Density-based User Representation using Gaussian Process Regression for Multi-interest Personalized Retrieval

Haolun Wu [1 2] Ofer Meshi [3] Masrour Zoghi [3] Fernando Diaz [4] Xue Liu [1 2] Craig Boutilier [3]
Maryam Karimzadehgan [3]

arXiv:submit/5453936 [cs.IR] 6 Mar 2024

Abstract

Accurate modeling of the diverse and dynamic interests of users remains a significant challenge in the design of personalized recommender systems. Existing user modeling methods, like single-point and multi-point representations, have limitations w.r.t. accuracy, diversity, and adaptability. To overcome these deficiencies, we introduce *density-based user representations (DURs)*, a novel method that leverages Gaussian process regression (GPR) for effective multi-interest recommendation and retrieval. Our approach, GPR4DUR, exploits DURs to capture user interest variability without manual tuning, incorporates uncertainty-awareness, and scales well to large numbers of users. Experiments using real-world offline datasets confirm the adaptability and efficiency of GPR4DUR, while online experiments with simulated users demonstrate its ability to address the exploration-exploitation trade-off by effectively utilizing model uncertainty.

1. Introduction

With the proliferation of online platforms, users have ready access to content, products and services drawn from a vast corpus of candidates. Personalized *recommender systems (RSs)* play a vital role in reducing information overload and helping users navigate this space. It is widely recognized that users rarely have a single intent or interest when interacting with an RS (Weston et al., 2013; Pal et al., 2020; Cen et al., 2020). To enhance personalization, recent work focuses on discovering a user's multiple interests and recommending items that attempt to span their interests (Cen et al., 2020; Tan et al., 2021; Li et al., 2019). However, this is challenging for two reasons. First, user interests are diverse and dynamic: diversity makes it hard to detect all interests, while their dynamic nature renders determining which user interest is active at any given time quite difficult. Second, it is hard to retrieve items related to niche interests due to the popularity bias (Chen et al., 2020).

User representation is a fundamental design choice in any RS. A key component in capturing multiple interests is a representation that naturally encodes a user's diverse preferences. The most widely used strategy for user modeling is the *single-point user representation (SUR)*, which uses a single point in an item embedding space to represent the user. The user's affinity for an item is obtained using some distance measure (e.g., inner product, cosine similarity) with the point representing the item. However, SUR can limit the accuracy and diversity of item retrieval (Zhang et al., 2023); hence, most RSs generally use high-dimensional embedding vectors (with high computation cost).

To address the limitations of SUR, MaxMF (Weston et al., 2013) adopts a *multi-point user representation (MUR)*, where each user is represented using K points in the embedding space, each reflecting a different "interest". MaxMF uses a constant, uniform K across all users (e.g., $K = 4$), which is somewhat ad hoc and very restrictive. Subsequent research uses other heuristics (Liu et al., 2019; Cen et al., 2020; Wang et al., 2020a;b; Tan et al., 2021; Chen et al., 2021; Liu et al., 2022) or clustering algorithms (Li et al., 2019; Pal et al., 2020) to determine the number of interests per user. However, these all require the manual choice of K or a specific clustering threshold, limiting the adaptability of MUR methods, since interests generally have high variability across users. Moreover, uncertainty regarding a user's interests is not well-modeled by these methods, diminishing their ability to perform effective online exploration.

To address limitations of SUR and MUR point-based representations, we propose a user representation that emphasizes (i) *adaptability*, adapting to different interest patterns; (ii) *uncertainty-awareness*, modeling uncertainty in assessing user interests; and (iii) *efficiency*, avoiding high-dimensional embeddings. Specifically, we use a *density-based* represen-

[1]School of Computer Science, McGill University, Montreal, Canada. [2]Mila - Quebec AI Institute, Montreal, Canada. [3]Google Research, Mountain View, USA. [4]Google Research, Pittsburgh, USA. Correspondence to: Haolun Wu (This work was done when Haolun Wu was an intern at Google Research) <haolun.wu@mail.mcgill.ca>, Ofer Meshi <meshi@google.com>.

preprint version.

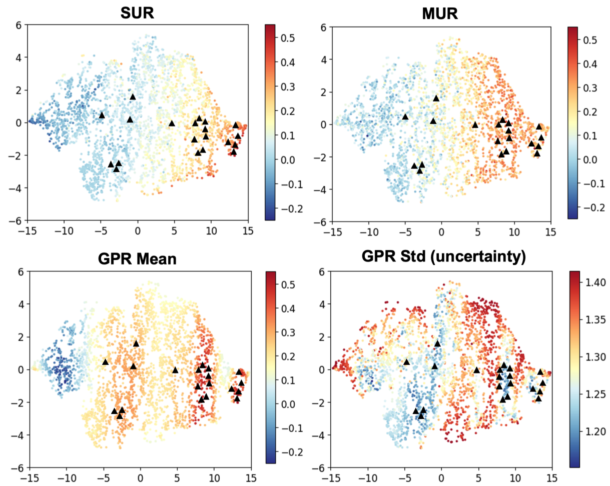


Figure 1: The t-SNE visualization of the prediction score between a picked user to all items in the MovieLens dataset. The score is computed as the inner-product between the user embedding and item embedding. The triangles (▲) indicate the latest 20 items interacted by the user. We use Matrix Factorization (MF) to obtain embeddings in this toy example. As depicted, only density-based method (bottom row) can well capture user interests with uncertainty.

tation, where the user's preferences are encoded using a function over the item embedding space. Under this representation, the relevance score for user-item pairs should be higher in regions of embedding space where a user demonstrated more interest in the corresponding items, and lower in regions where users have shown limited interest. We propose the *density-based user representation (DUR)*, a novel user modeling method which exploits *Gaussian process regression (GPR)* (Rasmussen, 2004; Rasmussen & Williams, 2005), a well-studied Bayesian approach to non-parametric regression using *Gaussian processes (GPs)* to extrapolate from training to test data. GPR has been applied across a wide range of domains, though it has been under-explored in user modeling. Given a sequence of user interactions, GPR predicts the probability of a user's interest in any item using its posterior estimates. This allows us to maintain a unique GP regressor for each user, effectively capturing their evolving preferences and assessing uncertainty. Top-N item retrieval is performed using bandit algorithms, such as UCB (Auer, 2003) or Thompson sampling (Russo et al., 2018), based on GPR posterior estimates.

To illustrate, consider Fig. 1, which shows the prediction score (*i.e.*, the inner-product between user and item embeddings) between a user and all movies from the MovieLens 1M dataset (Harper & Konstan, 2015) (reduced to 2D for visualization purposes). We examine 20 movies from the recent history of a particular user, shown as triangles (▲). We see that these movies lie in several different regions of the embedding space. However, when we fit either SUR or MUR ($K = 4$) models, they fail to capture the user's multiple interests and instead assign high scores only to movies from a single region (Fig. 1, top row). By contrast, we see in Fig. 1 (bottom left) that GPR fits the data well, assigning high values to all regions associated with the user's recent watches (interests). Fig. 1 (bottom right) shows that our approach can also capture uncertainty in our estimates of a user's interests, assigning high uncertainty to regions in embedding space with fewer samples.

Our approach has various desirable properties. First, it adapts to different interest patterns, since the number of interests for any given user is not set manually, but determined by GPR, benefiting from the non-parametric nature of GPs. Second, the Bayesian nature of GPs measures uncertainty with the posterior standard deviation of each item. This supports the incorporation of bandit algorithms in the recommendation and training loop to address the *exploration-exploitation trade-off* in online settings. Finally, our method can effectively retrieve items spanning multiple user interests, including "niche" interests, while using a lower-dimensional embedding relative to SUR and MUR.

To summarize, our work makes the following contributions:

- We develop *GPR4DUR*, a density-based user representation method, for personalized multi-interest retrieval. It is the first use of GPR for user modeling in this setting.
- We propose new evaluation protocols and metrics for multi-interest retrieval that measure the extent to which a model captures a user's multiple interests.
- We conduct comprehensive experiments on real-world offline datasets showing the adaptability and efficiency of GPR4DUR. Online experiments with simulated users show the value of GPR4DUR's uncertainty representation in balancing exploration and exploitation.

2. Related Work

Learning high-quality user representations is central to good RS performance. The *single-point user representation (SUR)* is the dominant approach, where a user is captured by a single point in some embedding space (Ricci et al., 2011; Rendle et al., 2009), for example, as employed by classical (Koren, 2008; Rendle, 2010; Chen et al., 2018) and neural (He et al., 2017) collaborative filtering (CF) methods. While effective and widely used, SUR cannot reliably capture a user's multiple interests.

To address this limitation, the *multi-point user representation (MUR)* (Weston et al., 2013) has been proposed, where a user is represented by multiple points in embedding space, each corresponding to a different "primary" interest. Selecting a suitable number of points K is critical in MUR. Existing algorithms largely use heuristic methods, e.g., choosing a global constant K for all users (Weston et al., 2013; Liu et al., 2019; Wang et al., 2020a; Cen et al., 2020; Wang et al., 2020b; Tan et al., 2021; Chen et al., 2021; Liu et al., 2022). Other methods personalize K by setting it to the logarithm of the number of items with which a user has interacted (Li et al., 2019). More recently, Ward clustering of a user's past items has been proposed, with a user's K determined by the number of such clusters (Pal et al., 2020). This too requires manual tuning of clustering thresholds.

