Bayesian Optimization with LLM-Based Acquisition Functions for Natural Language Preference Elicitation

David Eric Austin*
deaustin@uwaterloo.ca
University of Waterloo
Waterloo, Ontario, Canada

Anton Korikov*
Armin Toroghi
Scott Sanner
anton.korikov@mail.utoronto.ca
University of Toronto
Toronto, Ontario, Canada

ABSTRACT

Designing preference elicitation (PE) methodologies that can quickly ascertain a user's top item preferences in a cold-start setting is a key challenge for building effective and personalized conversational recommendation (ConvRec) systems. While large language models (LLMs) enable fully natural language (NL) PE dialogues, we hypothesize that monolithic LLM NL-PE approaches lack the multi-turn, decision-theoretic reasoning required to effectively balance the exploration and exploitation of user preferences towards an arbitrary item set. In contrast, traditional Bayesian optimization PE methods define theoretically optimal PE strategies, but cannot generate arbitrary NL queries or reason over content in NL item descriptions – requiring users to express preferences via ratings or comparisons of unfamiliar items. To overcome the limitations of both approaches, we formulate NL-PE in a Bayesian Optimization (BO) framework that seeks to actively elicit NL feedback to identify the best recommendation. Key challenges in generalizing BO to deal with natural language feedback include determining: (a) how to leverage LLMs to model the likelihood of NL preference feedback as a function of item utilities, and (b) how to design an acquisition function for NL BO that can elicit preferences in the infinite space of language. We demonstrate our framework in a novel NL-PE algorithm, PEBOL, which uses: 1) Natural Language Inference (NLI) between user preference utterances and NL item descriptions to maintain Bayesian preference beliefs, and 2) BO strategies such as Thompson Sampling (TS) and Upper Confidence Bound (UCB) to guide LLM query generation. We numerically evaluate our methods in controlled simulations, finding that after 10 turns of dialogue, PEBOL can achieve an MRR@10 of up to 0.27 compared to the best monolithic LLM baseline's MRR@10 of 0.17, despite relying on earlier and smaller LLMs.[1]

*Both authors contributed equally to this research.
[1]Our code is publically available at https://github.com/D3Mlab/llm-pe.

RecSys '24, October 14–18, 2024, Bari, Italy

ACM ISBN 979-8-4007-0505-2/24/1
https://doi.org/10.1145/3640457.3688142

CCS CONCEPTS

• **Information systems** → **Recommender systems**; **Personalization**; *Language models*.

KEYWORDS

Conversational Recommendation, Preference Elicitation, Bayesian Optimization, Online Recommendation, Query Generation

ACM Reference Format:
David Eric Austin, Anton Korikov, Armin Toroghi, and Scott Sanner. 2024. Bayesian Optimization with LLM-Based Acquisition Functions for Natural Language Preference Elicitation. In *18th ACM Conference on Recommender Systems (RecSys '24), October 14–18, 2024, Bari, Italy*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3640457.3688142

1 INTRODUCTION

Personalized conversational recommendation (ConvRec) systems require effective natural language (NL) preference elicitation (PE) strategies that can efficiently learn a user's top item preferences in cold start settings, ideally requiring only an arbitrary set of NL item descriptions. While the advent of large language models (LLMs) has introduced the technology to facilitate NL-PE conversations [14, 23] we conjecture that monolithic LLMs have limited abilities to strategically conduct active, multi-turn NL-PE dialogues about a set of arbitrary items. Specifically, we hypothesize that LLMs lack the multi-turn decision-theoretic reasoning to interactively generate queries that avoid over-exploitation or over-exploration of user-item preferences, thus risking over-focusing on already revealed item preferences or wastefully exploring preferences over low-value items. Further challenges faced by monolithic LLM NL-PE approaches include the need to jointly reason over large, potentially unseen sets of item descriptions, and the lack of control and interpretability in system behaviour even after prompt engineering or fine-tuning [28].

In contrast, conventional PE algorithms [21, 22, 27, 39, 40], including Bayesian optimization methods [2, 5, 12, 32, 36], establish formal decision-theoretic policies such as Thompson Sampling (TS) and Upper Confidence Bound (UCB) [16] to balance exploration and exploitation with the goal of quickly identifying the user's most preferred items. However, these techniques typically assume a user can express preferences via direct item ratings or comparisons – an unrealistic expectation when users are unfamiliar with most items [1]. While recent work has extended Bayesian PE to a fixed set of template-based queries over pre-defined keyphrases [36], no

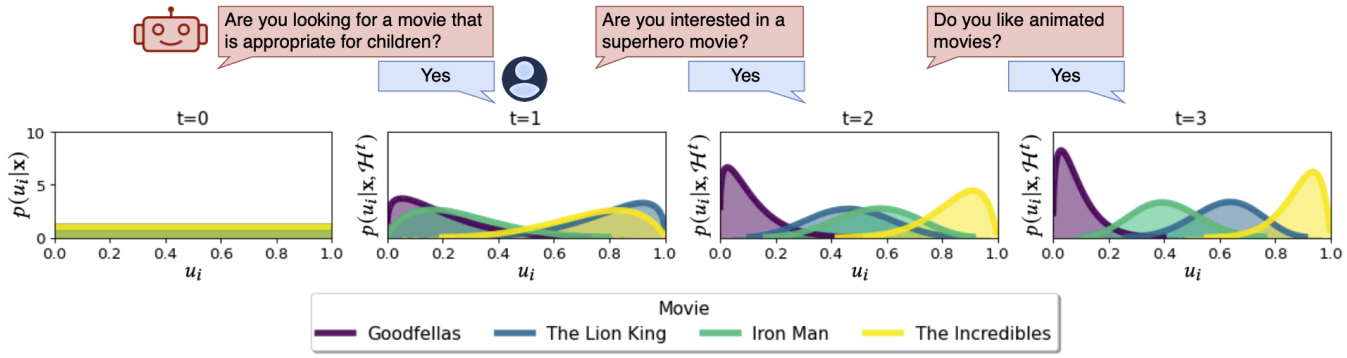

Figure 1: PEBOL's belief updates over a cold-start user's item utilities during three turns of NL dialogue. Bayesian preference beliefs not only facilitate recommendation, but also enable Bayesian optimization policies to guide LLM query generation, avoiding over-exploration (asking about clearly low-value items) and over-exploitation (over-focusing on known preferences).

existing work extends Bayesian methodologies to generative NL-PE over a set of generic NL item descriptions.

In this paper, we make the following contributions:

- We introduce the first Bayesian optimization formalization of NL-PE for arbitrary NL dialogue over a generic set of NL item descriptions – establishing a new framework for research on steering LLMs with decision-theoretic reasoning.
- We present PEBOL (**P**reference **E**licitation with **B**ayesian **O**ptimization augmented **LL**Ms), a novel NL-PE algorithm which 1) infers item preferences via Natural Language Inference (NLI) [37] between dialogue utterances and item descriptions to maintain Bayesian preference beliefs and 2) introduces LLM-based acquisition functions, where NL query generation is guided by decision-theoretic strategies such as TS and UCB over the preference beliefs.
- We numerically evaluate PEBOL against monolithic GPT-3.5 and Gemini-Pro NL-PE methods via controlled NL-PE dialogue experiments over multiple NL item datasets and levels of user noise.
- We observe that after 10 turns of dialogue, PEBOL can achieve a mean MRR@10 of up to 0.27 compared to the best monolithic LLM baseline's MRR@10 of 0.17, despite relying on earlier and smaller LLMs.

2 BACKGROUND AND RELATED WORK

2.1 Bayesian Optimization

Given an objective function $f : \mathcal{X} \to \mathbb{R}$, (standard) optimization systematically searches for a point $x^* \in \mathcal{X}$ that maximizes[2] f. Bayesian optimization focuses on settings where f is a black-box function which does not provide gradient information and cannot be evaluated exactly – rather, f must be evaluated using indirect or noisy observations which are expensive to obtain [10, 30]. To address these challenges, Bayesian optimization maintains probabilistic beliefs over $f(x)$ and its observations to guide an uncertainty-aware optimization policy which decides where to next observe $f(x)$.

Bayesian optimization begins with a *prior* $p(f)$ which represents the beliefs about f before any observations are made. Letting y_i represent a noisy or indirect observation of $f(x_i)$, and collecting a sequence of observations into a dataset $\mathcal{D} = (\mathbf{x}, \mathbf{y})$, an *observation model* defines the *likelihood* $p(\mathcal{D}|f)$. We then use the observed data and Bayes theorem to update our beliefs and obtain the *posterior*

$$p(f|\mathcal{D}) = \frac{p(f)p(\mathcal{D}|f)}{p(\mathcal{D})}. \tag{1}$$

This posterior informs an *acquisition function* $\gamma(x|\mathcal{D})$ which determines where to next observe $f(x)$ in a way that balances exploitation (focusing observations where f is likely near its maximum) with exploration (probing areas where f has high uncertainty).

2.2 Preference Elicitation

PE has witnessed decades of research, and includes approaches based on Bayesian optimization (e.g., [3, 8, 11, 13, 18]), Bandits (e.g., [5, 24, 25, 40]), constrained optimization [29], and POMDPs [2]. In the standard PE setting, a user is assumed to have some hidden utilities $\mathbf{u} = [u_1, ..., u_N]$ over a set $\mathcal{I}$ of N items, where item i is preferred to item j if $u_i > u_j$. The goal of PE is typically to search for an item $i^* \in \arg\max_i u_i$ that maximizes user utility in a minimal number of PE queries, which most often ask a user to express item preferences as item ratings (e.g., [3, 5, 24, 25, 40]) or relative preferences between item pairs or sets (e.g., [2, 8, 11, 12, 14, 32]). An alternative form of PE asks users to express preferences over predefined item features, also through rating- or comparison-based queries [22, 27, 39]. Central to the above PE methods are query selection strategies that balance the exploration and exploitation of user preferences, with TS and UCB algorithms (cf. Sec. 4.2) often exhibiting strong performance [5, 27, 36, 39, 40]. However, none of these methods are able to interact with users through NL dialogue or reason about NL item descriptions.

2.3 Language-Based Preference Elicitation

Yang *et al.* [36] introduce Bayesian PE strategies using TS and UCB for keyphrase rating queries, where keyphrases are first mined from NL item reviews and then co-embedded with user-item preferences in a recommendation system. Handa *et al.* [14] propose using LLMs to interface with a conventional Bayesian PE system, suggesting a preprocessing step to extract features from NL descriptions and a verbalization step to fluidly express pairwise item comparison

[2]We take the maximization direction since this paper searches for items with maximum utility for a person.

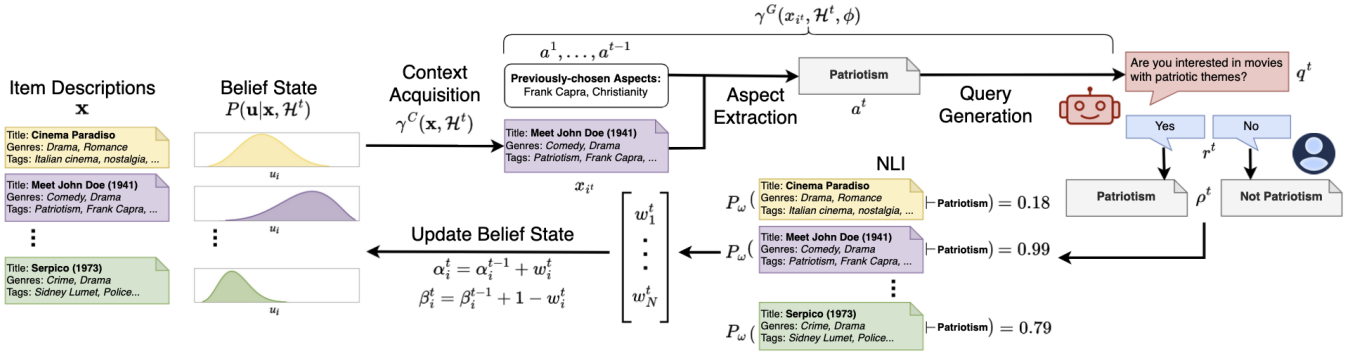

Figure 2: The PEBOL NL-PE algorithm, which maintains a Bayesian belief state over a user's item preferences given an arbitrary set of NL item descriptions x. This belief is used by a decision-theoretic policy to balance the exploration and exploitation of preferences by strategically selecting an item description x_{i^t} as the basis for LLM query generation. Belief updates are computed through Bayesian inference with NLI entailment scores between item descriptions and query-response pairs.

queries. Li *et al.* [23] prompt an LLM to generate PE queries for some specific domain (e.g., news content, morals), observe user responses, and evaluate LLM relevance predictions for a single item. While these works make progress towards NL-PE, they do not study how LLM query generation can strategically explore user preferences towards an arbitrary item set outside the realm of item-based or category-based feedback.

