Learning Personalized Item-to-Item Recommendation Metric via Implicit Feedback

Trong Nghia Hoang, Anoop Deoras
AWS AI, Amazon

Tong Zhao, Jin Li
Personalization, Amazon

George Karypis
AWS AI, Amazon

Abstract

This paper studies the item-to-item recommendation problem in recommender systems from a new perspective of metric learning via implicit feedback. We develop and investigate a personalizable deep metric model that captures both the internal contents of items and how they were interacted with by users. There are two key challenges in learning such model. First, there is no explicit similarity annotation, which deviates from the assumption of most metric learning methods. Second, these approaches ignore the fact that items are often represented by multiple sources of meta data and different users use different combinations of these sources to form their own notion of similarity.

To address these challenges, we develop a new metric representation embedded as kernel parameters of a probabilistic model. This helps express the correlation between items that a user has interacted with, which can be used to predict user interaction with new items. Our approach hinges on the intuition that similar items induce similar interactions from the same user, thus fitting a metric-parameterized model to predict an implicit feedback signal could indirectly guide it towards finding the most suitable metric for each user. To this end, we also analyze how and when the proposed method is effective from a theoretical lens. Its empirical effectiveness is also demonstrated on several real-world datasets.

1 Introduction

Item recommendation is one of the fundamental tasks in a recommender system which is applicable to many scenarios such as *you may also like* on e-commerce platforms (e.g., Amazon, Alibaba) or *because you watched* on content streaming services (e.g., Netflix). These include two specific use cases of (1) user-centric and (2) item-centric recommendations. In user-centric recommendation, the focus is on recommending items that fit best with a profile of a target user. In item-centric recommendation, the focus is instead on recommending items that are similar to a target item.

To date, most recommendation methods have focused on the user-centric context of recommending relevant items to a target user. However, such user-centric methods are often not suitable for item-centric use cases since (as mentioned above) their focus is to find items that fit best with a user's profile but might not necessarily be similar to the target item that the user is currently interested in. To the best of our knowledge, there are very few works devised to tackle item-item recommendation directly. Most notable works among those are *sparse linear method* (SLIM) [29], which was adapted from user-centric collaborative filtering (CF) methods [11, 27, 29, 34, 47], and *semi-parametric embedding* (SPE) [19], which combines elements of both CF- and content-based methods [25, 28, 30]. But, SLIM [29] does not make use of meta information while SPE [19] ignores different similarity notions that we mentioned above. In both cases, there is a need to develop a personalizable item-to-item distance metric that not only capture the similarities between items across different sources of meta data but also *how* these channels are perceived by different users.

This leads us to the problem of metric learning [1, 3, 21, 24, 44] which aims to learn a distance measure on the feature space of items that can capture well the semantic similarities of items on their original input space [2, 7, 26, 45] (see Section 2). However, most (deep) metric learning methods were developed outside the context of a practical recommendation system [8, 11, 20, 29, 36] where class labels or even co-view signals of commercial items are not fine-grained enough to determine whether two items are similar or not. For example, two movies might belong to the same genre

Proceedings of the 25[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.

but they are not considered similar due to other traits such as cast, producer and plot. On the other hand, co-viewed movies might be driven by an exploration behavior rather than by their intrinsic similarities [43]. This invalidates vanilla application of existing supervised metric learning algorithms. Furthermore, users tend to have different preferences in forming their own notion of similarity from different meta information channel of commercial items. For example, some users would consider movies from the same genres or thematic content to be similar, while others might prefer movies from specific cast. This invalidates direct uses of unsupervised methods seeking to preserve a single geometry of item-to-item similarity since there can be as many as the number of different users.

Motivation. This motivates us to develop a multi-channel metric representation that can be learned via implicit feedback on how good it is towards a downstream prediction task. To achieve this, we first propose and investigate an ensemble construction of multiple Siamese Twin segments [2] such that each segment comprises two identical towers that capture the embedded similar or dissimilar traits between two input items for each source of meta data describing a certain aspect of their internal contents. As there is no direct feedback to train this metric ensemble, we then use a surrogate prediction task to provide a self-supervised feedback for training, which can also be used to personalize the metric for a user given his or her interaction data.

Key Idea & **Contributions.** This is achieved by embedding the ensemble representation as part of the kernel parameters expressing the correlation between items within a prediction model, which is fitted to predict how a user would interact with an item based on its correlation with previous items that the user has interacted with. Our approach hinges on the intuition that similar items would induce similar interactions from the same user, thus learning an interaction (e.g., rating) prediction model based on the metric representation can implicitly guide it towards capturing the right metric for each user. In particular, we contribute:

1. An adapted Gaussian process (GP) [33] regression model whose kernel function is parameterized by an ensemble of Siamese Twin segments (Section 3.1). The GP model is fitted to predict the average user rating of an unseen item given items with observed ratings. As the correlation is expressed in terms of metric representation, a well-fitted GP would be encouraged to find a well-behaved metric that correctly preserves the averaged similarity geometry of items.

2. A personalization scheme that warps the averaged similarity geometry of items into a personalized one that better fits each specific user (Section 3.2) via optimizing the combining coefficient of the ensemble. This makes sense since the content extracted from different meta-data channels is user-agnostic, leaving only the combining parameters user-dependent.

3. A theoretical analysis (Section 4) that analyzes the effectiveness of the proposed self-supervised metric learning algorithm in terms of the statistical relevance between the surrogate prediction task (e.g., rating prediction) and the true (unknown) item-to-item metric. Our results (Theorems 1 and 2) show that under reasonable assumptions, the learned metric is close to the true metric with high probability if there is a sufficient amount of observations from the surrogate function.

4. An empirical evaluation of the proposed method on an experimentation benchmark comprising the several public datasets such as the MovieLens [9] and Yelp Review datasets (Section 5).

2 Related Work

2.1 Metric Learning and Siamese Network

One prominent line of research in metric learning focuses on supervised methods [1, 5, 10, 22, 24, 44, 48, 49] which assume there exist training examples of similar and dissimilar items are available. The metric learning task is thus reduced to learning a scoring function that pushes down on similar pairs while pushing up on dissimilar pairs. One notable example of such metric learning method is the Siamese network which can be learned via optimizing a contrastive loss.

Siamese Network. As developed in [2], Siamese network has a two-tower architecture that was specifically devised for contrastive learning. In a nutshell, a Siamese net is expected to accept a pair of input items $(\mathbf{x}_a, \mathbf{x}_b)$ and output a numeric distance between them or a probability that they are dissimilar. For a pair of similar items, we expect this distance or probability to be small and conversely, for dissimilar items, we expect it to be sufficiently large (i.e., above a certain margin).

To achieve this, the Siamese net has two identical network segments, $\mathbf{F}_a(\mathbf{x};\boldsymbol{\gamma}) \equiv \mathbf{F}(\mathbf{x};\boldsymbol{\gamma})$ and $\mathbf{F}_b(\mathbf{x};\boldsymbol{\gamma}) \equiv \mathbf{F}(\mathbf{x};\boldsymbol{\gamma})$, whose outputs, $\mathbf{z}_a = \mathbf{F}(\mathbf{x}_a;\boldsymbol{\gamma})$ and $\mathbf{z}_b = \mathbf{F}(\mathbf{x}_b;\boldsymbol{\gamma})$ reside in a metric space $\mathbb{R}^p$ equipped with a parameterized distance $\mathbf{D}(\mathbf{z}_a, \mathbf{z}_b) = (\mathbf{z}_a - \mathbf{z}_b)^\top \boldsymbol{\Lambda}(\mathbf{z}_a - \mathbf{z}_b)$ where $\boldsymbol{\Lambda} = \mathrm{diag}[\lambda_1, \lambda_2, \ldots, \lambda_p]$. Thus, given a pair $(\mathbf{x}_a, \mathbf{x}_b)$, the output of the Siamese net is $\mathbf{D}(\mathbf{x}_a, \mathbf{x}_b) =$

$$\Big(\mathbf{F}\Big(\mathbf{x}_a;\boldsymbol{\gamma}\Big) - \mathbf{F}\Big(\mathbf{x}_b;\boldsymbol{\gamma}\Big)\Big)^\top \boldsymbol{\Lambda} \Big(\mathbf{F}\Big(\mathbf{x}_a;\boldsymbol{\gamma}\Big) - \mathbf{F}\Big(\mathbf{x}_b;\boldsymbol{\gamma}\Big)\Big) \quad (1)$$

Then, suppose training examples $((\mathbf{x}_a^i, \mathbf{x}_b^i), y_{ab}^i)_{i=1}^n$ are available where $y_{ab}^i = 0$ indicate $(\mathbf{x}_a^i, \mathbf{x}_b^i)$ is a pair of similar items and otherwise for $y_{ab}^i = 1$. The parameterization of the metric net, $\boldsymbol{\gamma}$ and $\boldsymbol{\Lambda}$, can be learned

via optimizing the following contrastive loss,

$$\mathbf{L}\Big(\boldsymbol{\gamma}, \boldsymbol{\Lambda}\Big) \;=\; \sum_{i=1}^{n}\left[\Big(1-y^i_{ab}\Big)\mathbf{D}\left(\mathbf{x}^i_a, \mathbf{x}^i_b\right)\right] + \sum_{i=1}^{n}\left[y^i_{ab}\max\Big(0, \tau - \mathbf{D}\left(\mathbf{x}^i_a, \mathbf{x}^i_b\right)\Big)\right] \quad (2)$$

where τ is a contrastive margin such that the distance for dissimilar pairs are encouraged to be increased up to τ but not more than that. This implies forcing the distance between a dissimilar pair to be more than τ only yields diminishing gain in improving the discriminative capacity of the model. Thus, the loss is zeroed out in such cases to focus on minimizing the gap between the other similar pairs.

2.2 Gaussian Processes

A Gaussian process [33] defines a probabilistic prior over a random function $g(\mathbf{x})$. This prior is in turn defined by a mean function $m(\mathbf{x}) = 0$[1] and a kernel function $k(\mathbf{x}, \mathbf{x}')$. Such prior stipulates that for an arbitrary finite subset of inputs $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, the corresponding output vector $\mathbf{g} = [g(\mathbf{x}_1)\ g(\mathbf{x}_2) \ldots\ g(\mathbf{x}_n)]^\top$ is distributed by a multivariate Gaussian, $\mathbf{g} \sim \mathbb{N}(\mathbf{0}, \mathbf{K})$.

Here, the entries of the covariance matrix $\mathbf{K}$ are computed using the aforementioned kernel function. That is, $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ where common examples of $k(\mathbf{x}_i, \mathbf{x}_j)$ are detailed in [33] but in practice, the exact choice of the kernel function usually depends on the application. To predict with GP, let $\mathbf{x}_*$ be an unseen input whose corresponding output $g_* = g(\mathbf{x}_*)$ we wish to predict. Then, assuming a noisy setting where we only observe a noisy observation $r(\mathbf{x}) \sim \mathbf{N}(g(\mathbf{x}), \sigma^2)$ instead of $g(\mathbf{x})$ directly, the predictive distribution of g_* is:

$$g_* \;\triangleq\; g(\mathbf{x}_*) \mid \mathbf{r} \;\sim\; \mathbb{N}\Big(\mathbf{k}_*^\top(\mathbf{K}+\sigma^2\mathbf{I})^{-1}\mathbf{r},\ k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top(\mathbf{K}+\sigma^2\mathbf{I})^{-1}\mathbf{k}_*\Big)\ , \quad (3)$$

where $\mathbf{r} = [r(\mathbf{x}_1) \ldots r(\mathbf{x}_n)]^\top$. The defining parameter $\boldsymbol{\psi}$ of $k(\mathbf{x}, \mathbf{x}')$ is crucial to the predictive performance and needs to be optimized via minimizing the negative log likelihood (NLL) of $\mathbf{r}$ [33],

$$\ell(\boldsymbol{\psi}) \;\propto\; \frac{1}{2}\log\Big|\mathbf{K}_{\boldsymbol{\psi}} + \sigma^2\mathbf{I}\Big| + \frac{1}{2}\mathbf{r}^\top\Big(\mathbf{K}_{\boldsymbol{\psi}} + \sigma^2\mathbf{I}\Big)^{-1}\mathbf{r} \quad (4)$$

In the above, we use the subscript $\boldsymbol{\psi}$ to indicate that $\mathbf{K}$ is parameterized by $\boldsymbol{\psi}$ which, in our case, is the parameterization of a Siamese network (Section 2.1). In practice, both training $\boldsymbol{\Theta}$ and prediction incur $\mathbf{O}(n^3)$ cost. For better scalability, there have been numerous developments on sparse GPs [12, 14, 15, 16, 23, 42] whose computation are only linear in n (see Appendix G).