At inference time MUR is similar to SUR, computing the inner-product of the user embedding(s) and item embedding. Some methods compute K inner products, one per interest, and use the maximum as the recommendation (and the predicted score for that item) (Weston et al., 2013). Others first retrieve the top-N items for each interest ($N \times K$ items), then recommend the top-N items globally (Cen et al., 2020; Pal et al., 2020). None of these methods capture model uncertainty w.r.t. a user's interests, hence they lack the ability to balance exploration and exploitation in online recommendation in a principled way (Chen, 2021).

Our density-based user representation, and our proposed GPR4DUR, differs from prior work w.r.t. both problem formulation and methodology. Most prior MUR methods focus on *next-item prediction* (Cen et al., 2020; Liu et al., 2022), implicitly assuming a single-stage RS, where the trained model is the main recommendation engine. However, many practical RSs consist of two stages: *candidate selection* (or *retrieval*) followed by *ranking* (Covington et al., 2016b; Eksombatchai et al., 2018). This naturally raises the question: ***are the selected candidates diverse enough to cover a user's interests or intents?*** This is especially relevant when a user's dominant interest at the time of recommendation is difficult to discern with high probability; hence, it is important that the ranker have access to a diverse set of candidates that cover the user's *range of potential currently active interests*. In this paper, we focus on this *retrieval task*. As for methodology, almost all prior work uses SUR or MUR point-based representations, (Weston et al., 2013; Pal et al., 2020; Cen et al., 2020; Li et al., 2019)—these fail to satisfy all the desiderata outlined in Sec. 1. Instead, we propose DUR), a novel method satisfying these criteria, and, to the best of our knowledge, the first to adopt GPR for user modeling in multi-interest recommendation/retrieval.

A related non-parametric recommendation approach is k-nearest-neighbors (kNN), where users with similar preferences are identified (e.g., in user embedding space), and their ratings generate recommendations (see, e.g., Grcar (2004)). Our approach differs by not using user embeddings, but fitting a GPR model directly to item embeddings (see Sec. 4.4), allowing for uncertainty in the user model.

3. Formulation and Preliminaries

In this section, we outline our notation and multi-interest retrieval problem formulation, and provide some background on GPR, which lies at the core of our DUR model.

Notation. We consider a scenario where each recommended item is associated with *category* information (e.g., genre for movies). Denote the set of all *users*, *items*, and *categories* by $\mathcal{U}$, $\mathcal{V}$, and $\mathcal{C}$, respectively. For each $u \in \mathcal{U}$, whose interaction history has length l_u, we partition the sequence of items $\mathcal{V}_u$ in u's history into two disjoint lists based on the interaction timestamp (which are monotonic increasing): (i) the *history set* $\mathcal{V}_u^{\mathrm{h}} = [v_{u,1}, v_{u,2}, ..., v_{u,\ell_u}]$ serves as the model input; and (ii) the *holdout set* $\mathcal{V}_u^{\mathrm{d}} = [v_{u,\ell_u+1}, v_{u,\ell_u+2}, ..., v_{u,l_u}]$ is used for evaluation. We define u's *interests* $\mathcal{C}(\mathcal{V}_u)$ to be the *set of categories associated with all items in u's history*. Our notation is summarized in Table 4 in the appendix.

Problem Formulation. We formulate the *multi-interest retrieval problem* as follows: given $\mathcal{V}_u^{\mathrm{h}}$, we aim to retrieve the top N items $\mathcal{R}_u$ (i.e., $|\mathcal{R}_u| = N$) w.r.t. some *matching metric* connecting $\mathcal{R}_u$ and $\mathcal{V}_u^{\mathrm{d}}$ that measures personalized retrieval performance. We expect $\mathcal{R}_u$ to contain relevant items, and given our focus on multi-interest retrieval, $\mathcal{R}_u$ should ideally cover all categories in $\mathcal{C}(\mathcal{V}_u^{\mathrm{d}})$.

We note that our problem is related to *sequential recommendation*, where the input is a sequence of interacted items sorted by the timestamp per user, and the goal is to predict the next item with which the user will interact. However, here we focus on retrieving a set of items that cover a user's diverse interests. We defer the task of generating a precise recommendation list to a downstream ranking model.

GPR. The core of our model is *Gaussian process regression (GPR)*. A *Gaussian Process (GP)* is a non-parametric model for regression and classification. GPR models a *distribution over functions*, providing not only function value estimates for any input, but also uncertainty quantification via predictive variance, enhancing robust decision-making and optimization (Williams & Rasmussen, 1995).

The key components of the GP are the mean function and the covariance (kernel) function, which capture the prior assumptions about the function's behavior and the relationships between input points, respectively (Rasmussen & Williams, 2005). Let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \in \mathbb{R}^{n \times d}$ be a set of input points and $\mathbf{y} = \{y_1, \ldots, y_n\} \in \mathbb{R}^n$ be the corresponding output values. A GP is defined as:

$$f \sim \mathcal{GP}(\mu, k), \tag{1}$$

where $\mu(\mathbf{x})$ is the mean and $k(\mathbf{x}, \mathbf{x}')$ the covariance function. The joint distribution of the observation and the output at a new point is derived using a Gaussian distribution, as illustrated in the Appendix.

4. Methodology

We now outline our density-based user representation using GPR and its application to multi-interest retrieval.

4.1. GPR for Density-based User Representation

We use GPR to construct a novel density-based user representation (DUR) for multi-interest modeling in RSs.

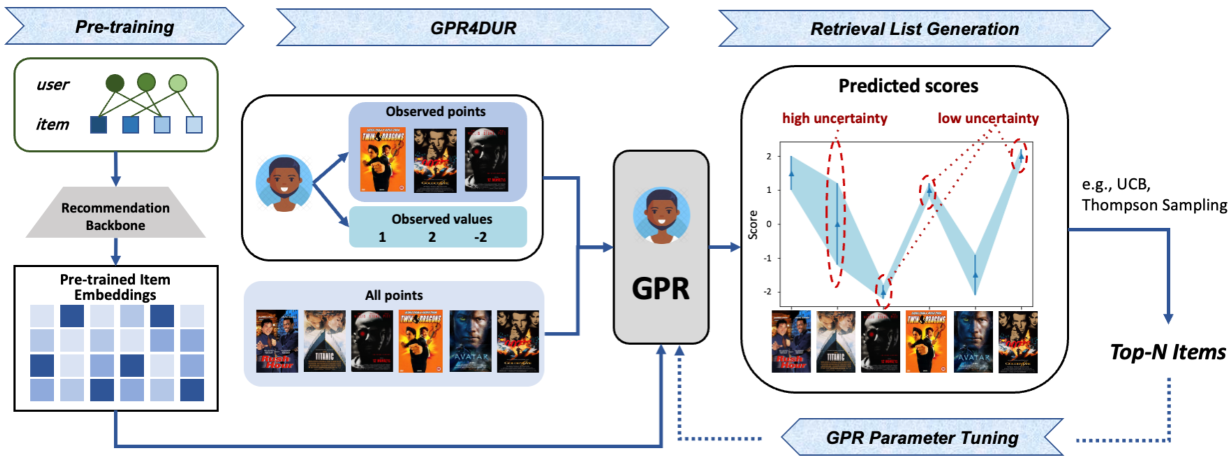


Figure 2: The architecture of GPR4DUR: an example of a movie recommendation for a single user.

Our key insight involves using GPR to learn a DUR, using a user's interaction history, that naturally embodies their diverse interest patterns. For any user u, let $\mathbf{V}_u = [\mathbf{v}_{u,1}, \mathbf{v}_{u,2}, \ldots, \mathbf{v}_{u,l_u}] \in \mathbb{R}^{l_u \times d}$ be the embeddings of all items in their history. This is derived from their interaction list $\mathcal{V}_u$ and an item embedding matrix $\mathbf{V}$ (we describe how to obtain $\mathbf{V}$ in Sec. 4.4). Let $\mathbf{o}_u = [o_{u,v} | v \in \mathcal{V}_u] \in \mathbb{R}^{l_u}$ be the vector of u's *observed interactions* with items in $\mathcal{V}_u$. We employ a GP to model u's interests given the input points $\mathbf{V}_u$ and corresponding observations $\mathbf{o}_u$ (analogous to Eq. 1), $g_u \sim \mathcal{GP}(\mu_u, k_u)$, where, g_u, μ_u and k_u are u's personalized GP regressor, mean, and kernel function, respectively. The joint distribution of observation $\mathbf{o}_u$ and the predicted observation of a novel item v_* is:

$$\begin{bmatrix} \mathbf{o}_u \\ g_u(\mathbf{v}_*) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}_u(\mathbf{V}_u) \\ \mu_u(\mathbf{v}_*) \end{bmatrix}, \begin{bmatrix} \mathbf{K}(\mathbf{V}_u, \mathbf{V}_u) + \sigma^2 \mathbf{I} & \mathbf{k}(\mathbf{V}_u, \mathbf{v}_*) \\ \mathbf{k}(\mathbf{v}_*, \mathbf{V}_u) & k(\mathbf{v}_*, \mathbf{v}_*) + \sigma_*^2 \end{bmatrix} \right). \tag{2}$$