2.4 Conversational Recommendation

Recent work on ConvRec uses language models[3] to facilitate NL dialogue while integrating calls to a recommender module which generates item recommendations based on user-item interaction history [4, 26, 33, 35]. He *et al.* [15] report that on common datasets, zero-shot GPT-3.5/4 outperforms these ConvRec methods, which generally use older language models and require user-item interaction history for their recommendation modules.

2.5 Natural Language Inference

Binary Natural Language Inference (NLI) [37] models predict the likelihood that one span of text called a premise is *entailed by* (i.e., can be inferred from) a second span called the hypothesis. For example, an effective NLI model should predict a high likelihood that the premise *"I want to watch Iron Man"* entails the hypothesis *"I want to watch a superhero movie"*. As illustrated by this example, the hypothesis typically must be more general than the premise. NLI models are trained by fine-tuning encoder-only LLMs on NLI datasets [6, 31, 34], which typically consist of short text spans for the premise and hypothesis – thus enabling relatively efficient performance on similar tasks with a fairly small number LLM parameters.

3 PROBLEM DEFINITION

We now present a Bayesian optimization formulation of NL-PE. The goal of NL-PE is to facilitate a NL dialogue which efficiently discovers a user's most preferred items out of a set of N items. Each item $i \in \mathcal{I}$ has a NL description x_i, which might be a title, long-form description, or even a sequence of reviews, with the item set $\mathcal{I}$ collectively represented by $\mathbf{x} \in \mathcal{X}$ with $\mathbf{x} = [x_1, ..., x_N]$. We assume the user has some (unknown) utility function $f : \mathcal{X} \to \mathbb{R}$ establishing hidden utilities $\mathbf{u} = f(\mathbf{x})$ so that item i is preferred to item j if $u_i > u_j$. Our goal is to find the most preferred item(s):

$$i^* \in \arg\max_{i \in \mathcal{I}} u_i. \tag{2}$$

In contrast to standard Bayesian PE formalisms (c.f. Sec 2.2), we do not assume that the user can effectively convey direct *item-level* preferences by either: 1) providing item ratings (i.e., utilities) or 2) pairwise or listwise item comparisons. Instead, we must infer user preferences by observing utterances during a NL system-user dialogue. At turn t of a dialogue, we let q^t and r^t be the system and user utterance, respectively, with $\mathbf{q}^t = [q^1, ..., q^t]$ and $\mathbf{r}^t = [r^1, ..., r^t]$ representing all system and user utterances up to t. In this paper, we call q^t the *query* and r^t the *response*, though extensions to more generic dialogues (e.g., when users can also ask queries) are discussed in Section 7. We let $\mathcal{H}^t = (\mathbf{q}^t, \mathbf{r}^t)$ be the conversation history at turn t.

To formulate NL-PE as a Bayesian optimization problem, we place a prior belief on the user's utilities $p(\mathbf{u}|\mathbf{x})$, potentially conditioned on item descriptions since they are available before the dialogue begins. We then assume an observation model that gives the likelihood $p(\mathbf{r}^t|\mathbf{x}, \mathbf{u}, \mathbf{q}^t)$, letting us define the posterior utility belief as

$$p(\mathbf{u}|\mathbf{x}, \mathcal{H}^t) \propto p(\mathbf{r}^t|\mathbf{x}, \mathbf{u}, \mathbf{q}^t) p(\mathbf{u}|\mathbf{x}). \tag{3}$$

This posterior informs an acquisition function $\gamma(\mathbf{x}, \mathcal{H}^t)$ which generates[4] a new NL query

$$q^{t+1} = \gamma(\mathbf{x}, \mathcal{H}^t), \tag{4}$$

to systematically search for i^*. The preference beliefs also let us define an Expected Utility (EU) μ_i^t for every item as

$$\mu_i^t = \mathbb{E}_{p(\mathbf{u}|\mathbf{x}, \mathcal{H}^t)}[u_i], \tag{5}$$

which allows the top-k items to be recommended at any turn based on their expected utilities.

Our Bayesian optimization NL-PE paradigm lets us formalize several key questions, including:

[3] Earlier systems (e.g. [4, 26]) use relatively small RNN-based language models.

[4] To represent the *generative acquisition* of NL outputs, we deviate from the conventional definition of acquisition functions as mapping to $\mathbb{R}$.

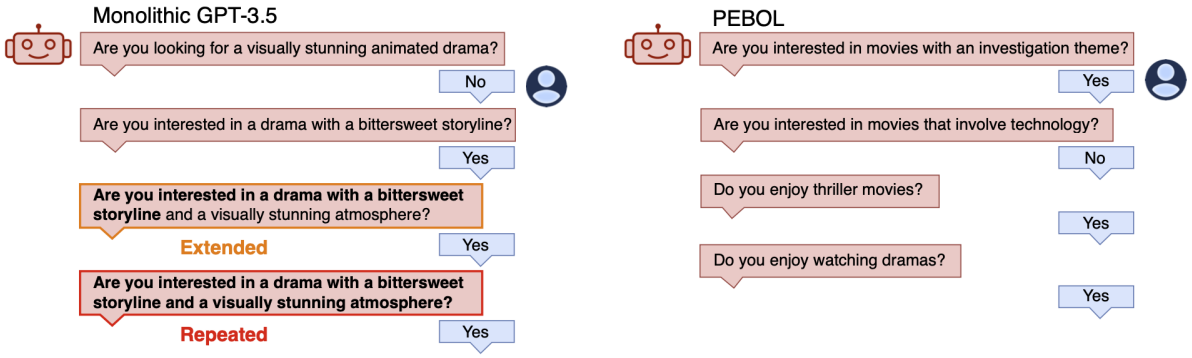

Figure 3: Cherry-picked system-generated dialogues from our NL-PE experiments. The Monolithic GPT-3.5 dialogue (left) demonstrates over-exploitation, with q^3 directly extending q^2 after a positive user preference is observed and leading to the extreme case of query repetition ($q^4 = q^3$). In contrast, PEBOL (right) continues exploring even after a positive response, while focusing on promising aspects (three out of four queries elicit a positive response) by using UCB-guided query generation.

(1) How do we represent beliefs $p(\mathbf{u}|\mathbf{x}, \mathcal{H}^t)$ in user-item utilities $\mathbf{u}$, given NL item descriptions $\mathbf{x}$ and a dialogue $\mathcal{H}^t$?
(2) What are effective models for the likelihood $p(\mathbf{r}^t|\mathbf{x}, \mathbf{u}, \mathbf{q}^t)$ of observed responses $\mathbf{r}^t$ given $\mathbf{x}$, $\mathbf{q}^t$, and user utilities $\mathbf{u}$?
(3) How can our beliefs inform the generative acquisition of NL queries q^{t+1} given $\mathcal{H}^t$ to strategically search for i^*?

These questions reveal a number of novel research directions discussed further in Section 7. In this paper, we present PEBOL, a NL-PE algorithm based on the above Bayesian optimization NL-PE formalism, and numerically evaluate it against monolithic LLM alternatives through controlled, simulated NL dialogues (cf. Sec. 6).

4 METHODOLOGY

Limitations of Monolithic LLM Prompting. An obvious NL-PE approach, described further as baseline in Section 5.1, is to prompt a monolithic LLM with all item descriptions $\mathbf{x}$, dialogue history $\mathcal{H}^t$, and instructions to generate a new query at each turn. However, providing all item descriptions $[x_1, ..., x_N]$ in the LLM context window is very computationally expensive for all but the smallest item sets. While item knowledge could be internalized through fine-tuning, each item update would imply system retraining. Critically, an LLM's preference elicitation behaviour cannot be controlled other than by prompt-engineering or further fine-tuning, with neither option offering any guarantees of predictable or interpretable behaviour that balances the exploitation and exploration of user preferences.

PEBOL Overview. We propose to addresses these limitations by augmenting LLM reasoning with a Bayesian Optimization procedure in a novel algorithm, PEBOL, illustrated in Figure 2. At each turn t, our algorithm maintains a probabilistic belief state over user preferences as a Beta belief state (cf. Sec. 4.1). This belief state guides an LLM-based acquisition function to generate NL queries explicitly balancing exploration and exploitation to uncover the top user preferences (cf. Sec. 4.2). In addition, our acquisition function reduces the context needed to prompt the LLM in each turn from all N item descriptions $\mathbf{x}$ to a single strategically selected item description x_{i^t}. PEBOL then uses NLI over elicited NL preferences and item descriptions to map dialogue utterances to numerical observations (c.f. Sec 4.3).

4.1 Utility Beliefs

4.1.1 Prior Beliefs. Before any dialogue, PEBOL establishes an uninformed prior belief $p(\mathbf{u})$ on user-item utilities. We assume item utilities are independent so that

$$p(\mathbf{u}) = \prod_{i=1}^{N} p(u_i), \tag{6}$$

and that the prior for each utility u_i is a Beta distribution

$$p(u_i) = \text{Beta}(\alpha_i^0, \beta_i^0). \tag{7}$$

Since this paper focuses on fully cold start settings, we assume a uniform Beta prior with $(\alpha_i^0, \beta_i^0) = (1, 1)$. Beta distributions, illustrated in Figure 1, lie in the domain $[0, 1]$ – a normalized interval for bounded ratings in classical recommendation systems. We can thus interpret utility values of $u_i = 1$ or $u_i = 0$ to represent a complete like or dislike of item i, respectively, while values $u_i \in (0, 1)$ provide a strength of preference between these two extremes.

4.1.2 Observation Model. To perform a posterior update on our utility beliefs given observed responses $\mathbf{r}^t$, we need an observation model that represents the likelihood $p(\mathbf{r}^t|\mathbf{x}, \mathbf{u}, \mathbf{q}^t)$. Modelling the likelihood of $\mathbf{r}^t$ is a challenging task, so we will require some simplifying assumptions. Firstly, we assume that the likelihood of a single response r^t is independent from any previous dialogue history $\mathcal{H}^{t-1}$, so that:

$$p(\mathbf{r}^t|\mathbf{x}, \mathbf{u}, \mathbf{q}^t) = \prod_{t'=1}^{t} p(r^{t'}|\mathbf{x}, \mathbf{u}, q^{t'}). \tag{8}$$

Note that this independence assumption will allow incremental posterior belief updates, so that

$$p(\mathbf{u}|\mathbf{x}, \mathcal{H}^t) \propto p(r^t|\mathbf{x}, \mathbf{u}, q^t) p(\mathbf{u}|\mathbf{x}, \mathcal{H}^{t-1}). \tag{9}$$

4.1.3 Binary Item Response Likelihoods and Posterior Update. With the factorized distributions over item utilities and observational

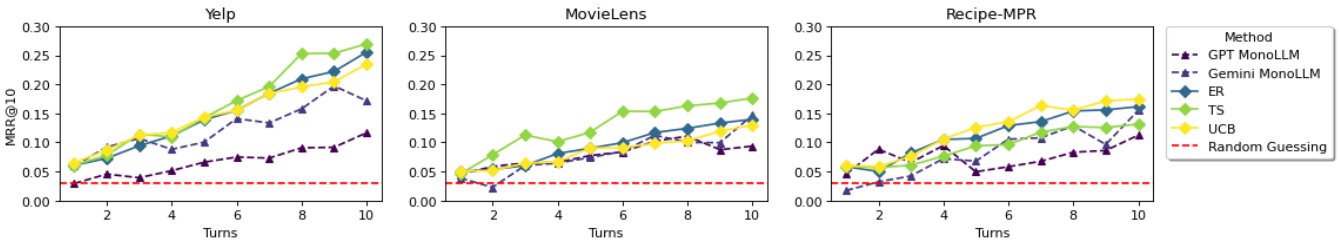

Figure 4: MRR@10 for MonoLLM and PEBOL-P with uncertainty-informed policies (UCB, TS, ER). All methods show preference learning over time and MonoLLM is generally outperformed by PEBOL.

likelihood history now defined, we simply have to provide a concrete observational model of the response likelihood conditioned on the query, item descriptions, and latent utility: $p(r^t|\mathbf{x}, \mathbf{u}, q^t)$.