[1] For simplicity, we assume a zero mean function since we can always re-center the training outputs around 0.

3 Metric Learning via Gaussian Process with Siamese Kernel

We will formalize our intuition (Section 1) of self-supervised learning (SSL) of item-to-item metric, namely the *SSL metric*, in Section 3.1. We will then show how such *SSL metric* can also be personalized for each user via minimizing a new loss function as proposed in Section 3.2.

3.1 Self-Supervised Metric Learning with Gaussian Processes

Let $r(\mathbf{x})$ denote a related prediction target of item $\mathbf{x}$ whose training examples $\{(\mathbf{x}_i, r(\mathbf{x}_i))\}_{i=1}^n$ are readily available from our data. For example, $r(\mathbf{x})$ can be an averaged review score for $\mathbf{x}$, which aggregates the ratings given to $\mathbf{x}$ by the users who interacted with it. Our goal is to build a prediction model such that for an unseen item $\mathbf{x}_*$, its prediction $\widehat{r}(\mathbf{x}_*)$ is largely based on its metric-based correlation with the training items $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$. This is the standard prediction pattern of kernel-based methods such as Gaussian process (GP). To substantiate this, we parameterize the kernel function of the GP prior using the aforementioned Siamese network (Section 2.1) as detailed below,

$$k\Big(\mathbf{x}_a, \mathbf{x}_b; \boldsymbol{\psi}\Big) \;=\; \exp\left(-\frac{1}{2}\mathbf{D}\left(\mathbf{x}_a, \mathbf{x}_b\right)\right) \quad (5)$$

where $\mathbf{D}(\mathbf{x}_a, \mathbf{x}_b)$ is defined in Eq. (1). Here, the kernel parameterization $\boldsymbol{\psi} = \{\boldsymbol{\Lambda}, \boldsymbol{\gamma}\}$ consists of two parts. First, $\boldsymbol{\gamma}$ denotes the defining parameters of the network segment that maps $\mathbf{x}$ to a vector in a metric space. Second, $\boldsymbol{\Lambda}$ specifies the correlation and unit scales across different dimensions of the metric space. For example, if we choose $\boldsymbol{\Lambda}$ to be the diagonal matrix, then the dimensions of the metric space are uncorrelated and their unit scales are the elements on the diagonal of $\boldsymbol{\Lambda}$. Then, given training examples $\{(\mathbf{x}_i, r_i)\}_{i=1}^n$ where $r_i \sim \mathbb{N}(r(\mathbf{x}_i), \sigma^2)$ are the noisy observations of $r(\mathbf{x}_i)$ perturbed with Gaussian noises, the metric parameters can be optimized via minimizing

$$\ell(\boldsymbol{\psi}) \;=\; \frac{1}{2}\log\Big|\mathbf{K}_{\boldsymbol{\psi}} + \sigma^2\mathbf{I}\Big| + \frac{1}{2}\mathbf{r}^\top\Big(\mathbf{K}_{\boldsymbol{\psi}} + \sigma^2\mathbf{I}\Big)^{-1}\mathbf{r} \quad (6)$$

with respect to $\boldsymbol{\psi}$ and σ where $\mathbf{r} = [r_1\ r_2 \ldots r_n]^\top$ and $\boldsymbol{\psi} \triangleq \{\boldsymbol{\Lambda}, \boldsymbol{\gamma}\}$. Here, Eq. (6) is the same as Eq. (4) except for that entries of $\mathbf{K}_{\boldsymbol{\psi}}$ were computed by Eq. (5) above. We will demonstrate later in Section 5 that training $\{\boldsymbol{\Lambda}, \boldsymbol{\gamma}\}$ using Eq. (6) is more effective than fitting them using Eq. (2) which requires direct feedback that cannot be acquired without incurring considerable label noise.

Key Result: We will also show in Section 4 below (see Theorem 1 and Theorem 2) that assuming a certain

statistical relationship (see **A1-A3**) between $\mathbf{r}$ and the true metric $\mathbf{D}_*(\mathbf{x}, \mathbf{x}')$, the learned *SSL metric* $\mathbf{D}(\mathbf{x}, \mathbf{x}')$ via fitting Eq. (6) is arbitrarily close to $\mathbf{D}_*(\mathbf{x}, \mathbf{x}')$ if the number of observations n is sufficiently large.

3.2 Learning Personalizable Metric with Gaussian Processes

Furthermore, to account for multiple different sources of meta data describing an item that lead to different user preferences over their uses in combination, we first extend the above Siamese network architecture into an ensemble construction of multiple Siamese segments.

Siamese Ensemble. Let $\mathbf{x} = (\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(p)})$ denote its multi-view representation across p different meta channels where $\mathbf{x}^{(i)}$ denote $\mathbf{x}$'s description in channel i. The Siamese ensemble is $\mathbf{D}\left(\mathbf{x}_a, \mathbf{x}_b\right) =$

$$\sigma\left(\left[\mathbf{D}_1\left(\mathbf{x}_a^{(1)}, \mathbf{x}_b^{(1)}\right), \ldots, \mathbf{D}_p\left(\mathbf{x}_a^{(p)}, \mathbf{x}_b^{(p)}\right)\right]^\top \mathbf{h} + b\right) \quad (7)$$

where $\sigma(z) = 1/(1 + \exp(-z))$ denotes the sigmoid function while $\mathbf{w} = (\mathbf{h}, b)$ where $\mathbf{h} \in \mathbb{R}^p$ and $b \in \mathbb{R}$ are learnable weights that aggregate the individual Siamese distances into a single distance metric. In our personalized context, the metric computation for each user u shares the same set of Siamese individual distance functions (and their parameterizations) but differs in how these individual Siamese distances were combined via different choices of $\mathbf{w} \leftarrow \mathbf{w}_u = (\mathbf{h}_u, b_u)$.

Learning Personalizable Metric. First, we observe that the parameterization $\boldsymbol{\psi}_i = (\boldsymbol{\gamma}_i, \boldsymbol{\Lambda}_i)$ of each Siamese segment $\mathbf{D}_i$ is user-agnostic since the intrinsic similarities between items across single channels are not user-dependent. The personalization must therefore concern only ensemble parameterization $\mathbf{w} \leftarrow \mathbf{w}_u$. This raises the question of how can we build a personalizable parameterization which can be fast adapted to an arbitrary user with limited personal data?

To address this question, one ad-hoc choice is to reuse the self-supervised learning recipe in Section 3.1 to optimize for $\mathbf{w} = (\mathbf{h}, b)$, which can be achieved by reconfiguring the kernel function $k(\mathbf{x}, \mathbf{x}')$ in Eq. (5) with the ensemble distance $\mathbf{D}$ in Eq. (7) and minimizing the following NLL loss $\ell(\boldsymbol{\psi}, \mathbf{w}) =$

$$\frac{1}{2}\log\left|\mathbf{K}_{(\boldsymbol{\psi},\mathbf{w})} + \sigma^2\mathbf{I}\right| + \frac{1}{2}\mathbf{r}^\top\left(\mathbf{K}_{(\boldsymbol{\psi},\mathbf{w})} + \sigma^2\mathbf{I}\right)^{-1}\mathbf{r}\ . \quad (8)$$

with respect to $\mathbf{w}$, while fixing $\boldsymbol{\psi} = (\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_p)$. The resulting $\mathbf{w}$ can then be re-fitted for each user u via another pass of the algorithm in Section 3.1 using only observations $\mathbf{r}_u$ from u's individual surrogate function $r_u(\mathbf{x})$. This approach however does not optimize for how fast $\mathbf{w}$ can be adapted (on average) for a random user u with limited data. To account for this, we instead minimize the following *post-update, personalized* loss function over q users – see its intuition below Eq. (10),

$$\mathbf{L}(\mathbf{w}) \ = \ \frac{1}{q}\sum_{u=1}^{q}\Big[\ell_u(\boldsymbol{\psi}, \kappa_u(\mathbf{w}))\Big] \quad (9)$$

where $\ell(\boldsymbol{\psi}, \mathbf{w})$ is defined in Eq. (8) above and $\ell_u(\boldsymbol{\psi}, \kappa_u(\mathbf{w}))$ is identical in form to $\ell(\boldsymbol{\psi})$ except for the fact that it is parameterized by $\mathbf{w}_u = \kappa_u(\mathbf{w})$ (instead of $\mathbf{w}$) and computed based on local observations $\mathbf{r}_u$ (instead of $\mathbf{r}$). Here, $\kappa_u(\mathbf{w})$ denotes a personalization procedure that minimizes $\ell_u(\boldsymbol{\psi}, \mathbf{w})$ for each user u, which can be represented in the following form,

$$\begin{aligned}\kappa_u(\mathbf{w}) \ = \ \kappa_u^{(t)}(\mathbf{w}) \ &\triangleq \ \kappa_u^{(t-1)}(\mathbf{w}) \\ &+ \ \omega\cdot\nabla_{\mathbf{w}}\ell\left(\kappa_u^{(t-1)}(\mathbf{w})\right)\end{aligned} \quad (10)$$

and $\kappa_u^{(0)}(\mathbf{w}) = \mathbf{w}$. Here, we drop $\boldsymbol{\psi}$ from the argument of $\ell_u(\boldsymbol{\psi}, \mathbf{w})$ since it is clear from context and is fixed. This encompasses the t-step gradient update procedure that aims to numerically minimize $\ell_u(\mathbf{w})$ with $\mathbf{w}$ being the initializer and ω denote the learning rate. Intuitively, minimizing Eq. (9) means finding a vantage point $\mathbf{w}$ that are most effective for personalization. That is, starting at $\mathbf{w}$, the local update procedure $\kappa_u(\mathbf{w})$ can arrive at an effective parameter configuration that reduces the local loss $\ell_u(\mathbf{w})$ the most. This generic form, however, poses a challenge since the gradient $\nabla_{\mathbf{w}}\mathbf{L}$ might not be tractable since $\kappa_u(\mathbf{w})$ might not exist in closed-form.

Key Idea: To address this, note that the vector-value function $\kappa_u(\mathbf{w})$ can be represented as $\kappa_u(\mathbf{w}) = [\kappa_u^1(\mathbf{w}) \ldots \kappa_u^{p+1}(\mathbf{w})]$ where we have $\dim(\mathbf{w}) = \dim(\mathbf{h}) + \dim(b) = p + 1$. We can then approximate each component with a 2^{nd}-order Taylor expansion around $\mathbf{0}$ and show that under such expansion, $\nabla_{\mathbf{w}}\mathbf{L}$ can be computed, which allows $\mathbf{L}$ to be minimized via Lemma 1.

