Adaptive Pointwise-Pairwise Learning-to-Rank for Content-based Personalized Recommendation

Yagmur Gizem Cinar
yagmur.cinar@imag.fr
Naver Labs Europe and Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG

Jean-Michel Renders
jean-michel.renders@naverlabs.com
Naver Labs Europe

ABSTRACT

This paper extends the standard pointwise and pairwise paradigms for learning-to-rank in the context of personalized recommendation, by considering these two approaches as two extremes of a continuum of possible strategies. It basically consists of a surrogate loss that models how to select and combine these two approaches adaptively, depending on the context (query or user, pair of items, etc.). In other words, given a training instance, which is typically a triplet (a query/user and two items with different preferences or relevance grades), the strategy adaptively determines whether it is better to focus on the "most preferred" item (pointwise - positive instance), on the "less preferred" one (pointwise - negative instance) or on the pair (pairwise), or on anything else in between these 3 extreme alternatives. We formulate this adaptive strategy as minimizing a particular loss function that generalizes simultaneously the traditional pointwise and pairwise loss functions (negative log-likelihood) through a mixture coefficient. This coefficient is formulated as a learnable function of the features associated to the triplet. Experimental results on several real-world news recommendation datasets show clear improvements over several pointwise, pairwise, and listwise approaches.

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

learning-to-rank, recommender systems, news recommendation

ACM Reference Format:
Yagmur Gizem Cinar and Jean-Michel Renders. 2020. Adaptive Pointwise-Pairwise Learning-to-Rank for Content-based Personalized Recommendation. In *Fourteenth ACM Conference on Recommender Systems (RecSys '20), September 22–26, 2020, Virtual Event, Brazil.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3383313.3412229

1 INTRODUCTION

Ranking corresponds to arranging a list of items from first to last according to a notion of relevance, which improves the utility of this list, and it is widely used in information retrieval, recommendation and natural language processing applications. Typically learning-to-rank approaches, *i.e.* machine learning approaches applied to ranking task, are used to learn a scoring function, which is then used to rank these lists of items. Learning-to-rank approaches are distinguished according to their surrogate loss: pointwise, pairwise, and listwise [18]. Pointwise approaches estimate direct relevance of an item by formulating ranking as a regression or classification task [6]. Pairwise approaches [9, 15] articulate the ranking problem in terms of pairwise preferences, and listwise approaches [2, 4, 30] address ranking by directly optimizing a loss defined over an entire ranked list.

When designing search or recommender systems and, especially, "learning to rank" strategies, one typically faces the issue of choosing which one of the "pointwise", "pairwise" and "listwise" approaches should be adopted, depending on the data distribution (e.g. sparsity level, label noise, input feature distribution, ...). Each approach has its own advantages and drawbacks, in terms of ranking performance, robustness to noise, and computational complexity. To alleviate their weaknesses, researchers in each approach have developed "ad-hoc" improvements, often based on heuristics. For instance, listwise approaches model ranking in a more natural way but might require higher computational resources (e.g. ListNET [4] or ListMLE [29]), and pointwise approaches can be improved by specific instance-weighting schemes and by using user-level feature normalisation (e.g. removing the "user rating bias"). In the same vein, pairwise approaches, which are very sensitive to label noise [5, 18] and to the choice of the "negative" instances, are enhanced by specific sampling strategies when feeding the "learning to rank" algorithm with training "pairs" or "triplets" [23].

In this paper, we focus on the pointwise and pairwise approaches, leaving the combination with listwise methods for further work. Even if pointwise and pairwise approaches have their own complementary advantages and drawbacks, many studies focus on only one approach and try to mitigate their drawbacks in some ad-hoc way. The main research question that we are addressing is: *Can a new surrogate loss combine the two losses optimally and adaptively?* The main contributions of this paper are i) the introduction of a new surrogate loss which is an optimal and adaptive combination of the two approaches, a continuum between pointwise and pairwise losses, so that the precise balance between pointwise and pairwise contributions could depend on the particular pair or triplet instance; ii) experimental results that are obtained by using the proposed surrogate loss on several personalized recommendation datasets with implicit feedback, yielding significant improvements over pointwise, pairwise, listwise and other (combined) pointwise-pairwise learning-to-rank approaches.