For simplicity, we assume $\boldsymbol{\mu}_u = \mathbf{0}$ and a commom kernel function and variance across all users. Assuming implicit feedback, we have $\mathbf{o}_u = \mathbf{1}$, i.e., u shows "interest" in all interacted items. Thus, the posterior (prediction) $g_u(\mathbf{v})$ for any $v \in \mathcal{V}$ is:

$$g_u(\mathbf{v}) | \mathbf{o}_u \sim \mathcal{N}(\bar{g}_u, \mathrm{cov}(g_u)), \tag{3}$$

where the GP mean $\bar{g}_u$ and variance $\mathrm{cov}(g_u)$ are:

$$\bar{g}_u = \mathbf{k}(\mathbf{v}, \mathbf{V}_u)[\mathbf{K}(\mathbf{V}_u, \mathbf{V}_u) + \sigma^2 \mathbf{I}]^{-1} \mathbf{o}_u, \tag{4}$$

$$\begin{aligned} \mathrm{cov}(g_u) = & k(\mathbf{v}, \mathbf{v}) + \sigma_*^2 \\ & - \mathbf{k}(\mathbf{v}_*, \mathbf{V}_u)[\mathbf{K}(\mathbf{V}_u, \mathbf{V}_u) + \sigma^2 \mathbf{I}]^{-1} \mathbf{k}(\mathbf{V}_u, \mathbf{v})^T. \end{aligned} \tag{5}$$

4.2. Retrieval List Generation

After obtaining a DUR g_u for $u \in \mathcal{U}$ using GPR, we generate the *retrieval list* using the posterior $g_u(\mathbf{v})$ over all unobserved items. The top N items with the highest values form our retrieval list. We consider two methods for selection: (i) *Thompson sampling (TS)*, a probabilistic method that selects items based on posterior sampling (Russo et al., 2018) and (ii) *Upper Confidence Bound (UCB)*, a deterministic method that selects items based on their estimated rewards and uncertainties (Auer, 2003; Auer et al., 2002).

4.3. GPR Parameter Tuning

The free parameters in our model are the kernel function $\mathbf{K}$ and variance σ^2 in Eq. 2. We treat these as hyperparameters of GPR which are optimized using evaluation on a separate holdout set. Specifically, to generate $\mathcal{R}_u$ for user u, we fit the GPR model not to the complete interaction history $\mathcal{V}_u$, but to the reduced history $\mathcal{V}_u^{\mathrm{h}}$, using the item embeddings and observed ratings. We assess retrieval performance using specific metrics (see Sec. 5.3) on both $\mathcal{R}_u$ and the holdout set $\mathcal{V}_u^{\mathrm{d}}$. We tune GPR parameters using these criteria (we detail the adjustable parameters in Sec. 5.2).

4.4. Item Embedding Pre-training

Following Pal et al. (2020), we assume that item embeddings are fixed and precomputed: this ensures rapid computation and real-time updates at serving time. For item embedding pre-training, we use *extreme multi-class classification*, a method widely used in prior work on sequential recommendation (Cen et al., 2020; Kang & McAuley, 2018; Hidasi et al., 2016): given a training sample (u_i, v_j), we first compute the likelihood of u_i interacting with v_j, i.e., $p(v_j|u_i) = \exp(\mathbf{u}_i^\intercal \mathbf{v}_j) / \sum_{v' \in \mathcal{V}} \exp(\mathbf{u}_i^\intercal \mathbf{v}')$, where $\mathbf{u}_i$ and $\mathbf{v}_j$ are embeddings of u_i and v_j. Our objective is to maximize the log-likelihood of a user interacting with their items. Moreover, we want to ensure that item embeddings align with their categories since categories explicitly indicate a user's interests. We capture item-category information by computing the likelihood that an item belongs to a category in a similar way: $p(c_k|v_j) = \exp(\mathbf{v}_j^\intercal \mathbf{c}_k) / \sum_{c' \in \mathcal{C}} \exp(\mathbf{v}_j^\intercal \mathbf{c}')$.

The overall objective for pre-training combines the two negative log-likelihoods using a scaling factor γ:

$$\mathcal{L} = -\sum_{u_i} \sum_{v_j \in \mathcal{V}_{u_i}} \log p(v_j|u_i) - \gamma \sum_{v_j} \sum_{c_k \in \mathcal{C}_{v_j}} \log p(c_k|v_j), \tag{6}$$

where $\mathcal{V}_{u_i}$ is the set of items that u_i interacted with and $\mathcal{C}_{v_j}$ is the set of categories that v_j belongs to. The full item embedding matrix $\mathbf{V}$ is jointly learned and fixed after the pre-training phase. Notice that we do not use the user and category embeddings after pre-training, and other schemes for computing item embeddings are possible. For example, an alternative approach is to use item co-occurrence information to derive item embeddings only, without user or category embeddings. The full architecture for GPR4DUR is depicted in Figure 2.

5. Offline Experiments

We next evaluate our GPR4DUR on three real-world datasets, and compare our DUR technique with other state-of-the-art methods for multi-interest retrieval.

5.1. Datasets

We use three widely studied datasets: *Amazon* (He & McAuley, 2016), *MovieLens* (Harper & Konstan, 2015), and *Taobao* (Zhu et al., 2018). We adopt the same setting as previous work (Li et al., 2020; Wang et al., 2019), filtering out items with fewer than ten appearances in the dataset, and users with fewer than 25 item interactions. This ensures users have sufficient history to indicate multiple interests. Each interest corresponds to a category: for Amazon, we use the single principle category of each item; for MovieLens, the 18 movie genres are categories—each movie can belong to multiple categories; and for Taobao, we cluster items into 20 categories using Ward clustering on the pretrained item embeddings, a technique shown to be effective by Pal et al. (2020). We use the same categories during training and inference. Dataset statistics shown in Table 5 (See Appendix). An ablation analysis of various clustering algorithms and numbers of clusters is shown in Table 6 (See Appendix).

5.2. Experiment Setup

Most prior work on recommendation and retrieval evaluates model performance on generalization to new items per user. Instead, similar to recent work (Liang et al., 2018a; Cen et al., 2020), we assess model performance on a more challenging task: its ability to generalize to new users. We split users into disjoint subsets: *training users* ($\mathcal{U}^{\text{train}}$), *validation users* ($\mathcal{U}^{\text{val}}$), and *test users* ($\mathcal{U}^{\text{test}}$) at a ratio of 8:1:1. We treat the last 20% of each user's full interaction sequence as a *holdout set* for evaluation, and the first 80% as a *history set* used to fit the GPR model. We cap the history length to 60, 160, and 100 for the three datasets, aligning with the average count of user interactions per dataset.

For the item embedding pre-training phase, we train the recommendation backbones using the *history set* of all training users and tune the parameters based on the performance on the *holdout set* for validation users. The maximum number of training iterations is 100,000. Early stopping is used if performance on the validation set does not improve for 50 successive iterations. We tune GPR hyperparameters by fitting the GP regressor to the *history set* of all training and validation users, to learn their user representations, then tune the parameters (i.e., kernel k, variance σ) using the *holdout set*. To compare all methods fairly, we report metrics on the holdout set for all test users.

5.3. Metrics

Metrics used in prior work on sequential recommendation are not well-suited to assess performance in our multi-interest retrieval task, for several reasons: (i) Conventional metrics, such as *precision* and *recall*, often employed in multi-interest research, do not adequately quantify whether an item list reflects the full range of a user's (multiple) interests. A model may primarily recommend items from a narrow range of highly popular categories and still score high on these metrics while potentially overlooking less popular or niche interests. (ii) Metrics like precision and recall are overly stringent, only recognizing items in the recommendation list that appear in the holdout set. We argue that credit should also be given if a similar, though not identical, item is recommended (e.g., *Iron Man 1* instead of *Iron Man 2*). This requires a soft version of these metrics. (iii) Multi-interest retrieval systems should expose users to niche content to address their diverse interests. However, conventional metrics may neglect the item perspective, potentially underserving users with specialized interests. Consequently, we propose the use of the following four metrics that encompassing the aforementioned factors.

Interest-wise Coverage (IC) This metric is similar to *subtopic-recall* (Zhai et al., 2003), which directly measures whether the model can comprehensively retrieve all user interests reflected in the holdout set. The higher the value of this metric the better:

$$\text{IC}@k = \frac{1}{|\mathcal{U}^{\text{test}}|} \sum_{u \in \mathcal{U}^{\text{test}}} \frac{|\mathcal{C}(\mathcal{V}_u^{\text{d}}) \cap \mathcal{C}(\mathcal{R}_u^{1:k})|}{|\mathcal{C}(\mathcal{V}_u^{\text{d}})|}. \tag{7}$$

Interest-wise Relevance (IR) To further measure the relevance of retrieved items, we introduce a "soft" recall metric, calculating the maximum cosine similarity between items in the retrieval list and the holdout set within the same category. The motivation for IR is that the success of a retrieval or recommendation list often depends on how satisfying the most relevant item is:

$$\text{IR}@k = \frac{1}{|\mathcal{U}^{\text{test}}|} \sum_{u \in \mathcal{U}^{\text{test}}} \frac{\sum_{c \in \mathcal{C}(\mathcal{V}_u^{\text{d}})} \max_{v_i \in \mathcal{V}_u^{\text{d}}, v_j \in \mathcal{R}_u^{1:k}} S(v_i, v_j)}{|\mathcal{C}(\mathcal{V}_u^{\text{d}})|}, \tag{8}$$
$$\text{s.t. } \mathcal{C}(v_i) = \mathcal{C}(v_j) = c,$$

where $S(v_i, v_j)$ is the cosine similarity between item v_i and v_j. To obtain ground-truth similarities between items, and to mitigate the influence of the chosen pre-trained model, we pretrain the item embeddings using YoutubeDNN (Covington et al., 2016a) with a higher dimension size ($d = 256$) to compute a uniform $\mathbf{S}_{i,j}$ for any backbone. A higher value of this metric is better.

