Empowering Retrieval-based Conversational Recommendation with Contrasting User Preferences

Heejin Kook[*], Junyoung Kim[*], Seongmin Park, Jongwuk Lee[†]
Sungkyunkwan University, Republic of Korea
`{hjkook, junyoung44, psm1206, jongwuklee}@skku.edu`

Abstract

Conversational recommender systems (CRSs) are designed to suggest the target item that the user is likely to prefer through multi-turn conversations. Recent studies stress that capturing sentiments in user conversations improves recommendation accuracy. However, they employ a single user representation, which may fail to distinguish between contrasting user intentions, such as *likes and dislikes*, potentially leading to suboptimal performance. To this end, we propose a novel conversational recommender model, called ***CO**ntrasting user p**R**eference exp**A**nsion and **L**earning* (***CORAL***). Firstly, CORAL extracts the user's hidden preferences through *contrasting preference expansion* using the reasoning capacity of the LLMs. Based on the potential preference, CORAL explicitly differentiates the contrasting preferences and leverages them into the recommendation process via *preference-aware learning*. Extensive experiments show that CORAL significantly outperforms existing methods in three benchmark datasets, improving up to 99.72% in Recall@10. The code and datasets are available at `https://github.com/kookeej/CORAL`.

1 Introduction

Conversational recommender systems (CRSs) (Lu et al., 2021; Wang et al., 2022; Zhang et al., 2024; Li et al., 2024) deliver personalized recommendations by deeply understanding users' evolving contexts through multi-turn interactions. Typically, CRSs consist of the following two tasks: *recommendation task* to provide items by identifying the user's intent from the text, and *generation task* to offer a human-friendly response to the user. While recent LLMs have shown impressive performance in natural language generation (He et al., 2023; Huang et al., 2024), the recommendation task is yet challenging to address. In this paper, we thus focus on improving the recommendation task.

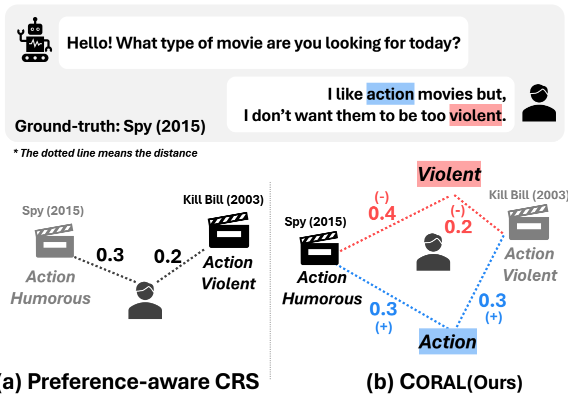


Figure 1: Comparison between traditional preference modeling and contrasting preference-aware modeling.

Existing CRS methods (Chen et al., 2019; Zhou et al., 2020; Wang et al., 2022) leverage external information (*e.g.*, knowledge graphs) or LLMs' reasoning capabilities to understand dialogue context and recommend user's preferable items. While user conversations often contain both positive and negative user preferences, they assume that the entities in the dialogue history are positive. Capturing various user sentiments or emotions is critical for understanding hidden user preferences in the decision-making process (Lerner et al., 2015; Zhang et al., 2024).

To address this issue, recent studies (Lu et al., 2021; Zhang et al., 2024; Xi et al., 2024; Xu et al., 2021; Zhao et al., 2023) introduce preference-aware CRS models that extract user preferences from conversations across multiple sentiments or emotions, leveraging them for more precise item recommendations. For example, Zhang et al. (2024) estimated probabilities for nine preference labels for each user utterance and integrated these probabilities into a unified user representation. However, they failed to model the complicated relationship among the user, items, and individual preferences

[*]Equal contribution.
[†]Corresponding author.

Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 7692–7707
April 29 - May 4, 2025 ©2025 Association for Computational Linguistics

by representing contrasting preferences with a single user representation.

In the conversation, users express various preferences with opposing intentions, such as like and dislike, which can be referred to as *contrasting preferences*. Figure 1 illustrates an example of the importance of differentiating contrasting preferences, where the number indicates the semantic distance between the user and the item. Although the user expresses a negative sentiment towards *"violent"*, existing preference-aware CRS models may still recommend *"Kill Bill (2003)"*, which contains a similar characteristic to the user utterance. In contrast, considering negative preferences directly enables us to recommend *"Spy (2015)"*, where the user may exhibit less negative sentiment toward *"violent"*.

Based on this observation, we ask the following key questions: *(i) How do we extract contrasting user preferences from the conversation? (ii) How do we learn the relationship between the contrasting preferences and the user/item?*

To this end, we propose a novel retrieval-based CRS framework, ***CO**ntrasting user p**R**eference exp**A**nsion and **L**earning (**CORAL**)*, which extracts and learns contrasting preferences. Specifically, it has two key components. Firstly, we utilize *contrasting preference expansion* using the advanced reasoning power of LLMs to accurately distinguish contrasting preferences within user-system dialogues into positive (*i.e.*, *like*) and negative (*i.e.*, *dislike*) preferences. Then, the extracted preferences are augmented for the recommendation task to elicit the user's potential preferences. In this process, we utilize a dense retrieval model to extract users, items, and preferences, thus representing them within the same representation space.

Secondly, *preference-aware learning* is used to capture two opposite user preferences to identify whether the user will like or dislike the given item, as depicted in Figure 1(b). Given the textual representations, which consist of dialogue and item descriptions, along with like/dislike preferences obtained from the contrasting preference expansion stage, we input them into an encoder and optimize item representations to be semantically closer to dialogue and like preferences while being pushed further apart from dislike preferences. We also use negative sampling to enable the model to distinguish between items that are difficult to classify based solely on conversation representation, enhancing preference representation. Therefore, it allows us to directly associate both preference types to calculate the recommendation scores of items.

The main contributions of our work are summarized as follows:

- **Recommendation-tailored Augmentation**: We extract contrasting user preferences to achieve effective user preference modeling. Leveraging the reasoning capabilities of LLMs, we identify complex preferences expressed in natural language within the dialogue and augment potential preferences using prompts tailored for recommendation tasks.
- **Preference-aware Recommender**: To directly involve contrasting user preferences, we explicitly model and learn the relationships among the users, preferences, and items. Explicitly separating both like and dislike preferences provides a rationale for the recommendations, enhancing the interpretability and transparency of our approach.
- **Comprehensive Validation**: CORAL outperforms seven baselines across three datasets, improving up to 99.72% in Recall@10. Notably, the ablation study demonstrates the effectiveness of learning preference relationships by separating like/dislike from user preferences.

2 Preliminaries

2.1 Problem Statement

Let u and i denote a user and an item of user set $\mathcal{U}$ and item set $\mathcal{I}$. Each item i contains a key-value list of metadata represented as $\{(a_m, v_m)\}_{m=1}^{|i|}$, where a_m and v_m denote the textual attribute (*e.g.*, Title) and the corresponding textual value (*e.g.*, Frozen(2013)) of m-th metadata, respectively. Here, $|i|$ represents the number of metadata entries associated with item i. The dialogue history of u is denoted as $c = \{(s_t, u_t)\}_{t=1}^{|c|}$, where u_t is the utterance at t-th turn, $|c|$ is the number of turns within c and s_t is the speaker at t-th turn, either the *user*, seeking an item or the *system*, providing personalized recommendations, respectively.

The goal of CRSs is to offer a set of candidate items to the user at the n-th turn, based on the dialogue history $c = \{(s_t, u_t)\}_{t=1}^{n-1}$ and the available metadata.

