Knowledge and Information Systems
https://doi.org/10.1007/s10115-021-01546-8

REGULAR PAPER

PPNW: personalized pairwise novelty loss weighting for novel recommendation

Kachun Lo[1] · Tsukasa Ishigaki[1]

Received: 10 January 2020 / Revised: 31 December 2020 / Accepted: 4 January 2021

Abstract

Most works of recommender systems focus on providing users with highly accurate item predictions based on the assumption that accurate suggestions can best satisfy users. However, accuracy-focused models also create great system bias towards popular items and, as a result, unpopular items rarely get recommended and will stay as "cold items" forever. Both users and item providers will suffer in such scenario. To promote item novelty, which plays a crucial role in system robustness and diversity, previous studies focus mainly on re-ranking a top-N list generated by an accuracy-focused base model. The re-ranking algorithm is thus completely independent of the base model. Eventually, these frameworks are essentially limited by the base model and the separated 2 stages cause greater complication and inefficiency in providing novel suggestions. In this work, we propose a personalized pairwise novelty weighting framework for BPR loss function, which covers the limitations of BPR and effectively improves novelty with negligible decrease in accuracy. Base model will be guided by the novelty-aware loss weights to learn user preference and to generate novel top-N list in only 1 stage. Comprehensive experiments on 3 public datasets show that our approach effectively promotes novelty with almost no decrease in accuracy.

Keywords Recommender systems · Collaborative filtering · Novel recommendation · Personalized recommendation · Loss weighting

1 Introduction

Recommender systems (RSs), studying how to effectively connect entities in the system, have played a crucial role in today's businesses. Especially in online e-commence where overwhelming number of items commonly exists, without proper recommendations, users would have to spend hours to discover things meeting their preferences.

✉ Tsukasa Ishigaki
isgk@tohoku.ac.jp

Kachun Lo
argent.lo.hk@gmail.com

[1] School of Economics and Management, Tohoku University, Sendai, Japan

Published online: 15 February 2021

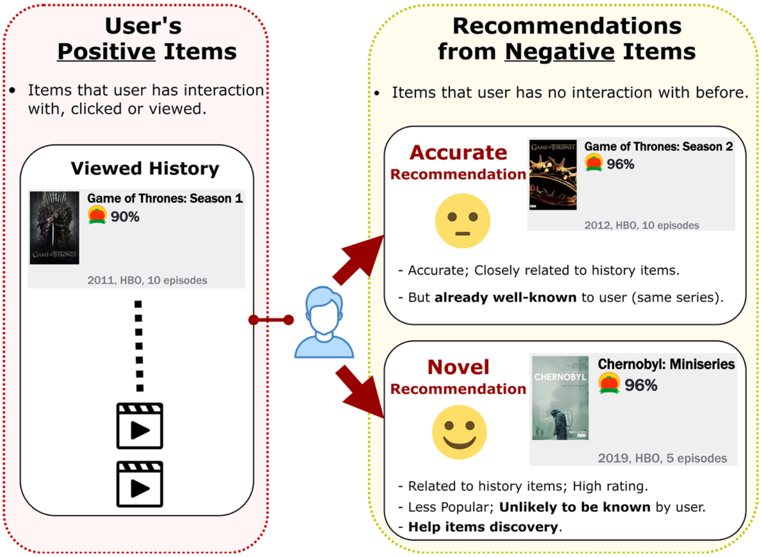

User's Positive Items
• Items that user has interaction with, clicked or viewed.
Viewed History
Game of Thrones: Season 1
90%
2011, HBO, 10 episodes
Recommendations from Negative Items
• Items that user has no interaction with before.
Accurate Recommendation
Game of Thrones: Season 2
96%
2012, HBO, 10 episodes
- Accurate; Closely related to history items.
- But already well-known to user (same series).
Novel Recommendation
Chernobyl: Miniseries
96%
2019, HBO, 5 episodes
- Related to history items; High rating.
- Less Popular; Unlikely to be known by user.
- Help items discovery.

Fig. 1 Illustration of two different strategies of recommendation based on a user's history in implicit feedback setting. Accuracy-focused recommendation might suggest items highly related but well-known to user. While novel recommendation aims at suggesting unpopular items surprising users and helping items discovery. Best view in color (movies information from *Rotten Tomatoes*)

In real-life RSs, "implicit feedback" dominates most applications [16,19]. Unlike explicit feedback, such as ratings, implicit feedback are binary, e.g., whether an item is clicked/viewed by a user or not. Specifically, implicit feedback is usually summarized as {0: **negative item** (unobserved interaction); 1: **positive item** (observed interaction)} [14]. Hence, implicit feedback are much easier to obtain but are weaker signals about users' preferences. In this work, our discussion focuses on the ubiquitous implicit feedback setting, but note that it generalizes to explicit feedback as well, since data like ratings can be easily binarized as implicit feedback. The objective of RSs in the implicit feedback setting is to recommend negative items (unobserved items) to users, based on their historical positive items (observed items) [3,32].

Most academic works of RSs mainly focus on improving accuracy in predicting user–item interactions, assuming that *accurate* recommendations can best satisfy users. Common metrics used to measure a system's accuracy include precision, hit ratio (recall) and normalized discounted cumulative gain (NDCG). Models that include more correct predicted items, which users actually interact later on or in the test set, in the top-N recommendation list are more accurate.

As an example shown in Fig. 1, given a user who has watched "GOT Season 1", an accuracy-focused system is very likely to recommend "GOT Season 2" to her. Since most people would watch season 2 after season 1, this prediction is quite safe for the system to be accurate.

However, focusing solely on accuracy can bring negative effects to both users and item providers, because accuracy-focused models introduce great system bias toward **popular items** [15,29,43,47]. Since popular items are likely to be interacted by most users, accuracy-focus systems would repeatedly promote them to maintain accuracy. In fact, users will encounter similar items which they are well aware of and have less chance to discovery

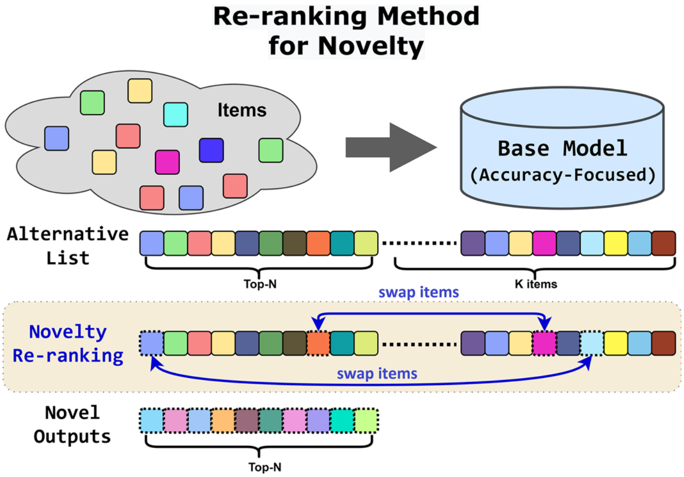


Fig. 2 Illustration of general two-stage re-ranking method for novel recommendation. Principally, an accuracy-based model will be trained first as usual. Then, $K + N$ number of items outputted from the base model are kept to perform re-ranking algorithm based on rules predefined by each work to rank novel items forward. Finally, top N items are kept as the top-N recommendation list. Best view in color

novel interesting items. On the other hand, item providers, especially those who sell unpopular (niche) items, will have a tough time trying to promote items and raise revenue.

A better recommender system should take *"item novelty"* into consideration [15,22,43,44]. In Fig. 1, a novel recommendation "Chernobyl: Miniseries" to this user might be more satisfactory. Not only is this TV show high-rated and related to the user history, which guarantees reasonable accuracy, but it is also less popular and unlikely to be known by the user before, which might surprise user and help her discover cold items among the enormous market.

Since novel recommendation has different objective from accuracy-focused recommendation, promoting novelty inevitably reduces accuracy, indicating a trade-off between novelty and accuracy [29,43]. For instance, on Amazon.com, recommending a popular item, like "bottled water", could safely be accurate, since it's so popular that almost every user would buy it. On the other hand, when promoting novelty, an unpopular novel item, such as a specialized sports beverage, might be suggested to users, which leads to a drop in accuracy as only few users would eventually buy it.

In fact, too accurate recommendations could bored users and undermine item providers, while promoting novel but inaccurate items would confuse users. The trade-off between novelty and accuracy needs to be optimized for a robust recommender system [44].

To construct a system with novel recommendation, most previous studies adopt a two-stage re-ranking style to balance the trade-off between novelty and accuracy [12,17,24].

As illustrated in Fig. 2, generally, re-ranking methods first train an individual accuracy-focused base model for generating a long list candidate items. Then, the candidate list is re-ranked according to predefined rules, which are designed to push novel items to better ranks and, finally, to provide the novel top-N list.

Though two-stage re-ranking algorithms are highly flexible, there are three key limitations:

- As multiple stages are involved, the item retrieving process is rather complicated and verbose.

- The second stage re-ranking algorithm is essentially confined by the separated base model. When the first stage base model is strongly biased to popular items, the alternative list it proposes contains only popular items and excludes novel items. In such case, novel items are blocked earlier in the first stage, causing the postprocessing re-ranking to be less effective.
- Since the two stages are completely independent of each other, the novelty-accuracy trade-off cannot be optimized, and the potential capability of the base model cannot be fully exploited. Ideally, a model, which is trained in one end-to-end step and optimized simultaneously for both novelty and accuracy during training, could better balance the aforementioned trade-off.

Furthermore, to provide novel recommendations given the implicit feedback setting, common loss functions for training base models have two limitations as well.

- In previous works, user's personal preference about novelty has never been explicitly included in loss functions. The majority of loss functions take only observed user–item interactions as inputs [5,10]. Models can learn user novelty preference easier if it is explicitly integrated in the loss function.
- Most loss functions are incapable of distinguishing two types of negative items, "unknown negatives" or "disliked negatives". Specifically, "unknown negatives" are unobserved because they are not popular enough to be discovered by most users, while "disliked negatives" are unobserved because some users dislike them. Ideally, recommended items should be retrieved only from unknown negatives. Since novel items are more likely to be unknown to users, by properly integrating "item novelty" into loss functions, a model would be able to distinguish between the two negatives and to recommend unknown negative rather than disliked negative [15,27].

  Detailed discussion about limitations of loss functions is in Sect. 2.2.2.

To effectively promote novelty as well as to deal with the problems of loss functions, we propose our **Personalized Pairwise Novelty Weighting (PPNW)** approach for one-stage novelty-promoting RSs. Our method naturally summarizes user and item novelty information and explicitly integrates them into a pair-wise loss function to help the base models to learn novelty preference efficiently.

Besides, it also alleviates the problem of loss functions that interesting "unknown negatives" are indistinguishable from "disliked negatives". Since user's unawareness of an item is proportional to the item's novelty [1], a novel item is more likely to be an "unknown" item rather than a "disliked" item. By introducing novelty to the loss function, our method is able to down-weight or up-weight the loss of a negative item according to its novelty level.