Exposure Deviation (ED) In addition to measuring performance from the user side, we also measure from the item side to test whether exposure of different categories in the retrieval list is close to that in the holdout set. We treat each occurrence of an item category as one unit of exposure, and compute the normalized exposure vectors $\epsilon_u^*, \epsilon_u^{1:k} \in \mathbb{R}^{|\mathcal{C}|}$

	Methods	Interest Coverage (IC@k) *The higher the better* ⇑			Interest Relevance (IR@k) *The higher the better* ⇑			Exp. Deviation (ED@k) *The lower the better* ⇓			Tail Exp. Improv. (TEI@k) *The higher the better* ⇑		
		k=20	k=50	k=100	k=20	k=50	k=100	k=20	k=50	k=100	k=20	k=50	k=100
Amazon	♣ Random	0.690	0.888*	**0.961**	0.251	0.428	0.558	0.513	0.483	0.472	-0.041	-0.041	-0.041
	♣ MostPop	0.704*	0.766	0.788	0.324	0.426	0.490	0.563	0.501	0.485	-0.045	-0.045	-0.045
	♦ YoutubeDNN	0.672	0.810	0.878	**0.432**	0.550*	0.623*	0.470*	0.439	0.423	**-0.040**	-0.041	-0.040
	♦ GRU4Rec	0.676	0.810	0.884	0.415	0.524	0.602	0.485	0.452	0.438	-0.040	-0.040	-0.040
	♠ MIND	0.650	0.787	0.861	0.390	0.503	0.575	0.509	0.477	0.461	-0.041	-0.041	-0.041
	♠ DGCF	0.632	0.705	0.829	0.348	0.441	0.521	0.564	0.506	0.493	-0.077	-0.087	-0.090
	♠ ComiRec	0.656	0.785	0.861	0.399	0.510	0.595	0.492	0.453	0.432	-0.040	-0.040	-0.040*
	♠ CAMI	0.640	0.710	0.833	0.373	0.483	0.521	0.522	0.493	0.473	-0.071	-0.088	-0.080
	♥ **GPR4DUR**	**0.739**	**0.895**	0.956	0.429	**0.560**	**0.643**	**0.458**	**0.423**	**0.412**	-0.041	**-0.039**	**-0.039**
MovieLens	♣ Random	0.835	0.961	**0.992**	0.272	0.422	0.534	0.252	0.229	0.221	-0.185	-0.184	-0.183
	♣ MostPop	0.914*	0.973*	0.986	0.498	0.655	0.727	0.261	0.223	0.217	-0.022	-0.013*	-0.028*
	♦ YoutubeDNN	0.879	0.938	0.974	0.722*	0.846*	0.873*	0.250	0.229	0.218	-0.017*	-0.031	-0.051
	♦ GRU4Rec	0.832	0.939	0.976	0.646	0.784	0.863	0.228	0.226	0.195	-0.070	-0.076	-0.084
	♠ MIND	0.869	0.951	0.981	0.653	0.786	0.863	0.245	0.223	0.212	-0.049	-0.058	-0.073
	♠ DGCF	0.832	0.942	0.958	0.621	0.743	0.821	0.278	0.246	0.206	-0.097	-0.097	-0.103
	♠ ComiRec	0.844	0.946	0.981	0.635	0.776	0.859	**0.227**	**0.203**	**0.192**	-0.069	-0.074	-0.082
	♠ CAMI	0.853	0.933	0.954	0.625	0.742	0.830	0.272	0.263	0.253	-0.082	-0.094	-0.101
	♥ **GPR4DUR**	**0.929**	**0.974**	0.973	**0.825**	**0.862**	**0.891**	0.252	0.222	0.201	**-0.011**	**-0.007**	**-0.016**
Taobao	♣ Random	0.302	0.563	0.873*	0.232	0.416	0.527	0.493	0.425	0.333	-0.059	-0.049	-0.031
	♣ MostPop	0.342*	0.583*	0.863	0.362	0.439	0.643	0.512	0.437	0.343	-0.076	-0.050	-0.036
	♦ YoutubeDNN	0.305	0.523	0.822	0.471	0.529	0.713	0.498	0.443	0.356	-0.054	-0.044	-0.030
	♦ GRU4Rec	0.295	0.503	0.810	0.492	0.533	0.724	0.469	0.413	0.300	-0.053	-0.041	-0.030
	♠ MIND	0.295	0.517	0.813	0.502	0.552	0.733	0.463	0.411	0.294	-0.052	-0.041	-0.029
	♠ DGCF	0.271	0.493	0.793	0.502	0.551	0.730	0.427	0.378	0.289	-0.051	-0.039	-0.028
	♠ ComiRec	0.284	0.501	0.807	0.509	0.561	0.731	0.433	0.380	0.291	-0.051	-0.042	-0.030
	♠ CAMI	0.296	0.522	0.814	0.518*	0.573*	0.734*	**0.424**	**0.363**	0.281	-0.052	**-0.039**	-0.027*
	♥ **GPR4DUR**	**0.363**	**0.601**	**0.891**	**0.609**	**0.624**	**0.781**	0.453	0.367	**0.273**	**-0.050**	-0.040	**-0.023**

Table 1: Result comparison on the retrieval task. For the same metric on each dataset, the best is **bold** and the second best is underlined. We use four different symbols to indicate the different categories of methods detailed in Sec. 5.4. Our model has a statistical significance for $p \leq 0.01$ compared to the best baseline (labelled with *) based on the paired t-test.

for u's holdout set and retrieval list, respectively. Lower values of this metric are better.

$$\mathrm{ED}@k = \frac{1}{|\mathcal{U}^{\text{test}}|} \sum_{u \in \mathcal{U}^{\text{test}}} ||\epsilon_u^* - \epsilon_u^{1:k}||_2^2,$$

$$\text{s.t., } \epsilon_{u,c}^* = \frac{\sum_{v \in \mathcal{V}_u^{\text{d}}} \mathbb{1}_{c \in \mathcal{C}(v)}}{\sum_{v \in \mathcal{V}_u^{\text{d}}} |\mathcal{C}(v)|}, \epsilon_{u,c}^{1:k} = \frac{\sum_{v \in \mathcal{R}_u^{1:k}} \mathbb{1}_{c \in \mathcal{C}(v)}}{\sum_{v \in \mathcal{R}_u^{1:k}} |\mathcal{C}(v)|}. \tag{9}$$

Tail Exposure Improvement (TEI) With respect to category exposure, it is crucial to ensure that niche interests are not under-exposed. To evaluate this, we select a subset of the least popular categories and measure their exposure improvement in the retrieval list versus that in the holdout set. A higher value indicates better performance, and a positive value indicates improvement:

$$\mathrm{TEI}@k = \frac{1}{|\mathcal{U}^{\text{test}}|} \sum_{u \in \mathcal{U}^{\text{test}}} \sum_{c \in \mathcal{C}^{\text{tail}}} (\epsilon_{u,c}^{1:k} - \epsilon_{u,c}^*) \mathbb{1}_{\epsilon_{u,c}^* > 0}. \tag{10}$$

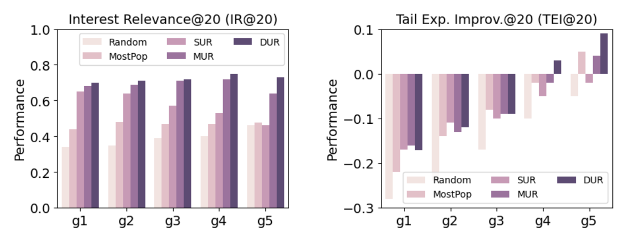
Here, $\mathcal{C}^{\text{tail}}$ refers to the set of niche categories (i.e., the last 50% long-tail categories), denoting those niche interests. $\mathbb{1}_{\epsilon_{u,c}^* > 0}$ indicates that we only compute the improvement for categories that appear in the user's holdout set, reflecting their true interests.

5.4. Methods Studied

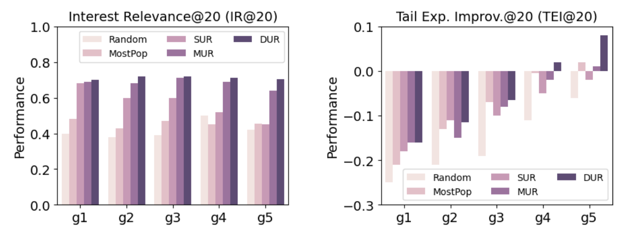
We study the following nine methods from four categories.

(i) Heuristic Methods ♣: **Random** recommends random items, **MostPop** recommends the most popular items.

(ii) SUR Methods ♦: **YoutubeDNN** (Covington et al., 2016a) is one of the most successful deep learning models for industrial recommendation platforms. **GRU4Rec** (Hidasi et al., 2016) is the first work to use recurrent neural networks for recommendation.