Lemma 1. *Assuming $\kappa_u^i(\mathbf{w})$ is twice-differentiable at $\mathbf{w} = \mathbf{0}$, the approximation of $\kappa_u^i(\mathbf{w})$ with its 2^{nd}-order Taylor expansion around $\mathbf{w} = \mathbf{0}$ induces $\nabla_{\mathbf{w}}\mathbf{L}(\mathbf{w}_+) =$*

$$\frac{1}{q}\sum_{u=1}^{q}\left(\Big[\mathbf{D}_{\mathbf{w}}\kappa_u(\mathbf{w}_+)\Big]\nabla_{\mathbf{w}}\ell_u\Big(\kappa_u(\mathbf{w}_+)\Big)\right) \quad (11)$$

with $\mathbf{D}_{\mathbf{w}}\kappa_u(\mathbf{w}_+) \triangleq \left[\nabla_{\mathbf{w}}^\top\kappa_u^1(\mathbf{w}_+); \ldots; \nabla_{\mathbf{w}}^\top\kappa_u^{p+1}(\mathbf{w}_+)\right]$ *whose rows are approximated via*

$$\nabla_{\mathbf{w}}\kappa_u^i\Big(\mathbf{w}_+\Big) \simeq \nabla_{\mathbf{w}}\kappa_u^i\Big(\mathbf{0}\Big) + \Big[\nabla_{\mathbf{w}}^2\kappa_u^i\Big(\mathbf{0}\Big)\Big]\,\mathbf{w}_+ \quad (12)$$

Lemma 1 implies that if $\nabla_{\mathbf{w}}\ell(\mathbf{w}_+)$ and $\nabla_{\mathbf{w}}\ell_u(\mathbf{w}_+)$ are tractable; and $\nabla_{\mathbf{w}}\kappa_u^i(\mathbf{w}_+)$ and $\nabla_{\mathbf{w}}^2\kappa_u^i(\mathbf{w}_+)$ are tractable at $\mathbf{w} = \mathbf{0}$ then the gradient of $\mathbf{L}(\mathbf{w})$ is also

approximately tractable at any $\mathbf{w}_+$, thus mitigating the lack of a closed-form expression for $\kappa_u(\mathbf{w})$. Here, the tractability of $\nabla_{\mathbf{w}}\ell(\mathbf{w}_+)$ and $\nabla_{\mathbf{w}}\ell_u(\mathbf{w}_+)$ is evident from their analytic form in Eq. (8) while the tractability of $\nabla_{\mathbf{w}}\kappa_u^i(\mathbf{w}_+)$ and $\nabla^2_{\mathbf{w}}\kappa_u^i(\mathbf{w}_+)$ at $\mathbf{w}_+ = \mathbf{0}$ along with the rest of the proof is deferred to Appendix B.

Note that for simple choices of $\kappa_u(\mathbf{w})$, $\mathbf{D}_{\mathbf{w}}\kappa_u(\mathbf{w}_+)$ can be computed analytically to bypass this approximation. For instance, in our experiment, we choose $\kappa_u(\mathbf{w}) = \mathbf{w} - \omega\nabla_{\mathbf{w}}\ell_u(\mathbf{w})$. It then follows that $\mathbf{D}_{\mathbf{w}}\kappa_u(\mathbf{w}_+) = \mathbf{I} - \omega\nabla^2_{\mathbf{w}}\ell_u(\mathbf{w}_+)$ which is exact and tractable. This leads to a simpler expression for Eq. (11), which also mimics the meta-update equation in meta learning [6].

4 Theoretical Analysis

This section analyzes the effectiveness of the proposed self-supervised metric learning algorithm (Section 3.1) from a theoretical lens, which aims to shed insights on when and how the induced metric approximates accurately. In essence, our main results, Theorems 1 and 2, show that under reasonable assumptions, the induced metric $\mathbf{D}(\mathbf{x}, \mathbf{x}')$ of our algorithm is arbitrarily close to the true (unknown) metric $\mathbf{D}_*(\mathbf{x}, \mathbf{x}')$ with high probability if it can observe a sufficiently large number n of observations from the surrogate training feedback $r(\mathbf{x})$. Our assumptions are first stated below.

Assumptions. Let $\mathbf{D}_*(\mathbf{x}, \mathbf{x}')$ denote the (unknown) true metric function and $k_*(\mathbf{x}, \mathbf{x}') = \exp(-0.5 \cdot \mathbf{D}_*(\mathbf{x}, \mathbf{x}'))$ denote the oracle kernel parameterized by the true metric. Then:

A1. There exists a constant value $d > 0$ for which $d \cdot \inf\, k_*(\mathbf{x}, \mathbf{x}') \geq \lambda_{\max}(\mathbf{K}_*)$ where $\lambda_{\max}(\mathbf{K}_*)$ denote the largest eigenvalue of the Gram matrix induced by k_* on $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$.

A2. Let $\widehat{\mathbf{r}} = [\widehat{r}(\mathbf{x}_1), \ldots, \widehat{r}(\mathbf{x}_n)]^\top$ denote the GP prediction of $\mathbf{r} = [r(\mathbf{x}_1), \ldots, r(\mathbf{x}_n)]^\top$ using the learned kernel function $k(\mathbf{x}, \mathbf{x}')$ – see Eq. (5)[2], there exists a non-negative constant α s.t. $\sup_{(\mathbf{x},\mathbf{x}')} |k(\mathbf{x}, \mathbf{x}') - k_*(\mathbf{x}, \mathbf{x}')| \leq$

$$1 \;-\; \alpha \;\cdot\; \left(\Big(\mathbf{r} - \widehat{\mathbf{r}}\Big)^\top \mathbf{A} \Big(\mathbf{r} - \widehat{\mathbf{r}}\Big)\right)^{-1} \tag{13}$$

where $\mathbf{A} = \dfrac{1}{n}\left(\mathbf{K} + \sigma^2\mathbf{I}\right)^2$ and $\mathbf{K}$ is the Gram matrix induced by $k(\mathbf{x}, \mathbf{x}')$ on $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$.

A3. Let $\mathbf{r} = [r(\mathbf{x}_1), r(\mathbf{x}_2), \ldots, r(\mathbf{x}_n)]^\top$ and $\mathbf{K}_*$ be defined as above. We assume $\mathbf{r} \sim \mathbb{N}(0, \mathbf{K}_*)$. This is key to establish our main results in Theorems 1 and 2.

[2] As it is clear from context, we drop the parameterization notation $\boldsymbol{\psi}$ from this point onward for simplicity.

Remark. Here, assumptions **A1** and **A2** state that the oracle kernel is bounded from below (**A1**) and the discrepancies between the approximate and oracle kernel are bounded above with a ceiling no more than 1 (**A2**). These are reasonable assumptions which can be realized in most cases given that by construction, the range of values for both kernel functions is between 0 and 1. Last, while **A3** imposes a stronger assumption on the statistical relationship between the surrogate training feedback $r(\mathbf{x})$ and oracle kernel $\mathbf{K}_*$, this is also not unreasonable given that in many situations, there also exists many feature signals that are both normally distributed and are directly related to the similarities across input instances, e.g. measurements of height/weight among people.

Under these assumptions, we can now state our key results which provably demonstrates how well the self-supervised learning metric can approximated the true metric, and under what conditions. Our strategy to address these questions are first described below.

Analysis Strategy. First, we aim to establish that if the maximum multiplicative error in approximating the oracle kernel (parameterized by the true metric) with the induced kernel (parameterized by the metric learned by our algorithm) can be made arbitrarily small, the same can also be said about the discrepancies between the true and learned metric (see Lemmas 2 and 3).

Then, we further establish that while the maximum multiplicative error between kernels is not always small with 100% certainty, the probability that it is large is vanishingly small as the size of the dataset increases (Theorem 1). This implies the results of Lemmas 2 and 3 can be invoked with high chance, guaranteeing that the metric discrepancies can be made vanishingly small with high probability. This is formalized in Theorem 2, which also details the least amount of data necessary for such event to happen.

Formal Results. To begin our technical analysis, we start with Lemma 2 below which shows that if the ratio between the true and approximate kernel values, $k_*(\mathbf{x}, \mathbf{x}')$ and $k(\mathbf{x}, \mathbf{x}')$, can be made arbitrarily close at $(\mathbf{x}, \mathbf{x}')$ then the approximated distance metric is also arbitrarily close at $(\mathbf{x}, \mathbf{x}')$.

Lemma 2. *Suppose* $(1 - \epsilon) \cdot k_*(\mathbf{x}, \mathbf{x}') \leq k(\mathbf{x}, \mathbf{x}') \leq (1 + \epsilon) \cdot k_*(\mathbf{x}, \mathbf{x}')$ *for* $\epsilon \in (0, 1)$ *then*

$$\Big|\mathbf{D}(\mathbf{x}, \mathbf{x}') - \mathbf{D}_*(\mathbf{x}, \mathbf{x}')\Big| \;\leq\; 2\log\left(\frac{1}{1-\epsilon}\right). \tag{14}$$

The result of Lemma 2, which is formally proved in Appendix C, suggests a direct strategy to guarantee the metric approximation is close to zero simultaneously for all pair $(\mathbf{x}, \mathbf{x}')$, as formalized in Lemma 3.

Lemma 3. *Suppose* $\sup_{(\mathbf{x},\mathbf{x}')} \left|\frac{k(\mathbf{x},\mathbf{x}') - k_*(\mathbf{x},\mathbf{x}')}{k_*(\mathbf{x},\mathbf{x}')}\right| \leq \epsilon$ *for* $\epsilon \in (0,1)$ *then*

$$\forall(\mathbf{x},\mathbf{x}') : \left|\mathbf{D}(\mathbf{x},\mathbf{x}') - \mathbf{D}_*(\mathbf{x},\mathbf{x}')\right| \leq 2\log\left(\frac{1}{1-\epsilon}\right) \quad (15)$$

Enforcing the premise of Lemma 3 – see its proof in Appendix D – however, is not always possible as it depends on the randomness in which we obtain our observations of the surrogate training feedback $r(\mathbf{x})$. This raises the following questions:

How likely this happens and how many observations are sufficient to guarantee that such premise would happen with high chance?

These are addressed in Theorems 1 and 2 below.

Theorem 1. *Let* $g(\tau) \triangleq \log(\tau) + (1/\tau) - 1$ *and* $c_\epsilon \triangleq \epsilon\lambda_{\max}(\mathbf{K}_*)/d$*, we have*

$$\begin{aligned}&\Pr\left(\sup_{(\mathbf{x},\mathbf{x}')}\left|\frac{k(\mathbf{x},\mathbf{x}') - k_*(\mathbf{x},\mathbf{x}')}{k_*(\mathbf{x},\mathbf{x}')}\right| \geq \epsilon\right)\\ \leq\ &\exp\left(-\frac{1}{2}n\cdot g\left(\frac{\sigma^4}{\alpha}\Big(1-c_\epsilon\Big)\lambda_{\max}\Big(\mathbf{K}_*\Big)\right)\right) \quad (16)\end{aligned}$$

where the constants d *and* α *are defined in* **A1** *and* **A2** *above respectively. The detailed proof of Theorem 1 is deferred to Appendix E. In addition, we note that* $g(\tau)$ *is non-negative so the RHS of Eq.* (16) *is no greater than* 1*, ensuring that the bound is not vacuous.*

Theorem 1 therefore establishes that the premise of Lemma 3 will happen with a high probability with a sufficiently large value of n, thus asserting its implication of Lemma 3 with high chance. This guarantees the discrepancy between the learned and true metric at any pair $(\mathbf{x},\mathbf{x}')$ can be made arbitrarily small with a large value of n. This is formalized below.

Theorem 2. *Let* $g(\tau) = \log(\tau) + (1/\tau) - 1$ *and* $g_\epsilon = g\left(\frac{\sigma^4}{\alpha}\left(1 - \lambda_{\max}\Big(\mathbf{K}_*\Big)\frac{\epsilon}{d}\right)\lambda_{\max}\Big(\mathbf{K}_*\Big)\right)$*. Then,*

$$\begin{aligned}\Pr\left(\sup_{(\mathbf{x},\mathbf{x}')}\left|\mathbf{D}(\mathbf{x},\mathbf{x}') - \mathbf{D}_*(\mathbf{x},\mathbf{x}')\right| \leq 2\log\left(\frac{1}{1-\epsilon}\right)\right)\\ \geq\ 1\ -\ \delta \quad (17)\end{aligned}$$

when $n \geq \frac{2}{g_\epsilon}\log\frac{1}{\delta}$ *and* $\delta \in (0,1)$ *is an arbitrarily small confidence parameter. Theorem 2 can be derived from Theorem 1 by setting the RHS of Eq.* (16) *to an arbitrarily small value* δ*, solving for* n *and following up with direct application of Lemma 3. Its proof is deferred to Appendix F.*

Theorem 2 thus concludes our analysis with the following take-home message: Under reasonable assumptions in **A1, A2** and **A3**, the metric induced by our self-supervised learning algorithm is vanishingly close to the true metric with arbitrarily high probability provided that we have access to a sufficiently large dataset of the surrogate training feedback $r(\mathbf{x})$. The statistical relation between this surrogate feedback and the true metric as stated in **A3** is key to establish this result.

5 Experiment

We evaluate our proposed self-supervised and personalized metric learning algorithms on the MovieLens [9] and Yelp review dataset[3]. A short description of the datasets is provided below.