2.2 Retrieval-based CRSs

Retrieval-based CRS models (Gupta et al., 2023) recommend items based on the similarity between

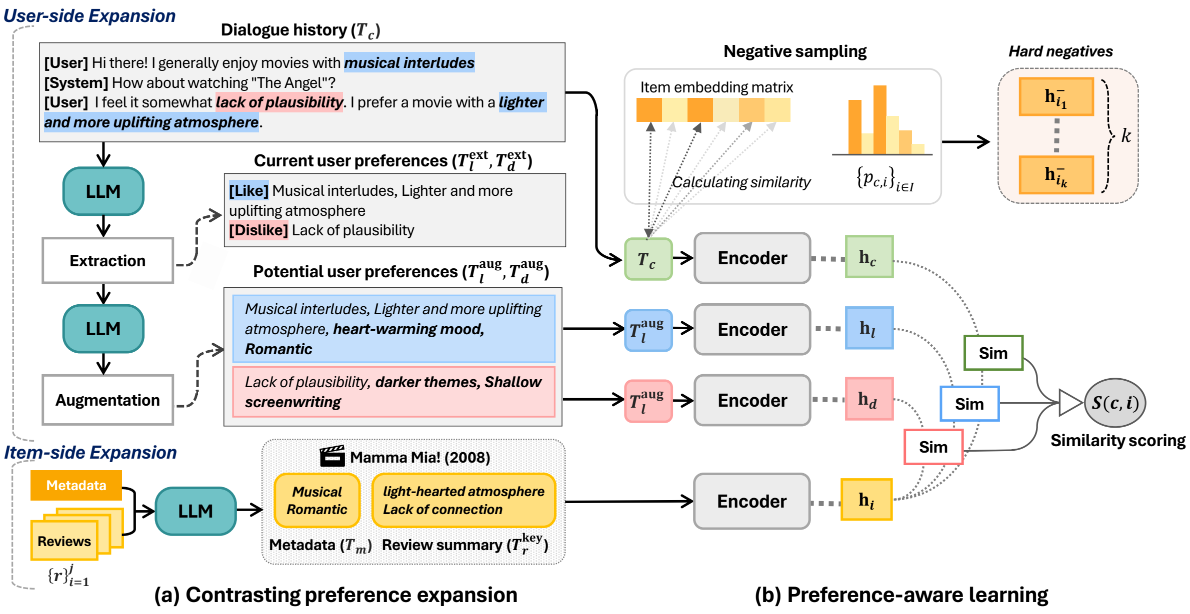


Figure 2: Overall architecture of CORAL. It comprises two components: (i) *Contrasting Preference Expansion*, which extracts superficial user preferences and augments potential preferences implicitly present in the conversation (Section 3.1); and (ii) *Preference-aware Learning*, which directly models the relationships among the user, contrasting preferences, and items (Section 3.2).

the textual representations of the dialogue T_c and the item T_m. Specifically, T_c is obtained by concatenating the speaker and utterance of every turn in the conversation (*e.g.*, T_c = *"[User] Hi there! ... [System] How about ..."*). The similarity score $\mathrm{sim}(c, i)$ between c and i is calculated as follows:

$$S(c, i) = \mathrm{sim}(\mathrm{Enc}(T_c), \mathrm{Enc}(T_m)), \qquad (1)$$

where sim denotes the similarity function (*e.g.*, dot product), and Enc denotes the encoder-based language models such as BERT (Devlin et al., 2019).

3 Proposed Model: CORAL

This section presents a novel CRS framework CORAL, as depicted in Figure 2, designed to differentiate between contrasting user preferences and effectively associate them with items.

3.1 Contrasting Preference Expansion

Figure 2(a) illustrates the overall process of contrasting preference expansion. We perform the user- and the item-side preference expansion. The user-side expansion aims to distinguish and enhance contrasting preferences into like/dislike ones, and the item-side expansion addresses the discrepancy between dialogue and item metadata.

3.1.1 User-side Expansion

User-side expansion distinguishes and infers user preferences embedded in the dialogue. We decompose the problem into two sub-tasks. First, we utilize an LLM to analyze the dialogue history and accurately extract the user's superficial preferences. These preferences serve as evidential input, enabling the LLM to infer additional implicit and potential preferences from the conversation (Gao et al., 2024; Madaan et al., 2023).

Step 1: Superficial preference extraction. For a given user u, we incorporate the dialogue into a prompt template P_{ext} and input it into LLMs. We provide a detailed example in the Appendix B.

$$T_l^{\text{ext}}, T_d^{\text{ext}} = f_{\text{LLM}}(P_{\text{ext}}, T_c), \qquad (2)$$

where T_l^{ext} and T_d^{ext} represent lists of keyphrases that capture the like and dislike preferences of user u, respectively. By leveraging the reasoning power of LLM, it is possible to analyze the context of dialogue history and accurately extract the user's meaningful surface preferences, allowing for precise distinction between likes and dislikes.

Step 2: Potential preference augmentation. To improve recommendation accuracy, we leverage the reasoning power of LLMs to augment the user's potential preferences. We use the extracted superficial preferences of user u as a rationale to infer potential preferences that the user might like, following self-feedback approaches in previous studies (Madaan et al., 2023; Gao et al., 2024).

$$T_l^{\text{aug}}, T_d^{\text{aug}} = f_{\text{LLM}}(P_{\text{aug}}, T_c, T_l^{\text{ext}}, T_d^{\text{ext}}), \quad (3)$$

where T_l^{aug} and T_d^{aug} denote lists of keyphrases that infer the potential positive and negative preferences of user u, respectively.

3.1.2 Item-side Expansion

To enhance the representation of an item, we adopted review summaries. Item metadata often diverges from the nature of conversations due to the absence of preferences. The preferences in the review bridge the gap between user conversation and item metadata, making the item representations more expressive.

Specifically, we crawled reviews for item i from IMDb[1] and selected the top-j reviews based on helpful votes. Using an LLM, we extracted and summarized common preferences from the reviews and identified keyphrases to optimize item expressiveness. This process condenses the representation, diminishes redundancy, and enhances contextual relevance.

$$T_r^{\text{sum}} = f_{\text{LLM}}(P_{\text{summary}}, T_{\text{m}}, \{r\}_{i=1}^{j}), \quad (4)$$

where r denotes individual reviews that have been crawled, and the top-j reviews are inserted into the prompt P_{summary}, which is designed to summarize multiple reviews. The common preferences summary for all the generated reviews is T_r^{sum}, which is then processed through an LLM to extract keyphrases using prompt template $P_{\text{keyphrase}}$.

$$T_r^{\text{key}} = f_{\text{LLM}}(P_{\text{keyphrase}}, T_r^{\text{sum}}). \quad (5)$$

Further explanation, along with concrete examples of this process, is provided in Appendix B.

3.2 Preference-aware Learning

Existing studies (Lu et al., 2021; Zhang et al., 2024) utilize differentiated user preferences but a single representation to represent contrasting preferences. In contrast, we explicitly represent these preferences separately from the conversation and learn the relationship between them to engage the preferences in item scoring directly.

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

3.2.1 Preference Modeling

Representation. The encoding function f_{Enc} takes the given dialogue history T_c as input and returns a vector representation of the dialogue history c, denoted as $\mathbf{h}_c = f_{\text{Enc}}(T_c) \in \mathbb{R}^d$. Specifically, T_c is tokenized and passed through a text encoder(Enc). The last hidden states of all tokens are subsequently mean-pooled, followed by L2 normalization to derive $\mathbf{h}_c$.

$$[t_1, \ldots, t_w] = \text{Tokenize}(T_c) \quad (6)$$

$$[\mathbf{o}_1, \ldots, \mathbf{o}_w] = \text{Enc}([t_1, \ldots, t_w]) \quad (7)$$

$$\mathbf{h}_c = \frac{\mathbf{o}_c}{\|\mathbf{o}_c\|_2}, \text{ where } \mathbf{o}_c = \frac{1}{w}\sum_{i=1}^{w} \mathbf{o}_i, \quad (8)$$

where t is the token, w is the length of tokenized T_c and $\mathbf{o}_i \in \mathbb{R}^d$ is the vector of last hidden state.