Furthermore, our one-stage end-to-end training design enables the base models to optimize directly the trade-off between novelty and accuracy during training. Eventually, PPNW aims at turning the trade-off into a win-win solution, namely recommending more novel items to users with negligible decrease in accuracy. In order to achieve this, our method leverages users' personal preference for novel items. Intuitively, PPNW only tries to recommend novel items to users who have strong preference for novelty, otherwise recommending novel items comes with tremendous cost of accuracy [21,48]. For example, recommending "Game of Thrones" to all users is generally better than recommending an unpopular but also high-rated "Chernobyl: Miniseries". However, for a user with rarefied taste in loving "niche documentary", recommending "Chernobyl: Miniseries" is more likely to satisfy this user than a too popular "Game of Thrones".

Specifically, the PPNW is built by first measuring novelty of users and items based on item popularity. Then, a modified Gaussian RBF kernel is adopted to model how an item

Table 1 List of notations

Symbol	Definition
U, I	Users, items set
u, i	A specific user, item
U_i, I_u	Interacted users set of i, interacted items set of u
i^+	Positive/observed item that a user has interacted with
i^-	Negative/unobserved item that a user hasn't interacted with
$r_{ui}, \hat{r}_{ui}$	True rating, predicted rating of u to i
θ	Novelty score
θ^N, θ^P	Normalized, personalized Novelty score
σ_{θ_u}	Standard deviation of u's Novelty score
$M(u, i)$	Novelty matching score of u and i
λ	Parameter to control novelty preference range
α	Parameter to emphasize novel item
γ	Parameter to control the scale of loss weights
w_{od}	Out-degree weight in graph embedding models

matches a user's novelty preference (or novelty tolerance). Next, we further emphasize novel items by upscaling item novelty score. Finally, to integrate all these information into the loss function, we devise two novelty weighting strategies, by which novelty information would be mapped to a proper scale for loss weighting. On the whole, our method intends to promote items that are both novel and matching specific user's novelty preference. Eventually, as to items, a novel item will be considered more important than a common item; as to users, a user prefers novelty would be recommended more novel items.

To summarize, this paper mainly makes the following contributions:

- We propose PPNW framework for 1-stage novelty-promoting recommender systems. A new loss weighting strategy utilizing novelty information enables end-to-end model training for direct optimization of the novelty-accuracy trade-off.
- The designed novelty matching strategies naturally integrate novelty information of both user and item into the loss function, encouraging the model to distinguish unknown items from disliked items. Moreover, by directly modelling users' personal preferences, the problem of uniform suppression of all negative items during training is alleviated.
- Comprehensive experiments on three public datasets is conducted to evaluate performances on accuracy and novelty. Experimental results show that the proposed method outperforms existing novel recommendation frameworks. In particular, our method improves novelty significantly with slight decrease in accuracy, which marks its promising adaptation for real industrial applications.

2 Preliminary

In this section, we first introduce our notation and then provide the definition and measurement of novelty we use in this paper. Following that, a brief discussion about merits and limitations of BPR loss function is covered.

Notation we use in this paper is summarized in Table 1.

2.1 Measuring novelty

According to previous studies [15,18], item novelty can be defined differently based on variously applications. In this work, we focus on long-existing but unpopular items. An item has been in the system for some time, but it is not popular enough to be seen by most users. Then, to these users, the item is novel [18,46]. Without further explanation, we use the terms "novel" and "unpopular" interchangeably. Since unpopular items make up the majority of many systems, by recommending these items to proper users the substantial amount of potential transactions can be achieved. Furthermore, promoting unpopular items benefits both users and item providers and improves market competition, leading to a more dynamic and robust system [44,47].

According to [1], the item novelty (or popularity) can be measured by its frequency $\theta_i^F = \frac{1}{|U_i|}$, number of users who have interaction with it. The idea is that an item is more likely to be novel when many users have no interaction with it. This raw frequency-based measure is the simplest way to obtain novelty score. However, since items' frequency can differ significantly, this unnormalized measure of score is not practical. A better frequency-based novelty score can be obtained by [20,45,48]:

$$\textbf{item: } \theta_i = \log\left(\frac{|U|}{|U_i|}\right); \textbf{ user: } \theta_u = \frac{1}{|I_u|}\sum_{i\in I_u} r_{ui}\theta_i. \tag{1}$$

For item θ_i, Eq. (1) normalizes the frequency based on the total number of users and then scales it logarithmically. For user θ_u, the rating r_{ui}, which reflects user's satisfaction with the item, is also incorporated. Therefore, the user's preference for novelty is proportional to his/her past items' novelty scores and is corrected by his/her ratings as well.

2.2 Bayesian personalized ranking (BPR)

Since BPR is the most popular and the default loss function for implicit feedback, in this section, we will briefly introduce the BPR and discuss its advantages as well as limitations. Our proposed loss weighting method (Sect. 3) will also be applied on the BPR to cover its shortages and to improve novelty in recommendation.

Before BPR, to train an implicit recommendation model, the typical approach is to frame the task as a regression problem and train it with a common pointwise mean square error (MSE) loss function:

$$\mathcal{L}^{\text{MSE}} = \sum_{i\in I}\sum_{u\in U}\left(r_{ui} - \hat{r}_{ui}\right)^2. \tag{2}$$

BPR, instead, transforms the implicit recommendation task into a classification problem [27]:

$$\mathcal{L}^{\text{BPR}} = \sum_{(u,i^+,i^-)} [\log \sigma(\hat{r}_{ui^+} - \hat{r}_{ui^-})]^2. \tag{3}$$

In BPR, $\sigma(\cdot)$ is sigmoid function and an input is a pair (u, i^+, i^-): a user u with a i^+ and a sampled i^-, where $i^+ \in I_u$ and $i^- \notin I_u$.

2.2.1 Merits over pointwise loss function

First, BPR optimizes the model on a pair level, using two items at a time, not on an instance level. The pairwise comparisons during training provide models with more contextual information. In addition, other than suppressing all predicted scores of negative items uniformly, BPR samples randomly negative items and evaluates the losses according to the relative differences between paired positive and negative items, which enables it to capture users personalized biases toward pairs of items. The idea of relative difference is decisive because the target value of negative items is no longer a fixed value, zero, like in MSE loss.

2.2.2 Limitations of BPR

However, BPR is not flawless. BPR punishes harder when a sampled negative item has high predicted rating during training. However, as discussed in Sect. 1, a negative item can be an "unknown negative", rather than a "disliked negative". In such case, high predicted rating indicates that the user might be interested in this unknown negative and it should be recommended.

To alleviate the problem of wrongly punishing interesting "unknown negatives", it's advantageous to introduce novelty score in BPR to distinguish "unknown" from "disliked" negatives. Moreover, user preference is not explicitly included in the BPR loss function. Models can learn user preference easier if it is explicitly integrated in the loss function.

To deal with these problems as well as to promote novelty in the system, we propose our **Personalized Pairwise Novelty Weighting (PPNW)** approach.

3 Our model

3.1 Novelty adjustment for user and item

In this part, we model user and item novelty representation and adjust them into the form suitable for the subsequent loss weighting.

To help gain a better understanding, the overall structure of PPNW is shown in Fig. 3.

3.1.1 User novelty preference

Each user is assumed to have certain novelty range of interest. For example, users who prefer niche products will have less interest in blockbuster movies. To model user's novelty preference, the nature way is using the mean and standard deviation.

$$\theta_u = \frac{1}{|I_u|} \sum_{i \in I_u} r_{ui}\theta_i \text{ and } \sigma_{\theta_u} = \sqrt{\frac{\sum_{i \in I_u} (\theta_i - \theta_u)^2}{|I_u|}}. \tag{4}$$

A greater value of θ_u corresponds to higher novelty preference. And the value of σ_{θ_u} shows how spread this user's interest of novelty is. Having these statistics of user novelty preference allows us to measure the novelty matching score between a user and an item. We employ a Gaussian Radial Basis Function (RBF) kernel with a controllable parameter λ to measure the Novelty Matching Score.

$$M(u,i) = \exp\left(-\frac{\|\theta_u - \theta_i\|^2}{2 \cdot \lambda \cdot \sigma_{\theta_u}^2}\right), \lambda \geq 1. \tag{5}$$

θ_u and θ_i are the novelty score of user and item. λ here is a hyperparameter, together with σ_{θ_u}, controls how spread the range of user novelty preference is. Most real-world datasets,

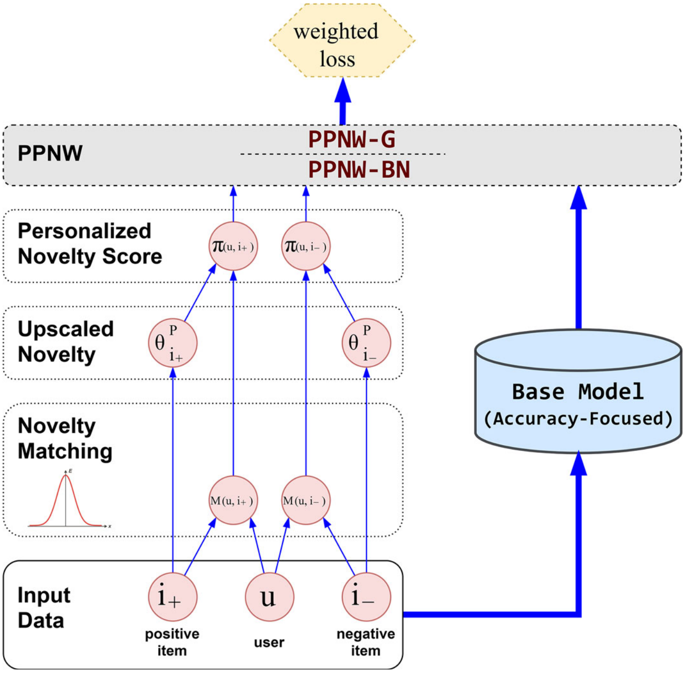


Fig. 3 Illustration of the overall structure of PPNW. The input data are a user and a pair of items, positive and negative, which are then fed into an accuracy-focused base model and the proposed PPNW simultaneously. In the end, PPNW weights the original losses from the base model by PPNW-G or PPNW-BN to guide the optimization toward novelty promoted recommendation

due to their extreme sparsity, consist of many users with narrow novelty preferences. $\lambda \geq 1$ can scale-up the computed novelty preference, allowing novel items to have higher matching score. Empirically, $\lambda \in [1.5, 2.5]$ works well in all our experiments. More detailed effects of lambda in the system is discussed in Sect. 4. The novelty matching score has value range $M(u, i) \in [0, 1]$.

3.1.2 Item novelty normalization

We normalize items' novelty scores using

$$\theta_i^z = \frac{\theta_i - \min(\theta_I)}{\max(\theta_I) - \min(\theta_I)}. \tag{6}$$

After normalization, the distribution is mapped to the bounded region [0, 1] and the scale is desirable for loss weighting.

3.2 Personalized novelty score

In this part, we introduce the personalized novelty score. Primarily, the personalized novelty score should satisfy two requirements:

1. An item should have greater personalized novelty score if it meets the user's preference better.

2. A novel item should be assigned a higher personalized novelty score than a popular item, when these two items have the same distance to a particular user.

Eventually, the personalized novelty score should both emphasize novel items and reflect correctly how well an item matches a user's novelty preference.