MovieLens Dataset [9]. The dataset comprises 26K+ items which were interacted with by 138K+ users. Each interaction is a triplet of user, item and timestamp which is measured in seconds with respect to a certain point of origin in 1970. There are about 20M of such interactions and in addition, the dataset also provides multiple channels of meta data per item in various formats such as categorical (genre), numerical (rating) and text (plot and title). Here, representations of categorical and text features are multi-hot and pre-trained BERT[4] [4] embedding vectors, respectively.

Yelp Review Dataset[3]. The dataset comprises 8M+ reviews given to businesses by customers. Here, we treat the businesses as items and customers as users. There are approximately 160K+ businesses (items) and about 2M+ customers (users). For each business, we have meta data regarding its averaged rating and business categories. The latter of which is represented as a multi-hot vector ranging over 1300+ categories (e.g., Burgers, Mexican and Gastropubs).

Both datasets were pre-processed using the same procedure as described in Section A.1. In what follows, our experiments aim to address the following key questions:

Q1. Does the induced metric via self-supervised learning (Section 3.1) improve over the vanilla metric induced from optimizing a Siamese network on noisy annotations of similar pairs of items?

Q2. Does such SSL induced metric can be further personalized (Section 3.2) to fit better with a user's personal notion of similarity, which often varies substantially across different users?

[3] https://www.yelp.com/dataset/download

[4] Text descriptions of title and plot were embedded separately into a 768-dimensional space by the Google's pre-trained BERT model [4] released at `https://huggingface.co/bert-base-uncased`.

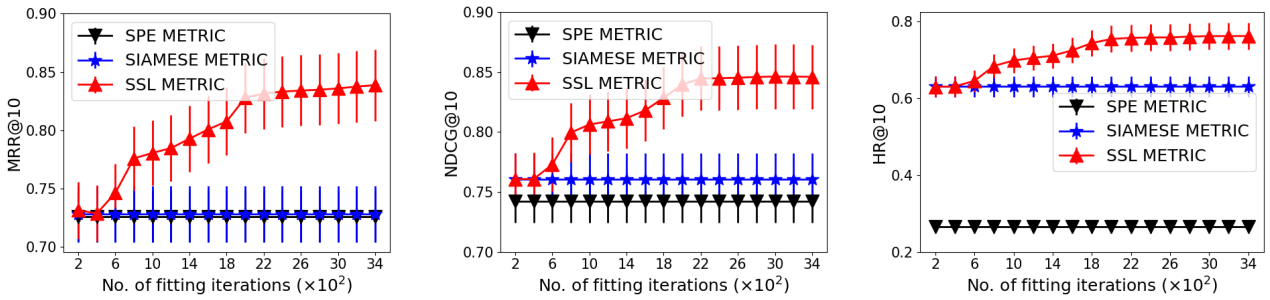

Figure 1: Plots of the averaged MRR, NDCG and HR measurements (over 5 independent SSL runs) of our SSL metric when it was used to produce a top-10 recommendation list for each unseen test item from the MovieLens dataset. The corresponding measurements of SPE and Siamese baselines were also included as references. These measurements however do not improve with more iterations as SPE and Siamese are non-SSL.

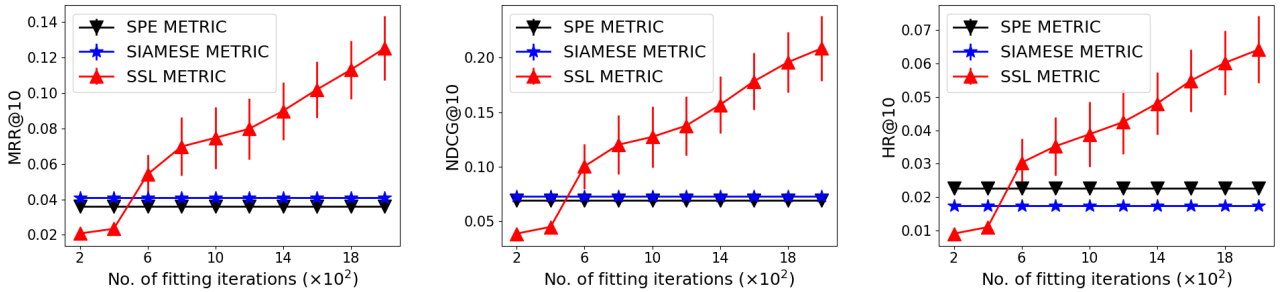

Figure 2: Plots of the averaged MRR, NDCG and HR measurements (over 5 independent SSL runs) of our SSL metric evaluated on YELP dataset. The performance of SPE and Siamese baselines were also included as references. These measurements however do not improve with more iterations as SPE and Siamese are non-SSL. Similar to our experiments with the MovieLens dataset, the MRR, NDCG and HR measurements are computed with respect to top-10 recommendation lists produced by the participating baselines on the same unseen test set.

Evaluation. Once learned, the item-to-item metric is used to rank the items in the list of candidates (i.e., the entire item catalogue) in the decreasing order of their similarities to a test item. For each test item, the quality of the resulting ranked list can then be assessed via standard ranking measurements such as mean reciprocal rank (MRR), hit rate (HR) and normalized discounted cumulative gain (NDCG). These are described below.

Measurement Description. To evaluate the efficiency of an item metric $\mathbf{D}$ on a specific item $\mathbf{x}$, we use it to compute the distance between $\mathbf{x}$ and every other item $\mathbf{x}'$ in the catalogue. The top $k = 10$ closest items $\{\mathbf{x}'_1, \ldots, \mathbf{x}'_k\}$ based on their computed distances to $\mathbf{x}$ are then extracted. Let $\mathbf{G}(\mathbf{x})$ denote the set of similar items to $\mathbf{x}$, the following quality measurements are computed to assess $\mathbf{D}$:

HR@K. The HR@K (or *hit rate* at k) measurement of $\mathbf{D}$ at test item $\mathbf{x}$ is

$$\mathbf{HR}_k\Big(\{\mathbf{x}'_i\}_{i=1}^k; \mathbf{x}\Big) \triangleq k^{-1} \sum_{i=1}^{k} \mathbb{I}\Big(\mathbf{x}'_i \in \mathbf{G}(\mathbf{x})\Big)$$

where $\{\mathbf{x}'_1, \ldots, \mathbf{x}'_k\}$ represent the top k items suggested by the recommender (in decreasing order of relevance to the test item $\mathbf{x}$). The average HR@K is computed by averaging over items in a test set.

MRR@K. The MRR@K (or *mean reciprocal rank* at k) measurement of $\mathbf{D}$ at $\mathbf{x}$ is

$$\mathbf{MRR}_k\Big(\{\mathbf{x}'_i\}_{i=1}^k; \mathbf{x}\Big) \triangleq \Big(\arg\min_{i=1}^k \{\mathbf{x}'_i : \mathbf{x}'_i \in \mathbf{G}(\mathbf{x})\}\Big)^{-1}$$

or 0 if none of the items in the recommendation is in $\mathbf{G}(\mathbf{x})$. The average MRR@K is then computed by averaging over items in a test set.

NDCG@K. First, the DCG@K – *discounted cumulative gain* at k – measurement of $\mathbf{D}$ at $\mathbf{x}$ is

$$\mathbf{NDCG}_k\Big(\{\mathbf{x}'_i\}_{i=1}^k; \mathbf{x}\Big) \triangleq \mathbb{I}\Big(\mathbf{x}'_1 \in \mathbf{G}(\mathbf{x})\Big) + \sum_{i=2}^{k} \mathbb{I}\Big(\mathbf{x}'_i \in \mathbf{G}(\mathbf{x})\Big) \cdot \log_2^{-1}[i]$$

The NDCG@K – *normalized discounted cumulative gain* at K – is then obtained by dividing DCG@K to

the maximum achievable DCG@K by a permutation of items in the recommendation list. The average NDCG@K is computed by averaging over the test set.

The overall assessment of the item-to-item metric is then computed by averaging the above measurements over the set of test items. Here, the test items are those whose 1st interaction happens after a timestamp (hence, not visible to the learning algorithms) which was set so that the test set comprises about 5% of the entire item catalogue.

To set the ground-truth for item-to-item recommendation, we deem two items similar if they were interacted with by the same users within a time horizon. Otherwise, they are deemed dissimilar. Here, we also note that unlike user-item truths, item-item truths acquired in this fashion are undeniably noisy so during training, we further make use of a downstream prediction task that (presumably) correlates well with the oracle item-item truths. To our intuition, rating prediction in the case of movies does appear to correlate with item similarities, which explains for the improved performance of our SSL method as reported in Section 5.1 below.

All experiments were run on a computing server with a Tesla V100 GPU with 16GB RAM. For more information regarding our experiment setup and data pre-processing, please refer to Appendix A.

5.1 Self-Supervised Metric Learning

To answer **Q1**, we evaluate the performance of the item-to-item metrics generated by (1) optimizing the vanilla Siamese ensemble (SIAMESE) combining the meta information channels (including *ratings*) of the items; (2) optimizing the more recently proposed SPE method [19] using *ratings* as side information and other channels as content; and (3) optimizing the GP with Siamese kernel (SSL), which is initialized with the Siamese ensemble generated in (1), using averaged *ratings* of the items. The results were averaged over 5 independent runs and reported in Figure 1 and Figure 2.

It is observed from both Figure 1 and Figure 2 that our SSL metric becomes increasingly better and outperforms both the semi-parametric embedding (SPE) and SIAMESE baselines significantly (across all measurements) after 3000 iterations. This provides strong evidence to support our intuition earlier (Section 1) that as we implicitly express the correlation between items in terms of the metric representation that parameterizes a GP model, a well-fitted GP would induce a well-shaped metric that preserves the averaged similarity geometry of items, which is consistent with our theoretical analysis in Section 4.

5.2 Personalized Metric Learning

Though we obtain positive evidence in Section 5.1 that our SSL-induced metric improves significantly over all baselines, this is measured on the average over the entire user population rather than on each individual user. In the former context, as long as two items A and B are both interacted with by a user in the population, they are considered similar. But, in the (latter) context of a single user, A and B might not be considered similar if the user never interacts with both of them. In this case, we are interested in understanding how well our SSL metric would perform *with* and *without* personalization (see **Q2**), especially on users with limited data. To shed light on this matter, we sample a subset of 20 users with fewer than 200 (but no fewer than 20) item interactions. We then compute a personalizable metric based on the previously computed SSL-induced metric via minimizing Eq. (9). The personalizable metric is then tuned to fit each user's individual rating data using the SSL recipe in Section 3.1.

The MRR, NDCG and HR measurements of the metric (before and after 2000 personalization iterations on the MovieLens dataset) are reported in Figure 3 and Figure 5. Our observations are as follows: (1) on average (over 20 users), the personalized metric consistently shows a sheer improvement over its SSL base in Figure 5; (2) on an individual basis, the personalized MRR, NDCG and HR measurements improve significantly on 16/20, 17/20 and 17/20 users, resulting in success rates of 80%, 85% and 85% (see Figure 3).

We also evaluate and report the individual personalized MRR, NDCG and HR measurements on the Yelp Review dataset for a sampled group of 20 users in Figure 4. In most individual cases, it can again be seen that the personalized metric also improves over the base metric (see Fig. 4). These observations are therefore all consistent with our early observations on the MovieLens dataset.

Remark. Note that, these are averaged measurements over the selected users. For each individual, the measurement is computed based *only* on the corresponding user's personal co-interaction data, rather than on the co-interaction data over the entire population. Thus, this is a stricter performance measurement in comparison to that of Section 5.1, and expectedly so to evaluate personalized performance.