Similarly, for T_l^{aug} and T_d^{aug}, we obtain the like/dislike preference representations, $\mathbf{h}_l$ and $\mathbf{h}_d$.

$$\mathbf{h}_l = f_{\text{Enc}}(T_l^{\text{aug}}) \text{ and } \mathbf{h}_d = f_{\text{Enc}}(T_d^{\text{aug}}). \quad (9)$$

Inspired by previous work (Hou et al., 2024; Lei et al., 2024), we concatenate item metadata and review information to obtain a review-enhanced item vector representation.

$$\mathbf{h}_i = f_{\text{Enc}}(T_m \oplus T_r^{\text{key}}), \quad (10)$$

where the $\oplus$ denotes concatenation.

Consequently, we obtain three user-side representations (*i.e.*, $\mathbf{h}_c$, $\mathbf{h}_l$, and $\mathbf{h}_d$) and one item-side representation (*i.e.*, $\mathbf{h}_i$).

Similarity scoring. To compute the final score, we linearly aggregate three scores—the similarity between the dialogue history and item, between like preference and item, and between dislike preference and item. The goal is to ensure that the desired item is close to the conversation context and the user's like preference while being far from the user's dislike preference. We extend Equation (1) as follows:

$$S(c, i) = \text{sim}(\mathbf{h}_c, \mathbf{h}_i) + \alpha \cdot \text{sim}(\mathbf{h}_l, \mathbf{h}_i) - \beta \cdot \text{sim}(\mathbf{h}_d, \mathbf{h}_i), \quad (11)$$

where $\alpha, \beta \in (0, 1]$ are hyperparameters representing the importance of the user's like and dislike preferences. Empirically, α is set to 0.5, and β is adjusted in [0.1, 0.3] depending on the dataset.

This approach is a simple yet effective way of reflecting contrasting preferences, requiring no additional parameters. Furthermore, it enhances interpretability by intuitively revealing which preferences the recommended items are derived from, demonstrated in a case study (Section 5.3).

3.2.2 Training

Hard negative sampling. Hard negative sampling directly impacts the convergence and performance of dense retrieval models (Xiong et al., 2021).

Our key contribution lies in utilizing hard negative sampling to enhance the representation of user preferences, especially for samples that are challenging to predict based on conversation alone. Specifically, we first compute the similarity between $\mathbf{h}_c$ and all item embeddings, then apply softmax to convert the similarity scores into a probability distribution. Using this distribution $\{p_{c,i}\}_{i\in I}$, we sample a set of k negative items $\mathcal{I}_c{}^-$, resulting in a set of hard negative samples, further enriching the model's understanding of user preferences.

$$p_{c,i} = \frac{\exp(\mathrm{sim}(\mathbf{h}_c, \mathbf{h}_i))}{\sum_{j\in\mathcal{I}} \exp((\mathrm{sim}(\mathbf{h}_c, \mathbf{h}_j))} \tag{12}$$

$$\begin{aligned}\mathcal{I}_c{}^- &= \{i_1, i_2, \ldots, i_k\} \\ &\sim \text{Multinomial}\left(k, \{p_{c,i}\}_{i\in\mathcal{I}}\right).\end{aligned} \tag{13}$$

Loss function. For training, we utilize the cross-entropy loss $\mathcal{L}$ as follows.

$$\mathcal{L} = -\log \frac{\exp\left(S\left(c, i^+\right)/\tau\right)}{\sum_{i\in\mathcal{I}_c{}^-} \exp\left(S\left(c, i\right)/\tau\right)}, \tag{14}$$

where i^+ is the positive item of c, and τ is a hyperparameter to adjust the temperature.

Training procedure. For training efficiency, we compute $\mathbf{h}_c$ and $\mathbf{h}_i$ for all items c and i in the training set at the beginning of each epoch to obtain hard negative samples for the entire training set. The pre-computed $\mathbf{h}_i$ values are stored in an item embedding table, allowing the model to retrieve item embeddings directly from the table during training without passing them through the encoder again. This approach reduces the time complexity and enables more efficient training.

4 Experimental Setup

4.1 Datasets

To evaluate the performance of CORAL, we utilize three benchmark datasets in the movie domain. INSPIRED (Hayati et al., 2020) and REDIAL (Li et al., 2018) are widely used datasets built through crowdsourcing on the Amazon Mechanical Turk (AMT) platform. PEARL (Kim et al., 2024) is a dataset constructed based on movie reviews, designed to reflect the user's persona in conversations. The dataset statistics are summarized in Table 1.

Dataset	#Dial.	#Items	#Likes	#Dislikes
PEARL	57,159	9,685	9.59	5.99
INSPIRED	2,017	1,058	11.11	5.65
REDIAL	31,089	5,896	10.99	1.00

Table 1: Data statistics. 'Dial.' represents dialogue history, while '# Likes' and '# Dislikes' refer to the average counts of the like and dislike preferences after the augmentation stage, respectively.

4.2 Evaluation Protocol

To evaluate the recommendation performance on the CRS models, we utilize the widely used ranking metrics NDCG@k and Recall@k (with $k = 10, 50$). Notably, previous research (He et al., 2023) has found that ground-truth items already seen in previous dialogue can lead to shortcuts. Therefore, we exclude these items from the ground-truth set to ensure a more accurate assessment following (He et al., 2023; Xi et al., 2024; He et al., 2024).

4.3 Baselines

We compare CORAL with seven baselines.

- **Traditional CRS models**: **UniCRS** (Wang et al., 2022), **RevCore** (Lu et al., 2021), and **ECR** (Zhang et al., 2024) leverage domain-specific knowledge via knowledge graphs. For each entity, RevCore categorizes sentiments into positive or negative, whereas ECR identifies nine distinct emotional responses.

- **LLM-based CRS models**: **Zero-shot** recommends based solely on dialogue history and internal knowledge of items. **ChatCRS** (Li et al., 2024) enhances the domain knowledge of LLM through a knowledge graph.

- **Retrieval-based CRS models**: **BM25** (Robertson and Walker, 1994) ranks items by term relevance from a static index, while **DPR** (Karpukhin et al., 2020) retrieves items based on the similarity with dense vectors of the dialogue context. We use T_c and T_m as the user and item textual representation.

Model		Traditional CRS			LLM-based CRS		Retrieval-based CRS			Gain
Dataset	Metric	RevCore	UniCRS	ECR	Zero-shot	ChatCRS	BM25	DPR	CORAL	
PEARL	R@10	0.0268	0.1156	0.0957	0.0767	0.0763	0.0026	0.0940	**0.1851***	60.07%
	R@50	0.0898	0.2624	0.2373	0.1129	0.1168	0.0123	0.2206	**0.3619***	37.94%
	N@10	0.0132	0.0642	0.0501	0.0468	0.0462	0.0014	0.0502	**0.1125***	75.17%
	N@50	0.0266	0.0958	0.0806	0.0560	0.0565	0.0033	0.0777	**0.1511***	57.74%
INSPIRED	R@10	0.0948	0.1113	0.1711	0.1436	0.1410	0.0429	0.1019	**0.3417***	99.72%
	R@50	0.3344	0.2528	0.2826	0.2436	0.2436	0.1210	0.2672	**0.5632***	68.45%
	N@10	0.0509	0.0642	0.1077	0.0927	0.0806	0.0202	0.0512	**0.1772***	64.52%
	N@50	0.1041	0.0952	0.1417	0.1175	0.1071	0.0373	0.0872	**0.2255***	59.07%
REDIAL	R@10	0.1739	0.1549	0.1685	0.1670	0.1666	0.0373	0.0774	**0.2182***	25.48%
	R@50	0.3034	0.3540	0.3793	0.2783	0.2824	0.0300	0.2138	**0.4741***	25.23%
	N@10	0.1053	0.0776	0.0805	0.0937	0.0893	0.0032	0.0403	**0.1128***	7.10%
	N@50	0.1337	0.1215	0.1293	0.1226	0.1191	0.0083	0.0713	**0.1724***	28.96%

Table 2: Overall performance. The best and second-best are **bold** and underlined. Gain measures the difference between CORAL and the best competitive baseline. '*' indicates statistically significant improvement ($p < 0.01$) for a paired t-test of CORAL compared to the best baseline, as conducted across 5 experiments.

