https://github.com/google-research-datasets/Taskmaster/tree/master/TM-2-2020>
- [7] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kavin Ethayarajah, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori

¹⁸We train on smaller datasets for ablations (100k conversations) than the main results (>1M conversations).

- Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshthe Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kudithipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Muniyikwa, Suraj Nair, Avani Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Nieves, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. 2021. On the Opportunities and Risks of Foundation Models. *arXiv:2108.07258* [CS.LG]
- [8] Geoffroy Bonnin and Dietmar Jannach. 2014. Automated Generation of Music Playlists: Survey and Experiments. *Comput. Surveys* 47, 2, Article 26 (nov 2014), 35 pages.
- [9] Marc Bron, Krisztian Balog, and Maarten de Rijke. 2013. Example Based Entity search in the Web of Data. In *Proceedings of the 35th European Conference on Advances in Information Retrieval (ECIR '13)*. 392–403.
- [10] Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - A Large-Scale Multi-Domain Wizard-of-Oz Dataset for Task-Oriented Dialogue Modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP '18)*. 5016–5026.
- [11] Joseph A. Calandrino, Ann Kilzer, Arvind Narayanan, Edward W. Felten, and Vitaly Shmatikov. 2011. "You Might Also Like:" Privacy Risks of Collaborative Filtering. In *Proceedings of the 2011 IEEE Symposium on Security and Privacy (SP '11)*. 231–246.
- [12] Arun Tejasvi Chaganty, Megan Leszczynski, Shu Zhang, Ravi Ganti, Krisztian Balog, and Filip Radlinski. 2023. Beyond Single Items: Exploring User Preferences in Item Sets with the Conversational Playlist Curation Dataset. *arXiv:2303.06791* [cs.IR]
- [13] Wen Chen, Pipei Huang, Jiaming Xu, Xin Guo, Cheng Guo, Fei Sun, Chao Li, Andreas Pfadler, Huan Zhao, and Binqiang Zhao. 2019. POG: Personalized Outfit Generation for Fashion Recommendation at Alibaba IFashion. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)*. 2662–2670.
- [14] Konstantina Christakopoulou, Alex Beutel, Rui Li, Sagar Jain, and Ed H. Chi. 2018. Q&R: A Two-Stage Approach Toward Interactive Recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18)*. 139–148.
- [15] Konstantina Christakopoulou, Filip Radlinski, and Katja Hofmann. 2016. Towards Conversational Recommender Systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining (KDD '16)*. 815–824.
- [16] Zhuyun Dai, Arun Tejasvi Chaganty, Vincent Y. Zhao, Aida Amini, Qazi Mamunur Rashid, Mike Green, and Kelvin Guu. 2022. Dialog Inpainting: Turning Documents into Dialogs. In *Proceedings of the 39th International Conference on Machine Learning (ICML '22)*. 4558–4586.
- [17] Arjen P. de Vries, Anne-Marie Vercouste, James A. Thom, Nick Craswell, and Mounia Lalmas. 2008. Overview of the INEX 2007 Entity Ranking Track. In *Proceedings of the 6th Initiative on the Evaluation of XML Retrieval (INEX '07)*. 245–251.
- [18] Maria del Carmen Rodríguez-Hernández, Sergio Ilarri, Ramón Hermoso, and Raquel Trillo-Lado. 2017. DataGenCARS: A Generator of Synthetic Data for the Evaluation of Context-Aware Recommendation Systems. *Pervasive and Mobile Computing* 38 (2017), 516–541.
- [19] Gianluca Demartini, Arjen P. de Vries, Tereza Iofciu, and Jianhan Zhu. 2009. Overview of the INEX 2008 Entity Ranking Track. In *Advances in Focused Retrieval: 7th International Workshop of the Initiative for the Evaluation of XML Retrieval (INEX '08)*. 243–252.
- [20] Gianluca Demartini, Tereza Iofciu, and Arjen P. de Vries. 2009. Overview of the INEX 2009 Entity Ranking Track. In *Proceedings of the Focused Retrieval and Evaluation, and 8th International Conference on Initiative for the Evaluation of XML Retrieval (INEX '09)*. 254–264.