To realize these effects, we first raise the normalized item novelty score up to a certain power Eq. 7, resulting in an upscale novelty score. Rather than the original linear scale, this upscale score emphasizes novel items in an exponential scale. Novel items compared with common items, thus, are assigned significantly higher values.

$$\theta_i^P = \left(\theta_i^z\right)^\alpha, \quad \alpha \geq 1. \tag{7}$$

The personalized novelty score is then computed by merging "upscale novelty score" and "novelty matching score":

$$\pi\left(u, i\right) = \theta_i^P \cdot M\left(u, i\right). \tag{8}$$

The personalized novelty score incorporates two decisive factors: emphasis of novelty and personal novelty preference. The two requirements are, thus, achieved by this single score.

3.3 Personalized pairwise novelty weighting (PPNW)

In this section, we convert the personalized novelty score $\pi\left(u, i\right)$ into loss weights W_{loss} for BPR. When $W_{\text{loss}} = 1$, no loss weight is used. Empirically, loss weights having most of its values fall into region $W_{\text{loss}} \in [0, 2]$ is desired while using larger loss weights, like $W_{\text{loss}} \in [1, 10]$, will end up overemphasizing original losses. In addition, negative loss weights should be clipped to 0 because the resulting directions of gradients should not be changed.

We propose two strategies for converting the $\pi\left(u, i\right)$ to loss weight for BPR. In the original BPR without weights, negative item with high predicted rating will be punished badly and positive item with low predicted rating will be encouraged strongly to have higher rating. On the contrary, our weighted BPR further adjusts the losses according to both novelty matching score $M\left(u, i\right)$ and item upscale novelty θ_i^P.

The first loss weighting strategy, we call **"Gamma Matching (PPNW-G)"**, because of its gamma parameter, is highly flexible, allowing preferable performance to be achieved with two more hyperparameters. The second strategy is called **"Batch-Normalization Matching (PPNW-BN)"** which uses batch-normalization to generate weights for loss weighting and is practically free of hyperparameters.

3.3.1 Gamma loss weighting (PPNW-G)

We first model the personalized novelty scores for both positive and negative item for each user, $\pi\left(u, i^+\right)$ and $\pi\left(u, i^-\right)$, as described in Sect. 3.2. Then, the gamma loss weight is obtained by the following equation.

$$w^G_{\left(u, i^+, i^-\right)} = 1 + \gamma \cdot \left[\pi\left(u, i^+\right) - \pi\left(u, i^-\right)\right]. \tag{9}$$

The hyperparameter γ here is to control the strength of loss weighting. Note that when gamma is set to 0, the weighted BPR reduces to the original BPR loss function. In the case where most values of $\left[\pi\left(u, i^+\right) - \pi\left(u, i^-\right)\right]$ are small, say 0.005, the loss weighting has marginal or no effect in the training and a greater value of γ can help amplify the strength.

The gamma loss weights $w^{G}_{(u,i^{+},i^{-})}$ are then integrated in BPR

$$\mathcal{L}^{\text{PPNW}-\text{G}} = \sum_{(u,i^{+},i^{-})} w^{G}_{(u,i^{+},i^{-})} \cdot \log \sigma \left(\hat{r}_{ui^{+}} - \hat{r}_{ui^{-}}\right)^{2} \tag{10}$$

to complete the loss weighting.

While the gamma loss weighting strategy introduces a parameter γ in addition to the previous α and λ, it is able to achieve higher performance.

3.3.2 Batch-normalization loss weighting (PPNW-BN)

Instead of relying on γ to adjust the the strength of loss weighting, PPNW-BN employs batch normalization to automatically execute the adjustment.

$$w^{\text{BN}}_{(u,i+,i-)} = 1 + \text{BN}\left[\pi\left(u, i^{+}\right) - \pi\left(u, i^{-}\right)\right]. \tag{11}$$

BN$(\cdot)$ denotes the mini-batch normalization operation. BN($\mathbf{X}$) normalizes a vector $\mathbf{X}$ to BN($\mathbf{X}$) $\sim (0, 1)$ with most values falling in the region of $[-1, 1]$ and are not likely to be too small or too great. The resulting weights, thus, distribute mainly within the region $w^{\text{BN}}_{(u,i+,i-)} \in [0, 2]$.

$$\mathcal{L}^{\text{PPNW}-\text{BN}} = \sum_{(u,i^{+},i^{-})} w^{BN}_{(u,i^{+},i^{-})} \cdot \log \sigma \left(\hat{r}_{ui^{+}} - \hat{r}_{ui^{-}}\right)^{2}. \tag{12}$$

Moreover, as our experiment implies, PPNW-BN maintains highly stable performance regardless of the choice of α, meaning the PPNW-BN is practically hyperparameter-free. The reason is that α controls the exponential scale of item novelty level θ_{i}^{P}; however, batch normalization maps whatever distribution to a standard form, causing the upscale effect negligible.

3.3.3 Two situations

In this subsection, we discuss, with intuitive examples, the two situations covered by the loss weighting strategies.

- $\pi\left(u, i^{+}\right) \geq \pi\left(u, i^{-}\right)$: indicating the sampled negative item is not better than the positive item in terms of novelty matching ($M\left(u, i\right)$) or novelty level (θ_{i}^{P}). In this situation, to punish this negative item more severely for this user, the paired loss is then emphasized by up-weighting ($w^{G}_{(u,i^{+},i^{-})} \geq 1$ or $w^{\text{BN}}_{(u,i^{+},i^{-})} \geq 1$).
- $\pi\left(u, i^{+}\right) < \pi\left(u, i^{-}\right)$: indicating the sampled negative item is more preferable to this user's novelty taste than the positive item and it's likely to be an "interesting but unknown item". To promote this promising negative item, PPNW down-weights the paired loss to avoid suppressing this negative item too much ($w^{G}_{(u,i^{+},i^{-})}$ or $w^{\text{BN}}_{(u,i^{+},i^{-})} < 1$).

3.4 Scalability and complexity analysis

We analyze the scalability and complexity of PPNW in this section. Since datasets could easily be of extremely large scale in modern RSs, model's capability to scale up is crucial to efficiently provide novel recommendations in real applications.

In fact, PPNW, which only integrates novelty statistics into loss function, can be regarded as a light "plug-in" to the base model.

- Before training, PPNW summarizes novelty statistics of the dataset, i.e., user novelty θ_u and item novelty θ_i, which takes $O(|U| + |I|)$ for processing the whole dataset.
- During training the base model, given one observed positive data, PPNW computes $\pi\left(u, i^+\right)$ $(O(1))$ and $\pi\left(u, i^-\right)$ $(O(1))$ to obtain loss weight $w_{(u,i^+,i^-)}$. Thus, when the number of observed interactions in a dataset is $|I^+|$, the overall complexity of PPNW during training is $O(|I^+|)$.

Note that, the extra complexity that PPNW adds to base model is almost **negligible**. Empirically, in our experiments, the pre-processing time of PPNW on ML-1M dataset is about 5s. And the extra training time that PPNW adds to base model (CMN) is about 6s per epoch (total epoch time is 3.5 min). More detailed experimental settings are in Sect. 4. The reason is that current base models' architectures are very deep [5,38], and their complexity could rise to $O(|I^+| \cdot d^2)$, where d is the embedding size and normally chosen as 150.

3.5 Relation to other loss weighting approach: AllRank-Pop

The proposed method is a general pairwise loss weighting strategy for promoting novelty. Both novelty and personal preference are considered in the loss function, allowing the base model to learn a better trade-off between prediction accuracy and recommending novelty.

A previous study, "AllRank-Pop" [30], utilizing also loss weighting scheme for improving unpopularity in recommendation, is similar to our method conceptually. In fact, AllRank-Pop is a modified version of a more generic "AllRank" loss weighting model [8]. It aims at designing an "unbiased popularity stratified test" and the loss weighting for novelty is not the main focus. In AllRank-Pop, only one kind of item, positive or negative, will be weighted proportional to the inverse of the number of past users, while the left kind of items is weighted by a fixed value.

The proposed model differs from AllRank-Pop in three ways. First, we integrate the loss weights into the pairwise loss function, while they use the pointwise MSE. Second, our model does not treat positive items and negative items differently and every item will be weighted in the same way. In AllRank-Pop, however, either positive or negative items will be assigned a fixed weight. Finally, personal preference is taken into account in our method. Yet, AllRank-Pop weights loss on a global level and personal preference is excluded.

To further compare two methods, our experiments include AllRank-Pop as well. More discussion and experimental details are in Sect. 4.6.

3.6 PPNW on graph embedding base models

In this final part of the section, we first introduce a special type of accuracy-based models, the "graph-based embedding model", and then discuss PPNW's relation with it. Following the relation, we propose to apply PPNW, with merely a few modifications of the base model's objective function, on graph-based model to improve novelty.

3.6.1 Graph-based embedding models for recommendation

Collaborative filtering (CF) assuming that similar users behave similarly in many aspects has arguably become the most important technique in constructing a recommender system. Previously, most researches extend the basic CF on a model/structure level by adding specialized layers to capture complex non-linear relation between users and items. Since DeepWalk [25], LINE [31] and Node2Vec [7] successfully applied embedding learning on graph, graph-based embedding model has caught great attentions of researchers due to its extreme flexibility in dealing most complicated data structures and decent performance.

Recently, **graph-based embedding** has also been studied for doing recommendation and has provided state-of-the-art accuracy performance. These include Hop-Rec [39], NGCF [38] and CSE [2]. In general, graph-based embedding models aim to learn representative embeddings for users and items for recommendation by maintaining proximity relations obtained from the graph during training phase.

3.6.2 PPNW on graph embedding model

To provide superior performance, graph embedding models commonly take into consideration the out-degree of node and use it as loss weight.

Commonly, given a node j (a user or an item), its out-degree weight w_{od_j} is:

$$w_{\mathrm{od}_j} \propto \frac{1}{|U_j|}. \tag{13}$$

The reason that w_{od_j} is inversely proportional to its out-degree is because, in graph information propagation, information from popular nodes (those with high out-degree) are less valuable than information from unpopular nodes (low out-degree). Intuitively, two users must have particular similarity to buy a very unpopular item ignored by most people.

Since novelty is measured by frequency in this work, namely "out-degree", and Eq. 1 is a special case of Eq. 13, the effect of w_{od_j} in graph embedding model has already been captured by PPNW when measuring novelty. More specifically, the upscale novelty score θ_j^P covers the function of the out-degree weight w_{od_j}. Therefore, to apply PPNW on graph-based embedding models requires merely the removal of the original out-degree weight and attaches the loss weight w^G or w^{BN} of PPNW.

4 Experiments

In this section, settings of experiments will be described and experimental results will be discussed in detail.

To support reproducibility, the implementation of our PPNW framework and datasets are publicly available on *Github*.[1]

We construct this section in the following order. First the experimental settings are explained in details, including three datasets, introduction to base models and baseline models. Next, the evaluation metrics and formulas are covered. Finally, we evaluation the effectiveness of the proposed PPNW method and show the experimental results with discussions.

[1] https://github.com/ArgentLo/PPNW-KAIS.

4.1 Datasets

Three publicly available datasets are used to evaluate the effectiveness of our model: Citeulike-a, Pinterest and MovieLens-1M. A description of these datasets are summarized in Table 2. Citeulike-a and Pinterest are extremely sparse settings and ML-1M is denser.