4.4 Implementation Details

We utilize `gpt-4o-mini` for contrasting preference expansion. We used `gpt-4o-mini-2024-07-18` in all of our experiments including baselines. We initialize the model parameters with NV-Embed (Lee et al., 2024) (NV-Emb.) where $d = 4096$, and we applied LoRA (Hu et al., 2022), a parameter-efficient fine-tuning technique for training. The parameter α was set to 0.5, and β was set to 0.3, 0.1, and 0.2 for the INSPIRED, REDIAL, and PEARL dataset, respectively. We set the batch size to 8 for PEARL, and 10 for INSPIRED and REDIAL. For negative samples, we used 24 for PEARL and 16 for INSPIRED and REDIAL. We use Adam optimizer (Kingma and Ba, 2015) with a learning rate of 5e-5 for PEARL and REDIAL, and 1e-4 for INSPIRED. We adopt early stopping based on NDCG@10, with a patience of 3 for PEARL, and 5 for INSPIRED and REDIAL. The temperature parameter τ is set to 0.05. We set the maximum sequence length to 512 for items and conversations and 256 for likes and dislikes. The warm-up steps are set to 10% of one epoch. We set $k = 3$ for the review summarization, using 3 reviews per item. The prompts used in Contrasting Preference Expansion and examples are provided in Appendix B and detailed configurations for the baselines are provided in Appendix C.

5 Results and Analysis

5.1 Overall Performance

Table 2 compares CORAL against seven baselines across three datasets. CORAL achieves state-of-the-art performance, improving Recall@50 and

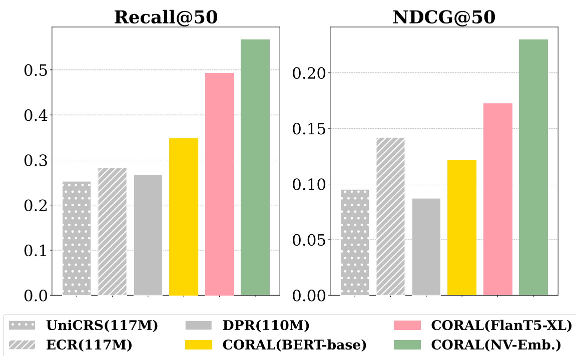


Figure 3: Performance according to various model sizes in INSPIRED.

NDCG@50 over the best competitive baseline by an average of 43.87% and 48.59%, respectively. This demonstrates that it consistently captures and models user preferences across datasets with diverse characteristics. Notably, CORAL outperforms ECR (Zhang et al., 2024), which enhances a single user representation with sentiment. This indicates that explicitly learning the relationships between preferences and users/items is more effective for user preference modeling.

RevCore (Lu et al., 2021) and ECR (Zhang et al., 2024), which utilize user sentiment, outperform UniCRS (Wang et al., 2022) in INSPIRED and REDIAL but not in PEARL. One possible reason is relatively longer dialogues in PEARL, which demand a more contextual ability to capture conversational context. RevCore and ECR focus on sentiment extraction at the entity and utterance levels, respectively, making it challenging to capture sentiment considering the full context. In contrast, CORAL identifies user sentiment at the conversation

Dataset		PEARL				INSPIRED				REDIAL			
Retriever	L, D	R@10	R@50	N@10	N@50	R@10	R@50	N@10	N@50	R@10	R@50	N@10	N@50
BM25	w/o	0.0053	0.0269	0.0031	0.0076	0.0390	0.1486	0.0216	0.0444	0.0091	0.0361	0.0044	0.0104
	w/	**0.0079**	**0.0340**	**0.0045**	**0.0099**	**0.0448**	**0.1714**	**0.0297**	**0.0580**	**0.0235**	**0.0704**	**0.0113**	**0.0219**
BERT	w/o	**0.0040**	0.0203	**0.0015**	0.0049	0.0057	0.0410	0.0017	0.0100	0.0069	0.0270	0.0039	0.0084
	w/	0.0031	**0.0225**	**0.0015**	**0.0056**	**0.0086**	**0.0648**	**0.0032**	**0.0152**	**0.0087**	**0.0302**	**0.0048**	**0.0096**
NV-Emb.	w/o	0.0454	0.1323	0.0210	0.0397	0.1648	0.3943	0.0856	0.1374	0.0767	0.1885	0.0368	0.0613
	w/	**0.0569**	**0.1508**	**0.0286**	**0.0489**	**0.2276**	**0.4048**	**0.1140**	**0.1538**	**0.0872**	**0.2190**	**0.0408**	**0.0711**

Table 3: The zero-shot performance of various language models depending on the presence or absence of the user's potential preference. L and D mean T_l^{aug} and T_d^{aug}, respectively.

Dataset	INSPIRED			
Variants	R@10	R@50	N@10	N@50
CORAL	**0.3481**	0.5667	**0.1827**	**0.2297**
w/o L, D	0.3248	**0.5767**	0.1668	0.2226
w/o R	0.3167	0.5348	0.1710	0.2193
w/o L, D, R	0.2948	0.5633	0.1595	0.2196
w/o $Aug.$	0.3016	0.5379	0.1663	0.2202
w/o $Neg.$	0.2974	0.5692	0.1520	0.2115
w/o PL	0.1847	0.3616	0.1107	0.1491

Table 4: Ablation study of CORAL in INSPIRED. The best scores are in **bold**. L, D and R denote T_l^{aug}, T_d^{aug}, and T_r^{key}, respectively. Also, *Aug.*, *Neg.*, and *PL* mean potential preference augmentation, hard negative sampling, and preference-aware learning, respectively.

level, achieving a more comprehensive understanding of user preferences throughout the dialogue.

5.2 In-depth Analysis

5.2.1 Performance by Model Size

Figure 3 presents the performance of CORAL with different backbone model sizes. We have two observations as follows. (i) The performance increases as the model size increases, *i.e.*, BERT-base (110M) → FlanT5-XL (1.5B) → NV-Embed (7B). This is because larger models are better at capturing complex semantic relationships in text. Additionally, CORAL leverages a dense retriever structure to fully exploit the capabilities of PLMs. (ii) CORAL significantly improves performance even with a relatively small model. Both DPR and CORAL (BERT-base) use the same backbone. CORAL (BERT-base) shows a 39% performance gain in NDCG@50 and achieves comparable performance to similarly sized baselines, such as UniCRS and ECR. These reveal that CORAL is a highly scalable, universal, and effective framework that can be applied to any model size.

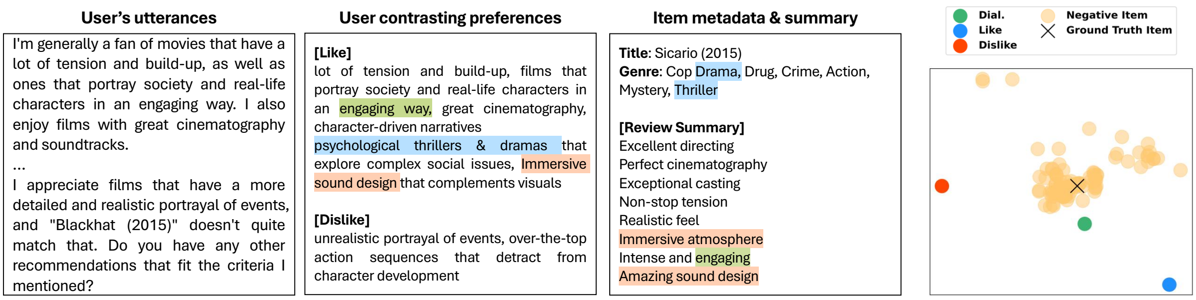
5.2.2 Zero-shot Performance

To investigate the effectiveness of user-side expansion, we compare the zero-shot performance of various models with and without leveraging the user's potential preferences. We evaluate zero-shot performance on a sparse retriever and two dense retrievers with varying sizes, as shown in Table 3. For '*w/o L, D*', we only utilize T_c for user-side textual representation without T_l^{aug}, and T_d^{aug}. '*w/ L, D*' utilizes $T_c, T_l^{\text{aug}}, T_d^{\text{aug}}$ and the score is computed as Equation (11). For the item-side textual representation, we concatenated T_m with ${T_r}^{\text{key}}$. CORAL significantly improves performance across different backbones, demonstrating a 37% average gain in the zero-shot setting. These results confirm that contrasting preference expansion effectively improves recommendation performance by inferring potential user preferences. Appendix A.1 provides more detailed ablation for user-side expansion.