- [21] Jesse Dodge, Andreea Gane, Xiang Zhang, Antoine Bordes, Sumit Chopra, Alexander H. Miller, Arthur Szlam, and Jason Weston. 2016. Evaluating Prerequisite Qualities for Learning End-to-End Dialog Systems. In *Proceedings of the 4th International Conference on Learning Representations (ICLR '16)*.
- [22] Ahmed Elgohary, Denis Peskov, and Jordan Boyd-Graber. 2019. Can You Unpack That? Learning to Rewrite Questions-in-Context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP '19)*. 5918–5924.
- [23] Anna Volodkevich Petr Chunaev Klavdiya Bochenina Dmitry Bugaychenko Elizaveta Stavina, Alexander Grigorievskiy. 2022. Synthetic Data-Based Simulators for Recommender Systems: A Survey. *arXiv:2206.11338* [cs.IR]
- [24] Chongming Gao, Wenqiang Lei, Xiangnan He, Maarten de Rijke, and Tat-Seng Chua. 2021. Advances and Challenges in Conversational Recommender Systems: A Survey. *AI Open* 2 (2021), 100–126.
- [25] Luyu Gao, Zhuyun Dai, Tongfei Chen, Zhen Fan, Benjamin Van Durme, and Jamie Callan. 2021. Complement Lexical Retrieval Model with Semantic Residual Embeddings. In *Advances in Information Retrieval: 43rd European Conference on IR Research (ECIR '21)*. 146–160.
- [26] Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, and Jiawei Zhang. 2023. Chat-REC: Towards Interactive and Explainable LLMs-Augmented Recommender System. *arXiv:2303.14524* [cs.IR]
- [27] Jean Garcia-Gathright, Brian St. Thomas, Christine Hosey, Zahra Nazari, and Fernando Diaz. 2018. Understanding and Evaluating User Satisfaction with Music Discovery. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18)*. 55–64.
- [28] Shijie Geng, Shuchang Liu, Zuhui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022. Recommendation as Language Processing (RLP): A Unified Pretrain, Personalized Prompt & Predict Paradigm (P5). In *Proceedings of the 16th ACM Conference on Recommender Systems (RecSys '22)*. 299–315.
- [29] Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldrige, Eugene Ie, and Diego Garcia-Olano. 2019. Learning Dense Representations for Entity Retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL '19)*. 528–537.
- [30] Carlos A. Gomez-Urbe and Neil Hunt. 2016. The Netflix Recommender System: Algorithms, Business Value, and Innovation. *ACM Transactions on Management Information Systems* 6, 4 (2016).
- [31] Christina Göpfert, Yinlam Chow, Chih-Wei Hsu, Ivan Vendrov, Tyler Lu, Deepak Ramachandran, and Craig Boutilier. 2022. Discovering Personalized Semantics for Soft Attributes in Recommender Systems Using Concept Activation Vectors. In *Proceedings of the ACM Web Conference 2022 (WWW '22)*. 2411–2421.
- [32] Javeria Habib, Shuo Zhang, and Krisztian Balog. 2020. IAI MovieBot: A Conversational Movie Recommender System. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management (CIKM '20)*.
- [33] Shirley Anugrah Hayati, Dongyeop Kang, Qingxiaoyang Zhu, Weiyan Shi, and Zhou Yu. 2020. INSPIRED: Toward Sociable Recommendation Dialog Systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP '20)*. 8142–8152.
- [34] Katja Hofmann, Lihong Li, and Filip Radlinski. 2016. Online Evaluation for Information Retrieval. *Foundations and Trends® in Information Retrieval* 10, 1 (2016), 1–117.
- [35] Qingqing Huang, Aren Jansen, Joonseok Lee, Ravi Ganti, Judith Yue Li, and Daniel P. W. Ellis. 2022. MuLan: A Joint Embedding of Music Audio and Natural Language. In *Proceedings of the 23rd International Society for Music Information Retrieval Conference (ISMIR '22)*.
- [36] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised Dense Information Retrieval with Contrastive Learning. *arXiv:2112.09118* [cs.IR]
- [37] Dietmar Jannach. 2022. Evaluating Conversational Recommender Systems. *Artificial Intelligence Review* (2022).
- [38] Dietmar Jannach, Ahtsham Manzoor, Wanling Cai, and Li Chen. 2021. A Survey on Conversational Recommender Systems. *Comput. Surveys* 54, 5 (2021).