RecSys '20, September 22–26, 2020, Virtual Event, Brazil

ACM ISBN 978-1-4503-7583-2/20/09…$15.00
https://doi.org/10.1145/3383313.3412229

2 RELATED WORK

Even if pointwise and pairwise approaches have their own complementary advantages and drawbacks, many pieces of work focus on only one approach, trying to alleviate their drawbacks. For pointwise approaches, in order to handle label ambiguity, especially with implicit feedback, different non-uniform weighting strategies have been developed (see [20] as the seminal work in this direction and [11] as a typical representative of the most recent work); in the same vein, a recent elegant alternative [28] was proposed to use Noise Contrastive Estimation to artificially generate "negative" examples, in order to avoid any assumption on the implicitly negative (i.e. not clicked) ones. Closer to our work is the "BPR-max" approach [12] that extends the standard BPR-loss by weighting the contribution of the negative samples using a softmax score. This allows the gradient to give more importance to "high score" – and so harder – negatives in a listwise manner.

For pairwise approaches, the focus has been put mainly on designing efficient sampling strategies, to select the most useful (and less noisy) training triplets. Examples of the family of sampling strategies goes from the seminal work of [23, 24] to most recent pieces work such as [7, 8], passing through famous techniques such as "hard negative mining" often used in the Computer Vision community [10]. In a slightly different vein, [5] proposed to use a sigmoid loss layer that follows an initial ranker to improve the robustness of the ranker.

To our knowledge, this study is the first one to propose a surrogate loss that combines pointwise and pairwise ranking surrogate losses as a continuum. The most related pieces of work that we identified simply add the two loss functions corresponding to the pointwise and pairwise approaches, and solve the problem either jointly (gradient descent on the total loss at every iteration) [27] or alternately (gradient descent on the pointwise loss followed by gradient descent on the pairwise loss) [17]. In the same vein, recently, a combined framework was proposed to mix pointwise and pairwise approaches [31] by simply considering a global loss defined as weighted sum between a pointwise reconstruction squared error and a smooth, differentiable approximation of the listwise Average Precision metrics; the combination weight is tuned as an hyperparameter. Another older related work is [19], which optimizes an isotonic regression loss that incorporates tied pairwise and pointwise constraints in a listwise manner.

The main novelty here is to introduce a new surrogate loss which a continuum between pointwise and pairwise losses, and can be formulated as an adaptive weighting of pointwise and pairwise losses, so that the precise balance between their contributions could depend on the particular pair or triplet instance. Adaptive weighting for a particular training instance can be found similar to mixture-of-experts [14], which is an ensemble method combining different learners scores based on the input instance. However, our proposed method defines a surrogate loss that leads to a single model, not an ensemble of models.

3 STANDARD PERSONALIZED RANKING MODELS FROM IMPLICIT FEEDBACK

3.1 Pointwise Personalized Ranking from Implicit Feedback

Pointwise ranking directly estimates the relevance of an item i for a user u, considering the relevance estimation as a classification problem or as an ordinal regression problem (for graded relevances). Groundtruth relevance labels in the training set are given either explicitly by manual relevance assessments, or by user clicks (implicit user feedback). In the latter case, some propensity weighting strategy should be used to compensate for the different biases (*e.g.* position or layout biases). Most of the recommendation models assume that the relevance probability of an item i for user u can be expressed in the following form:

$$p(i|u) = \sigma(f(u, i|\theta)) \tag{1}$$

where σ is typically the sigmoid function ($\sigma(x) = 1/(1 + e^{-x})$), and $f(u, i|\theta)$ is a scoring function, with learnable parameters θ, that estimates the relevance score of the item i for the user u. As mentioned before, the scoring function f and its associated parameters θ can be learned by solving a classification task, which classifies the item i as positive class ($y_{u,i} = 1$) if it is clicked by the user u (or manually assessed as relevant) or as negative class ($y_{u,i} = 0$) if it is not clicked by the user u (or manually assessed as irrelevant). The parameters θ of the classification function can be learned by maximizing the likelihood of correct relevance of an item i for a user u:

$$\arg\max_{\theta} \prod_{(u,i)\in D,\, y_{u,i}=1} p(i|u) \prod_{(u,i)\in D,\, y_{u,i}=0} (1 - p(i|u)) \tag{2}$$

This amounts to minimizing the binary cross entropy loss: $\mathcal{L}_{\text{pointwise}}$, in Equation 3.

$$\mathcal{L}_{\text{pointwise}}(\theta) = -\sum_{(u,i)\in D} \Big(y_{u,i} \log \sigma\big(f(u, i|\theta)\big) + (1 - y_{u,i}) \log \Big(1 - \sigma\big(f(u, i|\theta)\big)\Big)\Big) \tag{3}$$

where D denotes the set of pairs $< u, i >$ made up of a user u and all the items i that were proposed to her [1]. For clarity reasons, we dropped the regularisation term on θ. Obviously, in practice, this regularisation term is included in the complete loss.

3.2 Pairwise Personalized Ranking from Implicit Feedback

Pairwise ranking focuses on the relative order between a pair of items to achieve correct ranking of this set of items. One specific version of pairwise ranking is Bayesian Personalized Ranking (BPR) [24] which optimizes the model parameters (during training) by maximizing the probability of an item i to be preferred over an item j for a user u. This probability is typically formulated as:

$$p(i > j|u) = p(j < i|u) = \sigma(f(u, i|\theta) - f(u, j|\theta)) \tag{4}$$

[1] In this paper, we assume to be in the usual recommendation setting, where the user is presented with a list of recommended (aka impressed) items and where these lists are known in the training data. In this case, the "negative" items are restricted to the impressed items that were not clicked.

where $f(u, i|\theta)$ is the scoring function that assigns a relevance score of the item i for the user u and σ is the sigmoid function. Let's introduce the variable $y_{u,i>j}$, with $y_{u,i>j} = 1$ when user u prefers item i over j, and $y_{u,i>j} = 0$ otherwise. We expect $p(i > j|u)$ to be high when $y_{u,i>j}$=1, and to be low in the opposite case ($y_{u,i>j} = 0$).

Maximizing the user preference over different items corresponds to maximizing the likelihood of correct ordering for any item pair (i, j) given a user u:

$$\arg\max_{\theta} \prod_{(u,i,j)\in D', y_{u,i>j}=1} p(i > j|u) \prod_{(u,i,j)\in D', y_{u,i>j}=0} \big(1-p(i > j|u)\big) \tag{5}$$

where D' is the set of all pairs with different relevance labels (hence, pairs with one clicked item and one non-clicked item). Maximizing this likelihood amounts to minimizing the following binary cross entropy loss:

$$\mathcal{L}_{\text{pairwise}}(\theta) = -\sum_{(u,i,j)\in D'} \Big(y_{u,i>j} \log \sigma\big(f(u, i|\theta) - f(u, j|\theta)\big) + (1 - y_{u,i>j}) \log\big(1 - \sigma\big(f(u, i|\theta) - f(u, j|\theta)\big)\big)\Big) \tag{6}$$

4 ADAPTIVE POINTWISE-PAIRWISE PERSONALIZED RANKING FROM IMPLICIT FEEDBACK

The pairwise ranking approach better formulates the ranking problem by using relative preference (ordering) between two items. In particular, it does not require any inter-user normalisation, which might be necessary in pointwise approaches to mitigate the inter-user variance. Pairwise approaches are less sensitive to class imbalance than pointwise approaches [26] but are more sensitive to label noise [5]. Implicit feedback naturally brings some noise since some irrelevant items could be clicked by mistake or some relevant items might not be clicked by the user.