Citeulike-a Citeulike-a consists of implicit feedback data and are collected from CiteULike, an academic papers management website providing its user convenience of saving and sharing academic papers [35].

Pinterest Pinterest is also a dataset containing implicit feedback data of users' interaction with images. The Pinterest dataset is provided by [6] and it's originally used for testing performance of visual recommendation.

MovieLens-1M(ML-1M) ML-1M are constructed by rating data showing users ' explicit preferences to movies. Every user has at least 20 historical ratings and all ratings have values from 1 to 5 with one-star increment [9]. To convert into implicit feedback data, we binarize all entries. In the end, the value of 1 indicates positive interaction and 0 shows no interaction.

4.2 Base models and settings

To evaluate the generalization of the proposed approach, we select 5 accuracy-focused models as base models and our method will be applied upon them. All base models are trained by optimizing the BPR loss function.

Non Graph-based Models

- **Generalized Matrix Factorization (GMF)** [10,26] is one-layer, non-linear generalized model, capable of learning latent representations for users and items.
- **Neural Matrix Factorization (NMF)** [10] is deep network incorporating GMF and multilayer perceptron and jointly training the two parts to capture complicated users and items relationship.
- **Collaborative Memory Network (CMN)** [5] is one of the state-of-the-art models that leverages the advantages of both latent feature and neighborhood representation.

Graph-based Embedding Models

- **High-Order Proximity Recommendation (HOP-Rec)** [39] is a state-of-the-art graph-based CF model which merges the graph information with collaborative relations. The high-order proximity is obtained via data sampling.
- **Neural Graph Collaborative Filtering (NGCF)** [38] is also a state-of-the-art graph-based model which explicitly models the information propagation by stacking "propagation layer" recursively.

Parameter settings For base models, the mini-batch size of both ML-1M and Pinterest is set to 256 and Citeulike-a is set to 128. We use a negative ratio of 6 for all experiments in the BPR loss function. The number of hidden features of GMF and CMN is fixed to 50. On the other hand, for NMF, the last layer has 32 hidden features. In terms of hidden layers, NMF is trained with three layers; CMN with two hops; GMF with one by default. Without further mention, all models are trained using BPR loss function. All graph-based models have embedding size set to 64 as default.

For our proposed method **PPNW**, we set parameters based only on dataset regardless of base model. α is set to 1.5 for Citeulike-a. For Pinterest and ML-1M, average user have

Table 2 Datasets

Dataset	Sparsity (%)	Users	Items	Interactions
Citeulike-a	99.78	5551	16,980	204,987
Pinterest	99.73	55,187	9916	1,500,809
ML-1M	95.53	6040	3706	1,000,209

relatively narrower novelty preference (smaller σ_{θ_u}). To enable user to have wider attention to item, we set the α to 2.5. In BN loss weighting **PPNW-BN**, we use $\lambda = \{2, 2, 7\}$ for ML-1M, Pinterest and Citeulike-a, respectively. In Gamma loss weighting **PPNW-G**(λ), for ML-1M, we set $\gamma = \{40, 125\}$ with PPNW-G($\{2, 4\}$); For Pinterest, we set $\gamma = \{25, 100\}$ with PPNW-G($\{2, 4\}$); for Citeulike-a, we set $\gamma = \{40, 75\}$ with PPNW-G($\{7, 9\}$), respectively. As mentioned in Sect. 3.3.1, these values are chosen empirically to guarantee most weights falling in range of [0, 2]. For simplicity, negative sampling is conducted by uniformly sampling negative items for each user during training and the default negative ratio is set to 6 meaning 6 negative items will be sampled to reinforce 1 positive items.

4.3 Baseline models and settings

Two re-ranking methods and one loss weighting method are used for conducting comparative experiments.

The two **re-ranking methods** are:

- **Personalized Ranking Adaptation (PRA)** [17] is a generic re-ranking framework, applied upon accuracy-focused models, for general purposes, e.g., novelty, diversity and others. In PRA, user preference and item characteristic will be first estimated according to prespecified target or criterion. Then, PRA keeps the first top-N items and the next M items outputted by base model to perform re-ranking. In our experiment, the target of PRA is novelty and the quantity of novelty tendency is measured by using its mean-and-standard-deviation method. Regarding to the parameter settings, we follow the experiments done by Zolaktaf et al. [17] [48]. The sample size of user is set to be $S_u \in \min\left(\left|\mathcal{I}_u^{\mathcal{R}}\right|, 10\right)$; the length of alternative list is $M = 25$; and the maximum steps of swapping is $MaxSteps = 25$.
- **Long-tail Resources Mining (5D)** [12] is a re-ranking framework particularly for long-tail item promotion. To balance various criteria of the re-ranked list, the framework merges multiple targets into a single score, called 5D-score, which summarizes performances in five dimensions, including accuracy, coverage, balance, quantity and quality. The re-ranking takes two steps. First, missing ratings for all possible user–item pairs are predicted by base model and ratings will then be used to distribute limited resources to each item. Second, by taking into consideration of user relative preferences and the item resources, 5D-score is computed and the re-ranking is performed according to it. In this paper, two experiments of 5D are conducted. "5D-Pop" generates top-N based solely on the 5D-score and thus promotes long-tail item remarkably. "5D-RR_ACC" performs 2 extra algorithms, rank of rankings (RR) and accuracy filtering (ACC), after "5D-Pop" to maintain accuracy from base model. We use the default settings of [12], and in the maximization problem, we follow [48] that the parameters are set as: $K = 3|I|$ and $q = 1$.

The novelty **loss weighting method** is:

- **AllRank-Pop** [8,30] is, as discussed in Sect. 3.5, a loss weighting method applied on pointwise loss function, MSE. The loss weight is designed to be proportional to item novelty. In our experiment, we use the "decreasing strategy" variant of the model, which suppresses weights of popular positive items. For negative items, we use a fixed weight $W_{\text{neg}} = 0.005$ and the imputed missing rating is set as $R_{\text{neg}} = 0.4$, given all experiments are based on implicit feedback. Two experiments are implemented, $\beta_w = \{0.9, 0.95\}$, respectively. Higher β_w is supposed to recommend items with greatrer novelty.

  Furthermore, while the AllRank-Pop weights the MSE loss function, the proposed PPNW weights a pairwise loss function. For fair comparisons, we use "**SacrificeRatio**"

$$\text{SacrificeRatio}_{\text{LT}} = \frac{\Delta\text{HR}(\%)}{\Delta\text{L\_Tail}(\%)},$$
$$\text{SacrificeRatio}_{\text{NS}} = \frac{\Delta\text{HR}(\%)}{\Delta\text{Nov\_Score}(\%)}$$

  to measure to what percent of accuracy, HR@10 specifically, a method needs to sacrifice to increase 1% of L_Tail@10 or Nov_Score@10. This gives us a fair comparison when the base loss functions are different.

4.4 Evaluation metrics

Except for graph embedding base models, since deep-learning models require more training data, all our experiments adopt "Leave-one-out" evaluation strategy, following the setting from previous works [5,10,11,27]. In this setup, one positive item from each user is held out randomly for evaluation while other positive items are used for training. During evaluation, as [5,10], the test set is constructed by the holdout item and 100 randomly sampled negative items for each user. The task is to rank the 101 items for each user.

On the other hand, for graph embedding base models, we use the typical 80/20 split to randomly select 80% users' interacted items for training and 20% for testing. To validate both accuracy and novelty of our proposed method, we use the following metrics:
Accuracy metrics

- **Hit Ratio (HR)** is the ratio of the holdout item being included in the top-N list.
- **Normalized Discounted Cumulative Gain (NDCG)** measures accuracy like HR and penalizes greater when the positive item is ranked lower in the list [28].

Novelty metrics

- **Long Tail Ratio (L_Tail)** is the ratio of long-tail items in the top-N list. According to the Pareto principle (or the 80/20 Rule) [40], long-tail items are of the top 80% unpopular items contrary to the 20% most popular items.
- **Average Novelty Score (Nov_Score)** is the average novelty score of the top-N recommended items [15].

All our experimental results are evaluated on the top-10 list.

4.5 Comparison with re-ranking methods

In this section, models are conducted on 3 datasets: Citeulike-a, ML-1M and Pinterest. We compare our methods, PPNW-BN and PPNW-G, with PRA, 5D-Pop and 5D-RR_ACC.

Except for our PPNW methods, all comparative methods use re-ranking strategy after the three base models are trained. To make the comparison clear, we compute the changes from base model in percentage and record them in brackets.

Base models: non-graph embedding models Table 3 shows the experimental results of base models being non-graph embedding models.

In a sparse and smaller Citeulike dataset setting, PPNW generally improves novelty recommendation from base model with marginal decrease in accuracy. Among variants of PPNW, PPNW-BN losses relativelly more in accuracy with competitive novelty score. Especially in GMF-Base, the PPNW-BN provides the second best L_Tail and Nov_Score. PPNW-G, as mentioned in Sect. 3.3.1, provides high flexibility, enabling better performance to realize. Both variants of PPNW-G give the lowest decreases in accuracy among all experiments. It's also worth noting that PPNW-G(9) on GMF actually promotes the accuracy for the base models. 5D-Pop focuses exclusively on improving long tail and novelty score and sacrifices great amounts of accuracy. 5D-RR_ACC balances the trade-off, but the resulting performances are not competitive with any PPNW model.

ML-1M is a denser and larger dataset. Overall, PPNW improves dramatically both 2 novelty measurements, up to 41.5% in L_Tail and 16% in Nov_Score, respectively. Again 5D-Pop can boost the novelty, but reduces accuracy dramatically. Though PRA deceases L_Tail, it promotes Nov_Score on GMF and NMF.

Finally, Pinterest is sparse and the largest dataset in our experiments. All methods do not improve novelty metrics as greatly as they perform on the other two datasets. PPNW variants, in this setting, raise novelty metrics with the lowest losses in accuracy (none of the decreases is lower than -4% in HR and -5.5% in NDCG) compared with the results on the other two datasets. PRA increases Nov_Score but reduces L_Tail as well as on ML-1M.

Base models: graph embedding models To examine the generalization of PPNW on various types of base models, we observe and discuss the comparative results of PPNW on graph embedding base models. Table 3 shows the experimental results with the base models being graph embedding style.

We observe that the overall performances of PPNW on graph embedding base models are very similar to its performances on non-graph embedding models. These experimental results support that the PPNW is capable of consistently generating stable and well-balanced novel recommendations regardless of various types of base models.

In summary, the performances of the proposed PPNW and re-ranking methods are generally independent of base model and depend more on the characteristics of different datasets, e.g., sparsity and size. In addition, PPNW outperforms all re-ranking methods in terms of accuracy and novelty, except 5D-Pop. 5D-Pop increases the novelty and reduces accuracy dramatically. 5D-RR_ACC balances accuracy and novelty better than 5D-Pop, but its results are not competitive with PPNW. PRA re-ranks the recommendation list to improve particularly Nov_Score. As a whole, PPNW is capable of improving novelty as well as maintaining accuracy, while re-ranking methods focus on a single target "novelty" and can hardly balance the trade-off between novelty and accuracy well.