Because the prior is factorized over conditionally independent u_i (cf. (6)), we can likewise introduce individual per-item factorized binary responses $r_i^t \in \{0(dislike), 1(like)\}$ to represent the individual relevance of each item i to the preference elicited at turn t. Critically, we won't actually require an individual response per item — this will be computed by a natural language inference (NLI) model [6] to be discussed shortly — but we'll begin with an individual binary response model for r_i^t for simplicity:

$$p(r_i^t|x_i, u_i, q^t) = \text{Bernoulli}(u_i). \tag{10}$$

With our response likelihood defined, this now leads us to our first pass at a full posterior utility update that we term PEBOL-B for observed *Binary* rating feedback. Specifically, given observed binary ratings r_i^t, the update at $t = 1$ uses the Beta prior (7) with the Bernoulli likelihood (10) to form a standard Beta-Bernoulli conjugate pair and compute the posterior utility belief

$$p(u_i|x_i, \mathcal{H}^1) \propto p(u_i|x_i)p(r_i^1|x_i, u_i, q^t) \tag{11}$$

$$= \text{Beta}(\alpha_i^1, \beta_i^1), \tag{12}$$

where $\alpha_i^1 = \alpha_i^0 + r_i^1$, $\beta_i^1 = \beta_i^0 + (1 - r_i^1)$. Subsequent incremental updates updates follow Eq. (9) and use the same conjugacy to give

$$p(u_i|x_i, \mathcal{H}^t) = \text{Beta}(\alpha_i^t, \beta_i^t), \tag{13}$$

where $\alpha_i^t = \alpha_i^{t-1} + r_i^t$, $\beta_i^t = \beta_i^{t-1} + (1 - r_i^t)$.

4.1.4 Natural Language Inference and Probabilistic Posterior Update. As hinted above, effective inference becomes slightly more nuanced since we don't need to observe an explicit binary response *per item* in our PEBOL framework. Rather, we receive general preference feedback r^t on whether a user generically prefers a text description q^t and then leverage an NLI model [6] to *infer* whether the description x_i of item i would be preferred according to this feedback. For instance, for a (q^t, r^t) pair (*"Want to watch a children's movie?","Yes"*), NLI should infer a rating of $r_1^t = 1$ for x_1 = *"The Lion King"* and $r_2^t = 0$ for x_2 = *"Titanic"*.

To deal with the fact that NLI models actually return an *entailment probability*, our *probabilistic* observation variant, PEBOL-P leverages the probability that item description x_i entails q_t, which we denote as w_i^t. We provide a full graphical model and derivation of the Bayesian posterior update given this entailment probability in the Supplementary Material, but note that we can summarize the final result as a *relaxed* version of the binary posterior update of (13) that replaces the binary observation $r_i \in \{0, 1\}$ with the entailment probability $w_i^t \in [0, 1]$, i.e., $\alpha_i^t = \alpha_i^{t-1} + w_i^t$, $\beta_i^t = \beta_i^{t-1} + (1 - w_i^t)$.

To visually illustrate how this posterior inference process works in practice, Figure 1 shows the effect of PEBOL's posterior utility belief updates based on NLI for three query-response pairs – we can see the system gaining statistical knowledge about useful items for the user from the dialogue.

4.2 LLM-Based Acquisition Functions

Recall from Sec. 2.1 that in Bayesian optimization, the posterior informs an acquisition function which determines where to make the next observation. PEBOL generates a new query q^t with a two-step acquisition function γ, first using Bayesian Optimization policies (step 1) based on the posterior utility beliefs $p(\mathbf{u}|\mathbf{x}, \mathcal{H}^t)$ to select NL context, and then using this selected context to guide LLM prompting (step 2). We express the overall acquisition function $\gamma = \gamma^G \circ \gamma^C$ as a composition of a *context* acquisition function γ^C (cf. Sec. 4.2.1) and a NL *generation* function γ^G (cf. Sec. 4.2.2).

4.2.1 Context Acquisition via Bayesian Optimization Policies. First, PEBOL harnesses Bayesian optimization policies to select an item description x_{i^t} which will be used to prompt an LLM to generate a query about an aspect described by x_{i^t} (cf. Sec. 4.2.2). Selecting an item i^t whose utility u_{i^t} is expected to be near the maximum, u_{i^*}, will generate *exploitation queries* asking about properties of items that are likely to be preferred by the user. In contrast, selecting an item i^t associated with high uncertainty in its utility u_i^t will generate *exploration queries* that probe into properties of items for which user preferences are less known. Thus, strategically selecting x_{i^t} allows PEBOL to balance the exploration and exploitation behaviour of NL queries, decreasing the risks of becoming stuck in local optima (over-exploitation) or wasting resources exploring low utility item preferences (over-exploration). We define the item selected by the context acquisition function as

$$i^t = \gamma^C(\mathbf{x}, \mathcal{H}^t), \tag{14}$$

and list several alternatives for γ^C, including the well-known strategies of TS and UCB [30]:

(1) **Thompson Sampling (TS)**: First, a sample of each item's utility $\hat{u}_i^t$ is taken from the posterior, $\hat{u}_i^t \sim p(u_i|x_i, \mathcal{H}^t)$. Then, the item with the highest sampled utility is selected:

$$i^t = \arg\max_i \hat{u}_i^t. \tag{15}$$

TS explores more when beliefs have higher uncertainty and exploits more as the system becomes more confident.

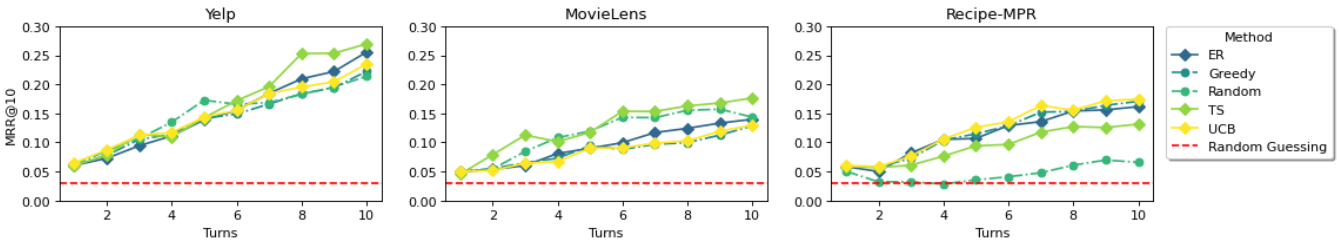

Figure 5: MRR@10 for PEBOL-P with various context acquisition policies.

(2) **Upper Confidence Bound (UCB)**: Let $P_k(\alpha, \beta)$ represent the k'th percentile of Beta(α, β), which provides a confidence bound on the posterior. UCB selects the item with the highest confidence bound

$$i^t = \arg\max_i P_k(p(u_i|x_i, \mathcal{H}^t)), \tag{16}$$

following a balanced strategy because confidence bounds are increased by both high utility and high uncertainty.

(3) **Entropy Reduction (ER)**: An explore-only strategy that selects the item with the most uncertain utility:

$$i^t = \arg\max_i \text{Var}(p(u_i|x_i, \mathcal{H}^t)). \tag{17}$$

(4) **Greedy**: An exploit-only strategy that selects the item with the highest expected utility μ_i^t (Eq. 5):

$$i^t = \arg\max_i \mu_i^t. \tag{18}$$

(5) **Random**: An explore-only heuristic that selects the next item randomly.

4.2.2 Generating Short, Aspect-Based NL Queries. Next, PEBOL prompts an LLM to generate a NL query q^t based on the selected item description x_{i^t} while also using the dialogue history $\mathcal{H}^t$ to avoid repetitive queries. We choose to generate "yes-or-no" queries asking if a user prefers items with some aspect a^t, which is a short text span extracted dynamically from x_{i^t} to be different from any previously queried aspects $a^1, ..., a^{t-1}$. We adopt this query generation strategy to: 1) reduce cognitive load on the user, who may be frustrated by long and specific queries about unfamiliar items and 2) better facilitate NLI through brief, general phrases [37]. Letting ϕ represent the query generation prompt, we let

$$q^t, a^t = \gamma^G(x_{i^t}, \mathcal{H}^t, \phi) \tag{19}$$

be the LLM generated query and aspect at turn t, with prompting details discussed in Section 5.2.3. An example of such a query and aspect (bold) is *"Are you interested in movies with **patriotic** themes?"*, generated by PEBOL in our movie recommendation experiments and shown in Figure 2.

4.3 NL Item-Preference Entailment

4.3.1 Preference Descriptions from Query Response Pairs. Next, PEBOL receives a NL user response r^t, which it must convert to individual item preference observations. Since the LLM is instructed to generate "yes-or-no" queries q^t asking a user if they like aspect a^t, we assume the user response will be a *"yes"* or a *"no"*, and create a NL description of the users preference ρ^t, letting $\rho^t = a^t$ if $r^t =$*"yes"*, and $\rho^t = \text{concat}($*"not "*$, a^t)$ if $r^t =$ *"no"*. For example, given a query that asks if the user prefers the aspect "patriotism" in an item, if the user response is *"yes"*, then the user preference ρ^t is "patriotism", and "not patriotism" otherwise. This approach produces short, general preference descriptions that are well suited for NLI models [37].

4.3.2 Inferring Item Ratings from NL Preferences. Given a NL preference ρ^t, PEBOL must infer whether the user would like an item described by x_i. Specifically, PEBOL acquires ratings $\mathbf{w}^t = [w_1^t, ..., w_N^t]$ (cf. Sec. 4.1.4) by using NLI to predict whether an item description x_i entails (i.e., implies) the preference ρ^t. For example, we expect that an NLI model would predict that $x_i =$"*The Lion King*" entails $\rho^t =$***"animated"*** while $x_j =$*"Titanic"* does not, inferring that a user who expressed preference ρ^t would like item i but not j. We use an NLI model $P_\omega(x_i, \rho^t)$ to predict the probability w_i^t that x_i entails ρ^t, and return $r_i^t = \lfloor w_i^t \rceil$ in the case of binary observations (PEBOL-B) and w_i^t in the case of probabilistic observations (PEBOL-P).

4.4 The Complete PEBOL System

This concludes the PEBOL specification – the entire process from prior utility belief to the LLM-based acquisition function generation of a query to the posterior utility update is illustrated in Figure 2.

5 EXPERIMENTAL METHODS

We numerically evaluate our PEBOL variations through controlled NL-PE dialogue experiments across multiple datasets and response noise levels – comparing against two monolithic LLM (MonoLLM) baselines. Specifically, these baselines directly use GPT-3.5-turbo-0613 (GPT MonoLLM) or Gemini-Pro (Gemini MonoLLM) as the NL-PE system, as described in Section 5.1. We do not compare against ConvRec methods [4, 26, 33, 35] because they are not cold-start systems, requiring observed user-item interactions data to drive their recommendation modules. We also do not base our experiments on ConvRec datasets such as ReDIAL [26], since they are made up of *pre-recorded* conversation histories and cannot be used to evaluate active, cold-start NL-PE systems.