6 Conclusion

This paper develops a self-supervised learning method for item-item metric distance in the context of a recommendation system where direct training examples are not readily available. The developed method instead

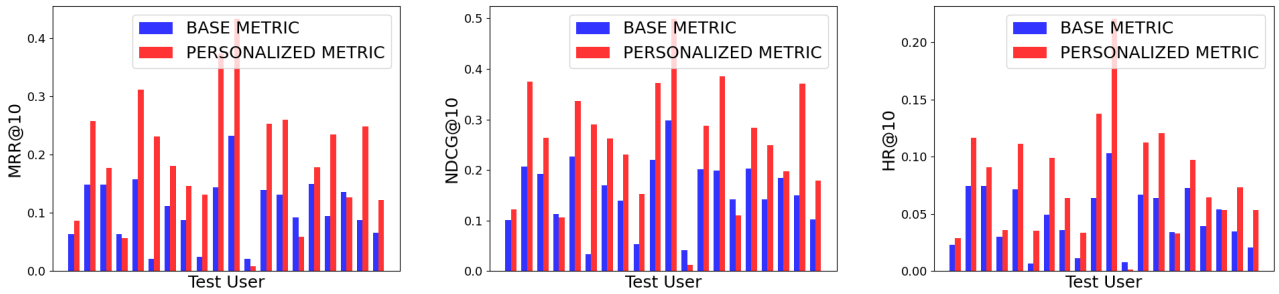

Figure 3: Bar charts of the individual MRR, NDCG and HR measurements of a set of 20 randomly sampled users with fewer than 200 interactions with items in the catalogue. Red (blue) columns reflect the measurement (MRR, NDCG and HR) of the personalized (base) SSL metric on those users.

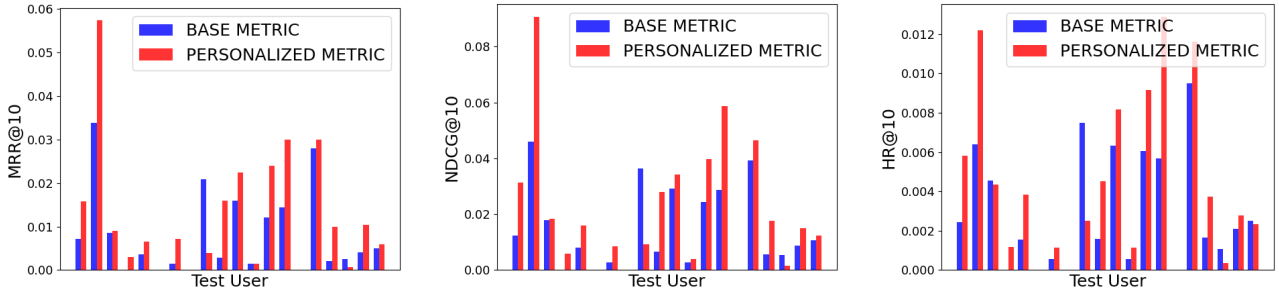

Figure 4: Bar charts of the individual MRR, NDCG and HR measurements of our SSL metric on a set of randomly sampled users with fewer than 200 interactions with items in the catalogue. Red (blue) columns reflect the quality measurement (MRR, NDCG and HR) of the personalized (base) SSL metric on those users.

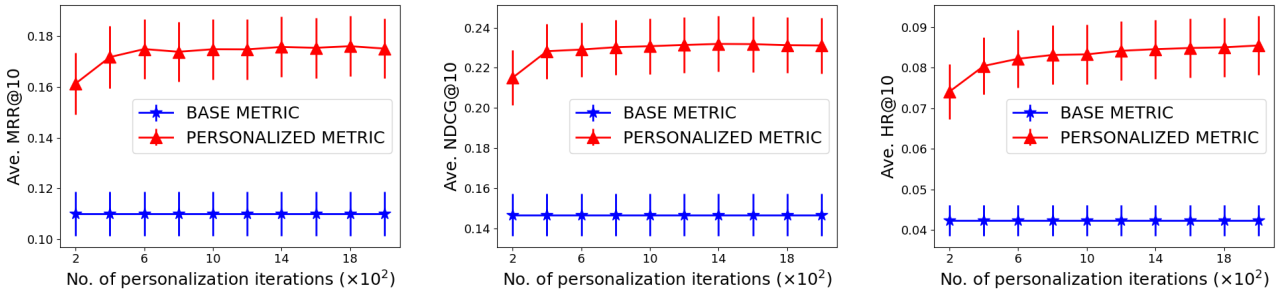

Figure 5: Plots of the MRR, NDCG and HR measurements of our personalized metric over 2000 personalization iterations on the MovieLens dataset. Our personalized metric was evaluated separately on each user and the plotted results were averaged over 20 users. Here, its performance measurements on each individual user is computed using only the user's personal co-interaction data, rather than the co-interaction data over the entire population, which is stricter and expectedly so to evaluate personalized performance.

embed the metric representation as kernel parameters of a Gaussian process prediction model, which is then fitted to predict a user-item interaction function whose training examples are more readily available.

Our approach builds on the intuition that similar items would induce similar interactions from the same user, thus learning an interaction prediction model that expresses its prediction in terms of item-item similarities can implicitly guide it towards capturing the right metric. This also reveals a principled method to personalize the item-to-item metric for each user. We show that theoretically, our learning model can recover the right metric up to a certain error threshold with high probability in the limit of data. Its empirical effectiveness is also demonstrated on several real-world datasets.

Societal Impact. Though applications of our work to real data could result in ethical considerations, this is an unpredictable side-effect of our work. We use sanitized public datasets to evaluate its performance. No ethical considerations are raised.

References

[1] A. Bar-Hillel, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations. In *Proc. ICML*, 2003.

[2] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively with application to face verification. In *Proc. CVPR*, pages 539–546, 2005.

[3] J. V. Davis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *Proc. ICML*, pages 209–216, 2003.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[5] C. Domeniconi and D. Gunopulos. Adaptive nearest neighbor classification using support vector machines. In *Proc. NIPS*, 2002.

[6] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proc. ICML*, 2017.

[7] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighborhood components analysis. In *Proc. NIPS*, 2005.

[8] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. Deepfm: a factorization-machine based neural network for ctr prediction. In *Proc. IJCAI*, pages 1725–1731, 2017.

[9] F. M. Harper and J. A. Konstan. The MovieLens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TIIS)*, 5(4):1–19, 2015.

[10] T. Hastie and R. Tibshirani. Discriminant adaptive nearest neighbor classification. *IEEE Pattern Analysis and Machine Intelligence*, 18, 1996.

[11] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proc. WWW*, pages 173–182, 2017.

[12] J. Hensman, N. Fusi, and N. D. Lawrence. Gaussian processes for big data. In *Proc. UAI*, pages 282–290, 2013.

[13] Q. M. Hoang, T. N. Hoang, and K. H. Low. A generalized stochastic variational Bayesian hyperparameter learning framework for sparse spectrum Gaussian process regression. In *Proc. AAAI*, pages 2007–2014, 2017.

[14] Q. M. Hoang, T. N. Hoang, H. Pham, and D. P. Woodruff. Revisiting the sample complexity of sparse spectrum approximation of gaussian processes. In *Proc. NeurIPS*, 2020.

[15] T. N. Hoang, Q. M. Hoang, and K. H. Low. A unifying framework of anytime sparse Gaussian process regression models with stochastic variational inference for big data. In *Proc. ICML*, pages 569–578, 2015.

[16] T. N. Hoang, Q. M. Hoang, and K. H. Low. A distributed variational inference framework for unifying parallel sparse Gaussian process regression models. In *Proc. ICML*, pages 382–391, 2016.

[17] T. N. Hoang, Q. M. Hoang, K. H. Low, and J. P. How. Collective online learning of Gaussian processes in massive multi-agent systems. In *Proc. AAAI*, 2019.

[18] T. N. Hoang, Q. M. Hoang, O. Ruofei, and K. H. Low. Decentralized high-dimensional bayesian optimization with factor graphs. In *Proc. AAAI*, 2018.

[19] P. Hu, R. Du, Y. Hu, and N. Li. Hybrid item-item recommendation via semi-parametric embedding. In *Proc. IJCAI*, pages 2521–2527, 2019.

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

[21] B. Kulis. *Metric Learning: A Survey*. Foundations and Trends in Machine Learning, 2012.

[22] J. T. Kwok and I. W. Tsang. Learning with idealized kernels. In *Proc. ICML*, 2003.

[23] M. Lázaro-Gredilla, J. Quiñonero-Candela, C. E. Rasmussen, and A. R. Figueiras-Vidal. Sparse spectrum Gaussian process regression. *Journal of Machine Learning Research*, pages 1865–1881, 2010.

[24] G. Lebanon. Flexible metric nearest neighbor classification. In *Proc. IJCAI*, 2003.

[25] P. Lops, M. D. Gemmis, and G. Semeraro. Content-based recommender systems: State of the art and trends. In *Recommender Systems Handbook*, pages 73–105, 2011.

[26] J. Lu, J. Hu, and J. Zhou. Deep metric learning for visual understanding: An overview of recent advances. *IEEE Signal Processing Magazine*, 34:76–84, 2017.

[27] A. Mnih and R. Salakhutdinov. Probabilistic matrix factorization. In *Proc. NIPS*, pages 1257–1264, 2007.

[28] R. J. Mooney and L. Roy. Content-based book recommending using learning for text categorization. In *Proc. 5th ACM Conference on Digital Libraries*, pages 195–204, 2000.

[29] X. Ning and G. Karypis. Slim: Sparse linear methods for top-n recommender systems. In *Proc. ICDM*, 2011.

[30] M. J. Pazzani and D. Billsus. Content-based recommendation systems. In *Proc. The Adaptive Web, Methods and Strategies of Web Personalization*, pages 325–341, 2007.

[31] J. Quiñonero-Candela and C. E. Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005.

[32] J. Quiñonero-Candela, C. E. Rasmussen, and C. K. I. Williams. Approximation methods for gaussian process regression. *Large-Scale Kernel Machines*, pages 203–223, 2007.

[33] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[34] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proc. ICML*, pages 880–887, 2008.

[35] A. Schwaighofer and V. Tresp. Transductive and inductive methods for approximate Gaussian process regression. In *Proc. NIPS*, pages 953–960, 2003.

[36] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. Autorec: Autoencoders meet collaborative filtering. In *Proc. WWW*, pages 111–112, 2015.

[37] M. Seeger, C. K. I. Williams, and N. D. Lawrence. Fast forward selection to speed up sparse Gaussian process regression. In *Proc. AISTATS*, 2003.

[38] A. J. Smola and P. L. Bartlett. Sparse greedy Gaussian process regression. In *Proc. NIPS*, pages 619–625, 2001.

[39] E. Snelson and Z. Gharahmani. Sparse Gaussian processes using pseudo-inputs. In *Proc. NIPS*, pages 1259–1266, 2005.

[40] E. L. Snelson. *Flexible and efficient Gaussian process models for machine learning*. Ph.D. Thesis, University College London, London, UK, 2007.

[41] E. L. Snelson and Z. Ghahramani. Local and global sparse Gaussian process approximation. In *Proc. AISTATS*, 2007.

[42] M. K. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *Proc. AISTATS*, 2009.

[43] Romain Warlop, Alessandro Lazaric, and Jérémie Mary. Fighting boredom in recommender systems with linear reinforcement learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[44] E. Xing, A. Ng, M. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. In *Proc. NIPS*, 2003.

[45] X. Yang, P. Zhou, and M. Wang. Person reidentification via structural deep metric learning. *IEEE Transaction on Neural Network Learning System*, pages 1–12, 2018.

[46] Haibin Yu, Trong Nghia Hoang, Kian Hsiang Low, and Patrick Jaillet. Stochastic variational inference for fully bayesian sparse gaussian process regression models. *CoRR*, abs/1711.00221, 2017.

[47] K. Yu, J. Lafferty, S. Zhu, and Y. Gong. Large-scale collaborative prediction using a nonparametric random effects model. In *Proc. ICML*, pages 1185–1192, 2009.

[48] K. Zhang, M. Tang, and J. T. Kwok. Applying neighborhood consistency for fast clustering and kernel density estimation. In *Proc. CVPR*, pages 1001–1007, 2005.

[49] Z. Zhang, J. Kwok, and D. Yeung. Parametric distance metric learning with label information. In *Proc. IJCAI*, 2003.

Supplementary Material: Learning Personalized Item-to-Item Recommendation Metric via Implicit Feedback

A Additional Experimental Details

A.1 Experiment Setup

In our experiment setup, each dataset is pre-processed in the following forms that describe separately the item's behavior data (e.g., user-item interactions) and its content information (e.g., the meta data of an item that is user-independent). This includes:

Interaction Data. This is a collection of tuples (*user*, *item*, *rating*, *timestamp*), each of which describes an event where a *user* gives an *item* a certain *rating* at a certain time marked by *timestamp*. Here, the *timestamp* is in seconds counting from a certain origin in the past. For example, the time origin of the MovieLens dataset is at midnight (UTC) of January 1, 1970.