4.6 Comparison with loss weighting methods

In this section, we conduct comparative experiments for loss weighting methods, PPNW and AllRank-Pop. Because different loss functions are used, in order to make a fair comparison, we compute the "SacrificeRatio" to measure the trade-off between novelty and accuracy made by each model. Experimental results are shown in Table 5.

Table 3 Comparison with Re-ranking (non-graph base model)

		HR	NDCG	L_Tail	Nov_Score
Citeulike-a (@10)	**GMF-Base**	0.8418	0.6226	0.6867	6.1633
	+PRA	0.5221 (−37.9%)	0.3936 (−36.7%)	0.4870 (−29%)	6.0475 (−1.8%)
	+5D-Pop	0.0982 (−88.3%)	0.0610 (−90.2%)	**0.9677 (+40.9%)**	**6.8180 (+10.6%)**
	+5D-RR_ACC	0.5063 (−39.8%)	0.3622 (−41.8%)	0.7420 (+8.0%)	6.2630 (+1.6%)
	+PPNW-BN	0.7968 (−5.3%)	0.5386 (−13.4%)	0.8444 (+22.9%)	6.4869 (+5.2%)
	+PPNW-G(7)	0.8370 (−0.5%)	0.6197 (−0.4%)	0.7129 (+3.8%)	6.2196 (+0.9%)
	+PPNW-G(9)	**0.8449 (+0.3%)**	**0.6243 (+0.2%)**	0.7099 (+3.3%)	6.2342 (+1.1%)
	NMF-Base	0.8721	0.6384	0.6802	6.1425
	+PRA	0.5672 (−34.9%)	0.4189 (−34.3%)	0.4912 (−27.7%)	6.0725 (−1.1%)
	+5D-Pop	0.0977 (−88.7%)	0.0618 (−90.3%)	**0.9593 (+41.0%)**	**6.6285 (+7.9%)**
	+5D-RR_ACC	0.5201 (−40.3%)	0.3799 (−40.4%)	0.7218 (+6.1%)	6.2475 (+1.7%)
	+PPNW-BN	0.8311 (−4.7%)	0.6017 (−5.7%)	0.7354 (+8.1%)	6.2296 (+1.4%)
	+PPNW-G(7)	0.8637 (−0.9%)	**0.6184 (−3.1%)**	0.7607 (+11.8%)	6.3387 (+3.1%)
	+PPNW-G(9)	**0.8710 (−0.1%)**	0.6123 (−4.0%)	0.7625 (+12.0%)	6.3383 (+3.1%)
	CMN-Base	0.8910	0.6615	0.6746	6.1372
	+PRA	0.5893 (−33.8%)	0.4345 (−34.3%)	0.4836 (−28.3%)	5.8673 (−4.3%)
	+5D-Pop	0.1012 (−88.6%)	0.0687 (−89.6%)	**0.9352 (+38.6%)**	**6.5235 (+6.2%)**
	+5D-RR_ACC	0.5061 (−43.1%)	0.3905 (−40.9%)	0.7213 (+6.9%)	6.2344 (+1.5%)
	+PPNW-BN(7)	0.8580 (−3.7%)	0.6456 (−2.4%)	0.7254 (+7.5%)	6.2593 (+1.9%)
	+PPNW-G(7)	0.8712 (−2.2%)	0.6499 (−1.7%)	0.7514 (+11.3%)	6.3394 (+3.2%)
	+PPNW-G(9)	**0.8800 (−1.2%)**	**0.6590 (−0.3%)**	0.7411 (+9.8%)	6.3419 (+3.3%)

Table 3 continued

		HR	NDCG	L_Tail	Nov_Score
ML-1M (@10)	**GMF-Base**	0.6925	0.4191	0.3825	2.4451
	+PRA	0.4654 (− 32.7%)	0.2857 (− 31.8%)	0.2780 (− 27.3%)	2.6696 (+ 9.1%)
	+5D-Pop	0.0839 (− 87.8%)	0.0423 (− 89.9%)	**0.9036 (136.2%)**	**3.3740 (+ 37.9%)**
	+5D-RR_ACC	0.4394 (− 36.5%)	0.2798 (− 33.2%)	0.3154 (− 17.5%)	2.3418 (− 4.2%)
	+PPNW-BN	0.6611 (− 4.5%)	0.3912 (− 6.6%)	0.5053 (+ 32.1%)	2.8369 (+ 16%)
	+PPNW-G(2)	0.6654 (− 3.9%)	0.3921 (− 6.4%)	0.5137 (+ 34.3%)	2.7800 (+ 13.6%)
	+PPNW-G(4)	**0.6700 (− 3.2%)**	**0.3976 (− 5.1%)**	0.4637 (+ 21.2%)	2.7687 (+ 13.2%)
	NMF-Base	0.7064	0.4216	0.3658	2.4384
	+PRA	0.4930 (− 30.2%)	0.2941 (− 30.2%)	0.2824 (− 22.7%)	2.5224 (+ 3.4%)
	+5D-Pop	0.0841 (− 88.0%)	0.0469 (− 88.8%)	**0.8924 (+ 143.9%)**	**3.3455 (+ 37.2%)**
	+5D-RR_ACC	0.4606 (− 34.7%)	0.2956 (− 29.8%)	0.3187 (− 12.8%)	2.1811 (− 10.5%)
	+PPNW-BN	0.6720 (− 4.8%)	0.4083 (− 3.1%)	0.4869 (+ 33.1%)	2.8058 (+ 15%)
	+PPNW-G(2)	**0.6814 (− 3.5%)**	**0.4132 (− 1.9%)**	0.4997 (+ 36.6%)	2.6152 (+ 7.2%)
	+PPNW-G(4)	0.6791 (− 3.8%)	0.4099 (− 2.7%)	0.4952 (+ 35.3%)	2.6191 (+ 7.4%)
	CMN-Base	0.6875	0.4083	0.3544	2.3516
	+PRA	0.4507 (− 34.4%)	0.2759 (− 32.4%)	0.2802 (− 20.9%)	2.3255 (− 1.1%)
	+5D-Pop	0.0989 (− 85.6%)	0.0473 (− 88.4%)	**0.9021 (+ 154.5%)**	**3.1990 (+ 36%)**
	+5D-RR_ACC	0.4148 (− 39.6%)	0.2608 (− 36.1%)	0.3298 (− 6.9%)	2.1786 (− 7.3%)
	+PPNW-BN	0.6513 (− 5.2%)	**0.3755 (− 8.0%)**	0.4966 (+ 40.1%)	2.7985 (+ 19.0%)
	+PPNW-G(2)	0.6586 (− 4.2%)	0.3718 (− 8.9%)	0.5015 (+ 41.5%)	2.6237 (+ 11.5%)
	+PPNW-G(4)	**0.6651 (− 3.2%)**	**0.3753 (− 8.0%)**	0.4510 (+ 27.2%)	2.6866 (+ 14.2%)
Pinterest (@10)	**GMF-Base**	0.8486	0.5489	0.6298	5.8122
	+PRA	0.5855 (− 31.0%)	0.3848 (− 29.8%)	0.5289 (− 16.0%)	5.9360 (+ 2.1%)
	+5D-Pop	0.1043 (− 87.7%)	0.0622 (− 88.6%)	**0.8914 (+ 41.5%)**	**6.1985 (+ 6.6%)**

Table 3 continued

	HR	NDCG	L_Tail	Nov_Score
+5D-RR_ACC	0.4977 (−41.3%)	0.3865 (−29.5%)	0.6903 (+9.6%)	5.8767 (+1.1%)
+PPNW-BN	0.8330 (−1.8%)	0.5188 (−5.4%)	0.6840 (+8.6%)	5.9742 (+2.7%)
+PPNW-G(2)	**0.8449 (−0.4%)**	**0.5398 (−1.6%)**	0.6548 (+3.9%)	5.8654 (+0.9%)
+PPNW-G(4)	0.8418 (−0.8%)	0.5373 (−2.1%)	0.6541 (+3.8%)	5.9113 (+1.7%)
NMF-Base	0.8711	0.5509	0.6211	5.8207
+PRA	0.5887 (−32.4%)	0.3911 (−29.0%)	0.5763 (−7.2%)	5.9221 (+1.7%)
+5D-Pop	0.1140 (−86.9%)	0.0647 (−88.2%)	**0.9013 (+45.1%)**	**6.2386 (+7.1%)**
+5D-RR_ACC	0.5013 (−42.4%)	0.3927 (−28.7%)	0.7012 (+12.8%)	6.0104 (+3.2%)
+PPNW-BN	0.8399 (−3.5%)	0.5274 (−4.2%)	0.6648 (+7.0%)	5.9725 (+2.6%)
+PPNW-G(2)	0.8512 (−2.2%)	0.5365 (−2.6%)	0.6504 (+4.7%)	5.9810 (+2.7%)
+PPNW-G(4)	**0.8551 (−1.8%)**	**0.5411 (−1.7%)**	0.6547 (+5.4%)	6.0006 (+3.0%)
CMN-Base	0.8801	0.5641	0.6226	5.8271
+PRA	0.6027 (−31.5%)	0.3987 (−29.3%)	0.5724 (−8.0%)	5.9041 (+1.3%)
+5D-Pop	0.1148 (−86.9%)	0.0664 (−88.2%)	**0.9055 (+45.4%)**	**6.2760 (+7.7%)**
+5D-RR_ACC	0.4723 (−46.3%)	0.3643 (−35.4%)	0.7093 (+13.9%)	6.0458 (+3.7%)
+PPNW-BN	0.8495 (−3.4%)	0.5384 (−4.5%)	0.6778 (+8.8%)	5.9767 (+2.5%)
+PPNW-G(2)	0.8557 (−2.7%)	**0.5496 (−2.5%)**	0.6632 (+6.5%)	6.1250 (+5.1%)
+PPNW-G(4)	**0.8585 (−2.4%)**	0.5401 (−4.2%)	0.6628 (+6.4%)	5.9826 (+2.6%)

Values shown in the brackets are changes in percentage compared with base model. Best results of each metric are in bold. Generally, 5D-Pop always provides highest novelty with great decrease in accuracy. The proposed PPNW improves novelty significantly with almost no accuracy loss

Table 4 Comparison with Re-ranking (graph embedding base model)