5.1 MonoLLM Baseline

A major challenge of using MonoLLM for NL-PE is that item descriptions $\mathbf{x}$ either need to be internalized through training or be provided in the context window (cf. Sec. 4) – since we focus on fully cold-start settings, we test the latter approach as a baseline. In each turn, given the full conversation history $\mathcal{H}^t$ and $\mathbf{x}$, we prompt the MonoLLM to generate a new query to elicit user preferences – all prompts are shown in the Supplementary Materials. We evaluate

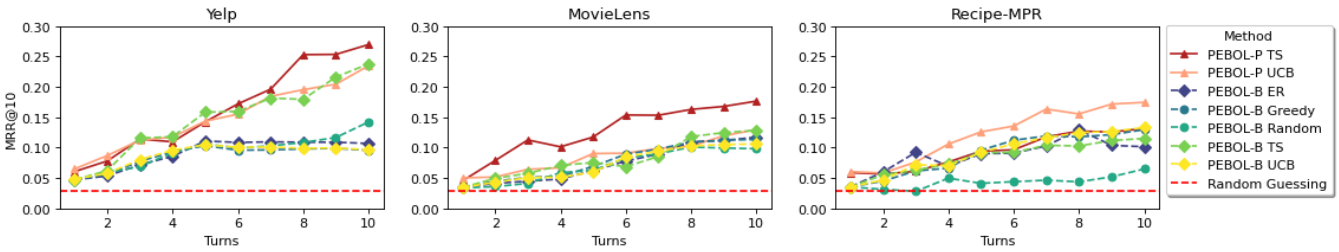

Figure 6: MRR@10 for PEBOL using binary vs. probabilistic entailment scores. PEBOL-P with the best policy (TS on Yelp and MovieLens, UCB on Recipe-MPR) generally outperforms PEBOL-B.

recommendation performance after each turn by using another prompt to recommend a list of ten item names from $\mathbf{x}$ given $\mathcal{H}^t$. Due to context window limits, this MonoLLM approach is only feasible for small item sets with short item descriptions; thus, we have to limit $|\mathcal{I}|$ to 100 for fair comparison to the MonoLLM baseline.

5.2 Simulation Details

We test PEBOL and MonoLLM through NL-PE dialogues with LLM-simulated users, where the simulated users' item preferences remain hidden from the system. We evaluate recommendation performance over 10 turns of dialogue.

5.2.1 User Simulation. For each experiment, we simulate 100 users, each of which likes a single item $i \in \mathcal{I}$. Each user is simulated by GPT-3.5-turbo-0613, which is given item description x_i and instructed to provide only "yes" or "no" responses to a query q^t as if it was a user who likes item i.

5.2.2 Evaluating Recommendations. We evaluate the top-10 recommendations in each turn using the Mean Reciprocal Rank (MRR@10) of the preferred item, which is equivalent to MAP@10 for the case of a single preferred item.

5.2.3 PEBOL Query Generation. In turn t, given an item description x_i and previously generated aspects $(a^1, ..., a^{t-1})$, an LLM (GPT-3.5-turbo-0613)[5] is prompted to generate an aspect a^t describing the item i that is no more than 3 words long. The LLM is then prompted again to generate a "yes-or-no" query asking if a user prefers a^t.

5.2.4 NLI. We use the 400M FAIR mNLI[6] model to predicts logits for *entailment*, *contradiction*, and *neutral*, and divide these logits by an MNLI temperature $T \in \{1, 10, 100\}$ As per the FAIR guidelines, we pass the temperature-scaled *entailment* and *contradiction* scores through a softmax layer and take the *entailment* probabilities. We report PEBOL results using the best MNLI temperature for the most datasets.

5.2.5 User Response Noise. We test three user response noise levels $\in \{0, 0.25, 0.5\}$ corresponding to the proportion or user responses that are randomly selected between "yes" and "no".

5.2.6 Omitting Query History Ablation. We test how tracking query history in PEBOL effects performance with an ablation study that removes previously generated aspects $(a^1, ..., a^{t-1})$ from the aspect extraction prompt.

5.3 Datasets

We obtain item descriptions from three real-world datasets: MovieLens25M[7], Yelp[8], and Recipe-MPR [38] (example item descriptions from each shown in Table 1 in the Supplementary Materials). After the filtering steps below for Yelp and MovieLens, we randomly sample 100 items to create $\mathbf{x}$. For Yelp, we filter restaurant descriptions to be from a single major North American city (Philadelphia) and to have at least 50 reviews and five or more category labels. For MovieLens,[9] we filter movies to be in the 10% by rating count with at least 20 tags, and let movie descriptions use the title, genre labels, and 20 most common user-assigned tags.

5.4 Research Questions

Our experiments explore the following research questions (RQs):

- **RQ1:** How does PEBOL perform against the MonoLLM baselines?
- **RQ2:** Does PEBOL perform better with binary or probabilistic observations, and how sensitive is the latter to temperature?
- **RQ3:** How do PEBOL and MonoLLM perform under user response noise?
- **RQ4:** How do the context selection policies of TS, UCB, ER, Greedy, and Random effect PEBOL performance?
- **RQ5:** How much does PEBOL performance depend on access to the query history during query generation?

6 EXPERIMENTAL RESULTS

6.1 RQ1 - PEBOL vs. MonoLLM

Figure 4 shows MRR@10 over 10 dialogue turns for MonoLLM and PEBOL (UCB,TS,ER),[10] with 95% confidence intervals (CIs) at turn 10 shown in Figure 8 (see Supplementary Materials for CIs for all turns and experiments). All methods start near random guessing, reflecting a cold start, and show clear preference learning over time.

[5]Experiments with newer LLMs such as Gemini or GPT4 for PEBOL query generation are left for future work due to the API time and cost requirements needed to simulate the many variants of PEBOL reported in Section 6. We do, however, compare PEBOL against a Gemini-MonoLLM baseline.
[6]https://huggingface.co/facebook/bart-large-mnli

[7]https://grouplens.org/datasets/movielens/25m/
[8]https://www.yelp.com/dataset
[9]For all experiments with MovieLens, we use the 16k version of GPT-3.5-turbo-0613, due to MonoLLM requiring extra context length for $\mathbf{x}$.
[10]For each PEBOL policy, we use the MNLI temperature that performed best on the most datasets with continuous responses (see Supplementary Materials).

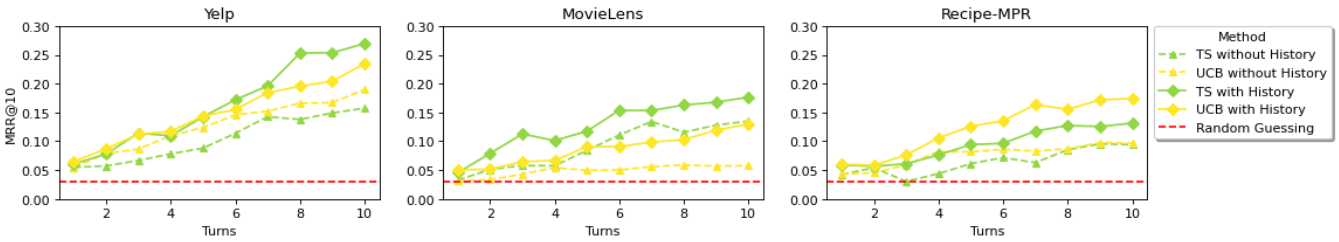

Figure 7: The effect of including the generated aspect history in the aspect generation prompt. Including the history improves performance, which we hypothesize is due to reducing repeated or uninformative queries.

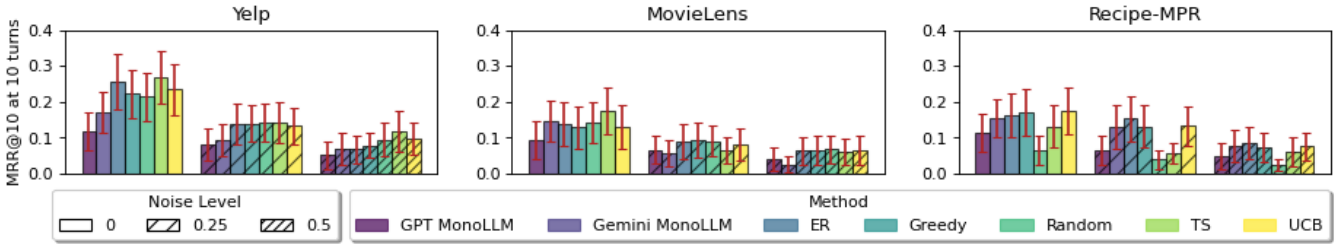

Figure 8: The effect of user response noise on MRR@10 – error bars are 95% confidence intervals.

Compared to GPT-MonoLLM, which uses the same LLM (GPT-3.5-turbo-0613) as PEBOL does for query generation, after 10 turns of dialogue PEBOL achieves: a mean MRR@10 of 0.27 vs. GPT-MonoLLM's MRR@10 of 0.12 on Yelp; 0.18 vs 0.09 on MovieLens; and 0.17 vs 0.11 on Recipe-MPR, respectively. Compared to Gemini-MonoLLM, which uses a newer generation LLM (Gemini-Pro) than PEBOL for query generation, after 10 turns PEBOL still achieves a higher mean MRR@10 of 0.27 vs. Gemini-MonoLLM's mean MRR@10 of 0.17 on Yelp; 0.18 vs. 0.15 on MovieLens, and 0.17 vs 0.16 on Recipe-MPR, respectively. While we did not have the resources to test PEBOL with Gemini query generation, we hypothesize that using a newer LLM for query generation can further improve PEBOL performance, since using the newer LLM (Gemini) shows performance improvements for the MonoLLM baseline.

6.2 RQ2 - Binary vs. Probabilistic Responses

Figure 6 compares PEBOL performance using binary (PEBOL-B) vs. continuous (PEBOL-P) feedback, and shows that performance is typically better when continuous responses are used – indicating that binary feedback models discard valuable information from the entailment probabilities.

6.3 RQ3 - Effect of User Response Noise

Figure 8 shows the impact of user response noise on MRR@10 at turn 10 – PEBOL generally continues to outperform MonoLLM under user response noise. Specifically, at all noise levels, both MonoLLM baselines are outperformed by all PEBOL-P variants on Yelp, and by at least one PEBOL-P variant on MovieLens and Recipe-MPR.

6.4 RQ4 - Comparison of Context Acquisition Policies

Figure 5 compares the performance of various PEBOL context acquisition policies – all policies show active preference learning, other than random item selection on RecipeMPR. There is considerable overlap between methods, however for most turns TS does well on Yelp and MovieLens while being beaten by Greedy, ER, and UCB on Recipe-MPR. As expected due to the randomness in sampling, TS performance is correlated with random item selection, while UCB performs quite similarly to greedy.

6.5 RQ5 - Effect of Aspect History in Query Generation

As shown in Figure 7, we see improvements in PEBOL performance from including a list of previously generated aspects in the aspect generation prompt. For instance, the differences in mean MRR@10 from including vs. excluding the query history for TS after 10 turns were: 0.27 vs 0.16 for Yelp; 0.18 vs 0.14 for MovieLens, and 0.13 vs 0.09 for Recipe-MPR, respectively. Practically, including the aspect generation history also helps to avoid repeat queries, which gain no information and could frustrate a user.

7 CONCLUSION AND FUTURE WORK

This paper presents a novel Bayesian optimization formalization of natural language (NL) preference elicitation (PE) over arbitrary NL item descriptions, as well as introducing and evaluating PEBOL, an algorithm for NL **P**reference **E**licitation with **B**ayesian **O**ptimization augmented **LL**Ms. As discussed below, our study also presents many opportunities for future work, including for addressing some of the limitations of PEBOL and our experiment setup.

User Studies. Firstly, our experiments limited by their reliance on LLM-simulated users. While the dialogue simulations indicate reasonable behaviour in the observed results, such as initial recommendation performance near random guessing, preference learning over time, and coherent user responses in logs such as those shown in Figure 7, future work would benefit from human user studies.

Multi-Item Belief Updates. While the assumption that item utilities can be updated independently allows the use of a simple and interpertable Beta-Bernouilli update model for each item, it also requires a separate NLI calculation to be performed for each item, which is computationally expensive. A key future direction is thus to explore alternative belief state forms which enable the joint updating of beliefs over all items from a single NLI computation.

Collaborative Belief Updates. Since PEBOL does not leverage any historical interactions with other users, an important future direction is to study NL-PE which leverages collaborative, multi-user data. One possibility is to initialize a cold start user's prior beliefs based on interaction histories with other users. Another direction is adapting collaborative filtering based belief updating, such as the methods used in item-based feedback PE techniques (e.g., [5]), to NL-PE.