5.2.3 Ablation Study

To understand the impact of each component of CORAL on performance, we conducted an ablation study on INSPIRED, as illustrated in Table 4. First, we validate the effectiveness of contrasting preference expansion. '*w/o* L, D' means that T_l^{aug} and T_d^{aug} were not used in both train and inference. We can see that all three expansions (*i.e.*, L, D, and R) contribute to performance, especially at a high ranking. It indicates that the expansions allow us to distinguish subtle preferences. We then investigate the effect of augmenting user's potential preferences. For '*w/o Aug.*', we use $T_l^{\text{ext}}, T_d^{\text{ext}}$ as the user's positive and negative preferences with only superficial preference extraction. We find that augmenting the potential preference significantly improves performance, yielding up to a 15.41% increase in Recall@10. It highlights the importance of underlying preference within the dialogue.

Lastly, we explore the effects of our proposed



(a) Case Study (b) Visualization

Figure 4: Case study and visualization of CORAL in PEARL dataset.

preference-aware learning. '*w/o Neg.*' is a variant that uses in-batch negative instead of hard negative, and '*w/o PL*' is a variant without preference modeling and hard negative, which utilizes single user representation by concatenating $T_c, T_l^{\text{aug}}, T_d^{\text{aug}}$. Compared to CORAL, removing negative sampling or preference modeling leads to a significant drop in performance. Hence, preference-aware learning effectively learns the relationship between conversation, preferences, and items. Refer to Appendix A.2 for the results of the ablation study on BERT, and Appendix A.3 for the results using different LLMs in Contrasting Preference Expansion.

5.3 Case Study and Visualization

Figure 4(a) shows a case study of the user's dialogue, the augmented preferences, and the ground truth item from PEARL. The highlighted phrases in the same color represent phrases that belong to the same concept and are related to the ground-truth item. In particular, the blue, green, and orange preferences mean newly augmented through contrasting preference expansion.

Firstly, CORAL effectively inferred the user's preference for *Thriller* and *Drama* genres based on the dialogue. This demonstrates that leveraging the augmentation effectively makes it possible to predict user preferences that are difficult to infer solely from the given dialogue history. Also, it can be observed that the user's preferences align with the item's review summary (*e.g.*, *immersive sound design* in user preference, *immersive atmosphere* in item review summary). This indicates that the review summary successfully bridges the gap between the preferences expressed in the user conversation and the item. Additionally, CORAL's contrasting preference expansion serves as a rationale for the recommendation results, thereby providing explainability.

We then visualized the corresponding example in the embedding space using t-SNE in Figure 4(b), to illustrate how CORAL utilizes preference. Green, blue, and red dots are the embedding vectors of h_c, h_l, h_d for the dialogue shown in Figure 4(a). The X marks indicate the ground truth item and the orange dots represent negative items that are not the ground truth. The example shows incorrect items, such as those close to the dialogue but positioned toward the "dislike" items, that may be selected when only the conversation is used. This is because user conversations contain contrasting preferences. However, since CORAL explicitly differentiates like/dislike preferences and models the relationship between the user and the item, it successfully recommended the correct item by distinguishing subtle differences.

6 Related Work

Traditional conversational recommender systems (CRSs) (Jannach et al., 2021) increasingly harness external information, such as knowledge graphs (Chen et al., 2019; Zhou et al., 2020; Petruzzelli et al., 2024; Wang et al., 2022; Zhou et al., 2023) and metadata (Yang et al., 2022), to enhance domain knowledge. With the substantial impact of large language models (LLMs) demonstrating exceptional world knowledge, recent studies (He et al., 2023; Liu et al., 2023; Li et al., 2024; Spurlock et al., 2024) have focused on utilizing LLMs as standalone recommenders. In particular, several approaches (Zhao et al., 2021; Li et al., 2024) have been proposed to integrate the strong contextual understanding of LLMs with knowledge graphs to address gaps in domain-specific knowledge to improve the system's overall performance.

However, these studies commonly neglect the diversity of users' emotions and attitudes toward entities in dialogues, which undermines the conversation's complexity and degrades user experience.

Preference-aware CRSs. Several studies have focused on improving user preferences at the entity level by considering user emotions using knowledge graphs (Lu et al., 2021; Zhang et al., 2024). RevCore (Lu et al., 2021) classifies entities from conversations as positive or negative and retrieves emotion-related reviews to enhance user expressiveness. ECR (Zhang et al., 2024) leverages LLMs to categorize entities into nine specific emotional categories, segmenting the user preferences. MemoCRS (Xi et al., 2024) employs a memory-enhanced approach to ensure preference continuity by tracking sequential user preferences. Although these studies consider user preferences at various levels of granularity and context, they still overlook the existence of contrasting preferences.

Retrieval-based CRSs. Recent work (Hou et al., 2024; Lei et al., 2024; Gupta et al., 2023; Kemper et al., 2024) reformulated recommender systems as item retrieval tasks, fully utilizing the semantic understanding and matching capabilities of language models. In light of this, a few studies (Gupta et al., 2023; Kemper et al., 2024) leveraging retrievers have been introduced to enhance CRS tasks. Specifically, they treat the conversation as a query and items as documents and utilize text-matching techniques such as BM25 (Lin et al., 2021), offering high generalizability and scalability.

7 Conclusion

In this paper, we proposed a new CRS model CORAL that distinguishes users' ambiguous preferences, implying conflicting intentions inherent in their likes and dislikes. To support the recommendation task, CORAL expands users' preferences and leverages the reasoning capabilities of LLMs to learn the relationships between conversations, preferences, and items. Extensive experiments are conducted to evaluate the effectiveness of our preference expansion and learning strategy, confirming that our approach surpasses all baseline models in enhancing recommendation performance (Table 2). CORAL can also be robustly and universally applied across various language models (Figure 3), and operates effectively in a zero-shot setting, demonstrating the reliability of our augmented user potential preferences (Tables 3 and 5).

Limitation

The limitations of this study can be categorized into aspects: (i) training and inference inefficiency and (ii) reliance on large language models (LLMs).

Firstly, in the contrasting preference expansion, extracting user preferences from dialogue heavily depends on the reasoning power of LLMs, which presents a challenge. The accuracy of extracted preferences and the quality of potential preferences are significantly influenced by the performance, size, bias, and knowledge scope of the LLM. Consequently, it is essential to closely examine the impact of these LLM characteristics on the learning outcomes. Secondly, in this work, we trained NV-Embed (7B) with only 10M parameters using the parameter-efficient training technique LoRA (Hu et al., 2022) and enhanced training efficiency through negative sampling. Despite these efforts, training time and computational costs remain high when using LLMs.

Acknowledgments

This work was partly supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP)-ICT Creative Consilience Program grant funded by the Korea government (MSIT) (No. RS-2019-II190421, RS-2022-II220680, IITP-2025-RS-2020-II201821, IITP-2025-RS-2024-00360227).

References

Qibin Chen, Junyang Lin, Yichang Zhang, Ming Ding, Yukuo Cen, Hongxia Yang, and Jie Tang. 2019. Towards knowledge-based recommender dialog system. *CoRR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *CoRR*.

Jinglong Gao, Xiao Ding, Yiming Cui, Jianbai Zhao, Hepeng Wang, Ting Liu, and Bing Qin. 2024. Self-evolving GPT: A lifelong autonomous experiential learner. In *ACL*, pages 6385–6432.

Raghav Gupta, Renat Aksitov, Samrat Phatale, Simral Chaudhary, Harrison Lee, and Abhinav Rastogi. 2023. Conversational recommendation as retrieval: A simple, strong baseline. *CoRR*.