- [39] Arjan J. P. Jeckmans, Michael Beye, Zekeriya Erkin, Pieter Hartel, Reginald L. Legendijk, and Qiang Tang. 2013. Privacy in Recommender Systems. *Social Media Retrieval* (2013), 263–281.
- [40] Meihuizi Jia, Ruixue Liu, Peiying Wang, Yang Song, Zexi Xi, Haobin Li, Xin Shen, Meng Chen, Jinhui Pang, and Xiaodong He. 2022. E-ConvRec: A Large-Scale Conversational Recommendation Dataset for E-Commerce Customer Service. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference (LREC '22)*. 5787–5796.
- [41] Ray Jiang, Sven Gouw, Yuqiu Qian, Timothy Mann, and Danilo J. Rezende. 2019. Beyond Greedy Ranking: Slate Optimization via List-CVAE. In *International Conference on Learning Representations (ICLR '19)*.
- [42] Satyen Kale, Lev Reyzin, and Robert E. Schapire. 2010. Non-Stochastic Bandit Slate Problems. In *Proceedings of the 23rd International Conference on Neural Information Processing Systems (NeurIPS '10)*. 1054–1062.
- [43] Dongyeop Kang, Anusha Balakrishnan, Pararth Shah, Paul Crook, Y-Lan Boureau, and Jason Weston. 2019. Recommendation as a Communication Game: Self-Supervised Bot-Play for Goal-oriented Dialogue. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP '19)*. 1951–1961.
- [44] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP '20)*. 6769–6781.

- [45] J. F. Kelley. 1984. An Iterative Design Methodology for User-Friendly Natural Language Office Information Applications. *ACM Transactions on Office Information Systems* 2, 1 (1984), 26–41.
- [46] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (2009), 30–37.
- [47] Yehuda Koren, Steffen Rendle, and Robert Bell. 2022. Advances in Collaborative Filtering. *Recommender Systems Handbook* (2022), 91–142.
- [48] Antonios Minas Krasakis, Andrew Yates, and Evangelos Kanoulas. 2022. Zero-Shot Query Contextualization for Conversational Search. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*. 1880–1884.
- [49] Taku Kudo. 2018. Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL '18)*. Melbourne, Australia, 66–75.
- [50] Harsh Lara and Manoj Tiwari. 2022. Evaluation of Synthetic Datasets for Conversational Recommender Systems. arXiv:2212.08167 [cs.CL]
- [51] Martin Josifoski Sebastian Riedel Luke Zettlemoyer Ledell Wu, Fabio Petroni. 2020. Zero-shot Entity Linking with Dense Entity Retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP '20)*. 6397–6407.
- [52] Wenqiang Lei, Xiangnan He, Yisong Miao, Qingyun Wu, Richang Hong, Min-Yen Kan, and Tat-Seng Chua. 2020. Estimation-action-reflection: Towards deep interaction between conversational and recommender systems. In *Proceedings of the 13th International Conference on Web Search and Data Mining (WSDM '20)*. 304–312.
- [53] Megan Leszczynski, Daniel Fu, Mayee Chen, and Christopher Re. 2022. TABi: Type-Aware Bi-Encoders for Open-Domain Entity Retrieval. In *Findings of the Association for Computational Linguistics: ACL 2022 (ACL Findings '22)*. 2147–2166.
- [54] Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. 2018. Towards Deep Conversational Recommendations. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems (NeurIPS '18)*. 9748–9758.
- [55] Sheng-Chieh Lin, Jheng-Hong Yang, Rodrigo Nogueira, Ming-Feng Tsai, Chuan-Ju Wang, and Jimmy Lin. 2020. Multi-Stage Conversational Passage Retrieval: An Approach to Fusing Term Importance Estimation and Neural Query Rewriting. arXiv:2005.02230 [cs.CL]
- [56] Fan Liu, Zhiyong Cheng, Huilin Chen, Yinwei Wei, Liqiang Nie, and Mohan Kankanhalli. 2022. Privacy-Preserving Synthetic Data Generation for Recommendation Systems. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*. 1379–1389.
- [57] Zeming Liu, Haifeng Wang, Zheng-Yu Niu, Hua Wu, and Wanxiang Che. 2021. DuRecDial 2.0: A Bilingual Parallel Corpus for Conversational Recommendation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP '21)*. 4335–4347.