Diverse Query Forms. While PEBOL uses a pointwise query generation strategy that selects one item description at a time for LLM context, future work can explore LLM-based acquisition functions with pairwise and setwise context selection. Such multi-item context selection would enable *contrastive* query generation that could better *discriminate* between item preferences.

NL-PE in ConvRec Architectures. Another direction for future research is the integration of NL-PE methodologies such as PEBOL into conversational recommendation (ConvRec) system architectures (e.g., [7, 9, 17, 20]), which must balance many tasks including recommendation, explanation, and personalized question answering. Thus, in contrast to PEBOL's pointwise queries and *"yes-or-no"* user responses, the use of PE in ConvRec systems implies that future algorithms will need to elicit preferences based on *arbitrary* pairs of NL system-user utterances. In these potential extensions, aspect-based NLI could be enabled by extracting aspects from utterances with LLMs [19].

REFERENCES

[1] Erdem Biyik, Fan Yao, Yinlam Chow, Alex Haig, Chih-wei Hsu, Mohammad Ghavamzadeh, and Craig Boutilier. 2023. Preference Elicitation with Soft Attributes in Interactive Recommendation. *ArXiv* abs/2311.02085 (2023). https://api.semanticscholar.org/CorpusID:265034238

[2] Craig Boutilier. 2002. A POMDP formulation of preference elicitation problems. In *AAAI/IAAI*. Edmonton, AB, 239–246.

[3] Eric Brochu, Tyson Brochu, and Nando De Freitas. 2010. A Bayesian interactive optimization approach to procedural animation design. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 103–112.

[4] Qibin Chen, Junyang Lin, Yichang Zhang, Ming Ding, Yukuo Cen, Hongxia Yang, and Jie Tang. 2019. Towards Knowledge-Based Recommender Dialog System. 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)*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, Hong Kong, China, 1803–1813. https://doi.org/10.18653/v1/D19-1189

[5] 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* (San Francisco, California, USA) *(KDD '16)*. Association for Computing Machinery, New York, NY, USA, 815–824.

[6] Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges Workshop*. Springer, 177–190.

[7] Yashar Deldjoo, Zhankui He, Julian McAuley, Anton Korikov, Scott Sanner, Arnau Ramisa, René Vidal, Maheswaran Sathiamoorthy, Atoosa Kasirzadeh, and Silvia Milano. 2024. A Review of Modern Recommender Systems Using Generative Models (Gen-RecSys). In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain*.

[8] Brochu Eric, Nando Freitas, and Abhijeet Ghosh. 2007. Active preference learning with discrete choice data. *Advances in Neural Information Processing Systems* 20 (2007).

[9] Luke Friedman, Sameer Ahuja, David Allen, Terry Tan, Hakim Sidahmed, Changbo Long, Jun Xie, Gabriel Schubiner, Ajay Patel, Harsh Lara, et al. 2023. Leveraging Large Language Models in Conversational Recommender Systems. *arXiv preprint arXiv:2305.07961* (2023).

[10] Roman Garnett. 2023. *Bayesian optimization*. Cambridge University Press.

[11] Javier González, Zhenwen Dai, Andreas Damianou, and Neil D Lawrence. 2017. Preferential bayesian optimization. In *International Conference on Machine Learning*. PMLR, 1282–1291.

[12] Shengbo Guo and Scott Sanner. 2010. Real-time multiattribute Bayesian preference elicitation with pairwise comparison queries. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings, 289–296.

[13] Shengbo Guo, Scott Sanner, and Edwin V Bonilla. 2010. Gaussian process preference elicitation. *Advances in Neural Information Processing Systems* 23 (2010).

[14] Kunal Handa, Yarin Gal, Ellie Pavlick, Noah Goodman, Jacob Andreas, Alex Tamkin, and Belinda Z. Li. 2024. Bayesian Preference Elicitation with Language Models. arXiv:2403.05534 [cs.CL]

[15] 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. *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management* (2023).

[16] Giannis Karamanolakis, Kevin Raji Cherian, Ananth Ravi Narayan, Jie Yuan, Da Tang, and Tony Jebara. 2018. Item recommendation with variational autoencoders and heterogeneous priors. In *Proceedings of the 3rd Workshop on Deep Learning for Recommender Systems*. 10–14.

[17] 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 *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Washington, DC, USA) *(SIGIR '24)*. ACM, New York, NY, USA.

[18] Mohammad M. Khajah, Brett D. Roads, Robert V. Lindsey, Yun-En Liu, and Michael C. Mozer. 2016. Designing Engaging Games Using Bayesian Optimization. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 5571–5582. https://doi.org/10.1145/2858036.2858253

[19] Anton Korikov, George Saad, Ethan Baron, Mustafa Khan, Manav Shah, and Scott Sanner. 2024. Multi-Aspect Reviewed-Item Retrieval via LLM Query Decomposition and Aspect Fusion. In *Proceedings of the 1st SIGIR'24 Workshop on Information Retrieval's Role in RAG Systems, July 18, 2024, Washington D.C., USA*.

[20] Anton Korikov, Scott Sanner, Yashar Deldjoo, Francesco Ricci, Zhankui He, Julian McAuley, Arnau Ramisa, Rene Vidal, Maheswaran Sathiamoorthy, Atoosa Kasirzadeh, and Silvia Milano. 2024. Large Language Model Driven Recommendation. *arXiv preprint arXiv:2404.XXXXX* (2024).

[21] Hoyeop Lee, Jinbae Im, Seongwon Jang, Hyunsouk Cho, and Sehee Chung. 2019. MeLU: Meta-Learned User Preference Estimator for Cold-Start Recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Anchorage, AK, USA) *(KDD '19)*. Association for Computing Machinery, New York, NY, USA, 1073–1082. https://doi.org/10.1145/3292500.3330859

[22] Wenqiang Lei, Gangyi Zhang, Xiangnan He, Yisong Miao, Xiang Wang, Liang Chen, and Tat-Seng Chua. 2020. Interactive Path Reasoning on Graph for Conversational Recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Virtual Event, CA, USA) *(KDD '20)*. Association for Computing Machinery, New York, NY, USA, 2073–2083. https://doi.org/10.1145/3394486.3403258

[23] Belinda Z. Li, Alex Tamkin, Noah Goodman, and Jacob Andreas. 2023. Eliciting Human Preferences with Language Models. arXiv:2310.11589 [cs.CL]

[24] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web*. 661–670.

[25] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. 2011. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms.

In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*. 297–306.

[26] Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. 2018. Towards deep conversational recommendations. *Advances in Neural Information Processing Systems* 31 (2018).

[27] Shijun Li, Wenqiang Lei, Qingyun Wu, Xiangnan He, Peng Jiang, and Tat-Seng Chua. 2021. Seamlessly Unifying Attributes and Items: Conversational Recommendation for Cold-start Users. *ACM Trans. Inf. Syst.* 39, 4, Article 40 (aug 2021), 29 pages. https://doi.org/10.1145/3446427

[28] Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On Faithfulness and Factuality in Abstractive Summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (Eds.). Association for Computational Linguistics, Online, 1906–1919. https://doi.org/10.18653/v1/2020.acl-main.173

[29] Francesca Rossi and Allesandro Sperduti. 2004. Acquiring both constraint and solution preferences in interactive constraint systems. *Constraints* 9, 4 (2004), 311–332.

[30] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. 2016. Taking the Human Out of the Loop: A Review of Bayesian Optimization. *Proc. IEEE* 104, 1 (2016), 148–175. https://doi.org/10.1109/JPROC.2015.2494218

[31] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a Large-scale Dataset for Fact Extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, Marilyn Walker, Heng Ji, and Amanda Stent (Eds.). Association for Computational Linguistics, New Orleans, Louisiana, 809–819. https://doi.org/10.18653/v1/N18-1074

[32] Ivan Vendrov, Tyler Lu, Qingqing Huang, and Craig Boutilier. 2020. Gradient-Based Optimization for Bayesian Preference Elicitation. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 06 (Apr. 2020), 10292–10301. https://doi.org/10.1609/aaai.v34i06.6592

[33] Xiaolei Wang, Kun Zhou, Ji-Rong Wen, and Wayne Xin Zhao. 2022. Towards unified conversational recommender systems via knowledge-enhanced prompt learning. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1929–1937.

[34] Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)* (New Orleans, Louisiana). Association for Computational Linguistics, 1112–1122. http://aclweb.org/anthology/N18-1101

[35] Bowen Yang, Cong Han, Yu Li, Lei Zuo, and Zhou Yu. 2022. Improving Conversational Recommendation Systems' Quality with Context-Aware Item Meta-Information. In *Findings of the Association for Computational Linguistics: NAACL 2022*. 38–48.

[36] Hojin Yang, Scott Sanner, Ga Wu, and Jin Peng Zhou. 2021. Bayesian Preference Elicitation with Keyphrase-Item Coembeddings for Interactive Recommendation. In *Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization*. 55–64.

[37] Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking Zero-shot Text Classification: Datasets, Evaluation and Entailment Approach. 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)*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, Hong Kong, China, 3914–3923. https://doi.org/10.18653/v1/D19-1404

[38] Haochen Zhang, Anton Korikov, Parsa Farinneya, Mohammad Mahdi Abdollah Pour, Manasa Bharadwaj, Ali Pesaranghader, Xi Yu Huang, Yi Xin Lok, Zhaoqi Wang, Nathan Jones, and Scott Sanner. 2023. Recipe-MPR: A Test Collection for Evaluating Multi-aspect Preference-based Natural Language Retrieval. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Taipei, Taiwan) *(SIGIR '23)*. Association for Computing Machinery, New York, NY, USA, 2744–2753. https://doi.org/10.1145/3539618.3591880

[39] Xiaoying Zhang, Hong Xie, Hang Li, and John C.S. Lui. 2020. Conversational Contextual Bandit: Algorithm and Application. In *Proceedings of The Web Conference 2020* (Taipei, Taiwan) *(WWW '20)*. Association for Computing Machinery, New York, NY, USA, 662–672. https://doi.org/10.1145/3366423.3380148

[40] Xiaoxue Zhao, Weinan Zhang, and Jun Wang. 2013. Interactive collaborative filtering. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*. 1411–1420.

Supplementary Materials

ABSTRACT

This document provides the supplementary materials for the paper titled "Bayesian Optimization with LLM Acquisition Functions for Natural Language Preference Elicitation".

A PROBABILISTIC GRAPHICAL MODEL FOR POSTERIOR UTILITY UPDATE

In this section, we present the probabilistic graphical model for the posterior utility updates introduced in our paper with a more detailed derivation of the posterior utility belief. As discussed in the paper, the objective of posterior inference is to update the prior belief maintained over the utility of each item $i \in \mathcal{I}$ denoted by $p(u_i)$ given the query q^t, item description x_i, and e_i^t, the binary observation variable representing whether the item description entails the user's response to the queried preference q_t.

Figure 1 shows the graphical model representation of these variables. The presented query q^t and the item description x_i are observed (shaded), while the relevance of item i to query q^t is latent (unshaded) and denoted by $r_i^t \in \{0, 1\}$ and conditioned on the latent item utility u_i. We observe whether the item x_i "truly" entails (i.e., e_i^t = True) the user's response to query q_t (as determined by the NLI entailment probability w_i^t if the item is relevant, i.e., $r_i^t = 1$). The conditional probability distributions in this graphical model are formally defined as follows:

$$p(u_i) = \text{Beta}(u_i; \alpha_i, \beta_i), \tag{1}$$

$$p(r_i^t | u_i) = \text{Bernoulli}(u_i), \tag{2}$$

$$p(e_i^t = \text{True} | r_i^t, q^t, x_i) = \begin{cases} w_i^t & r_i^t = 1 \\ 1 - w_i^t & r_i^t = 0 \end{cases}, \tag{3}$$

To further explain the rationale for Eq (3), we note that w_i^t is the natural language entailment probability that item description x_i entails the aspect queried in the user's response to q_t given that item i is relevant ($r_i^t = 1$). This entailment probability is obtained from the NLI model, which produces the probability that the entailment is *true*, hence the reason why e_i^t = True. If item i is instead irrelevant ($r_i^t = 0$) then we assume for simplicity that $1 - w_i^t$ is the probability of the true entailment.[1]

To obtain the posterior utility $p(u_i | x_i, q^t, e_i^t)$, we need to marginalize the joint distribution $p(u_i, r_i^t | x_i, q^t, e_i^t)$ over r_i^t. Formally,

$$p(u_i | x_i, q^t, e_i^t) = \sum_{r_i^t} p(u_i, r_i^t | x_i, q^t, e_i^t). \tag{4}$$

Considering the conditional independencies determined from the graphical model, the joint distribution factorizes as

$$p(u_i, r_i^t | x_i, q^t, e_i^t) = p(u_i) p(r_i^t | u_i) p(e_i^t | r_i^t, x_i, q^t). \tag{5}$$

[1]We note that an alternative approach (not used here) could attempt to use NLI to determine the probability of an incorrect true entailment (or confusion) given that item i is irrelevant ($r_i^t = 0$). That is, there is no inherent requirement for the two cases of Eq (3) to sum to 1 since r_i^t is on the conditional side.