Shirley Anugrah Hayati, Dongyeop Kang, Qingxiaoyang Zhu, Weiyan Shi, and Zhou Yu. 2020. Inspired: Toward sociable recommendation dialog systems. In *EMNLP*, pages 8142–8152.

Zhankui He, Zhouhang Xie, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Prasad Majumder, Nathan Kallus, and Julian Mcauley. 2023. Large language models as zero-shot conversational recommenders. In *CIKM*.

Zhankui He, Zhouhang Xie, Harald Steck, Dawen Liang, Rahul Jha, Nathan Kallus, and Julian McAuley. 2024. Reindex-then-adapt: Improving large language models for conversational recommendation. *CoRR*.

Yupeng Hou, Jiacheng Li, Zhankui He, An Yan, Xiusi Chen, and Julian J. McAuley. 2024. Bridging language and items for retrieval and recommendation. *CoRR*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *ICLR*.

Xu Huang, Jianxun Lian, Yuxuan Lei, Jing Yao, Defu Lian, and Xing Xie. 2024. Recommender ai agent: Integrating large language models for interactive recommendations. *CoRR*.

Dietmar Jannach, Ahtsham Manzoor, Wanling Cai, and Li Chen. 2021. A survey on conversational recommender systems. *ACM Computing Surveys*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense passage retrieval for open-domain question answering. *CoRR*.

Sara Kemper, Justin Cui, Kai Dicarlantonio, Kathy Lin, Danjie Tang, Anton Korikov, and Scott Sanner. 2024. Retrieval-augmented conversational recommendation with prompt-based semi-structured natural language state tracking. In *SIGIR*.

Minjin Kim, Minju Kim, Hana Kim, Beong-woo Kwak, SeongKu Kang, Youngjae Yu, Jinyoung Yeo, and Dongha Lee. 2024. Pearl: A review-driven persona-knowledge grounded conversational recommendation dataset. In *ACL*, pages 1105–1120.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2024. Nv-embed: Improved techniques for training llms as generalist embedding models. *CoRR*.

Yuxuan Lei, Jianxun Lian, Jing Yao, Mingqi Wu, Defu Lian, and Xing Xie. 2024. Aligning language models for versatile text-based item retrieval. *CoRR*.

Jennifer S Lerner, Ye Li, Piercarlo Valdesolo, and Karim S Kassam. 2015. Emotion and decision making. *Annual review of psychology*, pages 799–823.

Chuang Li, Yang Deng, Hengchang Hu, Min-Yen Kan, and Haizhou Li. 2024. Incorporating external knowledge and goal guidance for llm-based conversational recommender systems. *CoRR*.

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

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A Python toolkit for reproducible information retrieval research with sparse and dense representations. In *SIGIR*, pages 2356–2362.

Junling Liu, Chao Liu, Peilin Zhou, Renjie Lv, Kang Zhou, and Yan Zhang. 2023. Is chatgpt a good recommender? a preliminary study. *CoRR*.

Yu Lu, Junwei Bao, Yan Song, Zichen Ma, Shuguang Cui, Youzheng Wu, and Xiaodong He. 2021. Revcore: Review-augmented conversational recommendation. *CoRR*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. In *NeurIPS*.

Pablo N. Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. 2011. Dbpedia spotlight: shedding light on the web of documents. In *I-Semantics*, page 1–8.

Alessandro Petruzzelli, Alessandro Francesco Maria Martina, Giuseppe Spillo, Cataldo Musto, Marco De Gemmis, Pasquale Lops, and Giovanni Semeraro. 2024. Improving transformer-based sequential conversational recommendations through knowledge graph embeddings. In *UMAP*, page 172–182.

Stephen E. Robertson and Steve Walker. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR*, pages 232–241.

Kyle Dylan Spurlock, Cagla Acun, Esin Saka, and Olfa Nasraoui. 2024. Chatgpt for conversational recommendation: Refining recommendations by reprompting with feedback. *CoRR*.

Xiaolei Wang, Kun Zhou, Ji-Rong Wen, and Wayne Xin Zhao. 2022. Towards unified conversational recommender systems via knowledge-enhanced prompt learning. In *KDD*.

Yunjia Xi, Weiwen Liu, Jianghao Lin, Bo Chen, Ruiming Tang, Weinan Zhang, and Yong Yu. 2024. Memocrs: Memory-enhanced sequential conversational recommender systems with large language models. *CoRR*.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *ICLR*.

Kerui Xu, Jingxuan Yang, Jun Xu, Sheng Gao, Jun Guo, and Ji-Rong Wen. 2021. Adapting user preference to online feedback in multi-round conversational recommendation. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 364–372.

Bowen Yang, Cong Han, Yu Li, Lei Zuo, and Zhou Yu. 2022. Improving conversational recommendation systems' quality with context-aware item meta-information. In *NAACL*, pages 38–48.

Xiaoyu Zhang, Ruobing Xie, Yougang Lyu, Xin Xin, Pengjie Ren, Mingfei Liang, Bo Zhang, Zhanhui Kang, Maarten de Rijke, and Zhaochun Ren. 2024. Towards empathetic conversational recommender systems. In *RecSys*, page 84–93.

Mengyuan Zhao, Xiaowen Huang, Lixi Zhu, Jitao Sang, and Jian Yu. 2021. Knowledge graph-enhanced sampling for conversational recommender system. *CoRR*.

Sen Zhao, Wei Wei, Xian-Ling Mao, Shuai Zhu, Minghui Yang, Zujie Wen, Dangyang Chen, and Feida Zhu. 2023. Multi-view hypergraph contrastive policy learning for conversational recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 654–664.

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. *CoRR*.

Yuanhang Zhou, Kun Zhou, Wayne Xin Zhao, Cheng Wang, Peng Jiang, and He Hu. 2023. C2-crs: Coarse-to-fine contrastive learning for conversational recommender system. *CoRR*.

A Further Study

A.1 Effect of Preference Ablation on Zero-shot Performance

Table 5 shows the results of ablating the input information used as a query without training NV-Embed. *Avg. Gain* refers to the average performance gain when additional preferences are added compared to using only T_c. For the item-side textual representation, we concatenated T_m with ${T_r}^{\text{key}}$. Across all three benchmark datasets, using all of T_c, T_l^{aug}, and T_d^{aug} resulted in significant performance improvements. Additionally, consistently using either T_c and T_l^{aug} or T_c and T_d^{aug} also outperformed using only T_c. (i) This demonstrates that the augmented data is of high quality and contributes positively to recommendation performance. (ii) Furthermore, it confirms that the proposed separate scoring method aligns well with the user's intent.

A.2 Ablation Study on Small LM

Table 6 presents the results of an ablation study on BERT, a small language model with 110M parameters. Notably, the highest performance is achieved when all preference components are incorporated. The ablation study further confirms that the components of CORAL contribute consistently to performance improvement, even when applied to small language models.

A.3 Effect of LLM in Contrasting Preference Expansion

Table 7 shows the results of utilizing different LLMs in Contrasting Preference Expansion. CORAL$_{\text{Mistral}}$ and CORAL$_{\text{gpt-4o-mini}}$ are variants that utilize `Mistral-7B-Instruct-v0.2` and `gpt-4o-mini` as LLMs in the Contrasting Preference Expansion step, respectively, and CORAL$_{\text{w/o } L,D,R}$ is a variant that does not use an expanded preference (*i.e.*, T_l^{aug}, T_d^{aug}, T_r^{key}). Both variants utilizing different LLMs outperform UniCRS and ECR and generally achieve better performance than the variant without expanded preferences. These results highlight the effectiveness of CORAL and further validate its scalability and generalizability.