		HR	NDCG	L_Tail	Nov_Score
Citeulike-a (@20)	**HopRec-Base**	0.2549	0.3489	0.6934	6.1414
	+PRA	0.1696 (− 33.5%)	0.2216 (− 36.5%)	0.5170 (− 25.4%)	6.0790 (− 1.0%)
	+5D-Pop	0.0303 (− 88.1%)	0.0359 (− 89.7%)	**0.9537 (+ 37.5%)**	**6.8090 (+ 10.9%)**
	+5D-RR_ACC	0.1611 (− 36.8%)	0.2132 (− 38.9%)	0.7301 (+ 5.3%)	6.3146 (+ 2.8%)
	+PPNW-BN	0.2426 (− 4.8%)	0.3266 (− 6.4%)	0.8193 (+ 18.2%)	6.4092 (+ 4.4%)
	+PPNW-G(7)	0.2529 (− 0.8%)	0.3332 (− 4.5%)	0.7288 (+ 5.1%)	6.2354 (+ 1.5%)
	+PPNW-G(9)	**0.2539 (− 0.4%)**	**0.3353 (− 3.9%)**	0.7295 (+ 5.2%)	6.2444 (+ 1.7%)
	NGCF-Base	0.2138	0.3424	0.6882	6.1600
	+PRA	0.1408 (− 34.2%)	0.2388 (− 30.3%)	0.5108 (− 25.8%)	6.0620 (− 1.6%)
	+5D-Pop	0.0272 (− 87.3%)	0.0380 (− 88.9%)	**0.9377 (+ 36.3%)**	**6.6866 (+ 8.5%)**
	+5D-RR_ACC	0.1302 (− 39.1%)	0.2054 (− 40.0%)	0.7104 (+ 3.2%)	6.3417 (+ 2.9%)
	+PPNW-BN	0.2067 (− 3.3%)	0.3181 (− 7.1%)	0.7558 (+ 9.8%)	6.2861 (+ 2.0%)
	+PPNW-G(7)	0.2112 (− 1.2%)	**0.3294 (− 3.8%)**	0.7712 (+ 12.1%)	6.4002 (+ 3.9%)
	+PPNW-G(9)	**0.2117 (− 1.0%)**	0.3263 (− 4.7%)	0.7702 (+ 11.9%)	6.4251 (+ 4.3%)
ML-1M (@20)	**HopRec-Base**	0.1900	0.5629	0.4073	2.3240
	+PRA	0.1210 (− 36.3%)	0.3647 (− 35.2%)	0.3022 (− 25.8%)	2.7139 (+ 16.8%)
	+5D-Pop	0.0182 (− 90.4%)	0.0507 (− 91.0%)	**0.8831 (+ 116.8%)**	**3.2847 (+ 41.3%)**
	+5D-RR_ACC	0.1191 (− 37.3%)	0.3569 (− 36.6%)	0.3565 (− 12.5%)	2.3388 (+ 0.6%)

Table 4 continued

	HR	NDCG	L_Tail	Nov_Score
+PPNW-BN	0.1794 (− 5.6%)	**0.5269 (− 6.4%)**	0.5173 (+ 27.0%)	2.8205 (+ 21.4%)
+PPNW-G(7)	0.1805 (− 5.0%)	0.5207 (− 7.5%)	0.5204 (+ 27.8%)	2.7952 (+ 20.3%)
+PPNW-G(9)	**0.1832 (− 3.6%)**	0.5173 (− 8.1%)	0.4766 (+ 17.0%)	2.7780 (+ 19.5%)
NGCF-Base	0.2447	0.6524	0.3944	2.3865
+PRA	0.1646 (− 32.7%)	0.4330 (− 33.6%)	0.2807 (− 28.8%)	2.3386 (− 2.0%)
+5D-Pop	0.0286 (− 88.3%)	0.0711 (− 89.1%)	**0.8759 (+ 122.1%)**	**3.2044 (+ 34.3%)**
+5D-RR_ACC	0.1569 (− 35.9%)	0.3869 (− 40.7%)	0.3238 (− 17.9%)	2.5221 (+ 5.7%)
+PPNW-BN	0.2322 (− 5.1%)	0.6002 (− 8.0%)	0.5200 (+ 31.8%)	2.8203 (+ 18.2%)
+PPNW-G(7)	0.2334 (− 4.6%)	0.6028 (− 7.6%)	0.5163 (+ 30.9%)	2.6834 (+ 12.4%)
+PPNW-G(9)	**0.2354 (− 3.8%)**	**0.6054 (− 7.2%)**	0.4862 (+ 23.3%)	2.7055 (+ 13.4%)

Values shown in the brackets are changes in percentage compared with base model. Best results of each metric are in bold. Generally, the proposed PPNW improves novelty significantly with almost no accuracy loss

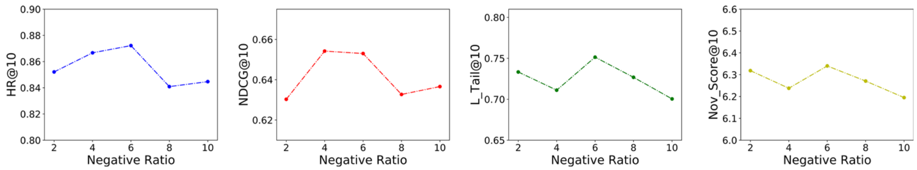


Fig. 4 Negative Ratio Experiment. Dataset is Citeulike-a and model is CMN-based PPNW-G(7). In terms of accuracy (HR and NDCG), negative ratio of 4 and 6 outperform the others. And for novelty (L_Tail and Nov_Score), negative ratio of 6 has best results

Among variants of PPNW, PPNW-G is always able to suppress the SacrificeRatios to be smaller than 0.1%, both $\text{SacrificeRatio}_{\text{LT}}$ and $\text{SacrificeRatio}_{\text{NS}}$. PPNW-BN, on the other hand, gives higher L_Tail and Nov_Score compared with PPNW-G. Among two AllRank-Pop methods, greater β_w promotes novelty metrics to higher level. In general, all PPNW methods outperform AllRank-Pop in respect of SacrificeRatio and, more specifically, both $\text{SacrificeRatio}_{\text{LT}}$ and $\text{SacrificeRatio}_{\text{NS}}$ are at least 10 times smaller the that of AllRank-Pop.

4.7 Recommendations visualization

To intuitively compare the results of accuracy-focused model and our PPNW, we collect in Table 6 the models' outputs (recommended movies) to 5 users on movies dataset ML-1M. All novelty scores reported in the table are normalized in range 0–1. Hence, user 608 with novelty score 0.039 has a strong preference for popular movies, while user 3997 (novelty score 0.676) favors novelty niche movies than popular blockbuster movies. Note that, since the ML-1M dataset only collects movies that are published before 2003, the recommended movies are mostly from the '90s as well.

As the results reveal, PPNW promotes average novelty for users of all novelty levels. Especially, for users having higher novelty taste, e.g., user 89 and 3997, the novelty gaps between CMN-Base and PPNW-G increase, indicating that PPNW promotes novelty much more aggressively when users are more likely to prefer niche novel movies. On the other hand, for users having low novelty taste, e.g., user 608 and 230, PPNW still provide reasonable improvement of novelty compared to the CMN-Base model.

4.8 Influence of negative ratio on PPNW

Since the proposed PPNW is a pairwise loss weighting method naturally utilizing negative examples during training, the effect of negative ratio needs to be analyzed. In this section, we examine the effects of various negative ratios from {2, 4, 6, 8, 10}. Because the previous two experiments show consistent results on different datasets and base model, we only choose CMN-based PPNW-G(7) on Citeulike-a in this experiment. As shown in Fig. 4, the four metrics vary on different negative ratios. When the accuracy is the main concern, negative ratio of {4, 6} performs better. When the novelty is the main concern, negative ratio of {6} outperforms others. The results are consistent with [5,10] that the accuracy of a simple base model is at peek when negative ratio is around 4–6 and drop from 8.

Table 5 Comparison with loss weighting methods on Citeulike-a and Pinterest

		HR	NDCG	L_Tail	Nov_Score	SacrificeRatio$_{LT}$	SacrificeRatio$_{NS}$
Citeulike-a (@10)	**GMF_AllRank**	0.6589	0.3930	0.6828	6.1547		
	+AllRank-Pop(0.9)	0.5072	0.2904	0.7303	6.2890	− 3.31%	− 10.55%
	+AllRank-Pop(0.95)	0.3046	0.1452	0.8527	6.4780	**− 2.16%**	**− 10.24%**
	GMF_BPR	0.8418	0.6226	0.6867	6.1633		
	+PPNW-BN	0.7968	0.5386	0.8444	6.4869	− 0.23%	− 1.02%
	+PPNW-G(7)	0.8370	0.6197	0.7129	6.2196	− 0.15%	− 0.62%
	+PPNW-G(9)	0.8449	0.6243	0.7099	6.2342	**+ 0.11%**	**+ 0.32%**
	CMN_AllRank	0.7402	0.4959	0.7430	6.3174		
	+AllRank-Pop(0.9)	0.6081	0.3686	0.7908	6.4902	**− 2.77%**	**− 6.52%**
	+AllRank-Pop(0.95)	0.3859	0.2542	0.8695	6.5191	− 2.81%	− 14.99%
	CMN_BPR	0.8910	0.6615	0.6746	6.1372		
	+PPNW-BN(7)	0.8580	0.6456	0.7254	6.2593	− 0.49%	− 1.86%
	+PPNW-G(7)	0.8712	0.6499	0.7514	6.3394	− 0.20%	− 0.67%
	+PPNW-G(9)	0.8800	0.6590	0.7411	6.3419	**− 0.13%**	**− 0.37%**
Pinterest (@10)	**GMF_AllRank**	0.6847	0.3685	0.6938	5.9069		

Table 5 continued

	HR	NDCG	L_Tail	Nov_Score	SacrificeRatio$_{\text{LT}}$	SacrificeRatio$_{\text{NS}}$
+AllRank-Pop(0.9)	0.5236	0.2789	0.7261	6.1877	− 5.05%	− **4.95%**
+AllRank-Pop(0.95)	0.2912	0.1271	0.8179	6.2186	− **3.21%**	− 10.89%
GMF_BPR	0.8486	0.5489	0.6298	5.8122		
+PPNW-BN(2)	0.8330	0.5188	0.6840	5.9742	− 0.21%	− 0.66%
+PPNW-G(2)	0.8449	0.5398	0.6548	5.8654	− **0.11%**	− 0.48%
+PPNW-G(4)	0.8418	0.5373	0.6541	5.9113	− 0.21%	− **0.47%**
CMN_AllRank	0.7744	0.4715	0.6721	6.0991		
+AllRank-Pop(0.9)	0.6085	0.3691	0.7275	6.1807	− **2.60%**	− **16.01%**
+AllRank-Pop(0.95)	0.3602	0.2701	0.8013	6.2755	− 2.78%	− 18.49%
CMN_BPR	0.8801	0.5641	0.6226	5.8271		
+PPNW-BN	0.8495	0.5384	0.6778	5.9767	− 0.39%	− 1.35%
+PPNW-G(2)	0.8557	0.5496	0.6632	6.1250	− 0.43%	− **0.54%**
+PPNW-G(4)	0.8585	0.5401	0.6628	5.9826	− **0.38%**	− 0.92%

Best results of SacrificeRatios for each model of different loss functions are in bold. In general, our method remarkably promotes long tail and novelty and has at least 10 times smaller SacrificeRatio than AllRank-Pop

Table 6 Top-5 recommendation outputs for users of different levels of novelty taste