Meta Data. This comprises multiple channels of different meta data. Each channel is a collection of pairs $(\mathbf{x}, \mathbf{v})$ where $\mathbf{x}$ is the item identification (e.g., a movie ID in a database) and $\mathbf{v}$ is either a scalar, multi-hot vector or a dense embedding vector representing a numerical feature, a categorical feature and an embedding of a text feature. For example, movie ratings would be represented as scalars, movie genres would be represented by multi-hot vectors, whose dimension is the total number of genres. Movie title and/or description would be represented by a 768-dim embedding vector generated by Google's pre-trained BERT [4] models.

Noisy Annotations of Similar (Dissimilar) Pairs. To train a Siamese Net that captures the similarities between items, the standard method is to acquire (noisy) annotations of similar/dissimilar pairs of items. Here, for each user $\mathbf{u}$, we collect and sort the items $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_q\}$ that $\mathbf{u}$ has interacted with in the increasing order of time. For an randomly sampled item $\mathbf{x}_i \sim \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_q\}$, we will sample (independently) h items $\{\mathbf{x}_1^+, \mathbf{x}_2^+, \ldots, \mathbf{x}_h^+\}$ within a forward κ-step window $\{\mathbf{x}_{i+1}, \mathbf{x}_{i+2}, \ldots, \mathbf{x}_{i+\kappa}\}$. This forms h positive (similar) pairs $\{(\mathbf{x}_i, \mathbf{x}_\iota^+)\}_{\iota=1}^h$ per user. We also sample randomly (over the item catalogue) another h items $\{\mathbf{x}_1^-, \ldots, \mathbf{x}_h^-\}$ to form another h negative pairs $\{(\mathbf{x}_i, \mathbf{x}_\iota^-)\}_{\iota=1}^h$.

Training Siamese Net Baseline. This results in a dataset $\{(\mathbf{x}_a, \mathbf{x}_b, y_{ab})\}$ where $y_{ab} = 0$ indicates $(\mathbf{x}_a, \mathbf{x}_b)$ is a pair of similar items and otherwise for $y_{ab} = 1$ (see Section 2.1). Here, the items $\mathbf{x}_a$ and $\mathbf{x}_b$ are represented as lists of meta vectors (one per channel), $\mathbf{x}_a = [\mathbf{v}_a^1, \ldots, \mathbf{v}_a^p]$ and $\mathbf{x}_b = [\mathbf{v}_b^1, \ldots, \mathbf{v}_b^p]$, respectively. This dataset will then be used to train a Siamese ensemble (see Section 3.2) which learns a separate individual Siamese distance $\mathbf{D}_i(\mathbf{v}_a^i, \mathbf{v}_b^i)$ per meta channel, and combines them via Eq. (6). $\mathbf{D}_i(\mathbf{v}_a^i, \mathbf{v}_b^i)$ is computed via Eq. (5) which is expressed in the abstract form of a feature embedding tower $\mathbf{F}_\mathbf{v}$ and a scale matrix $\mathbf{\Lambda}$. Both of which are detailed next in Appendix A.2.

Item Test Set. The above annotation is restricted to items that appears before a certain time T. An item is considered to appear before T if the earliest time it was interacted with by a user is before T. In our experiment, T is selected such that the no. of test item is about 5% of the item catalogue.

A.2 Model Parameterization

This section elaborates further on the Siamese ensemble architecture that we mention in the main text. First, this refers to Eq. (6) which breaks down the overall item metric into individual metric across multiple meta channels. Second, each individual metric is characterized in Eq. (2) which concerns a feature embedding tower $\mathbf{F}(\mathbf{x}; \boldsymbol{\gamma})$ that maps the meta data vector of a single channel[5] into a metric space equipped with a Mahalonobis distance parameterized by a diagonal scale matrix $\mathbf{\Lambda}$. Here, we set $\mathbf{\Lambda} = \mathbf{I}$ seeing that its values on the diagonal can be absorbed into the parameterization $\boldsymbol{\gamma}$ of $\mathbf{F}(\mathbf{x}; \boldsymbol{\gamma})$, which is parameterized separately for each channel, as detailed below.

For each channel m of meta data, the corresponding feature embedding tower $\mathbf{F}_m(\mathbf{x}; \boldsymbol{\gamma})$ is a feed-forward net comprising 3 dense layers with $2 \times h$, h and h hidden units where $h = 50$ in our experiments. The layers are activated by a ReLu (RL), sigmoid and tanh functions in that order. That is, $\mathbf{F}_m(\text{meta}_m(\mathbf{x}); \boldsymbol{\gamma}_m) \triangleq$

$$\tanh\left(\mathbf{W}_t^m \sigma\Big(\mathbf{W}_s^m \text{RL}\Big(\mathbf{W}_o^m \text{vec}(\mathbf{x}) + \mathbf{b}_o^m\Big) + \mathbf{b}_s^m\Big) + \mathbf{b}_t^m\right)$$

where $\mathbf{W}_o^m \in \mathbb{R}^{2h \times d}$, $\mathbf{W}_s^m \in \mathbb{R}^{h \times 2h}$ and $\mathbf{W}_t^m \in \mathbb{R}^{h \times h}$

[5] Here, we abuse the notation $\mathbf{x}$ to represent a single-channel meta vector whereas previously, $\mathbf{x}$ was also used to denote a list of such single-channel meta vectors. Nonetheless, we believe this notation abuse does not impact the readability of the section since the context is clear.

are learnable affine transformation weights. Likewise, $\mathbf{b}_o^m \in \mathbb{R}^{2h}$, $\mathbf{b}_s^m \in \mathbf{R}^h$ and $\mathbf{b}_t^m \in \mathbf{R}^h$ are learnable bias vectors. Here, the vec or Flatten operator reshapes the input tensor $\mathbf{x}$ into a column vector of d dimensions. The other tanh, sigmoid and ReLu are applied point-wise to components of their input vectors or matrices. Thus, in short, we have $\boldsymbol{\gamma}_m = \{(\mathbf{W}_t^m, \mathbf{b}_t^m), (\mathbf{W}_s^m, \mathbf{b}_s^m), (\mathbf{W}_o^m, \mathbf{b}_o^m)\}$ as learnable parameters for $\mathbf{F}_m$.

Furthermore, in addition to the above parametric embedding of each meta channel, we also have a non-parametric embedding tower that maps from each item ID (i.e., an integer scalar) to a unique continuous p-dimensional vector. This ID embedding tower is non-parametric since its number of parameters is proportional to the number of items in the catalogue,

$$\mathbf{F}_{\text{ID}}\left(\text{ID}\Big(\mathbf{x}\Big); \boldsymbol{\gamma}_{\text{ID}}\right) \triangleq \text{Flatten}\left(\text{Embedding}\Big(\text{ID}(\mathbf{x}); \boldsymbol{\gamma}\Big)\right)$$

Here, $\boldsymbol{\gamma}_{\text{ID}}$ comprises $p \times n$ learnable scalars where n is the total number of items. The Flatten operator again reshapes the output into a p-dimension column vector. In our experiment, $p = 30$. For more detail, our experimental code is also included in the supplement.

B Proof of Lemma 1

Lemma 1. Assuming $\kappa_u^i(\mathbf{w})$ is twice-differentiable at $\mathbf{w} = \mathbf{0}$, the approximation of $\kappa_u^i(\mathbf{w})$ with its 2^{nd}-order Taylor expansion around $\mathbf{w} = \mathbf{0}$ induces the following,

$$\nabla_{\mathbf{w}} \mathbf{L}(\mathbf{w}_+) = \frac{1}{q} \sum_{u=1}^{q} \left(\Big[\mathbf{D}_{\mathbf{w}} \kappa_u(\mathbf{w}_+)\Big] \nabla_{\mathbf{w}} \ell_u(\kappa_u(\mathbf{w}_+)) \right) \tag{18}$$

with $\mathbf{D}_{\mathbf{w}} \kappa_u(\mathbf{w}_+) \triangleq \Big[\nabla_{\mathbf{w}}^\top \kappa_u^1(\mathbf{w}_+); \ldots; \nabla_{\mathbf{w}}^\top \kappa_u^{p+1}(\mathbf{w}_+)\Big]$ whose rows are approximated via

$$\nabla_{\mathbf{w}} \kappa_u^i\Big(\mathbf{w}_+\Big) \simeq \nabla_{\mathbf{w}} \kappa_u^i\Big(\mathbf{0}\Big) + \Big[\nabla_{\mathbf{w}}^2 \kappa_u^i\Big(\mathbf{0}\Big)\Big] \ \mathbf{w}_+ \ . \tag{19}$$

Proof. First, it is straight-forward to see that Eq. (18) above can be derived by taking derivative on both sides of Eq. (9) and the RHS of Eq. (18) concerning the Jacobian $\mathbf{D}_{\mathbf{w}} \kappa_u(\mathbf{w}_+))$ is simply the result of the chain rule of differentiation.

Thus, what remains to be proved is how one arrive at Eq. (19) assuming the 2^{nd}-order Taylor expansion of $\kappa_u^i(\mathbf{w})$ exists around $\mathbf{w} = \mathbf{0}$ and can be used as a reasonable approximation. To see this, let us express the 2^{nd}-order Taylor of $\kappa_u^i(\mathbf{w})$ around $\mathbf{0}$ explicitly below:

$$\begin{aligned} \kappa_u^i\Big(\mathbf{w}\Big) &\simeq \kappa_u^i\Big(\mathbf{0}\Big) + \mathbf{w}^\top \Big[\nabla_{\mathbf{w}} \kappa_u^i\Big(\mathbf{0}\Big)\Big] \\ &+ \frac{1}{2} \mathbf{w}^\top \Big[\nabla_{\mathbf{w}}^2 \kappa_u^i\Big(\mathbf{0}\Big)\Big] \mathbf{w} \ . \end{aligned} \tag{20}$$

Taking derivative with respect to $\mathbf{w}$ on both sides of Eq. (21) and evaluating both at $\mathbf{w} = \mathbf{w}_+$ yield,

$$\nabla_{\mathbf{w}} \kappa_u^i\Big(\mathbf{w}\Big) \simeq \nabla_{\mathbf{w}} \kappa_u^i\Big(\mathbf{0}\Big) + \Big[\nabla_{\mathbf{w}}^2 \kappa_u^i\Big(\mathbf{0}\Big)\Big] \mathbf{w}_+ \ , \tag{21}$$

which is the desired approximation above. □

C Proof of Lemma 2

Lemma 2. Suppose $(1-\epsilon) \cdot k_*(\mathbf{x}, \mathbf{x}') \leq k(\mathbf{x}, \mathbf{x}') \leq (1+\epsilon) \cdot k_*(\mathbf{x}, \mathbf{x}')$ for $\epsilon \in (0, 1)$ then

$$\Big|\mathbf{D}(\mathbf{x}, \mathbf{x}') - \mathbf{D}_*(\mathbf{x}, \mathbf{x}')\Big| \quad \leq \quad 2 \log\left(\frac{1}{1-\epsilon}\right) \ . \tag{22}$$

Proof. From the above premises, we have

$$\begin{aligned} 1 + \epsilon &\geq k(\mathbf{x}, \mathbf{x}') / k_*(\mathbf{x}, \mathbf{x}') \\ &= \exp\left(\frac{1}{2} \cdot \Big(\mathbf{D}_*(\mathbf{x}, \mathbf{x}') - \mathbf{D}(\mathbf{x}, \mathbf{x}')\Big)\right) \end{aligned} \tag{23}$$

which immediately implies $\mathbf{D}_*(\mathbf{x}, \mathbf{x}') - \mathbf{D}(\mathbf{x}, \mathbf{x}') \leq 2\log(1+\epsilon)$. Likewise, repeating the same exercise for $k(\mathbf{x}, \mathbf{x}')/k_*(\mathbf{x}, \mathbf{x}') \geq 1 - \epsilon$ implies $\mathbf{D}_*(\mathbf{x}, \mathbf{x}') - \mathbf{D}(\mathbf{x}, \mathbf{x}') \geq -2\log(1/(1-\epsilon))$. Thus, combining the above, it follows that

$$\left|\mathbf{D}(\mathbf{x}, \mathbf{x}') - \mathbf{D}_*(\mathbf{x}, \mathbf{x}')\right| \quad \leq \quad 2 \log\left(\frac{1}{1-\epsilon}\right) \tag{24}$$

which holds because for $\epsilon \in (0, 1)$, we have $1/(1-\epsilon) \geq 1 + \epsilon$. □