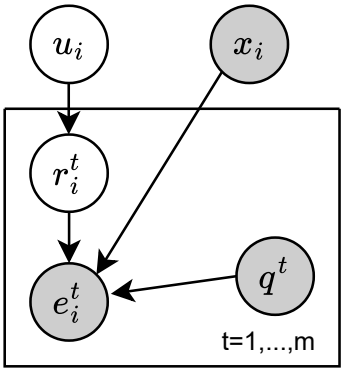

Figure 1: Graphical model used for posterior utility updates. u_i is the random variable representing the utility of item $i \in \mathcal{I}$ and x_i is the item description. q^t is the query presented at iteration t, r_i^t is the variable representing the latent (not directly observed) relevance of item i at step t, and e_i^t is the binary observation representing whether the item description entails the user preference. Unshaded and shaded nodes indicate unobserved and observed variables respectively.

Next, we replace the probability distribution of each factor according to Equations (1), (2), (3) in (4), to obtain

$$p(u_i | x_i, q^t, e_i^t = \text{True}) \propto \sum_{r_i^t \in \{0,1\}} u_i^\alpha (1 - u_i)^\beta \begin{pmatrix} u_i & r_i^t = 1 \\ 1 - u_i & r_i^t = 0 \end{pmatrix} \begin{pmatrix} w_i^t & r_i^t = 1 \\ 1 - w_i^t & r_i^t = 0 \end{pmatrix}. \tag{6}$$

Expanding the summation yields

$$\begin{aligned} p(u_i | x_i, q^t, e_i^t = \text{True}) &\propto w_i^t u_i^{\alpha+1} (1-u_i)^\beta + (1-w_i^t) u_i^\alpha (1-u_i)^{\beta+1} \\ &\propto w_i^t \text{Beta}(u_i; \alpha + 1, \beta) + (1 - w_i^t) \text{Beta}(u_i; \alpha, \beta + 1) \end{aligned} \tag{7}$$

The latter term represents a mixture of Beta distributions that is challenging to handle since multiple posterior updates would cause the number of components in the mixture to grow exponentially with the number of query observations m, leading to substantial computational and memory complexity.

To address this issue, several methods have been proposed for approximating the posterior distribution to allow for tractable computations. In this work, we use the Assumed Density Filtering (ADF) approach, a technique widely used in Bayesian filtering and tracking problems to project a complex posterior to an assumed simpler form (often the same form as the prior to maintain a closed-form). In our case, we project the Beta mixture posterior to a single Beta in order to maintain a closed-form Beta approximation of the posterior update matching the form of the Beta prior in Eq (1).

To apply ADF, we assume a Beta distribution with parameters α' and β' for the posterior, and approximate the original mixture of Beta's with this distribution by equating their first moments (i.e.,

their means):

$$\frac{\alpha'}{\alpha' + \beta'} = \frac{w_i^t(1+\alpha)}{(1+\alpha+\beta)} + \frac{\alpha(1-w_i^t)}{1+\alpha+\beta} \tag{8}$$

$$= \frac{\left[\alpha + w_i^t\right]}{\left[\alpha + w_i^t\right] + \left[\beta + (1-w_i^t)\right]}. \tag{9}$$

Equating the numerators yields

$$\alpha' = \alpha + w_i^t, \tag{10}$$

and replacing this α' in the equation of the denominators results in

$$\beta' = 1 + \beta - w_i^t. \tag{11}$$

Thus, the *"mean matched"* posterior is $\text{Beta}(u_i; \alpha + w_i^t, 1 + \beta - w_i^t)$.

Matching two distributions by equating their first moments is a special case of a more general technique called *"moment matching"*, which is widely used to approximate a complex probability distribution with a simpler one by equating their moments. In our work, we adopted a special case of this approach by matching the first moments, which we refer to as *"mean matching"* of the distributions that we used for its simplicity and intuitive interpretation. However, this is only one of the possible solutions, and a complete moment matching derivation results in a slightly different solution.

With this *"mean matching"* derivation and current item i posterior $Beta(u_i; \alpha, \beta)$ at step $t-1$, we can now perform an incremental posterior update after at step t given the probability w_i^t that the item description x_i entails preference query q_t yielding the closed-form Beta posterior $Beta(u_i; \alpha + w_i^t, 1 + \beta - w_i^t)$ as used in PEBOL-P.

Table 1: Examples of LLM Aspect-Based Query Generation from an Item Description

Dataset	Item Description d_i	Generated Aspect a^t	Generated Query q^t
MovieLens	Movie Title: Meet John Doe (1941) Genres: Comedy, Drama Tags: Christianity, Frank Capra, acting, anti-fascism, class issues, journalism, patriotic, pro american, thought provoking, AFI 100 (Cheers), BD-R, Barbara Stanwyck, Gary Cooper, baseball player, compare: This Is Our Land (2017), domain, funny, radio broadcast, reviewed in the NYer by Anthony Lanne (2018-04-30), suicide note	patriotism	Are you interested in movies with patriotic themes?
		classic	Do you enjoy classic movies?
Recipe-MPR	Spaghetti with mushrooms, onion, green pepper, chicken breasts, and alfredo sauce	alfredo sauce	Do you like alfredo sauce?
		chicken breast	Do you like chicken breasts?
Yelp	name: Le Pain Quotidien categories: Restaurants, Bakeries, Breakfast & Brunch, Coffee & Tea, Food, Belgian, French	bakery	Do you like bakeries?
		French pastries	Do you like French pastries?

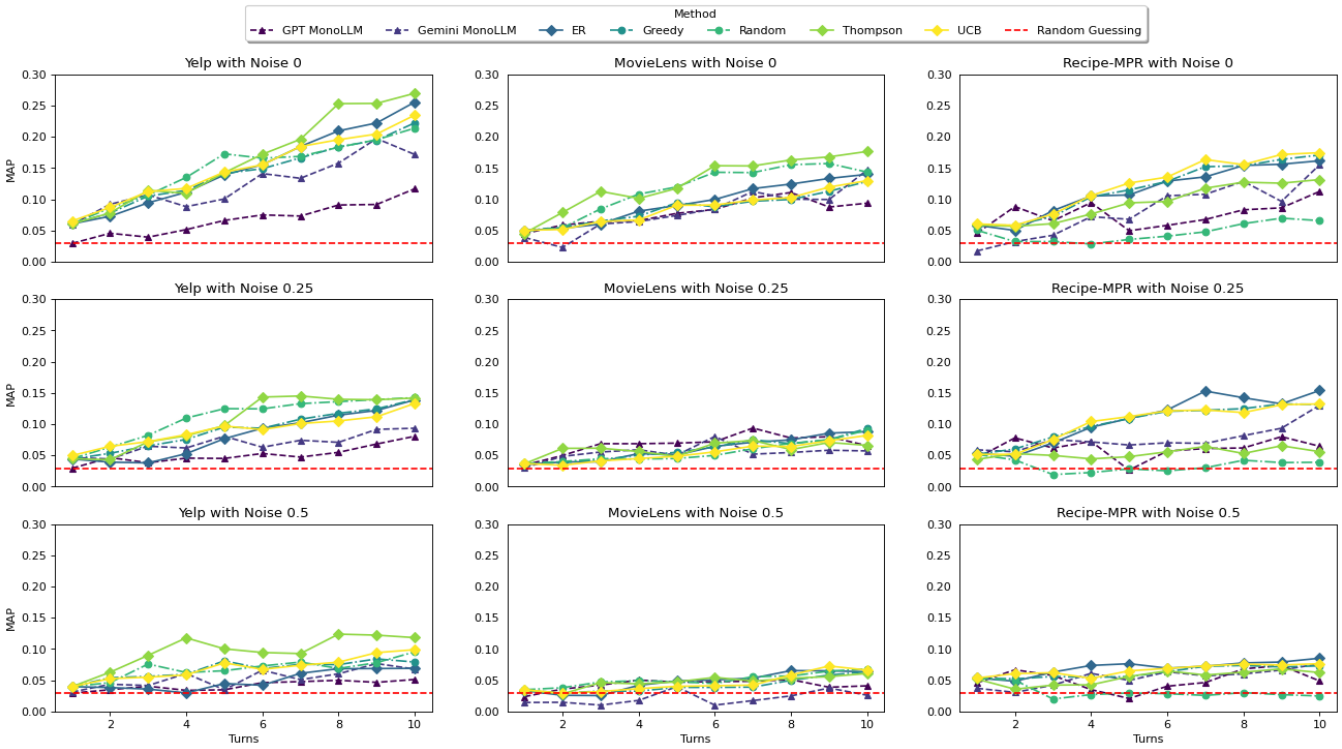

Figure 2: MAP@10 for all turns on all datasets and noise levels

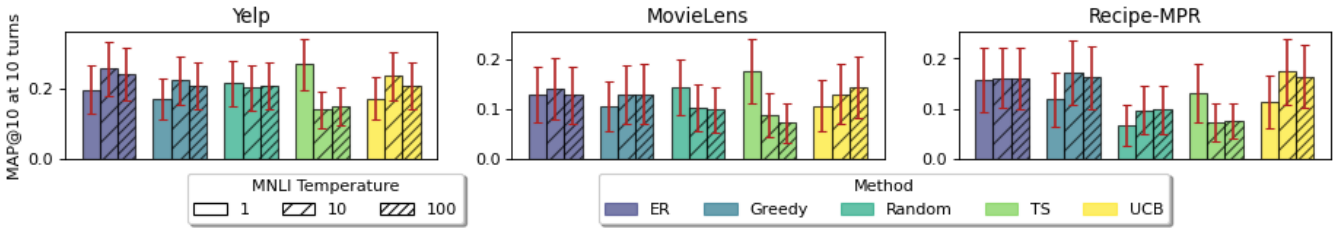

Figure 3: The effect of MNLI temperature on MAP@10 without noise – error bars are 95% C.I.s. Other results are reported using the temperatures that perform best across the most datasets for each policy.