(a) Users grouped by number of interactions



(b) Users grouped by number of interests

Figure 3: Methods comparison across different user groups on MovieLens. Best viewed in color.

(iii) MUR Methods ♠: **MIND** (Li et al., 2019) designs a multi-interest extractor layer based on the capsule routing, which is applicable for clustering past behaviors and extracting diverse interests. **DGCF** (Wang et al., 2020a) investigates the fine-grained user intention to enhance the ability of collaborative filtering by disentangling multiple latent factors for user representation. **ComiRec** (Cen et al., 2020)

	Amazon		*MovieLens*		*Taobao*	
	Recall	nDCG	Recall	nDCG	Recall	nDCG
♣ Random	0.836	0.848	0.685	0.716	0.838	0.866
♣ MostPop	0.867	0.867	0.710	0.741	0.868	0.896
♦ YoutubeDNN	0.875	0.888	0.715	0.748	0.875	0.904
♦ GRU4Rec	0.873	0.886	0.714	0.746	0.873	0.902
♠ MIND	0.842	0.885	0.713	0.745	0.872	0.901
♠ DGCF	0.867	0.880	0.709	0.741	0.867	0.896
♠ ComiRec	0.887	0.900	**0.731**	0.757	0.886	0.915
♠ CAMI	0.892	0.905	0.725	0.761	0.891	0.920
♥ **GPR4DUR**	**0.908**	**0.922**	0.730	**0.768**	**0.906**	**0.936**

Table 2: Result comparison on the ranking task measured at top-50 across all methods on three datasets.

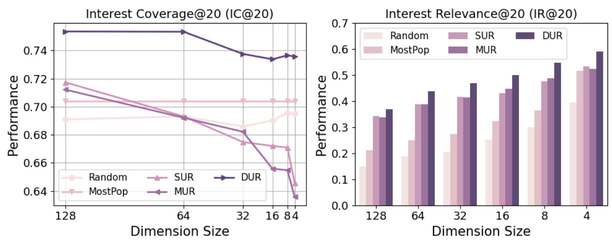


Figure 4: Robustness comparison across different dimension sizes on Amazon. Best viewed in color.

is one of the SOTA methods that captures multiple interests from user behavior sequences with a controller for balancing diversity. **CAMI** (Liu et al., 2022) uses a category-aware multi-interest model to encode users as multiple preference embeddings to represent user-specific interests.

(iv) DUR Method (Ours) ♥: **GPR4DUR** uses GPR as a density-based user representation tool for capturing users' diverse interests with uncertainty.

Overall Performance Our overall performance comparison addresses two central research questions (*RQs*): whether our proposed method, GPR4DUR, offers superior retrieval performance for users with multiple interests (*RQ1*); and whether it induces appropriate item-sided exposure w.r.t. both popular and niche interests (*RQ2*).

To assess *RQ1*, we evaluate the effectiveness of the retrieval phase. As shown in Table 1, GPR4DUR outperforms almost all baselines across all datasets w.r.t. Interest Coverage (IC@k) and Interest Relevance (IR@k), demonstrating high degrees of retrieval coverage and relevance, respectively. Specifically, on the Amazon dataset, GPR4DUR achieves the highest performance in 4 out of 6 interest metrics, in MovieLens it leads in 5 out of 6 metrics, and in Taobao, it excels across all 6 interest metrics. This shows that GPR4DUR covers a wide range of user interests while simultaneously maintaining excellent relevance.

For *RQ2*, our objective is to ascertain whether item exposure is suitably balanced. We measure this using the Exposure Deviation (ED@k) and Tail Exposure Improvement (TEI@k) metrics. Lower values of ED@k suggest more satisfying category exposure, while higher values of TEI@k are indicative of enhanced exposure in the long tail of item categories. GPR4DUR is highly effective on these metrics, consistently performing well across all datasets in most cases, and validating its ability to provide an optimal level of category exposure. Notably, GPR4DUR achieves the best results in eight of nine TEI@k metrics, indicating its superior ability to generate exposure to niche categories/interests. We point out, however, that all TEI@k values are negative, which suggests that none of the methods we tested improve exposure for niche categories relative to the exposure in the user holdout sets. This is explained by the well-known popularity bias inherent in most recommendation methods, and suggests that further work is required on diversification strategies to mitigate such effects.

While our primary emphasis is on the retrieval phase, we also present model comparisons for the ranking task. As illustrated in Table 2, GPR4DUR demonstrates competitive performance on the traditional relevance metrics relative to the baseline models. This further validates the strength of our proposed method.

Performance across User Groups *(RQ3)* We conduct a fine-grained analysis of GPR4DUR performance on users grouped into quantiles, $g1$ (lowest) through $g5$ (highest), based on the number of interaction or number of interests; results are provided in Fig. 3. Due to space constraints, we show results only for IR@20 and TEI@20 on MovieLens, and only plot the *best* SUR and MUR strategies; DUR denotes our GPR4DUR method. We see that GPR4DUR consistently outperforms the baselines w.r.t. relevance and exposure metrics. This improvement is more pronounced as the number of user interactions (resp., interests) increases. Interestingly, while no model enhances overall exposure of niche interests (see Table 1), GPR4DUR improves niche interest exposure considerably for users with large numbers of interactions (resp., interests); i.e., TEI@20 is positive for $g4$ and $g5$. These observations confirm the effectiveness of GPR4DUR in capturing a user's multiple interests—especially for those with non-trivial histories—and its potential to maintain fair exposure w.r.t. items.

Robustness to Dimension Size *(RQ4)* We underscored the importance of efficiency for a good user representation in Sec. 1. To shed light on this, we examine the IC@20 and IR@20 performance of various methods on the Amazon dataset. As shown in Fig. 4, GPR4DUR is very effective w.r.t. interest coverage even when operating with low-dimensional embeddings (i.e., d = 8 and d = 4). By contrast, the performance of the SUR and MUR methods degrades as the dimensionality decreases. GPR4DUR consistently outperforms other methods w.r.t. interest relevance across all dimensionalities examined. We note that lower dimensions facilitate higher interest relevance due to increased cosine similarity (Eq. 8), explaining the inverse correlation between IR@20 and dimension size in Figure 4 (right).

6. Online Simulation

To demonstrate the efficacy of GPR4DUR in capturing uncertainty in user interests and support exploration, we conduct an online simulation in a synthetic setting, using a specific model of stochastic user behavior to generate responses to recommendations.

Data Preparation. We assume $|\mathcal{C}| = 10$ interest clusters, each represented by a d-dim ($d = 32$) multivariate Gaussian. We randomly select a (user-specific) subset of these interests as the ground-truth interest set for each user. We set $|\mathcal{U}| = 1000$ and $|\mathcal{V}| = 3000$, with 300 items in each interest cluster. Each item belongs to a single cluster and its embedding is sampled from the corresponding interest distribution. Each user is modeled by a multi-modal Gaussian, a weighted sum of the corresponding ground-truth interest distributions. To simulate a sequence of item interactions $\mathcal{V}_u$ (i.e., user history), we follow Mehta et al. (2023), first running a Markov Chain using a predefined user interest transition matrix to obtain the user's interest interactions for $S = 10$ steps. We do not consider the cold-start problem in this experiment, so we simply recommend one item from each generated cluster to form the user history (i.e., $|\mathcal{V}_u| = S$). The observation $\mathbf{o}_u$ over items in $\mathcal{V}_u$ is set to 1 if the item belongs to a ground-truth cluster, and -1 otherwise.

GPR Fit and Prediction. After obtaining item embeddings $\mathbf{V}$, user history $\mathcal{V}_u$, and user observations $\mathbf{o}_u$, we use GPR4DUR to learn a DUR for each user, using the methods described in Sec. 4.

User History and Observation Update. Using a predetermined user browsing model, clicked and skipped items are generated and appended to the user interaction history (and the corresponding user observation is likewise updated, 1 for clicked, -1 for skipped). This process continues until a maximum iteration of $T = 10$ is reached. In this online setting, we adopt the *dependent click model (DCM)* of user browsing behavior, widely used in web search and recommendation (Cao et al., 2020; Chuklin et al., 2015). In the DCM, users begin by inspecting the top-ranked item, progressing down the list, engaging with items of interest and deciding to continue or terminate after each viewed item.

Experiment Results. We compare different recommendation policies by assessing various metrics at each iteration; due to space limitations, we report results only for interest coverage, see Table 3. Specifically, we compute a "cumulative" version of interest coverage by reporting the interest coverage averaged across all users on all previously recommended items prior to the current iteration. Our goal is to test whether policies that use uncertainty models outperform those that do not, specifically, whether such policies can exploit the inherent uncertainty representation of user interests offered by GPR4DUR.

Policy	t=1	t=2	t=3	t=4	t=5	t=6	t=7	t=8	t=9	t=10
Random	0.29	0.49	0.64	0.74	0.82	0.87	0.91	0.91	0.92	0.93
Greedy	0.71	0.88	0.90	0.90	0.91	0.91	0.91	0.92	0.92	0.92
UCB (β=1)	0.71	**0.89**	**0.91**	**0.91**	**0.92**	**0.92**	**0.92**	**0.93**	0.94	0.95
UCB (β=5)	**0.72**	0.88	0.90	0.90	0.91	0.91	**0.92**	0.92	0.93	0.94
Thompson	0.29	0.50	0.65	0.75	0.82	0.89	0.91	0.92	**0.95**	**0.98**

Table 3: Comparison between different policies in online setting. The reported values are the interest coverage averaged across all users on all *cumulative* recommended items up to each iteration. The highest value per column is bold.