A.4 Zero-shot Performance of Different LLMs compared to CORAL

Table 8 shows the performance comparison between CORAL and large language models `GPT-4o-mini` and `GPT-4o`. We used

Dataset	Input info.	R@10	R@50	N@10	N@50	Avg. Gain(%)
PEARL	C	0.0476	0.1230	0.0229	0.0395	-
	C, L	0.0560	0.1349	0.0303	0.0474	**19.91%**
	C, D	0.0481	0.1349	0.0224	0.0415	**3.40%**
	C, L, D	0.0573	0.1481	0.0311	0.0504	**26.05%**
INSPIRED	C	0.1837	0.4133	0.1038	0.1545	-
	C, L	0.2103	0.3949	0.1133	0.1534	**4.62%**
	C, D	0.2000	0.4154	0.1064	0.1527	**2.68%**
	C, L, D	0.2205	0.4205	0.1166	0.1597	**9.37%**
REDIAL	$C,$	0.0887	0.2067	0.0383	0.0640	-
	C, L	0.0954	0.2325	0.0415	0.0710	**9.83%**
	C, D	0.0910	0.2129	0.0400	0.0665	**3.48%**
	C, L, D	0.0986	0.2431	0.0427	0.0735	**13.78%**

Table 5: Zero-shot performance for preference input variant. C, L, and D denote T_c, T_l^{aug}, and T_d^{aug}, respectively.

Dataset	INSPIRED				REDIAL			
Variants	R@10	R@50	N@10	N@50	R@10	R@50	N@10	N@50
CORAL	**0.1219**	**0.3714**	0.0625	**0.1173**	**0.0856**	0.2255	**0.0446**	**0.0765**
w/o *L, D*	0.0962	0.3429	0.0429	0.0968	0.0775	**0.2287**	0.0407	0.0754
w/o *R*	0.1133	0.2695	**0.0752**	0.1103	0.0749	0.2052	0.0405	0.0702
w/o *L, D, R*	0.0876	0.3010	0.0450	0.0967	0.0764	0.2231	0.0403	0.0738
w/o *Neg.*	0.0867	0.2629	0.0371	0.0757	0.0725	0.2151	0.0375	0.0697
w/o *PL*	0.1076	0.2467	0.0516	0.0832	0.0774	0.2138	0.0403	0.0713

Table 6: Ablation study of CORAL in INSPIRED and REDIAL on BERT. The best scores are in **bold**. L, D and R denote T_l^{aug}, T_d^{aug}, and T_r^{key}, respectively. Also, *Neg.* and *PL* mean potential hard negative sampling and preference-aware learning, respectively.

Dataset	INSPIRED			
Model	R@10	R@50	N@10	N@50
UniCRS	0.1113	0.2528	0.0642	0.0952
ECR	0.1711	0.2826	0.1077	0.1417
CORAL $_{\text{w/o } L,D,R}$	0.2948	0.5633	0.1595	0.2196
CORAL $_{\text{Mistral}}$	0.3162	0.5410	0.1809	0.2326
CORAL $_{\text{gpt-4o-mini}}$	0.3417	0.5632	0.1772	0.2255

Table 7: Performance depending on the LLM utilized for Contrasting Preference Expansion. L, D and R denote T_l^{aug}, T_d^{aug}, and T_r^{key}, respectively.

Model: Dataset	Model: Metric	LLM zero-shot: 4o-mini	LLM zero-shot: 4o	CORAL
PEARL	R@10	0.0767	0.1398	**0.1851**
	R@50	0.1129	0.1869	**0.3619**
	N@10	0.0468	0.0743	**0.1125**
	N@50	0.0560	0.0843	**0.1511**
INSPIRED	R@10	0.1436	0.2092	**0.3417**
	R@50	0.2436	0.2704	**0.5632**
	N@10	0.0927	0.1223	**0.1772**
	N@50	0.1175	0.1344	**0.2255**
REDIAL	R@10	0.1670	**0.2490**	0.2182
	R@50	0.2783	0.3347	**0.4750**
	N@10	0.0937	**0.1180**	0.1128
	N@50	0.1226	0.1370	**0.1724**

Table 8: Zero-shot performance of different LLMs compared to CORAL. `4o-mini` and `4o` refer to GPT-4o-mini and GPT-4o, respectively.

Title: The Divergent Series: Insurgent (2015)
Genre: Action, Adventure, Sci-Fi
Director: Robert Schwentke
Cast:
Shailene Woodley, Ansel Elgort, Theo James

Like:
Intriguing plot, Shailene Woodley's performance, Surprising action sequences, Believable Tris and Four relationship, Unexpected ending, Impressive special effects, Good pacing

Dislike:
Lack of uniqueness, Dull and uninteresting, Repetitive shaky cam action scenes, Overemphasis on "special ones" trope

Figure 5: The examples of item-side expansion in Section 3.1

`gpt-4o-mini-2024-07-18` for `GPT-4o-mini` and `gpt-4o-2024-08-06` for `GPT-4o`, with the latter being a larger model than `GPT-4o-mini`. The experimental results demonstrate that CORAL outperforms existing large language models on both PEARL and INSPIRED, and in the case of REDIAL, it significantly surpasses the LLMs in R@50 and N@50 metrics.

B Contrasting Preference Expansion

B.1 User-side Expansion

The prompt used for *user-side expansion* is shown in Table 9. Figure 6 shows an example of the final results. Our augmentation method can enhance user preferences through reasoning power, even in cases where user preferences are scarcely revealed in the conversation. Experimentally, using the augmented potential preferences shows better performance than not using them, as demonstrated in Table 4. The datasets will be available upon acceptance.

B.2 Item-side Expansion

For item-side expansion, as mentioned in Section 3.1, it is necessary to extract review summaries from crawled reviews and convert them into keyphrases. The prompt used in this process focuses on summarizing the common like/dislike preference information that users commonly associate with the item. The prompt used for *item-side expansion* is shown in Table 10. These summaries are then transformed into keyphrases. Figure 5 provides several examples of the results.

C Implementation Details

Traditional CRS models. Unlike INSPIRED and REDIAL, PEARL does not provide the knowledge graph traditional CRS requires. Therefore, we used DBpedia Spotlight (Mendes et al., 2011) to extract entities from the dialogue, and movie entities were constructed using the movie entities from INSPIRED and REDIAL. While RevCore (Lu et al., 2021) utilizes review data crawled from IMDb, we used our own crawled review data to ensure a fair comparison in our experiments. ECR (Zhang et al., 2024) requires emotion labels. For REDIAL, we used the provided labels, whereas for PEARL and INSPIRED, which lack emotion labels, we inferred them using an LLM. Table 11 details the prompt used for this inference. Additionally, since PEARL and INSPIRED do not incorporate user feedback, we applied uniform weights to all items for feedback-aware item reweighting in ECR.

g **LLM-based CRS models.** We utilize ChatGPT[2] (`gpt-4o-mini`) as the backbone of LLM-based CRS. It generates item titles based on user dialogue without any task-specific fine-tuning. Tables 12 and 13 present the prompts used for the LLM-based approach. We computed the average performance across two types of prompts for evaluation, as used in (He et al., 2023; Li et al., 2024).

Retrieval-based CRS models. We implemented BM25 (Robertson and Walker, 1994) using Pyserini (Lin et al., 2021), and DPR (Karpukhin et al., 2020) was implemented with the BERT-base model (Devlin et al., 2019).

[2]https://openai.com/

Stage	Prompts
Superficial preference extraction	Given a dialogue history between the User and the System, find all aspects related to the movies the user currently seeks. Also, you must classify the preferences about each aspect, Like and Dislike. If there is nothing to mention about likes or dislikes, simply write "None." under the corresponding tag. Dialogue history: {***dialogHistory***} Response: [Like] {{keyphrases or descriptions separated by comma}} [Dislike] {{keyphrases or descriptions separated by comma}}
Potential preference augmentation	You are an advanced user's profile generator. Based on the conversation and the user's like/dislike preferences, use your reasoning to infer and expand upon the user's potential preferences. Augment key phrases related to the user's likes and dislikes, including preferences they may not have explicitly stated, to better guide personalized suggestions. If no explicit user preferences are provided, infer them from the conversation. Do not include any unrelated information; only state the user's preferences. User's preferences: {***extractedPreferences***} Conversation: {***dialogHistory***} Now, let's get started! [Like] {{Expanded keyphrases describing the user's likes}} [Dislike] {{Expanded keyphrases describing the user's dislikes}}

Table 9: Prompts for contrasting preference augmentation. Both ***dialogHistory*** and ***extractedPreferences*** are placeholders.