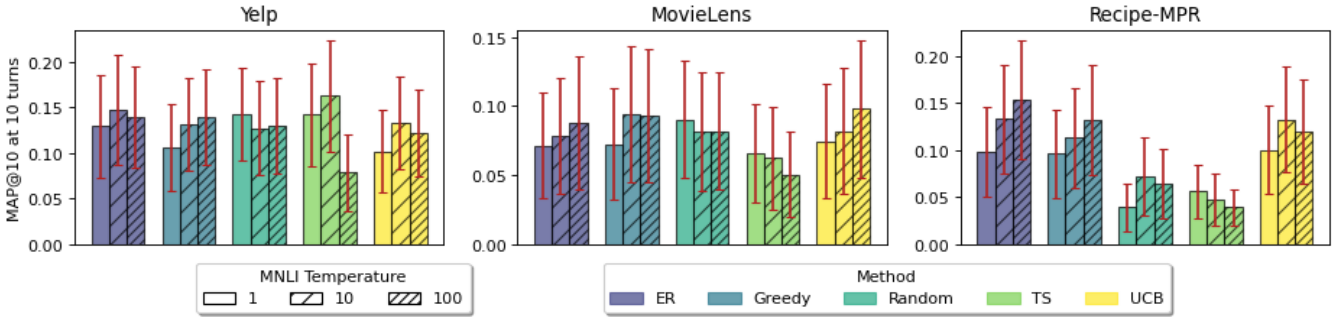

Figure 4: Effect of MNLI Temperature with noise 0.25

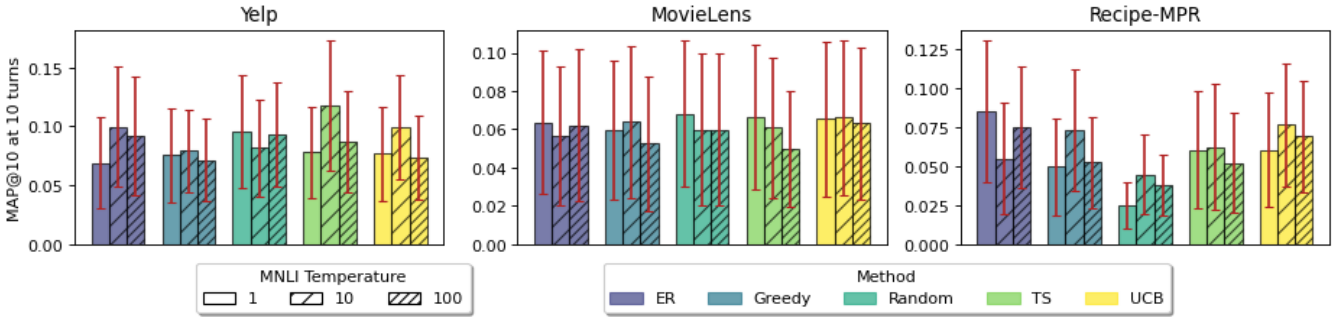

Figure 5: Effect of MNLI Temperature with noise 0.5

```
Given a set of item descriptions and a
conversation history,  generate a new
"yes or no" query to help determine the
user's preferences towards the items.

< 1 few-shot example >

Input:
''''
Item Descriptions:
****
{{ item descriptions }}
****
Conversation History:
++++
{{ past query and response pairs }}
++++

Output:
```

(a) Monolithic LLM query generation

```
Given a set of {{ domain }} descriptions,
a conversation history, and a list
length, generate a newline seperated list
of {{ domain }} names in order of most to
least relevant for the user.

< 1 few-shot example >

Input:
''''
Item Descriptions:
****
{{ item descriptions }}
****
Conversation History:
++++
{{ past query and response pairs }}
++++

List length: {{ # of items }}

Output:
```

(b) Monolithic LLM recommendation generation

```
Identify an important aspect of an item
given the item description. The aspect
should be different from any previously
identified aspects and must not exceed 3
words.

< 3 few-shot examples >

Input:
''''
Item Description:
****
{{ item description }}
****

Previously extracted aspects:
****
{{ previous aspects }}
****

Output:
```

(c) Aspect generation

```
Given a specific aspect of an item and
the item description, generate a "yes or
no" query to determine a user's
preferences towards that aspect.

< 2 few-shot examples >

Input:
''''
Item Description:
****
{{ item description }}
****

Aspect:
****
{{ aspect }}
****

Output:
```

(d) Aspect-based query generation

```
You are a user who is being asked yes or
no queries to elicit your preferences.
You must reply yes or no to the query
based on the description of your
preferred item.

< 2 few-shot examples >

Input:
''''
Item description:
****
{{ item description }}
****

Query:
****
{{ query }}
****
Response:
```

(e) User simulation

Figure 6: LLM Prompting Templates

Table 2: MRR@10 Mean and 95% C.I. for PEBOL-P vs MonoLLM on MovieLens without Response Noise

Turn Method	1	2	3	4	5	6	7	8	9	10
MonoLLM GPT Mean	0.04	0.06	0.06	0.07	0.08	0.08	0.10	0.11	0.09	0.09
MonoLLM GPT CI LB	0.01	0.02	0.02	0.02	0.03	0.03	0.05	0.06	0.04	0.04
MonoLLM GPT CI UB	0.08	0.10	0.11	0.11	0.12	0.13	0.15	0.16	0.14	0.15
MonoLLM Gemini Mean	0.04	0.02	0.06	0.06	0.07	0.08	0.11	0.10	0.10	0.15
MonoLLM Gemini CI LB	0.01	0.01	0.02	0.03	0.04	0.05	0.07	0.05	0.06	0.09
MonoLLM Gemini CI UB	0.07	0.04	0.10	0.10	0.11	0.12	0.16	0.15	0.14	0.20
ER Mean	**0.05**	0.05	0.06	0.08	0.09	0.10	0.12	0.12	0.13	0.14
ER CI LB	0.01	0.02	0.03	0.04	0.05	0.05	0.06	0.07	0.07	0.08
ER CI UB	0.08	0.09	0.09	0.12	0.13	0.15	0.17	0.18	0.19	0.20
Greedy Mean	**0.05**	0.05	0.07	0.07	0.09	0.09	0.10	0.10	0.11	0.13
Greedy CI LB	0.02	0.02	0.03	0.03	0.05	0.04	0.05	0.05	0.06	0.07
Greedy CI UB	0.09	0.09	0.10	0.11	0.14	0.14	0.15	0.15	0.17	0.19
Random Mean	**0.05**	0.05	0.08	0.11	0.12	0.14	0.14	0.16	0.16	0.14
Random CI LB	0.01	0.02	0.04	0.05	0.06	0.08	0.08	0.09	0.09	0.09
Random CI UB	0.08	0.09	0.13	0.16	0.18	0.21	0.20	0.22	0.22	0.20
TS Mean	**0.05**	**0.08**	**0.11**	**0.10**	**0.12**	**0.15**	**0.15**	**0.16**	**0.17**	**0.18**
TS CI LB	0.01	0.03	0.06	0.05	0.06	0.09	0.09	0.10	0.10	0.11
TS CI UB	0.08	0.13	0.17	0.15	0.17	0.22	0.21	0.23	0.23	0.24
UCB Mean	**0.05**	0.05	0.06	0.07	0.09	0.09	0.10	0.10	0.12	0.13
UCB CI LB	0.02	0.02	0.03	0.03	0.05	0.04	0.05	0.05	0.06	0.07
UCB CI UB	0.09	0.09	0.10	0.10	0.13	0.14	0.15	0.15	0.17	0.19

Table 3: MRR@10 Mean and 95% C.I. for PEBOL-P vs MonoLLM on Recipe-MPR without Response Noise

Turn Method	1	2	3	4	5	6	7	8	9	10
MonoLLM GPT Mean	0.05	**0.09**	0.07	0.09	0.05	0.06	0.07	0.08	0.09	0.11
MonoLLM GPT CI LB	0.01	0.05	0.03	0.05	0.02	0.02	0.03	0.04	0.04	0.06
MonoLLM GPT CI UB	0.08	0.13	0.10	0.14	0.08	0.10	0.11	0.13	0.13	0.17
MonoLLM Gemini Mean	0.02	0.03	0.04	0.07	0.07	0.11	0.11	0.13	0.10	0.16
MonoLLM Gemini CI LB	0.01	0.02	0.02	0.04	0.03	0.06	0.07	0.08	0.06	0.10
MonoLLM Gemini CI UB	0.03	0.05	0.07	0.10	0.10	0.15	0.15	0.18	0.13	0.21
ER Mean	**0.06**	0.05	**0.08**	**0.11**	0.11	0.13	0.14	0.15	0.16	0.16
ER CI LB	0.02	0.02	0.04	0.06	0.06	0.08	0.08	0.09	0.09	0.10
ER CI UB	0.10	0.08	0.13	0.16	0.16	0.18	0.19	0.22	0.22	0.22
Greedy Mean	**0.06**	0.06	0.07	0.10	0.11	0.13	0.15	0.15	0.16	0.17
Greedy CI LB	0.02	0.02	0.03	0.05	0.06	0.07	0.09	0.09	0.10	0.11
Greedy CI UB	0.10	0.09	0.11	0.15	0.17	0.18	0.21	0.21	0.23	0.24
Random Mean	0.05	0.03	0.03	0.03	0.04	0.04	0.05	0.06	0.07	0.07
Random CI LB	0.02	0.01	0.01	0.00	0.01	0.01	0.02	0.02	0.03	0.02
Random CI UB	0.08	0.06	0.06	0.05	0.06	0.07	0.08	0.10	0.11	0.11
TS Mean	**0.06**	0.06	0.06	0.08	0.09	0.10	0.12	0.13	0.13	0.13
TS CI LB	0.02	0.02	0.02	0.03	0.05	0.05	0.06	0.07	0.07	0.07
TS CI UB	0.10	0.09	0.10	0.12	0.14	0.14	0.17	0.19	0.18	0.19
UCB Mean	**0.06**	0.06	**0.08**	**0.11**	**0.13**	**0.14**	**0.16**	**0.16**	**0.17**	**0.17**
UCB CI LB	0.02	0.02	0.03	0.06	0.07	0.08	0.10	0.09	0.11	0.11
UCB CI UB	0.10	0.09	0.12	0.16	0.18	0.19	0.23	0.22	0.24	0.24

Table 4: MRR@10 Mean and 95% C.I. for PEBOL-P vs MonoLLM on Yelp without Response Noise

Turn Method	1	2	3	4	5	6	7	8	9	10
MonoLLM GPT Mean	0.03	0.05	0.04	0.05	0.07	0.07	0.07	0.09	0.09	0.12
MonoLLM GPT CI LB	0.01	0.01	0.01	0.02	0.03	0.03	0.03	0.04	0.04	0.06
MonoLLM GPT CI UB	0.05	0.08	0.07	0.08	0.10	0.12	0.12	0.14	0.14	0.17
MonoLLM Gemini Mean	0.06	**0.09**	**0.11**	0.09	0.10	0.14	0.13	0.16	0.20	0.17
MonoLLM Gemini CI LB	0.02	0.05	0.06	0.05	0.06	0.08	0.09	0.10	0.14	0.12
MonoLLM Gemini CI UB	0.10	0.14	0.15	0.13	0.14	0.20	0.18	0.21	0.26	0.23
ER Mean	0.06	0.07	0.09	0.11	0.14	0.16	0.18	0.21	0.22	0.26
ER CI LB	0.03	0.03	0.05	0.06	0.08	0.09	0.12	0.14	0.15	0.18
ER CI UB	0.10	0.11	0.14	0.16	0.20	0.22	0.25	0.28	0.29	0.33
Greedy Mean	0.06	0.08	**0.11**	0.11	0.14	0.15	0.17	0.18	0.19	0.22
Greedy CI LB	0.03	0.04	0.06	0.07	0.08	0.09	0.10	0.12	0.13	0.16
Greedy CI UB	0.10	0.12	0.15	0.16	0.20	0.21	0.23	0.25	0.26	0.29
Random Mean	0.06	**0.09**	**0.11**	**0.13**	**0.17**	0.17	0.17	0.18	0.20	0.21
Random CI LB	0.03	0.04	0.06	0.08	0.11	0.11	0.11	0.12	0.13	0.15
Random CI UB	0.10	0.13	0.16	0.19	0.23	0.23	0.23	0.24	0.26	0.28
TS Mean	0.06	0.08	**0.11**	0.11	0.14	**0.17**	**0.20**	**0.25**	**0.25**	**0.27**
TS CI LB	0.03	0.04	0.06	0.07	0.09	0.11	0.13	0.18	0.18	0.20
TS CI UB	0.10	0.12	0.17	0.15	0.20	0.23	0.26	0.33	0.32	0.34
UCB Mean	**0.07**	**0.09**	**0.11**	0.12	0.14	0.16	0.18	0.20	0.20	0.23
UCB CI LB	0.03	0.05	0.06	0.07	0.08	0.09	0.12	0.13	0.14	0.16
UCB CI UB	0.10	0.13	0.16	0.17	0.20	0.22	0.25	0.26	0.27	0.30

Table 5: MRR@10 Mean and 95% C.I. for PEBOL-P without Aspect History on MovieLens without Response Noise

Turn Method	1	2	3	4	5	6	7	8	9	10
TS Mean	**0.03**	**0.05**	**0.06**	**0.06**	**0.08**	**0.11**	**0.13**	**0.12**	**0.13**	**0.14**
TS CI LB	0.01	0.02	0.02	0.02	0.04	0.06	0.08	0.06	0.07	0.08
TS CI UB	0.06	0.09	0.09	0.09	0.13	0.16	0.19	0.17	0.18	0.19
UCB Mean	**0.03**	0.03	0.04	0.05	0.05	0.05	0.06	0.06	0.06	0.06
UCB CI LB	0.01	0.01	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02
UCB CI UB	0.05	0.06	0.07	0.09	0.08	0.08	0.09	0.10	0.09	0.09