We assume each policy recommends the top-10 items to each user at each iteration. Table 3 shows that methods using uncertainty (the bottom three rows) fairly reliably outperform those that do not (the top two rows, with *Greedy* being UCB with $\beta = 0$, where β is the scaling factor of the variance term in UCB). The observation confirms the benefit of using GPR4DUR to explicitly model uncertainty in a recommenders' estimates of a user's interests and to use this to drive exploration.

7. Conclusion and Discussion

In this paper, we introduced a density-based user representation model, GPR4DUR, marking the first application of Gaussian process regression for user modeling in multi-interest retrieval. This innovative approach inherently captures dynamic user interests, provides uncertainty-awareness, and proves to be more dimension-efficient than traditional point-based methods. We also establish a new evaluation protocol, developing new metrics specifically tailored to multi-interest retrieval tasks, filling a gap in the current evaluation landscape. Offline experiments validate the adaptability and efficiency of GPR4DUR, and demonstrate significant benefits relative to existing state-of-the-art models. Online simulations further highlight GPR4DUR's ability to drive user interest exploration by recommendation algorithms that can effectively leverage model uncertainty.

Discussion. While Gaussian process regression has a computational complexity of $O(N^3)$ with respect to the number of observations N, in our case this complexity is defined w.r.t. to a user's interaction history, not the entire set of items in the domain. In practice, given the typical sparseness of user interaction data relative to the total item count, this cubic complexity is generally manageable. Where needed, an effective way to tackle the computational demands of GPR is to impose a limit on the length of a user's interaction history used by GPR4DUR, using, say, the most recent interactions, or a representative set of interactions, if the length exceeds the threshold. This strategy not only helps to control the computational complexity, but helps ensure the model is primarily influenced by the most recent or relevant user interactions, thus making GPR more applicable and efficient in real-world recommender systems.

For future research, the incorporation of collaborative Gaussian processes presents a tantalizing prospect. Our current model primarily focuses on personalization, using the other users solely for the tuning of GPR hyperparameters. However, we postulate that harnessing the power of collaborative learning (e.g., Houlsby et al. (2012)) could further enhance the performance and effectiveness of our approach.

Impact Statements

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Auer, P. Using confidence bounds for exploitation-exploration trade-offs. *J. Mach. Learn. Res.*, 3:397–422, mar 2003. ISSN 1532-4435.

Auer, P., Cesa-Bianchi, N., and Fischer, P. Finite-time analysis of the multiarmed bandit problem. *Mach. Learn.*, 47(2–3):235–256, may 2002.

Cao, J., Sun, W., jun Max Shen, Z., and Ettl, M. Fatigue-aware bandits for dependent click models. *ArXiv*, abs/2008.09733, 2020.

Cen, Y., Zhang, J., Zou, X., Zhou, C., Yang, H., and Tang, J. Controllable multi-interest framework for recommendation. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pp. 2942–2951. ACM, 2020.

Chen, G., Zhang, X., Zhao, Y., Xue, C., and Xiang, J. Exploring periodicity and interactivity in multi-interest framework for sequential recommendation. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pp. 1426–1433. ijcai.org, 2021.

Chen, J., Dong, H., Wang, X., Feng, F., Wang, M., and He, X. Bias and debias in recommender system: A survey and future directions. *CoRR*, abs/2010.03240, 2020.

Chen, M. Exploration in recommender systems. In *RecSys '21: Fifteenth ACM Conference on Recommender Systems, Amsterdam, The Netherlands, 27 September 2021 - 1 October 2021*, pp. 551–553. ACM, 2021.

Chen, R., Hua, Q., Chang, Y., Wang, B., Zhang, L., and Kong, X. A survey of collaborative filtering-based recommender systems: From traditional methods to hybrid methods based on social networks. *IEEE Access*, 6:64301–64320, 2018.

Chuklin, A., Markov, I., and de Rijke, M. *Click Models for Web Search*. Morgan & Claypool, 2015. ISBN 9781627056489. doi: 10.2200/S00654ED1V01Y201507ICR043.

Covington, P., Adams, J., and Sargin, E. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*, pp. 191–198. ACM, 2016a.

Covington, P., Adams, J., and Sargin, E. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, pp. 191–198, 2016b.

Eksombatchai, C., Jindal, P., Liu, J. Z., Liu, Y., Sharma, R., Sugnet, C., Ulrich, M., and Leskovec, J. Pixie: A system for recommending 3+ billion items to 200+ million users in real-time. In *Proceedings of the 2018 world wide web conference*, pp. 1775–1784, 2018.

Grcar, M. User profiling: Collaborative filtering. In *Proceedings of SIKDD 2004 at Multiconference IS*, pp. 75–78, 2004.

Harper, F. M. and Konstan, J. A. The movielens datasets: History and context. 5(4), 2015. ISSN 2160-6455.

He, R. and McAuley, J. J. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *WWW*, pp. 507–517. ACM, 2016.

He, X., Liao, L., Zhang, H., Nie, L., Hu, X., and Chua, T. Neural collaborative filtering. In Barrett, R., Cummings, R., Agichtein, E., and Gabrilovich, E. (eds.), *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*, pp. 173–182. ACM, 2017.

Hidasi, B., Karatzoglou, A., Baltrunas, L., and Tikk, D. Session-based recommendations with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

Houlsby, N., Huszar, F., Ghahramani, Z., and Hernández-lobato, J. Collaborative gaussian processes for preference learning. *Advances in neural information processing systems*, 25, 2012.

Jin, X. and Han, J. *K-Means Clustering*, pp. 563–564. Springer US, Boston, MA, 2010. ISBN 978-0-387-30164-8. doi: 10.1007/978-0-387-30164-8_425. URL https://doi.org/10.1007/978-0-387-30164-8_425.

Kang, W. and McAuley, J. J. Self-attentive sequential recommendation. In *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17-20, 2018*, pp. 197–206. IEEE Computer Society, 2018.

Koren, Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In Li, Y., Liu, B., and Sarawagi, S. (eds.), *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008*, pp. 426–434. ACM, 2008.

Li, C., Liu, Z., Wu, M., Xu, Y., Zhao, H., Huang, P., Kang, G., Chen, Q., Li, W., and Lee, D. L. Multi-interest network with dynamic routing for recommendation at tmall. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pp. 2615–2623. ACM, 2019.

Li, J., Wang, Y., and McAuley, J. J. Time interval aware self-attention for sequential recommendation. In Caverlee, J., Hu, X. B., Lalmas, M., and Wang, W. (eds.), *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pp. 322–330. ACM, 2020.

Liang, D., Krishnan, R. G., Hoffman, M. D., and Jebara, T. Variational autoencoders for collaborative filtering. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pp. 689–698. ACM, 2018a.

Liang, D., Krishnan, R. G., Hoffman, M. D., and Jebara, T. Variational autoencoders for collaborative filtering. In Champin, P., Gandon, F., Lalmas, M., and Ipeirotis, P. G. (eds.), *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pp. 689–698. ACM, 2018b.

Liu, J., Dou, Z., Zhu, Q., and Wen, J. A category-aware multi-interest model for personalized product search. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pp. 360–368. ACM, 2022.

Liu, N., Tan, Q., Li, Y., Yang, H., Zhou, J., and Hu, X. Is a single vector enough?: Exploring node polysemy for network embedding. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pp. 932–940. ACM, 2019.

Mehta, N., Singh, A., Yi, X., Jain, S., Hong, L., and Chi, E. Density weighting for multi-interest personalized recommendation. In *arxiv eprint: arxiv 2308.01563*, 2023.

Pal, A., Eksombatchai, C., Zhou, Y., Zhao, B., Rosenberg, C., and Leskovec, J. Pinnersage: Multi-modal user embedding framework for recommendations at pinterest. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pp. 2311–2320. ACM, 2020.

Rasmussen, C. E. *Gaussian Processes in Machine Learning*, pp. 63–71. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

Rasmussen, C. E. and Williams, C. K. I. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005. ISBN 026218253X.

Rendle, S. Factorization machines. In Webb, G. I., Liu, B., Zhang, C., Gunopulos, D., and Wu, X. (eds.), *ICDM 2010, The 10th IEEE International Conference on Data Mining, Sydney, Australia, 14-17 December 2010*, pp. 995–1000. IEEE Computer Society, 2010.

Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L. BPR: bayesian personalized ranking from implicit feedback. In *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*, pp. 452–461. AUAI Press, 2009.

Ricci, F., Rokach, L., and Shapira, B. *Introduction to Recommender Systems Handbook*, pp. 1–35. Springer US, Boston, MA, 2011.

Russo, D. J., Van Roy, B., Kazerouni, A., Osband, I., and Wen, Z. A tutorial on thompson sampling. *Found. Trends Mach. Learn.*, 11(1):1–96, jul 2018.

Shi, H., Gu, Y., Zhou, Y., Zhao, B., Gao, S., and Zhao, J. Everyone's preference changes differently: A weighted multi-interest model for retrieval. In *International Conference on Machine Learning*, pp. 31228–31242. PMLR, 2023.