Stage	Prompts
Review summarization	Given some popular reviews about Kids {***title***}, describe what people liked and disliked about the movie under [Like] and [Dislike], respectively. If there is nothing to mention about like/dislike, simply write "None." under the corresponding tag. Here are some basic information about the movie and reviews about it: Title: {***title***} Genre: {***genres***} Cast: {***cast***} Director: {***director***} Reviews: {***reviews***}
Keyphrases generation	Below are the common [Like] and [Dislike] from users about the {***title***}. Based on this information, generate 5-8 keyphrases that represent user preferences and intentions for this movie, separated by commas. Do not include any other explanations or sentences. Here is some basic information about the movie and users' preferences information: {***userInformation***} The output format must strictly adhere to the following: [Like] keyphrases or descriptions separated by comma [Dislike] keyphrases or descriptions separated by comma

Table 10: Prompts for review summarization and keyphrases generation. Both ***title*** and ***userInformation*** are placeholders.

Prompt
You are an expert in emotion analysis. Given a target user's dialogue utterance and the dialogue history of the target user's dialogue utterance, identify the emotions expressed in the target user's dialogue utterance from the provided options. The options are as follows: a. like b. curious c. happy d. grateful e. negative f. neutral g. nostalgia h. agreement i. surprise. Output only the corresponding letter, and nothing else. Note that you only need to analyze the emotions in the target user's dialogue utterance, not the dialogue history. Dialogue history: {***dialogHistory***} Target user dialogue utterance: {***utternace***}

Table 11: Prompts for emotion classifier used to reproduce ECR. Both ***dialogHistory*** and ***utternace*** are placeholders.

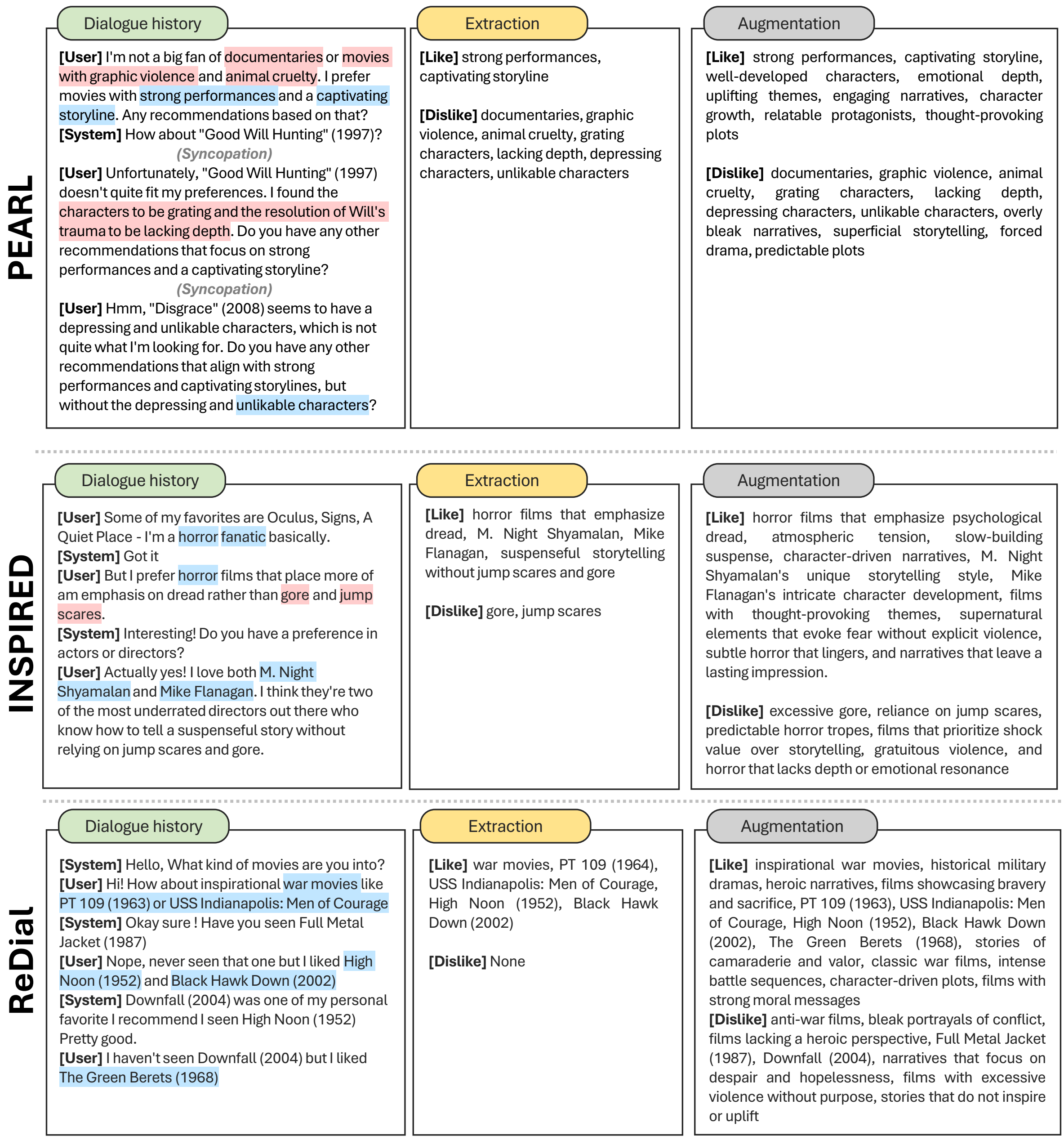


Figure 6: Step-by-step results applying the proposed contrasting preference expansion method to the PEARL, INSPIRED, and REDIAL datasets. "Extraction" corresponds to the superficial preference extraction phase, and "Augmentation" refers to the potential preference augmentation phase.

Reference	Prompts
Zero-shot (He et al., 2023)	Pretend you are a movie recommender system. I will give you a conversation between a user and you (a recommender system). Based on the conversation, you reply me with 50 recommendations without extra sentences. The format of the recommendation list is: no. title (year). Here is the conversation: {***dialogHistory***}
ChatCRS (Li et al., 2024)	You are an excellent conversational recommender that helps the user achieve recommendation-related goals through conversations. Given the dialogue history, your task is to generate appropriate item recommendations for the dialogue. You reply me with 50 recommendations without extra sentences. The format of the recommendation list is: no. title (year). Dialogue history: {***dialogHistory***}

Table 12: Prompts used for LLM zero-shot recommendation. Adapted from the prompts used for movie recommendation in (He et al., 2023; Li et al., 2024). We reported the average performance of these two prompts. Both ***dialogHistory*** and ***extractedPreferences*** are placeholders.

Reference	Prompts
Zero-shot (He et al., 2023)	Pretend you are a movie recommender system. I will give you a conversation between a user and you (a recommender system), along with the related knowledge triplets. Based on this information, you reply to me with 50 recommendations without extra sentences. The format of the recommendation list is: no. title (year). Here is the conversation:{***dialogHistory***} Here is the related knowledge triplets: [{***knowledgeTriplets***}]
ChatCRS (Li et al., 2024)	You are an excellent conversational recommender that helps the user achieve recommendation-related goals through conversations. Given the dialogue history and a knowledge triplets, your task is to genenrate appropriate item recommendations for the dialogue. You reply me with 50 recommendations without extra sentences. The format of the recommendation list is: no. title (year). Dialogue history: {***dialogHistory***} Knowledge triplets: [{***knowledgeTriplets***}]

Table 13: Prompts used for ChatCRS (Li et al., 2024) recommendation. Adapted from the prompts used for movie recommendation in (He et al., 2023; Li et al., 2024). We reported the average performance of these two prompts. Both ***dialogHistory*** and ***extractedPreferences*** are placeholders.