- [58] Zeming Liu, Haifeng Wang, Zheng-Yu Niu, Hua Wu, Wanxiang Che, and Ting Liu. 2020. Towards Conversational Recommendation over Multi-Type Dialogs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL '20)*. 1036–1049.
- [59] Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021. Sparse, Dense, and Attentional Representations for Text Retrieval. *Transactions of the Association for Computational Linguistics* 9 (2021), 329–345.
- [60] Ji Ma, Ivan Korotkov, Yinfei Yang, Keith Hall, and Ryan McDonald. 2021. Zero-shot Neural Passage Retrieval via Domain-targeted Synthetic Question Generation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL '21)*. 1075–1088.
- [61] Kelong Mao, Zhicheng Dou, and Hongjin Qian. 2022. Curriculum Contrastive Context Denoising for Few-Shot Conversational Dense Retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*. 176–186.
- [62] Seunghwan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. OpenDialKG: Explainable Conversational Reasoning with Attention-based Walks over Knowledge Graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL '19)*. 845–854.
- [63] Jianmo Ni, Chen Qu, Jing Lu, Zhuynun Dai, Gustavo Hernández Ábrego, Ji Ma, Vincent Y. Zhao, Yi Luan, Keith B. Hall, Ming-Wei Chang, and Yinfei Yang. 2021. Large Dual Encoders Are Generalizable Retrievers. arXiv:2112.07899 [cs.IR]
- [64] Javier Parapar and Filip Radlinski. 2021. Diverse User Preference Elicitation with Multi-Armed Bandits. In *Proceedings of the ACM international Conference on Web Search and Data Mining (WSDM '21)*. 130–138.
- [65] Marden Pasinato, Carlos Eduardo Mello, Marie-Aude Aufaure, and Geraldo Zimbrão. 2013. Generating Synthetic Data for Context-Aware Recommender Systems. In *2013 BRICS Congress on Computational Intelligence and 11th Brazilian Congress on Computational Intelligence (BRICS-CCI-CBIC '13)*. 563–567.
- [66] H. Polat and Wenliang Du. 2003. Privacy-Preserving Collaborative Filtering Using Randomized Perturbation Techniques. In *Third IEEE International Conference on Data Mining*. 625–628.
- [67] Vladimir Provalov, Elizaveta Stavina, and Petr Chunaev. 2021. SynEvaRec: A Framework for Evaluating Recommender Systems on Synthetic Data Classes. In *2021 International Conference on Data Mining Workshops (ICDMW '21)*. 55–64.
- [68] Filip Radlinski and Nick Craswell. 2017. A Theoretical Framework for Conversational Search. In *Proceedings of the 2017 Conference on Human Information Interaction and Retrieval (CHIIR '17)*. 117–126.
- [69] Filip Radlinski, Madhu Kurup, and Thorsten Joachims. 2008. How Does Click-through Data Reflect Retrieval Quality?. In *Proceedings of the 17th ACM Conference on Information and Knowledge Mining (CIKM '08)*.
- [70] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research* 21, 140 (2020), 1–67.
- [71] N. Ramakrishnan, B.J. Keller, B.J. Mirza, A.Y. Grama, and G. Karypis. 2001. Privacy Risks in Recommender Systems. *IEEE Internet Computing* 5, 6 (2001), 54–63.
- [72] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL '16)*. 1715–1725.
- [73] Minjoon Seo, Jinhyuk Lee, Tom Kwiatkowski, Ankur Parikh, Ali Farhadi, and Hannaneh Hajishirzi. 2019. Real-Time Open-Domain Question Answering with Dense-Sparse Phrase Index. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL '19)*. 4430–4441.
- [74] Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive Learning Rates with Sublinear Memory Cost. arXiv:1804.04235 [cs.LG]
- [75] Manel Slokom. 2018. Comparing Recommender Systems Using Synthetic Data. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys '18)*. 548–552.
- [76] Yueming Sun and Yi Zhang. 2018. Conversational recommender system. In *The 41st international acm sigir conference on research & development in information retrieval*. 235–244.