Shi, J. and Malik, J. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

Tan, Q., Zhang, J., Yao, J., Liu, N., Zhou, J., Yang, H., and Hu, X. Sparse-interest network for sequential recommendation. In *WSDM '21, The Fourteenth ACM International Conference on Web Search and Data Mining, Virtual Event, Israel, March 8-12, 2021*, pp. 598–606. ACM, 2021.

Wang, X., He, X., Wang, M., Feng, F., and Chua, T. Neural graph collaborative filtering. In *SIGIR*, pp. 165–174. ACM, 2019.

Wang, X., Jin, H., Zhang, A., He, X., Xu, T., and Chua, T. Disentangled graph collaborative filtering. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pp. 1001–1010. ACM, 2020a.

Wang, Y., Tang, S., Lei, Y., Song, W., Wang, S., and Zhang, M. Disenhan: Disentangled heterogeneous graph attention network for recommendation. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pp. 1605–1614. ACM, 2020b.

Ward, J. H. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963. ISSN 01621459. URL http://www.jstor.org/stable/2282967.

Weston, J., Weiss, R. J., and Yee, H. Nonlinear latent factorization by embedding multiple user interests. In *Seventh ACM Conference on Recommender Systems, RecSys '13, Hong Kong, China, October 12-16, 2013*, pp. 65–68. ACM, 2013.

Williams, C. and Rasmussen, C. Gaussian processes for regression. In Touretzky, D., Mozer, M., and Hasselmo, M. (eds.), *Advances in Neural Information Processing Systems*, volume 8. MIT Press, 1995. URL https://proceedings.neurips.cc/paper_files/paper/1995/file/7cce53cf90577442771720a370c3c723-Paper.pdf.

Zhai, C., Cohen, W. W., and Lafferty, J. D. Beyond independent relevance: methods and evaluation metrics for subtopic retrieval. In *SIGIR*, pp. 10–17. ACM, 2003.

Zhang, T., Ramakrishnan, R., and Livny, M. Birch: an efficient data clustering method for very large databases. *ACM sigmod record*, 25(2):103–114, 1996.

Zhang, X., Liu, J., Chang, S., Gong, P., Wu, Z., and Han, B. Mirn: A multi-interest retrieval network with sequence-to-interest em routing. *PLOS ONE*, 18(2):e0281275, 2023.

Zhu, H., Li, X., Zhang, P., Li, G., He, J., Li, H., and Gai, K. Learning tree-based deep model for recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pp. 1079–1088. ACM, 2018.

A. Appendix

A.1. Notations and Preliminaries

A.1.1. Notations

Notation	Description
$\mathcal{U}, \mathcal{V}, \mathcal{C}$	The set of all users, items, and (item) categories.
$\mathbf{U}, \mathbf{V}, \mathbf{C}$	Full embedding matrix of users, items, and categories.
$\mathcal{V}_u$	The user u's interaction history.
$\mathcal{V}_u^{\mathrm{h}}, \mathcal{V}_u^{\mathrm{d}}$	The history set and holdout set partitioned from $\mathcal{V}_u$.
$\mathbf{o}_u$	The observed rating scores of u on items in $\mathcal{V}_u$.
$t_{u,v}$	The time step when u interacted with v.
l_u	The length of $\mathcal{V}_u$.
ℓ_u	The length of user history for model input.
$\mathbf{V}_u$	Item embeddings for items in $\mathcal{V}_u$.
$\mathcal{R}_u$	The list of retrieved items to u.
$\mathcal{C}(\cdot)$	The set of categories of all items in the input sequence.

Table 4: Description of Notation.

A.1.2. Preliminaries on Gaussian Process Regression

Let $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\} \in \mathbb{R}^{n \times d}$ be a set of input points and $\mathbf{y} = \{y_1, \dots, y_n\} \in \mathbb{R}^n$ be the corresponding output values. A GP is defined as:

$$f \sim \mathcal{GP}(\mu, k), \tag{11}$$

where $\mu(\mathbf{x})$ is the mean function and $k(\mathbf{x}, \mathbf{x}')$ is the covariance function (kernel). Given a new point $\mathbf{x}_*$, the joint distribution of the observed outputs and the output at the new point is given by:

$$\begin{bmatrix} \mathbf{y} \\ f(\mathbf{x}_*) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}(\mathbf{X}) \\ \mu(\mathbf{x}_*) \end{bmatrix}, \begin{bmatrix} \mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I} & \mathbf{k}(\mathbf{X}, \mathbf{x}_*) \\ \mathbf{k}(\mathbf{x}_*, \mathbf{X}) & k(\mathbf{x}_*, \mathbf{x}_*) + \sigma_*^2 \end{bmatrix} \right), \tag{12}$$

where $\boldsymbol{\mu}(\mathbf{X})$ is the vector of mean values for the observed data points, $\mathbf{K}(\mathbf{X}, \mathbf{X})$ is the covariance matrix for the observed data points, $\mathbf{k}(\mathbf{X}, \mathbf{x}_*)$ is the vector of covariances between the observed data points and the new input point, and σ^2 and σ_*^2 are the noise variances. Without prior observations, $\boldsymbol{\mu}$ is generally set as $\mathbf{0}$.

The conditional distribution of $f(\mathbf{x}_*)$ given the observed data is:

$$f(\mathbf{x}_*) | \mathbf{y} \sim \mathcal{N}(\bar{f}_*, \mathrm{cov}(f_*)), \tag{13}$$

with the predictive mean and covariance given by:

$$\bar{f}_* = \mu(\mathbf{x}_*) + \mathbf{k}(\mathbf{x}_*, \mathbf{X})[\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I}]^{-1}(\mathbf{y} - \boldsymbol{\mu}), \tag{14}$$

$$\mathrm{cov}(f_*) = k(\mathbf{x}_*, \mathbf{x}_*) + \sigma_*^2 - \mathbf{k}(\mathbf{x}_*, \mathbf{X})[\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I}]^{-1} \mathbf{k}(\mathbf{X}, \mathbf{x}_*)^T. \tag{15}$$

Fig. 5 presents a visual illustration of GPR. The true underlying function is depicted in red, which is the function we aim to approximate through GPR. The observations, depicted as black crosses, represent known data points. As expected with GPR, where data points are observed, the uncertainty (represented by the shaded region) is minimal, signifying high confidence in predictions at those locations. On the other hand, in areas without observations, the uncertainty increases, reflecting less confidence in the model's predictions. The two dashed lines represent samples from the GP posterior. Around observed points, these sampled functions adhere closely to the actual data, representing the power and flexibility of GPR in modeling intricate patterns based on sparse data.

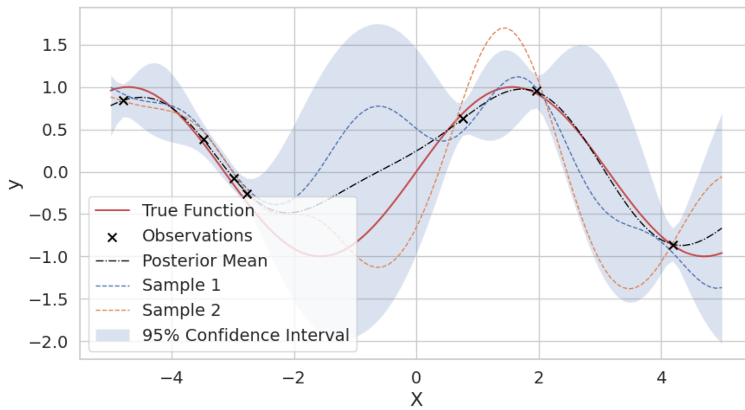


Figure 5: Illustration of Gaussian Process Regression in 1D. The true function is shown in red, observations are marked with black crosses, and the dashed lines represent two samples from the GP posterior. The dash-dot line represents the posterior mean, while the shaded region indicates the 95% confidence interval, showcasing the uncertainty associated with the GP predictions.

	# User	# Item	# Interac.	Density
Amazon	6,223	32,830	4M	0.18%
MovieLens	123,002	12,532	20M	1.27%
Taobao	756,892	570,350	70M	0.01%

Table 5: The statistics of datasets.

A.2. Dataset Statistics

The statistics of the three datasets, *Amazon* (He & McAuley, 2016), *MovieLens* (Harper & Konstan, 2015), and *Taobao*[1], is shown in Table 5. We report the number of users, number of items, number of total interactions, and density of the three datasets, respectively.

A.3. Ablation Analysis on Different Clustering Options

In this section, we provide the ablation analysis on using various clustering algorithms and different number of clusters on the Taobao dataset. Following Shi et al. (2023), we have selected four distinct clustering algorithms for our comparative analysis: (i) *Ward* (Ward, 1963): This hierarchical clustering technique focuses on minimizing the variance within each cluster, effectively reducing the overall sum of squared distances across all clusters. (ii) *K-Means* (Jin & Han, 2010): An iterative clustering algorithm that aims to minimize the sum of squared distances from each data point to the centroid of its assigned cluster, thereby reducing in-cluster variance. (iii) *Spectral Clustering* (Shi & Malik, 2000): This method performs clustering based on the eigenvectors of the normalized Laplacian, which is computed from the affinity matrix, facilitating the division based on the graph's inherent structure. (iv) *BIRCH* (Zhang et al., 1996): A hierarchical clustering approach that efficiently processes large datasets by incrementally constructing a Clustering Feature Tree, which groups data points based on their proximity and other features.