D Proof of Lemma 3

Lemma 3 Suppose $\sup_{(\mathbf{x}, \mathbf{x}')} \left|\frac{k(\mathbf{x}, \mathbf{x}') - k_*(\mathbf{x}, \mathbf{x}')}{k_*(\mathbf{x}, \mathbf{x}')}\right| \leq \epsilon$ for $\epsilon \in (0, 1)$ then

$$\forall (\mathbf{x}, \mathbf{x}') : \Big|\mathbf{D}(\mathbf{x}, \mathbf{x}') - \mathbf{D}_*(\mathbf{x}, \mathbf{x}')\Big| \leq 2 \log\left(\frac{1}{1-\epsilon}\right) \tag{25}$$

Proof. $\sup_{(\mathbf{x}, \mathbf{x}')} \left|\frac{k(\mathbf{x}, \mathbf{x}') - k_*(\mathbf{x}, \mathbf{x}')}{k_*(\mathbf{x}, \mathbf{x}')}\right| \leq \epsilon$ implies

$$\begin{aligned} \forall (\mathbf{x}, \mathbf{x}') : k_*(\mathbf{x}, \mathbf{x}')(1 - \epsilon) &\leq k(\mathbf{x}, \mathbf{x}') \\ &\leq k_*(\mathbf{x}, \mathbf{x}')(1 + \epsilon) \end{aligned} \tag{26}$$

Thus, by Lemma 2, Eq. (26) subsequently implies:

$$\forall (\mathbf{x}, \mathbf{x}') : \Big|\mathbf{D}(\mathbf{x}, \mathbf{x}') - \mathbf{D}_*(\mathbf{x}, \mathbf{x}')\Big| \leq 2 \log\left(\frac{1}{1-\epsilon}\right) \tag{27}$$

□

E Proof of Theorem 1

To prove Theorem 1, we need to first establish the following auxiliary results:

Lemma 4. *Let* $\mathbf{r} \sim \mathbb{N}(0, \mathbf{K}_*)$ *and* $\mathbf{A}$ *denote a positive semi-definite and symmetric matrix. We have:*

$$\forall \lambda > 0: \ \mathbb{E}\left[\exp\left(\lambda \mathbf{r}^\top \mathbf{A}\mathbf{r}\right)\right] \ = \ |\mathbf{I} - 2\lambda \mathbf{A}\mathbf{K}_*|^{-\frac{1}{2}} \quad (28)$$

where the expectation is over the distribution of $\mathbf{r}$.

Proof. Note that for any $\mathbf{r} \sim \mathbb{N}(0, \mathbf{B})$ where $\mathbf{B}$ is a symmetric, positive semi-definite matrix,

$$p(\mathbf{r}) \ = \ (2\pi)^{-\frac{n}{2}} |\mathbf{B}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\mathbf{r}^\top \mathbf{B}^{-1}\mathbf{r}\right) . \quad (29)$$

This also implies

$$\int_{\mathbf{r}} \exp\left(-\frac{1}{2}\mathbf{r}^\top \mathbf{B}^{-1}\mathbf{r}\right) \mathrm{d}\mathbf{r} \ = \ (2\pi)^{\frac{n}{2}} |\mathbf{B}|^{\frac{1}{2}} \quad (30)$$

since $p(\mathbf{r})$ must integrate to one. On the other hand, we can expand $\mathbb{E}\left[\exp\left(\lambda \mathbf{r}^\top \mathbf{A}\mathbf{r}\right)\right]$

$$\begin{aligned}
&= (2\pi)^{-\frac{n}{2}} |\mathbf{K}_*|^{-\frac{1}{2}} \int_{\mathbf{r}} \exp\left(\lambda \mathbf{r}^\top \mathbf{A}\mathbf{r}\right) \exp\left(-\frac{1}{2}\mathbf{r}^\top \mathbf{K}_*^{-1}\mathbf{r}\right) \mathrm{d}\mathbf{r} \\
&= (2\pi)^{-\frac{n}{2}} |\mathbf{K}_*|^{-\frac{1}{2}} \int_{\mathbf{r}} \exp\left(-\frac{1}{2}\mathbf{r}^\top \left(\mathbf{K}_*^{-1} - 2\lambda\mathbf{A}\right)\mathbf{r}\right) \mathrm{d}\mathbf{r} \\
&= (2\pi)^{-\frac{n}{2}} |\mathbf{K}_*|^{-\frac{1}{2}} (2\pi)^{\frac{n}{2}} |\mathbf{B}|^{\frac{1}{2}} = |\mathbf{K}_*|^{-\frac{1}{2}} |\mathbf{B}|^{\frac{1}{2}} \quad (31)
\end{aligned}$$

where $\mathbf{B} = (\mathbf{K}_*^{-1} - 2\lambda\mathbf{A})^{-1}$ and the second last step above follows from Eq. (30). Furthermore, since $\mathbf{B} = (\mathbf{K}_*^{-1} - 2\lambda\mathbf{A})^{-1} = ((\mathbf{I} - 2\lambda\mathbf{A}\mathbf{K}_*)\mathbf{K}_*^{-1})^{-1} = \mathbf{K}(\mathbf{I} - 2\lambda\mathbf{A}\mathbf{K}_*)^{-1}$, it follows that $|\mathbf{B}| = |\mathbf{K}_*||\mathbf{I} - 2\lambda\mathbf{A}\mathbf{K}_*|^{-1}$. Plugging this into Eq. (31) yields

$$\begin{aligned}
\mathbb{E}\left[\exp\left(\lambda \mathbf{r}^\top \mathbf{A}\mathbf{r}\right)\right] &= |\mathbf{K}_*|^{-\frac{1}{2}} |\mathbf{B}|^{\frac{1}{2}} \\
&= |\mathbf{K}_*|^{-\frac{1}{2}} |\mathbf{K}_*|^{\frac{1}{2}} |\mathbf{I} - 2\lambda\mathbf{A}\mathbf{K}_*|^{-\frac{1}{2}} \\
&= |\mathbf{I} - 2\lambda\mathbf{A}\mathbf{K}_*|^{-\frac{1}{2}} . \quad (32)
\end{aligned}$$

As the above is true for all $\lambda > 0$, our proof is completed. □

Lemma 5. *Let* $\mathbf{r} \sim \mathbb{N}(0, \mathbf{K}_*)$ *where* $\mathbf{r} = [r(\mathbf{x}_1), r(\mathbf{x}_2), \ldots, r(\mathbf{x}_n)]$. *Suppose we use* $\mathbf{r}$ *as observations of a surrogate feedback to fit our self-supervised GP model in Section 3.1 and let* $\widehat{\mathbf{r}}$ *denote the prediction made by the fitted GP at the same set of training inputs* $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, *we have*

$$(\mathbf{r} - \widehat{\mathbf{r}})^\top \mathbf{A} (\mathbf{r} - \widehat{\mathbf{r}}) \ = \ \sigma^4 \mathbf{r}^\top \left(\mathbf{K} + \sigma^2\mathbf{I}\right)^{-1} \mathbf{A} \left(\mathbf{K} + \sigma^2\mathbf{I}\right)^{-1} \mathbf{r} \quad (33)$$

where $\mathbf{A}$ *is a square matrix and* σ^2 *is the variance of the GP likelihood as defined in Eq.* (3).

Proof. Applying Eq. (3) on $\mathbf{x}_* = \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ and $\mathbf{y} = \mathbf{r}$ and collecting the results in a column vector $\widehat{\mathbf{r}}$ straight-forwardly yields

$$\widehat{\mathbf{r}} \ = \ \mathbf{K}\left(\mathbf{K} + \sigma^2\mathbf{I}\right)^{-1} \mathbf{r} , \quad (34)$$

which further implies

$$\begin{aligned}
\mathbf{r} - \widehat{\mathbf{r}} &= \mathbf{r} - \mathbf{K}\left(\mathbf{K} + \sigma^2\mathbf{I}\right)^{-1} \mathbf{r} \quad (35) \\
&= \mathbf{r} - \mathbf{K}\left(\mathbf{K} + \sigma^2\mathbf{I}\right)^{-1} \mathbf{r} - \sigma^2\mathbf{I}\left(\mathbf{K} + \sigma^2\mathbf{I}\right)^{-1} \mathbf{r} \\
&+ \sigma^2\mathbf{I}\left(\mathbf{K} + \sigma^2\mathbf{I}\right)^{-1} \mathbf{r} \quad (36) \\
&= \mathbf{r} - \left(\mathbf{K} + \sigma^2\mathbf{I}\right)\left(\mathbf{K} + \sigma^2\mathbf{I}\right)^{-1} \mathbf{r} \\
&+ \sigma^2\left(\mathbf{K} + \sigma^2\mathbf{I}\right)^{-1} \mathbf{r} \quad (37) \\
&= \mathbf{r} - \mathbf{r} + \sigma^2\left(\mathbf{K} + \sigma^2\mathbf{I}\right)^{-1} \mathbf{r} \\
&= \sigma^2\left(\mathbf{K} + \sigma^2\mathbf{I}\right)^{-1} \mathbf{r} . \quad (38)
\end{aligned}$$

Finally, plugging Eq. (38) into the expression of $(\mathbf{r} - \widehat{\mathbf{r}})^\top \mathbf{A} (\mathbf{r} - \widehat{\mathbf{r}})$ produces the desired result. □

Given the above results in Lemma 4 and Lemma 5, we are now ready to (re-)state and prove the key result in Theorem 1 below.

Theorem 1 Let $g(\tau) \triangleq \log(\tau) + (1/\tau) - 1$ and $c_\epsilon \triangleq \epsilon\lambda_{\max}\Big(\mathbf{K}_*\Big)/d$, we have

$$\begin{aligned}
&\Pr\left(\sup_{(\mathbf{x},\mathbf{x}')} \left|\frac{k(\mathbf{x},\mathbf{x}') - k_*(\mathbf{x},\mathbf{x}')}{k_*(\mathbf{x},\mathbf{x}')}\right| \ \geq \ \epsilon\right) \\
\leq \ &\exp\left(-\frac{1}{2}n \cdot g\left(\frac{\sigma^4}{\alpha}\Big(1 - c_\epsilon\Big)\lambda_{\max}\Big(\mathbf{K}_*\Big)\right)\right) \quad (39)
\end{aligned}$$

where d and α are defined in **A1** and **A2** above.