- [77] Peter Sunehag, Richard Evans, Gabriel Dulac-Arnold, Yori Zwols, Daniel Visentin, and Ben Coppin. 2015. Deep Reinforcement Learning with Attention for Slate Markov Decision Processes with High-Dimensional States and Actions. arXiv:1512.01124 [cs.AI]
- [78] Svitlana Vakulenko, Shayne Longpre, Zhucheng Tu, and Raviteja Anantha. 2021. Question Rewriting for Conversational Question Answering (WSDM '21). 355–363.
- [79] Daniel Valcarce, Alejandro Bellogín, Javier Parapar, and Pablo Castells. 2018. On the Robustness and Discriminative Power of Information Retrieval Metrics for Top-N Recommendation. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys '18)*. 260–268.
- [80] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation Learning with Contrastive Predictive Coding. arXiv:1807.03748 [cs.LG]
- [81] Mengting Wan, Di Wang, Jie Liu, Paul Bennett, and Julian McAuley. 2018. Representing and Recommending Shopping Baskets with Complementarity, Compatibility and Loyalty. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM '18)*. 1133–1142.
- [82] Lingzhi Wang, Huang Hu, Lei Sha, Can Xu, Daxin Jiang, and Kam-Fai Wong. 2022. RecnDial: A Unified Framework for Conversational Recommendation with Pretrained Language Models. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (AACL-IJCNLP '22)*. 489–500.
- [83] Tsung-Hsien Wen, David Vandyke, Nikola Mrksic, Milica Gasic, Lina M Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2016. A Network-Based End-to-End Trainable Task-Oriented Dialogue System. arXiv:1604.04562 [CS.CL]
- [84] Yikun Xian, Tong Zhao, Jin Li, Jim Chan, Andrey Kan, Jun Ma, Xin Luna Dong, Christos Faloutsos, George Karypis, S. Muthukrishnan, and Yongfeng Zhang. 2021. EX3: Explainable Attribute-Aware Item-Set Recommendations. In *Proceedings of the 15th ACM Conference on Recommender Systems (RecSys '21)*. 484–494.
- [85] Shi Yu, Jiahua Liu, Jingqin Yang, Chenyan Xiong, Paul Bennett, Jianfeng Gao, and Zhiyuan Liu. 2020. Few-Shot Generative Conversational Query Rewriting. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20)*. 1933–1936.
- [86] Shi Yu, Zhenghao Liu, Chenyan Xiong, Tao Feng, and Zhiyuan Liu. 2021. Few-Shot Conversational Dense Retrieval. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21)*. 829–838.
- [87] Hamed Zamani, Johanne R. Trippas, Jeff Dalton, and Filip Radlinski. 2022. Conversational Information Seeking. arXiv:2201.08808 [cs.IR]
- [88] Shuo Zhang and Krisztian Balog. 2017. EntiTables: Smart Assistance for Entity-Focused Tables. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '17)*. 255–264.
- [89] Shuo Zhang and Krisztian Balog. 2020. Evaluating Conversational Recommender Systems via User Simulation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '20)*. 1512–1520.

- [90] Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W. Bruce Croft. 2018. Towards Conversational Search and Recommendation: System Ask, User Respond. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM '18)*. 177–186.
- [91] Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W. Bruce Croft. 2018. Towards conversational search and recommendation: System ask, user respond. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM '18)*. 177–186.
- [92] Xiaoxue Zhao, Weinan Zhang, and Jun Wang. 2013. Interactive Collaborative Filtering. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management (CIKM '13)*. 1411–1420.
- [93] Kun Zhou, Xiaolei Wang, Yuanhang Zhou, Chenzhan Shang, Yuan Cheng, Wayne Xin Zhao, Yaliang Li, and Ji-Rong Wen. 2021. CRSLab: An Open-Source Toolkit for Building Conversational Recommender System. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations (ACL-IJCNLP '21)*. 185–193.
- [94] Kun Zhou, Yuanhang Zhou, Wayne Xin Zhao, Xiaoke Wang, and Ji-Rong Wen. 2020. Towards Topic-Guided Conversational Recommender System. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING '20)*. 4128–4139.
- [95] Jie Zou, Yifan Chen, and Evangelos Kanoulas. 2020. Towards Question-Based Recommender Systems. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20)*. 881–890.
- [96] Lixin Zou, Long Xia, Yulong Gu, Xiangyu Zhao, Weidong Liu, Jimmy Xiangji Huang, and Dawei Yin. 2020. Neural Interactive Collaborative Filtering. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20)*. 749–758.