A notable deviation from Shi et al. (2023) is in our handling of categories (interests) for training and inference phases; we maintain consistency in categories across both phases, in contrast to their separate categorization. We further refine our analysis by tuning the number of clusters from 10, 20, and 50.

Result and analysis. The results presented in the Table 6 indicate a nuanced understanding of the impact of clustering algorithms and the number of clusters on retrieval and ranking tasks in information retrieval systems. Across the board, the differences among the clustering algorithms—Ward, K-Means, Spectral, and BIRCH—are relatively small, suggesting that the choice of clustering algorithm might not be as critical as the configuration of the number of clusters within these algorithms for the tasks at hand.

[1] https://tianchi.aliyun.com/dataset/649?lang=en-us

	Num of clusters	Retrieval Task IR@50 *The higher the better* ⇑	Retrieval Task ED@50 *The lower the better* ⇓	Retrieval Task TEI@50 *The higher the better* ⇑	Ranking Task Recall@50 *The higher the better* ⇑	Ranking Task nDCG@50 *The higher the better* ⇑
Ward	10	0.602 ± 0.012	0.385 ± 0.011	-0.055 ± 0.001	0.892 ± 0.013	0.920 ± 0.012
	20	<u>0.624 ± 0.011</u>	<u>0.367 ± 0.011</u>	**-0.040 ± 0.001**	**0.906 ± 0.012**	<u>0.936 ± 0.011</u>
	50	**0.625 ± 0.011**	**0.365 ± 0.011**	<u>-0.041 ± 0.001</u>	**0.906 ± 0.012**	**0.938 ± 0.011**
K-Means	10	0.590 ± 0.013	0.392 ± 0.011	-0.060 ± 0.001	0.885 ± 0.013	0.912 ± 0.012
	20	<u>0.615 ± 0.012</u>	<u>0.370 ± 0.011</u>	**-0.045 ± 0.001**	<u>0.900 ± 0.012</u>	<u>0.928 ± 0.012</u>
	50	**0.620 ± 0.011**	**0.368 ± 0.011**	<u>-0.046 ± 0.001</u>	**0.904 ± 0.012**	**0.930 ± 0.011**
Spectral	10	0.605 ± 0.012	0.382 ± 0.011	-0.053 ± 0.001	0.895 ± 0.013	0.923 ± 0.012
	20	**0.625 ± 0.011**	<u>0.366 ± 0.011</u>	**-0.039 ± 0.001**	<u>0.908 ± 0.012</u>	**0.937 ± 0.011**
	50	<u>0.624 ± 0.011</u>	**0.364 ± 0.011**	**-0.039 ± 0.001**	**0.909 ± 0.011**	<u>0.936 ± 0.012</u>
BIRCH	10	0.580 ± 0.013	0.395 ± 0.012	-0.065 ± 0.002	0.880 ± 0.014	0.910 ± 0.013
	20	**0.610 ± 0.012**	<u>0.372 ± 0.011</u>	**-0.050 ± 0.001**	<u>0.899 ± 0.012</u>	<u>0.926 ± 0.012</u>
	50	**0.610 ± 0.011**	**0.370 ± 0.011**	<u>-0.051 ± 0.001</u>	**0.902 ± 0.012**	**0.928 ± 0.011**

Table 6: Result comparison on Taobao dataset using different cluster options. The Interest Coverage (IC) metric is omitted from this report as its value is significantly influenced by the chosen number of clusters, rendering it less relevant for comparative purposes. All reported metrics are evaluated at the top-50.

Notably, as the number of clusters increases from 10 to 20 and then to 50, there is a general trend of improvement in performance metrics. This improvement could be attributed to the algorithms' ability to capture more fine-grained user interests through a larger number of clusters, thereby enhancing both retrieval accuracy and ranking precision.

However, the marginal differences observed between the configurations of 20 and 50 clusters across all metrics indicate diminishing returns on further increasing the number of clusters beyond 20. This observation suggests that while increasing the number of clusters to 20 contributes to significant improvements, escalating to 50 clusters does not yield proportionally higher gains. Consequently, this study opts for a configuration of 20 clusters as the optimal balance between performance improvement and computational efficiency. The relatively stable standard deviations across different configurations further support the robustness of our findings, emphasizing the consistency of the performance improvements achieved with an increased number of clusters up to a certain point.

A.4. Latency and Performance Comparison

We show the efficiency comparison (with the performance comparison) across models in Fig 6. For a fair comparison, we run all experiments on a single NVIDIA A100 GPU with TensorFlow framework (version 1.12) without any further optimization on the computation. We choose the MovieLens dataset and set the batch as 1.

Result and analysis. In examining the efficiency of various recommendation models, including YoutubeDNN, GRU4Rec, MIND, DGCF, ComiRec, CAMI, and GPR4DUR, notable differences in training and inference times highlight the diverse computational demands across these models. Models like YoutubeDNN and GRU4Rec present a more efficient profile, offering lower latency during both training and inference phases, while perform the worst on the retrieval and ranking tasks. The variability in model efficiency, as seen in the standard deviations of training and inference times, indicates a potential fluctuation in latency that could impact real-world deployment. Our proposed model, GPR4DUR, stands out with its superior performance on the retrieval task (e.g., measured by interest relevance) and the ranking task (e.g., measured by recall). This suggests that for applications valuing enhanced retrieval and ranking accuracy, the trade-off of increased latency may be acceptable. Additionally, GPR4DUR exhibits a low standard deviation in its timing metrics. This consistency in processing times is advantageous for online deployment. It offers a predictable performance, which is crucial for developers when balancing between model efficiency and effectiveness.

Conclusively, GPR4DUR, along with the other baseline models evaluated, meets the latency requirements for online application without the need for additional computational and serving optimizations. While GPR4DUR exhibits a higher time cost relative to some baselines, its performance improvement offers a compelling advantage.

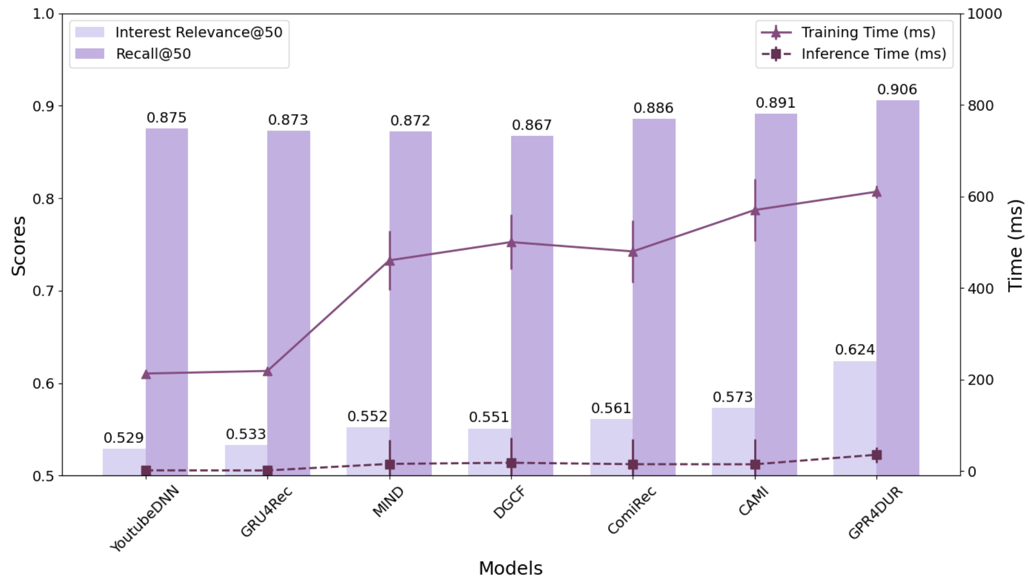


Figure 6: Latency and performance comparison across models using the MovieLens dataset. Training and inference latencies are measured in millisecond (ms), and the vertical short lines indicate the standard deviations.

Model	Retrieval Task				Ranking Task	
	IC@50	IR@50	ED@50	TEI@50	Recall@50	nDCG@50
	The higher the better ⇑	*The higher the better* ⇑	*The lower the better* ⇓	*The higher the better* ⇑	*The higher the better* ⇑	*The higher the better* ⇑
VAECF	0.783	0.482	0.546	-0.082	**0.911**	**0.931**
GPR4DUR	**0.895**	**0.560**	**0.423**	**-0.039**	0.908	0.922

Table 7: Result comparison between VAECF and GPR4DUR on the Amazon dataset.

A.5. Comparison with VAE-based Method

We notice that the VAE-based method is also a potential baseline, such as VAECF (Liang et al., 2018b), which assumes a distribution over user/item representations. We do not select it as a main baseline for comparison due to *(i)* VAECF is a parametric model while our model is non-parametric, and *(ii)* VAECF is not designed for multi-interest retrieval, which is the focus of this work.

To still satisfy this curiosity, we present a comparison between VAECF and GPR4DUR on the Amazon dataset for both the retrieval task and ranking task. As shown in Table 7, we observe that GPR4DUR outperforms VAECF on the retrieval task while being slightly worse than VAECF on the ranking task. Considering the gap on the ranking task compared to VAECF is small and the main focus of this work is the retrieval phase, we can still confirm the effectiveness of our model.