Table 6: MRR@10 Mean and 95% C.I. for PEBOL-P without Aspect History on Recipe-MPR without Response Noise

Turn Method	1	2	3	4	5	6	7	8	9	10
TS Mean	**0.04**	**0.05**	0.03	0.04	0.06	0.07	0.06	**0.09**	**0.10**	0.09
TS CI LB	0.01	0.02	0.01	0.01	0.03	0.03	0.03	0.04	0.05	0.05
TS CI UB	0.07	0.09	0.05	0.07	0.10	0.12	0.10	0.13	0.14	0.14
UCB Mean	**0.04**	**0.05**	**0.06**	**0.08**	**0.08**	**0.09**	**0.08**	**0.09**	**0.10**	**0.10**
UCB CI LB	0.01	0.01	0.02	0.04	0.04	0.04	0.04	0.04	0.05	0.05
UCB CI UB	0.07	0.08	0.09	0.12	0.12	0.13	0.12	0.13	0.14	0.14

Table 7: MRR@10 Mean and 95% C.I. for PEBOL-P without Aspect History on Yelp without Response Noise

Turn Method	1	2	3	4	5	6	7	8	9	10
TS Mean	**0.05**	0.06	0.07	0.08	0.09	0.11	0.14	0.14	0.15	0.16
TS CI LB	0.02	0.02	0.03	0.04	0.05	0.06	0.09	0.08	0.09	0.10
TS CI UB	0.09	0.09	0.10	0.11	0.13	0.16	0.20	0.19	0.21	0.22
UCB Mean	**0.05**	**0.08**	**0.09**	**0.11**	**0.12**	**0.15**	**0.15**	**0.17**	**0.17**	**0.19**
UCB CI LB	0.02	0.04	0.05	0.06	0.07	0.09	0.10	0.11	0.11	0.13
UCB CI UB	0.09	0.12	0.13	0.16	0.17	0.20	0.21	0.22	0.22	0.25

Table 8: MRR@10 Mean and 95% C.I. for PEBOL-B on MovieLens without Response Noise

Turn Method	1	2	3	4	5	6	7	8	9	10
ER Mean	**0.03**	0.04	0.04	0.05	0.06	0.08	0.09	0.11	0.11	0.12
ER CI LB	0.01	0.01	0.01	0.02	0.03	0.04	0.04	0.06	0.07	0.07
ER CI UB	0.06	0.07	0.08	0.08	0.10	0.12	0.14	0.16	0.16	0.16
Greedy Mean	**0.03**	0.04	0.05	0.05	0.07	**0.09**	**0.10**	0.11	0.11	0.11
Greedy CI LB	0.01	0.01	0.02	0.02	0.03	0.05	0.05	0.06	0.07	0.07
Greedy CI UB	0.06	0.08	0.08	0.09	0.11	0.13	0.14	0.16	0.16	0.16
Random Mean	**0.03**	0.04	0.04	0.06	0.06	0.08	0.09	0.10	0.10	0.10
Random CI LB	0.01	0.01	0.01	0.03	0.03	0.04	0.05	0.06	0.06	0.06
Random CI UB	0.06	0.06	0.07	0.09	0.10	0.12	0.13	0.14	0.14	0.14
TS Mean	**0.03**	**0.05**	**0.06**	**0.07**	**0.08**	0.07	0.09	**0.12**	**0.12**	**0.13**
TS CI LB	0.01	0.02	0.02	0.03	0.03	0.03	0.04	0.07	0.07	0.07
TS CI UB	0.06	0.08	0.09	0.11	0.12	0.11	0.13	0.17	0.18	0.18
UCB Mean	**0.03**	0.04	0.05	0.05	0.06	0.08	0.09	0.10	0.11	0.11
UCB CI LB	0.01	0.01	0.02	0.02	0.03	0.04	0.05	0.06	0.06	0.06
UCB CI UB	0.06	0.07	0.08	0.08	0.09	0.13	0.14	0.15	0.15	0.15

Table 9: MRR@10 Mean and 95% C.I. for PEBOL-B on Recipe-MPR without Response Noise

Turn Method	1	2	3	4	5	6	7	8	9	10
ER Mean	**0.04**	**0.06**	**0.09**	**0.07**	0.09	0.09	0.10	**0.13**	0.10	0.10
ER CI LB	0.00	0.02	0.05	0.03	0.04	0.04	0.05	0.07	0.05	0.05
ER CI UB	0.07	0.10	0.14	0.10	0.14	0.14	0.16	0.19	0.15	0.15
Greedy Mean	**0.04**	0.04	0.06	**0.07**	**0.10**	**0.11**	**0.12**	0.12	0.12	**0.13**
Greedy CI LB	0.00	0.01	0.02	0.03	0.05	0.06	0.06	0.06	0.06	0.07
Greedy CI UB	0.07	0.08	0.10	0.10	0.15	0.17	0.18	0.18	0.18	0.19
Random Mean	**0.04**	0.03	0.03	0.05	0.04	0.04	0.05	0.04	0.05	0.06
Random CI LB	0.00	0.00	0.01	0.01	0.01	0.01	0.01	0.01	0.02	0.02
Random CI UB	0.07	0.06	0.05	0.09	0.07	0.08	0.08	0.07	0.09	0.11
TS Mean	**0.04**	**0.06**	0.06	**0.07**	0.09	0.09	0.10	0.10	0.11	0.12
TS CI LB	0.00	0.02	0.02	0.03	0.05	0.04	0.05	0.05	0.06	0.06
TS CI UB	0.07	0.09	0.10	0.12	0.14	0.14	0.16	0.15	0.17	0.17
UCB Mean	**0.04**	0.05	0.07	**0.07**	0.09	0.11	**0.12**	0.12	**0.13**	**0.13**
UCB CI LB	0.00	0.02	0.03	0.03	0.04	0.05	0.06	0.07	0.07	0.08
UCB CI UB	0.07	0.08	0.11	0.11	0.14	0.16	0.17	0.18	0.18	0.19

Table 10: MRR@10 Mean and 95% C.I. for PEBOL-B on Yelp without Response Noise

Turn Method	1	2	3	4	5	6	7	8	9	10
ER Mean	**0.05**	0.05	0.07	0.09	0.11	0.11	0.11	0.11	0.11	0.11
ER CI LB	0.02	0.02	0.04	0.05	0.07	0.07	0.07	0.06	0.06	0.06
ER CI UB	0.08	0.09	0.11	0.12	0.15	0.15	0.15	0.15	0.15	0.15
Greedy Mean	**0.05**	**0.06**	0.08	0.09	0.10	0.10	0.10	0.10	0.10	0.10
Greedy CI LB	0.02	0.02	0.04	0.05	0.06	0.06	0.06	0.06	0.06	0.06
Greedy CI UB	0.08	0.09	0.11	0.13	0.14	0.13	0.13	0.14	0.14	0.13
Random Mean	**0.05**	**0.06**	0.07	0.09	0.11	0.10	0.10	0.11	0.12	0.14
Random CI LB	0.02	0.03	0.03	0.05	0.06	0.06	0.06	0.07	0.07	0.09
Random CI UB	0.08	0.09	0.11	0.14	0.15	0.14	0.14	0.15	0.16	0.19
TS Mean	**0.05**	**0.06**	**0.12**	**0.12**	**0.16**	**0.16**	**0.18**	**0.18**	**0.22**	**0.24**
TS CI LB	0.02	0.02	0.07	0.07	0.10	0.10	0.12	0.12	0.15	0.17
TS CI UB	0.08	0.10	0.17	0.17	0.22	0.22	0.24	0.24	0.28	0.31
UCB Mean	**0.05**	**0.06**	0.08	0.10	0.10	0.10	0.10	0.10	0.10	0.10
UCB CI LB	0.02	0.02	0.04	0.06	0.06	0.06	0.06	0.06	0.06	0.06
UCB CI UB	0.08	0.09	0.12	0.13	0.14	0.14	0.14	0.14	0.14	0.13

Table 11: MRR@10 Mean and 95% C.I. at Turn 10 for PEBOL-P and MonoLLM baselines on MovieLens with Different Levels of Response Noise

Noise Level Method	0	0.25	0.5
MonoLLM GPT Mean	0.09	0.07	0.04
MonoLLM GPT CI LB	0.04	0.03	0.01
MonoLLM GPT CI UB	0.15	0.10	0.07
MonoLLM Gemini Mean	0.15	0.06	0.03
MonoLLM Gemini CI LB	0.09	0.02	0.00
MonoLLM Gemini CI UB	0.20	0.09	0.05
ER Mean	0.14	**0.09**	0.06
ER CI LB	0.08	0.04	0.03
ER CI UB	0.20	0.14	0.10
Greedy Mean	0.13	**0.09**	0.06
Greedy CI LB	0.07	0.04	0.02
Greedy CI UB	0.19	0.14	0.10
Random Mean	0.14	**0.09**	**0.07**
Random CI LB	0.09	0.05	0.03
Random CI UB	0.20	0.13	0.11
TS Mean	**0.18**	0.07	0.06
TS CI LB	0.11	0.03	0.02
TS CI UB	0.24	0.10	0.10
UCB Mean	0.13	0.08	**0.07**
UCB CI LB	0.07	0.04	0.03
UCB CI UB	0.19	0.13	0.11

Table 12: MRR@10 Mean and 95% C.I. at Turn 10 for PEBOL-P and MonoLLM baselines on Recipe-MPR with Different Levels of Response Noise

Noise Level Method	0	0.25	0.5
MonoLLM GPT Mean	0.11	0.06	0.05
MonoLLM GPT CI LB	0.06	0.02	0.01
MonoLLM GPT CI UB	0.17	0.11	0.08
MonoLLM Gemini Mean	0.16	0.13	0.08
MonoLLM Gemini CI LB	0.10	0.07	0.03
MonoLLM Gemini CI UB	0.21	0.19	0.12
ER Mean	0.16	**0.15**	**0.09**
ER CI LB	0.10	0.09	0.04
ER CI UB	0.22	0.22	0.13
Greedy Mean	**0.17**	0.13	0.07
Greedy CI LB	0.11	0.07	0.03
Greedy CI UB	0.24	0.19	0.11
Random Mean	0.07	0.04	0.02
Random CI LB	0.02	0.01	0.01
Random CI UB	0.11	0.06	0.04
TS Mean	0.13	0.06	0.06
TS CI LB	0.07	0.03	0.02
TS CI UB	0.19	0.08	0.10
UCB Mean	**0.17**	0.13	0.08
UCB CI LB	0.11	0.08	0.04
UCB CI UB	0.24	0.19	0.12

Table 13: MRR@10 Mean and 95% C.I. at Turn 10 for PEBOL-P and MonoLLM baselines on Yelp with Different Levels of Response Noise

Noise Level Method	0	0.25	0.5
MonoLLM GPT Mean	0.12	0.08	0.05
MonoLLM GPT CI LB	0.06	0.03	0.01
MonoLLM GPT CI UB	0.17	0.13	0.09
MonoLLM Gemini Mean	0.17	0.09	0.07
MonoLLM Gemini CI LB	0.12	0.05	0.02
MonoLLM Gemini CI UB	0.23	0.14	0.11
ER Mean	0.26	**0.14**	0.07
ER CI LB	0.18	0.08	0.03
ER CI UB	0.33	0.19	0.11
Greedy Mean	0.22	**0.14**	0.08
Greedy CI LB	0.16	0.09	0.04
Greedy CI UB	0.29	0.19	0.11
Random Mean	0.21	**0.14**	0.10
Random CI LB	0.15	0.09	0.05
Random CI UB	0.28	0.19	0.14
TS Mean	**0.27**	**0.14**	**0.12**
TS CI LB	0.20	0.08	0.06
TS CI UB	0.34	0.20	0.17
UCB Mean	0.23	0.13	0.10
UCB CI LB	0.16	0.08	0.05
UCB CI UB	0.30	0.18	0.14