Proof. To prove the above result, note that

$$\begin{aligned}
&\Pr\left(\sup_{(\mathbf{x},\mathbf{x}')} \left|\frac{k(\mathbf{x},\mathbf{x}') - k_*(\mathbf{x},\mathbf{x}')}{k_*(\mathbf{x},\mathbf{x}')}\right| \ \geq \ \epsilon\right) \\
= \ &\Pr\left(\left|k(\mathbf{x}_a,\mathbf{x}_b) - k_*(\mathbf{x}_a,\mathbf{x}_b)\right| \ \geq \ \epsilon \cdot k_*(\mathbf{x}_a,\mathbf{x}_b)\right) \quad (40)
\end{aligned}$$

where $(\mathbf{x}_a, \mathbf{x}_b)$ is any pair of inputs for which the corresponding multiplicative approximation error meets the defined supremum value. Thus, let $\mathbb{E}$ be the event that $\left|k(\mathbf{x}_a,\mathbf{x}_b) - k_*(\mathbf{x}_a,\mathbf{x}_b)\right| \ \geq \ \epsilon \cdot k_*(\mathbf{x}_a,\mathbf{x}_b)$. By

assumptions **A1** and **A2**, we have $\Pr(\mathbb{E})$

$$\begin{aligned}
&\leq \Pr\left(\sup_{(\mathbf{x},\mathbf{x}')} \left|k(\mathbf{x},\mathbf{x}') - k_*(\mathbf{x},\mathbf{x}')\right| \;\geq\; \epsilon \cdot k_*(\mathbf{x}_a,\mathbf{x}_b)\right)\\
&\leq \Pr\left(\sup_{(\mathbf{x},\mathbf{x}')} \left|k(\mathbf{x},\mathbf{x}') - k_*(\mathbf{x},\mathbf{x}')\right| \;\geq\; \epsilon \cdot \frac{\lambda_{\max}\left(\mathbf{K}_*\right)}{d}\right)\\
&\leq \Pr\left(\frac{(\mathbf{r}-\widehat{\mathbf{r}})^\top \mathbf{A}(\mathbf{r}-\widehat{\mathbf{r}}) - \alpha}{(\mathbf{r}-\widehat{\mathbf{r}})^\top \mathbf{A}(\mathbf{r}-\widehat{\mathbf{r}})} \;\geq\; c_\epsilon\right)\\
&= \Pr\left((1-c_\epsilon)(\mathbf{r}-\widehat{\mathbf{r}})^\top \mathbf{A}(\mathbf{r}-\widehat{\mathbf{r}}) \;\geq\; \alpha\right) \qquad (41)
\end{aligned}$$

with $c_\epsilon = \epsilon \cdot \lambda_{\max}(\mathbf{K}_*)/d$ and $\mathbf{A} = (1/n)(\mathbf{K}+\sigma^2\mathbf{I})^2$ as defined in **A2**. Here, the second and third inequalities in the above follow from **A1** and **A2**, respectively. Next, applying Lemma 5 to the RHS of Eq. (41) with $\mathbf{A} = (1/n)(\mathbf{K}+\sigma^2\mathbf{I})^2$, it follows that $\forall \lambda > 0$:

$$\begin{aligned}
\Pr\Big(\mathbb{E}\Big) &\leq \Pr\left((1-c_\epsilon)(\mathbf{r}-\widehat{\mathbf{r}})^\top \mathbf{A}(\mathbf{r}-\widehat{\mathbf{r}}) \geq \alpha\right)\\
&= \Pr\left(\frac{\sigma^4}{n}(1-c_\epsilon)\mathbf{r}^\top\mathbf{r} \geq \alpha\right)\\
&= \Pr\left(\exp\left(\lambda \cdot \frac{\sigma^4}{n}(1-c_\epsilon)\mathbf{r}^\top\mathbf{r}\right) \geq \exp\left(\lambda\cdot\alpha\right)\right)\\
&\leq \mathbb{E}\left[\exp\left(\lambda \cdot \frac{\sigma^4}{n}(1-c_\epsilon)\mathbf{r}^\top\mathbf{r}\right)\right]\exp\left(-\lambda\cdot\alpha\right)\\
&= \left|\mathbf{I} - 2\lambda\cdot\frac{\sigma^4}{n}(1-c_\epsilon)\mathbf{K}_*\right|^{-\frac{1}{2}}\exp\left(-\lambda\cdot\alpha\right)\\
&= \exp\left(-\frac{1}{2}\log\left|\mathbf{I} - 2\lambda\cdot\frac{\sigma^4}{n}(1-c_\epsilon)\mathbf{K}_*\right| - \lambda\cdot\alpha\right)\\
&= \exp\left(\mathrm{F}(\lambda)\right) \qquad (42)
\end{aligned}$$

where $\mathrm{F}(\lambda) = -\frac{1}{2}\log\left|\mathbf{I} - 2\lambda\cdot\frac{\sigma^4}{n}(1-c_\epsilon)\mathbf{K}_*\right| - \lambda\cdot\alpha$. Here, the second step above follows from Lemma 5. The fourth step follows from the Markov inequality and the fifth step results from applying Lemma 4 when we replace λ (in Lemma 4) by $\lambda\cdot\frac{\sigma^4}{n}(1-c_\epsilon)$ and $\mathbf{A} = \mathbf{I}$.

Furthermore, we also note that Eq. (42) holds for all $\lambda > 0$, we can tighten the bound on the RHS by solving for λ that minimizes $\mathrm{F}(\lambda)$. To do this, we set $\mathrm{dF}(\lambda)/\mathrm{d}\lambda = 0$ and solve for λ which results in

$$\begin{aligned}
\lambda \;&=\; \frac{n}{2}\cdot\frac{1}{\sigma^4}\cdot\frac{1}{1-c_\epsilon}\cdot\frac{1}{\lambda_{\max}(\mathbf{K}_*)}\\
&\times\; \left(1 - \frac{\sigma^4}{\alpha}\cdot(1-c_\epsilon)\cdot\lambda_{\max}(\mathbf{K}_*)\right) \qquad (43)
\end{aligned}$$

Thus, plugging this optimal value of λ into the RHS of Eq. (42) yields

$$\Pr\Big(\mathbb{E}\Big) \;\leq\; \exp\left(-\frac{n}{2}\cdot g\left(\frac{\sigma^4}{\alpha}(1-c_\epsilon)\lambda_{\max}\Big(\mathbf{K}_*\Big)\right)\right) \qquad (44)$$

where $g(\tau) = \log(\tau) + (1/\tau) - 1$. From Eq. (40) and the definition of $\mathbb{E}$, we have

$$\Pr\left(\sup\left|\frac{k(\mathbf{x},\mathbf{x}') - k_*(\mathbf{x},\mathbf{x}')}{k_*(\mathbf{x},\mathbf{x}')}\right| \geq \epsilon\right) \;=\; \Pr\Big(\mathbb{E}\Big) \qquad (45)$$

Then, combining this with Eq. (44) above yields the desired result. □

F Proof of Theorem 2

Theorem 2 Let $g(\tau) = \log(\tau) + (1/\tau) - 1$ and $g_\epsilon = g\left(\frac{\sigma^4}{\alpha}\left(1 - \lambda_{\max}\Big(\mathbf{K}_*\Big)\frac{\epsilon}{d}\right)\lambda_{\max}\Big(\mathbf{K}_*\Big)\right)$. Then,

$$\begin{aligned}
\Pr\left(\sup_{(\mathbf{x},\mathbf{x}')}\left|\mathbf{D}(\mathbf{x},\mathbf{x}') - \mathbf{D}_*(\mathbf{x},\mathbf{x}')\right| \;&\leq\; 2\log\left(\frac{1}{1-\epsilon}\right)\right)\\
&\geq\; 1-\delta \qquad (46)
\end{aligned}$$

when $n \geq \frac{2}{g_\epsilon}\log\frac{1}{\delta}$ and $\delta \in (0,1)$ is an arbitrarily small confidence parameter.

Proof. Setting $\exp\left(-\frac{n}{2}\cdot g\left(\frac{\sigma^4}{\alpha}(1-c_\epsilon)\lambda_{\max}\Big(\mathbf{K}_*\Big)\right)\right) \leq \delta$ implies

$$\Pr\left(\sup_{(\mathbf{x},\mathbf{x}')}\left|\frac{k(\mathbf{x},\mathbf{x}') - k_*(\mathbf{x},\mathbf{x}')}{k_*(\mathbf{x},\mathbf{x}')}\right| \;\geq\; \epsilon\right) \;\leq\; \delta \qquad (47)$$

via Theorem 1 or equivalently,

$$\Pr\left(\sup_{(\mathbf{x},\mathbf{x}')}\left|\frac{k(\mathbf{x},\mathbf{x}') - k_*(\mathbf{x},\mathbf{x}')}{k_*(\mathbf{x},\mathbf{x}')}\right| \;\leq\; \epsilon\right) \geq 1-\delta \qquad (48)$$

That is, with probability at least $1-\delta$,

$$\sup_{(\mathbf{x},\mathbf{x}')}\left|\frac{k(\mathbf{x},\mathbf{x}') - k_*(\mathbf{x},\mathbf{x}')}{k_*(\mathbf{x},\mathbf{x}')}\right| \;\leq\; \epsilon \qquad (49)$$

which in turn implies

$$\sup_{(\mathbf{x},\mathbf{x}')}\left|\mathbf{D}(\mathbf{x},\mathbf{x}') - \mathbf{D}_*(\mathbf{x},\mathbf{x}')\right| \;\leq\; 2\log\left(\frac{1}{1-\epsilon}\right) \qquad (50)$$

via Lemma 3. This also means

$$\begin{aligned}
\Pr\left(\sup_{(\mathbf{x},\mathbf{x}')}\left|\mathbf{D}(\mathbf{x},\mathbf{x}') - \mathbf{D}_*(\mathbf{x},\mathbf{x}')\right| \;&\leq\; 2\log\left(\frac{1}{1-\epsilon}\right)\right)\\
&\geq\; 1-\delta\,. \qquad (51)
\end{aligned}$$

As this happens when

$$\exp\left(-\frac{n}{2}\cdot g\left(\frac{\sigma^4}{\alpha}(1-c_\epsilon)\lambda_{\max}\Big(\mathbf{K}_*\Big)\right)\right) \;\leq\; \delta\,, \qquad (52)$$

we have

$$\log\frac{1}{\delta} \;\leq\; \frac{n}{2}\cdot g\left(\frac{\sigma^4}{\alpha}(1-c_\epsilon)\lambda_{\max}\Big(\mathbf{K}_*\Big)\right) = \frac{n}{2}\cdot g_\epsilon \qquad (53)$$

which implies n must be at least $\frac{2}{g_\epsilon}\log\frac{1}{\delta}$ as desired. □

G Sparse Gaussian Processes

To improve the scalability of GP model, numerous sparse GP methods [13, 14, 17, 18, 23, 32, 40, 42, 46] have been proposed to reduce its cubic processing cost to linear in the size of data. One common recipe that is broadly adopted by these works is the exploitation of a low-rank approximate representation of the covariance (or Gram) matrix in characterizing the GP prior.

The work of [31] has in fact presented a unifying view of such methods, which share a similar structural assumption of conditional independence (albeit of varying degrees) based on a notion of inducing variables [35, 37, 38, 39, 41]. In particular, under this assumption, there exists a subset of supporting inputs $\mathbb{S} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$ such that conditioning on their (latent) outputs $\mathbf{g}_+ = [g(\mathbf{x}_1)) \ldots g(\mathbf{x}_m)]^\top$, the input space can be partitioned into regions such that if $\mathbf{x}$ and $\mathbf{x}'$ belong to two different regions, then $g(\mathbf{x})$ and $g(\mathbf{x}')$ are statistically independent.

Depending on the specific form the assumed conditional independence, which entails how the input space is partitioned and is different across different methods, the exact covariance matrix $\mathbf{K}$ can be shown to be equivalent to either $\mathbf{Q}$ [37, 38], $\mathbf{Q} - \text{diag}[\mathbf{Q} - \mathbf{K}]$ [39] or $\mathbf{Q} - \text{blkdiag}[\mathbf{Q} - \mathbf{K}]$ [35, 40] (among others). Here, common to all these approximations is the low-rank matrix $\mathbf{Q}$ whose entries $\mathbf{Q}_{ab} \triangleq \mathbf{k}_a^\top \mathbf{K}_{++}^{-1} \mathbf{k}_b$ where $\mathbf{k}_a \triangleq [k(\mathbf{x}_a, \mathbf{x}_1) \ldots k(\mathbf{x}_a, \mathbf{x}_m)]^\top$, $\mathbf{k}_b \triangleq [k(\mathbf{x}_b, \mathbf{x}_1) \ldots k(\mathbf{x}_b, \mathbf{x}_m)]^\top$ and $\mathbf{K}_{++}$ is the Gram matrix induced by the kernel function $k(\mathbf{x}, \mathbf{x}')$ on $\mathbb{S} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$.

By its construction, it is straight-forward to see that $\text{rank}(\mathbf{Q}) \leq m$ and as such, plugging $\mathbf{Q}$ into one of the above approximations of $\mathbf{K}$ and replacing $\mathbf{K}$ in Eq. (4) with the approximation would result in an expression that is computable in $\mathbf{O}(nm^2)$ – via the Woodburry matrix inversion identity – which is linear in n. For interested readers, the exact derivation of these approximations can be found in [31]. In our self-supervised metric learning experiment (Section 3.1), we adopt the simplest approximation with $\mathbf{Q}$ which is sufficient to scale with datasets spanning tens or even hundreds of thousands of items.

Remark. The approximation with $\mathbf{Q}$ [38] is later rediscovered as the result performing variational inference to approximate the tractable (but otherwise costly) GP prediction [42]. This results in an alternative variational perspective that unifies a broader spectrum (including the above) of sparse GPs. Under this new view, the above approximations can be reproduced as the exact result of performing variational inference on a GP with modified prior [15] or noise covariance [16].