User	Model	Top-5 Recommended Movies (Year) (Novelty)
UserId: 608 **Novelty**: 0.039 (**Low**)	CMN-Base (Accuracy-Focused)	[**Avg. Nov=0.089**] ① Wild Bill (1995) (0.091); ② Hate (1995) (0.031); ③ Mr. Wonderful (1993) (0.115); ④ Dead Man (1995) (0.078); ⑤ Angels and Insects (1995) (0.129)
	PPNW-G(4) (Ours)	[**Avg. Nov=0.160**] ① Mr. Wonderful (1993) (0.115); ② Infinity (1996) (0.235); ③ North (1994) (0.14); ④ Congo (1995) (0.084); ⑤ Species (1995) (0.228)
UserId: 230 **Novelty**: 0.159 (**Low**)	CMN-Base (Accuracy-Focused)	[**Avg. Nov=0.109**] ① Burnt By the Sun (1994) (0.089); ② To Die For (1995) (0.156); ③ Copycat (1995) (0.037); ④ Strawberry and Chocolate (1993) (0.120); ⑤ Swimming with Sharks (1995) (0.145)
	PPNW-G(4)(Ours)	[**Avg. Nov=0.199**] ① Firestorm (1998) (0.281); ② Burnt By the Sun (1994) (0.089); ③ True Crime (1995) (0.219); ④ Nina Takes a Lover (1994) (0.177); ⑤ Frankie Starlight (1995) (0.227)
UserId: 1388 **Novelty**: 0.262 (**Medium**)	CMN-Base (Accuracy-Focused)	[**Avg. Nov=0.192**] ① Walk in the Clouds (1995) (0.153); ② Last Dance (1996) (0.195); ③ Snowriders (1996) (0.174); ④ Something to Talk About (1995) (0.12); ⑤ Commandments (1997) (0.319)
	PPNW-G(4) (Ours)	[**Avg. Nov=0.271**] ① Braindead (1992) (0.167); ② Leaving Las Vegas (1995) (0.282); ③ Metisse (1993) (0.304); ④ Beans of Egypt, Maine (1994) (0.254); ⑤ Barney's Great Adventure (1998) (0.348)
UserId: 89 **Novelty**: 0.351 (**Medium**)	CMN-Base (Accuracy-Focused)	[**Avg. Nov=0.185**] ① Now and Then (1995) (0.097); ② Scarlet Letter (1995) (0.254); ③ Charade (1963) (0.179); ④ Naked in New York (1994) (0.21); ⑤ Silence of the Lambs (1991) (0.220)
	PPNW-G(4) (Ours)	[**Avg. Nov=0.346**] ① Kidnapped (1960) (0.487); ② Scarlet Letter (1995) (0.254); ③ Convent (1995) (0.371); ④ Farewell My Concubine (1993) (0.244); ⑤ Strike! (1998) (0.372)
UserId: 3997 **Novelty**: 0.676 (**High**)	CMN-Base (Accuracy-Focused)	[**Avg. Nov=0.181**] ① Curdled (1996) (0.218); ② Drop Dead Fred (1991) (0.269); ③ Boys on the Side (1995) (0.081); ④ Mille bolle blu (1993) (0.216); ⑤ Kim (1950) (0.119)
	PPNW-G(4) (Ours)	[**Avg. Nov=0.546**] ① Bachelor (1999) (0.722); ② House of Frankenstein (1944) (0.547); ③ On the Waterfront (1954) (0.41); ④ Romancing the Stone (1984) (0.494); ⑤ James Dean Story (1957) (0.558)

Movies dataset ML-1M is used and all novelty scores are normalized in range 0–1 As the results reveal, PPNW promotes novelty for users of all novelty levels. Especially, PPNW promotes more aggressively for users having high novelty taste, e.g., UserId 3997

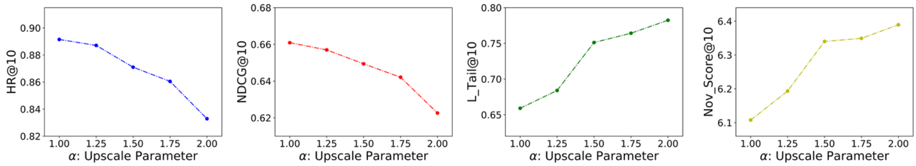


Fig. 5 Experiments of α's impact. Dataset is Citeulike-a and model is CMN-based PPNW-G(7). Experimental results show that α is the key hyperparameter of balancing accuracy and novelty. With greater α, accuracy consistently decreases and novelty increases

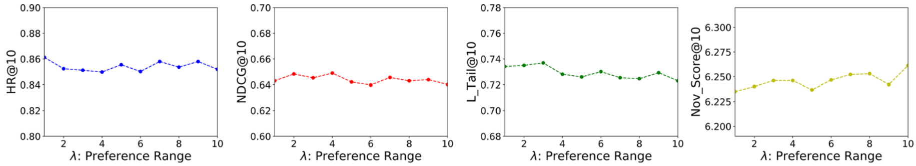


Fig. 6 λ's impact on PPNW-BN. Dataset is Citeulike-a and model is CMN-based PPNW-BN. Generally, different choices of λ have not impact on PPNW-BN. PPNW-BN performs stably and is insensitive to λ, marking itself as a "hyperparameter-free" model

4.9 Influence of alpha on PPNW

In this part, the influence of the hyperparameter in PPNW, α, is examined. We use CMN-based PPNW-G(7) on Citeulike-a in this experiment. As shown in 5, four metrics show consistent changes in direction with increasing value of α. Our experimental results show that α is the key hyperparameter of balancing accuracy and novelty. With greater α, accuracy consistently decreases and novelty increases. This observation suggests that PPNW introduces an important hyperparameter α, which can be utilized as a key factor to adjust for preferable level of novelty with acceptable sacrifice in accuracy.

4.10 Influence of lambda on PPNW-BN

To show that PPNW-BN is a "hyperparameter-free" model, we verify PPNW-BN's insensitivity to λ in this section. For the same reason as Sect. 4.8, we Choose CMN and Citeulike-a to conduct this experiment. Figure 6 shows the results on different λ values. The stable performances of both accuracy and novelty reflect that PPNW-BN is generally insensitive to different choices of λ. These results verify our argument of PPNW-BN's stability in Sect. 3.3.2 and its potential use in industrial applications.

4.11 Ablation study

There are two key components in PPNW, the novelty matching score $\pi_{u,i}$ and the upscale novelty score θ_i^P. To gain deeper understanding, we perform ablation study to analyze each effect. We choose GMF as base model and augment the base model with PPNW-BN. The experiments are conducted on 2 datasets, Citeulike-a and Pinterest.

- "Novelty Matching": PPNW-BN with only $\pi_{u,i}$.
- "Novel Scaling": PPNW-BN with only θ_i^P.
- "Full Model": PPNW-BN with $\pi_{u,i}$ and θ_i^P.

Table 7 Ablation study

Citeulike-a (@10)	HR	NDCG	L_Tail	Nov_Score
GMF-base	0.8418	**0.6226**	0.6867	6.1633
+ Novelty Matching	**0.8442**	0.6177	0.6741	6.1343
+ Novelty Scaling	0.8233	0.5987	0.7124	6.2579
+ **Full model**	0.7968	0.5386	**0.8444**	**6.4869**

Pinterest (@10)	HR	NDCG	L_Tail	Nov_Score
GMF-Base	0.8486	**0.5489**	0.6298	5.8122
+ Novelty Matching	**0.8547**	0.5331	0.5995	5.7467
+ Novelty Scaling	0.8287	0.5135	0.6695	**6.1477**
+ **Full model**	0.8330	0.5188	**0.6840**	5.9742

Table 7 shows the experimental results. It's interesting that by considering only user's preference ("Novelty Matching"), the accuracy is improved on both two datasets. This implies that user does have specific region of novelty preference and he/she tends to select items from this region.

As to "Novelty Scaling" model, it promotes novelty on both datasets with slight decease in HR and NDCG. However, note that in Sects. 4.10 and 3.3.2, we have discussed that the impact of upscale score might be alleviated by the batch-normalization and higher accuracy might be achieved by PPNW-G.

Lastly, the "Full Model" provides best L_Tail scores over all experiments and competitive Nov_Score. For the smaller dataset Citeulike-a, the "Full Model" prefers novelty at the cost of higher loss in accuracy. While for larger dataset Pinterest, it gives more balanced results. These might be caused by overfitting on smaller dataset.

5 Related work

As closely related work has been discussed in detail in Sects. 2 and 3.5, we here present a more general overview of recent related work.

5.1 Novelty-promoting recommender systems

The importance of novelty in RSs has been well acknowledged [15,29]. To promote novelty in the system, most works adopt re-ranking strategy, which post-processes the output of a base model. In [12], researchers translate the recommendation task to a resource allocation problem and re-rank the entire output user–item ratings matrix according to limited resources for each user. PRA [17] introduces a generic approach which promotes long-tail items by re-ranking the first top-N items and the next M candidate items. By considering only the first N+M items, PRA is able to maintain accuracy.

Besides the common frequency-based novelty measure, some researches use other novelty definitions and change methodology accordingly. In [42], novel items are restaurants having no previous interaction. The authors design a framework to recommend novel restaurants according to users' current novelty-seeking status. While in [24], gross movie earnings are used to approximate novelty and to compute users personal tendency. The novel recommendation is then provided by matching users and movies novelty tendency. Nakatsuji et al. [22] leverage taxonomy information of items to model novelty and to construct user similarity graph. The novel recommendations are then inferred by random walk with restart on the graph.

Our novel recommendation method, on the contrary, is a one-stage approach and requires only historical interaction data.

5.2 Loss weighting in recommender systems

Loss weighting is an intuitive and convenient way to adjust losses during training and has been wildly used to deal with other problems, like imbalanced classes problem [4,41]. In recommender systems, loss weighting is employed mainly to improve accuracy. Ma et al. [21] and Hu et al. [13] weight implicit feedback according to interaction frequency. Greater interaction frequency implies higher confidence level so the importance of this interaction should be emphasized by up-weighting its loss during training. In [36], rank-based weighting scheme is adopted to penalize positive items being ranked lower in the list. Since user clusters with various sparsity levels provide different information, Ning et al. [23] utilizes gradient information and converts it to loss weights to train different user clusters. For novel recommendation, detailed in section 3.5, Steck [30] weights losses based on item popularity without using user personal preference.

6 Conclusion

In the paper, we propose an efficient pairwise loss weighting framework for one-stage, end-to-end novel recommendation. Our method integrates both novelty information and user's personal preference into the BPR loss function for optimizing directly the novelty-accuracy trade-off during base model training. Specifically, our approach first summarizes and adjust the novelty scores of users and items. Then, in order to suggest items according to users' various novelty tastes, we design a novelty matching function, which is inspired by the Gaussian RBF kernel, to measure item-user novelty matching scores. After that, two loss weighting strategies are proposed to integrate all novelty information into the loss function. Eventually, by up-weighting or down-weighting losses, our PPNW is able to guide the model optimization toward a robust novelty-promoting system.

Extensive experiments are conducted. As experimental results reveal, not only does PPNW achieve better novelty-accuracy balance, but it also outperforms existing re-ranking and loss-weighting methods, validating its effectiveness and potential use in real-world applications.

Our approach differs from previous works in there aspects. (1) Instead of two-stage re-ranking, our approach adopts a one-stage end-to-end training style for novelty recommendation, which is more efficient and enables direct optimization of the novelty-accuracy trade-off. (2) Our approach takes into consideration the common limitations in loss functions and manages to alleviate them by injecting both novelty information and user preferences in loss function. (3) Our PPNW is a general and light "plug-in" to any accuracy-focused base model, functioning only in the loss function to adjust gradients. Any base model could be easily augmented with PPNW for a novelty-promoting system.

To encourage future applications and researches on PPNW, we summarize the limitations of PPNW as well as possible improvements as follows:

(1) The PPNW framework or our loss weighting strategies can be extended to promote other key factors of RSs as well, i.e., diversity, serendipity and coverage. For a robust RS, all key factors should be considered and balanced. Since PPNW is designed to optimize the trade-off between different factors, the potential extension is promising.
(2) We will attempt to use other advanced machine learning techniques to capture novelty information. For example, attention mechanisms [33] could be modified to measure user–item matching score on a feature level; more accurate novelty score might be obtained if additional data, like reviews and user profiles, are analyzed with advanced NLP techniques [34,37].
(3) The presented PPNW needs modifications in order to adopt to an explicit feedback setting. Though implicit feedback (clicked/viewed data) is the dominant data type in RSs, recommendations based on explicit feedback (ratings data) are also important in some applications [14,16]. The adaptation and performance evaluation on explicit feedback would be done as one of the future works.
(4) We also plan to extend PPNW for other granularity levels of novelty. For instance, "genre novelty" measures how a genre is unknown/novelty to a user. A system would be more competitive, if it is able to expand user interest by, for example, successfully recommending "horror movie" to an "action movie" fan.

Acknowledgements We thank anonymous reviewers for their very useful comments and suggestions. This work was supported by JSPS KAKENHI (Grant Numbers JP20K01983, JP18H00904, JP17H01001).

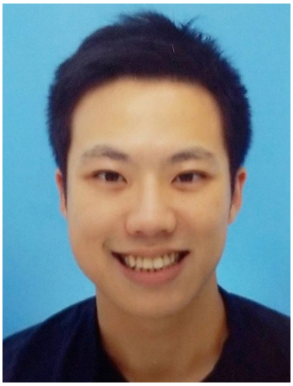
References

1. Adomavicius G, Kwon YO (2012) Improving aggregate recommendation diversity using ranking-based techniques. IEEE Trans Knowl Data Eng 24(5):896–911
2. Chen C-M et al (2019) Collaborative similarity embedding for recommender systems. In: The World Wide Web conference. ACM, pp 2637–2643
3. Cheng P et al (2017) Learning to recommend accurate and diverse items. In: Proceedings of the 26th international conference on World Wide Web. International World Wide Web Conferences Steering Committee, pp 183–192
4. Dong Q, Gong S, Zhu X (2019) Imbalanced deep learning by minority class incremental rectification. IEEE Trans Pattern Anal Mach Intell 41(6):1367–1381
5. Ebesu T, Shen B, Fang Y (2018) Collaborative memory network for recommendation systems. In: The 41st international ACM SIGIR conference on research & development in information retrieval. ACM, pp 515–524
6. Geng X et al (2015) Learning image and user features for recommendation in social networks. In: 2015 IEEE international conference on computer vision, ICCV 2015, Santiago, Chile, 7–13 Dec 2015, pp 4274–4282
7. Grover A, Leskovec J (2016) node2vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp 855–864
8. Harald S (2010) Training and testing of recommender systems on data missing not at random. In: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp 713–722
9. Harper FM, Konstan JA (2016) The MovieLens datasets: history and context. TiiS 5(4):19:1–19:19. https://doi.org/10.1145/2827872
10. He X et al (2017) Neural collaborative filtering. In: Proceedings of the 26th international conference on World Wide Web. International World Wide Web Conferences Steering Committee, pp 173–182
11. He X et al (2016) Fast matrix factorization for online recommendation with implicit feedback. In: Proceedings of the 39th international ACM SIGIR conference on research and development in information retrieval, SIGIR 2016, Pisa, Italy, 17–21 July 2016, pp 549–558
12. Ho Y-C, Chiang Y-T, Hsu JY-J (2014) Who likes it more? Mining worth-recommending items from long tails by modeling relative preference. In: Proceedings of the 7th ACM international conference on web search and data mining. ACM, pp 253–262

13. Hu Y, Koren Y, Volinsky C (2008) Collaborative filtering for implicit feedback datasets. In: Proceedings of the 8th IEEE international conference on data mining (ICDM 2008), 15–19 Dec 2008, Pisa, Italy, pp 263–272
14. Hu Y, Koren Y, Volinsky C (2008) Collaborative filtering for implicit feedback datasets. In: 2008 Eighth IEEE international conference on data mining. IEEE, pp 263–272
15. Hurley N, Zhang M (2011) Novelty and diversity in top-N recommendation—analysis and evaluation. ACM Trans Internet Technol (TOIT) 10(4):14:1–14:30
16. Joachims T et al (2017) Accurately interpreting clickthrough data as implicit feedback. ACM SIGIR Forum, vol 51, no 1. ACM, New York, pp 4–11
17. Jugovac M, Jannach D, Lerche L (2017) Efficient optimization of multiple recommendation quality factors according to individual user tendencies. Expert Syst Appl 81:321–331
18. Kapoor K et al (2015) I like to explore sometimes: adapting to dynamic user novelty preferences. In: RecSys 2015, Vienna, Austria, 16–20 Sept 2015, pp 19–26
19. Kelly D, Teevan J (2003) Implicit feedback for inferring user preference: a bibliography. ACM SIGIR forum, vol 37, no 2. ACM, New York, pp 18–28
20. Kotkov D, Veijalainen J, Wang S (2016) Challenges of serendipity in recommender systems. In: Proceedings of the 12th international conference on web information systems and technologies, WEBIST 2016, vol 2, Rome, Italy, 23–25 April 2016, pp 251–256
21. Ma C et al (2018) Point-of-interest recommendation: exploiting selfattentive autoencoders with neighbor-aware influence. In: Proceedings of the 27th ACM international conference on information and knowledge management, CIKM 2018, Torino, Italy, 22–26 Oct, 2018. ACM, pp 697–706
22. Nakatsuji M et al (2010) Classical music for rock fans? Novel recommendations for expanding user interests. In: Proceedings of the 19th ACM international conference on information and knowledge management. ACM, pp 949–958
23. Ning Y et al (2017) A gradient-based adaptive learning framework for efficient personal recommendation. In: Proceedings of the eleventh ACM conference on recommender systems. ACM, pp 23–31
24. Oh J et al (2011) Novel recommendation based on personal popularity tendency. In: 2011 IEEE 11th international conference on data mining. IEEE, pp 507–516
25. Perozzi B, Al-Rfou R, Skiena S (2014) Deepwalk: online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 701–710
26. Rendle S (2010) Factorization machines. In: ICDM 2010, The 10th IEEE international conference on data mining, Sydney, Australia, 14–17 Dec 2010, pp 995–1000
27. Rendle S et al (2009) BPR: Bayesian personalized ranking from implicit feedback. In: Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence. AUAI Press, pp 452–461
28. Ricci F, Rokach L, Shapira B (2011) Introduction to recommender systems handbook. In: Recommender systems handbook, pp 1–35
29. Ricci F, Rokach L, Shapira B (2015) Recommender systems handbook. Springer, Berlin
30. Steck H (2011) Item popularity and recommendation accuracy. In: Proceedings of the fifth ACM conference on recommender systems. ACM, pp 125–132
31. Tang J et al (2015) Line: large-scale information network embedding. In: Proceedings of the 24th international conference on world wide web. International World Wide Web Conferences Steering Committee, pp 1067–1077
32. Vargas S, Castells P (2014) Improving sales diversity by recommending users to items. In: Proceedings of the 8th ACM conference on recommender systems. ACM, pp 145–152
33. Vaswani A et al (2017) Attention is all you need. In: Advances in neural information processing systems, pp 5998–6008
34. Vijaikumar M, Shevade S, Murty MN (2019) SoRecGAT: leveraging graph attention mechanism for top-N social recommendation. In: Joint European conference on machine learning and knowledge discovery in databases. Springer, pp 430–446
35. Wang C, Blei DM (2011) Collaborative topic modeling for recommending scientific articles. In: Proceedings of the 17th ACM SIGKDD international conference on knowledge discovery and data mining, San Diego, CA, USA, 21–24 Aug 2011, pp 448–456
36. Wang Q et al (2018) Neural memory streaming recommender networks with adversarial training. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, KDD 2018, London, UK, 19–23 Aug 2018, pp 2467–2475
37. Wang X et al (2019) KGAT: knowledge graph attention network for recommendation. arXiv preprint arXiv:1905.07854
38. Wang X et al (2019) Neural graph collaborative filtering. arXiv preprint arXiv:1905.08108

39. Yang J-H et al (2018) HOP-rec: high-order proximity for implicit recommendation. In: Proceedings of the 12th ACM conference on recommender systems. ACM, pp 140–144
40. Yin H et al (2012) Challenging the long tail recommendation. PVLDB 5(9):896–907
41. Yue S (2017) Imbalanced malware images classification: a CNN based approach. In: CoRR abs/1708.08042
42. Zhang F et al (2015) A novelty-seeking based dining recommender system. In: Proceedings of the 24th international conference on World Wide Web, WWW 2015, Florence, Italy, 18–22 May 2015, pp 1362–1372
43. Zhang M, Hurley N (2008) Avoiding monotony: improving the diversity of recommendation lists. In: Proceedings of the 2008 ACM conference on recommender systems, RecSys 2008, Lausanne, Switzerland, 23–25 Aug 2008, pp 123–130
44. Zhang YC et al (2012) Auralist: introducing serendipity into music recommendation. In: Proceedings of the fifth international conference on web search and web data mining, WSDM 2012, Seattle, WA, USA, 8–12 Feb 2012, pp 13–22
45. Zhang YC et al (2012) Auralist: introducing serendipity into music recommendation. In: Proceedings of the fifth international conference on web search and web data mining, WSDM 2012, Seattle, WA, USA, February 8-12, 2012, pp 13–22
46. Zhao P, Lee DL (2016) How much novelty is relevant? It depends on your curiosity. In: Proceedings of the 39th international ACM SIGIR conference on research and development in information retrieval, SIGIR 2016, Pisa, Italy, 17–21 July 2016, pp 315–324
47. Ziegler C-N et al (2005) Improving recommendation lists through topic diversification. In: Proceedings of the 14th international conference on World Wide Web, WWW 2005, Chiba, Japan, 10–14 May 2005, pp 22–32
48. Zolaktaf Z, Babanezhad R, Pottinger R (2018) A generic top-N recommendation framework for trading-off accuracy, novelty, and coverage. In: 2018 IEEE 34th international conference on data engineering (ICDE). IEEE, pp 149–160

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Kachun Lo received his B.Sc. from Nankai University, Tianjin, China in 2016. He received his M.Sc. from Tohoku University, Sendai, Japan in 2019. Currently, he is working toward the doctoral degree at the Tohoku University, Sendai, Japan. He has worked in the academic field for more than 4 years. His research interests mainly include knowledge discovery, personalized recommendation and natural language processing.

Tsukasa Ishigaki received his Ph.D. in the Department of Statistical Science from the Graduate University for Advanced Studies (SOKENDAI), Japan, in 2007. He is currently an associate professor in the Graduate School of Economics and Management, Tohoku University, Japan. His research focuses on knowledge discovery through statistical modeling, data mining and machine learning in business, medical care, intelligent sensing system, service engineering, social science, etc.