We propose a learning strategy so that the model can decide itself, for each triplet $< u, i, j >$, which one of the pointwise and the pairwise approaches should be adopted. Actually, the surrogate loss we propose considers that there is a continuum between the pointwise and the pairwise approaches: how to position the cursor in this continuum should be learned from the data and should be dependent on the considered triplet. This can be done by utilizing a coefficient γ which softly determines the compromise between pointwise and pairwise ranking:

$$p(i > j|u) = \sigma(f(u, i|\theta) - \gamma f(u, j|\theta)) \tag{7}$$

where γ can take values in $[0, 1]$ and, instead of being a simple hyperparameter, we propose to compute it as a learnable function – that we call the "adaptive mixing function" – depending on user u, items i and j (possibly including the ranks of these items), with parameters θ_g: $\gamma = g(u, i, j|\theta_g)$. One possible formulation of g would be a similarity function (as described later, in Section 5.1.3).

Consequently, the negative log-likelihood loss function can be formulated as:

$$\mathcal{L}_{\text{adaptive}}(\theta, \theta_g) = -\sum_{(u,i,j)\in D'} \Big(y_{u,i>j} \log \sigma\big(f(u, i|\theta) - g(u, i, j|\theta_g) f(u, j|\theta)\big) + (1 - y_{u,i>j}) \log\Big(1 - \sigma\big(f(u, i|\theta) - g(u, i, j|\theta_g) f(u, j|\theta)\big)\Big)\Big) \tag{8}$$

Once again, for clarity reasons, we dropped the regularisation terms on θ and θ_g. In practice, the regularisation terms are included.

Equation 8 reduces to Equation 3 when γ, *i.e.* $g(u, i, j|\theta_g)$, is equal to 0, and to Equation 6 when γ is equal to 1. In particular, depending on the particular triplet $< u, i, j >$, the corresponding term in the loss function could correspond to (i) learning to classify the instance i as positive when $g(u, i, j|\theta_g)$ is close to 0 and $y_{u,i>j}$=1; (ii) learning to classify an instance i as negative when $g(u, i, j|\theta_g)$ is close to 0 and $y_{u,i>j}$=0; and (iii) learning to rank i higher than j when $g(u, i, j|\theta_g)$ is close to 1 and $y_{u,i>j}$=1.

Note that the complete loss function considers all "positive" pairs ($y_{u,i>j} = 1$) as well as all "negative" pairs ($y_{u,i>j} = 0$). While these two terms could seem to be redundant at first glance, this is not the case: only considering the first term would lead to a trivial solution where $g(u, i, j|\theta_g)$ is 0 everywhere and the problem would reduce to training a binary classifier with a loss function including only the positive examples (any model $f(.|\theta)$ that outputs a very high constant value would be satisfactory). The second term forces the loss function to include the "negative" examples as well, either in a pointwise way, or in a pairwise way.

5 EXPERIMENTS

5.1 Personalized Recommendation Ranking Model Details

5.1.1 User and Item Representations. We are considering here a personalized news recommendation task using implicit feedback. We adopt a content-based approach for solving the recommendation problem, taking as user features her previous interactions with the system. A user action history at time t with N interactions is denoted as $S_t^u = \big(I_1^u, I_2^u, \ldots, I_N^u\big)$. Each interaction of user u is defined at the impression level and consists of her (implicit) feedback on a list of K recommended items. We will denote this interaction by the list $I^u = \big((i_1, y_{u,i_1}), \ldots, (i_k, y_{u,i_k}), \ldots, (i_K, y_{u,i_K})\big)$, which contains clicked ($y_{u,i_k} = 1$) and non-clicked ($y_{u,i_k} = 0$) items from a list of K recommended items. A user is represented by features derived from her previous interactions (clicked and not clicked items) in the following way:

$$\mathbf{x}_{u_t} = \boldsymbol{\mu}^+_{u_t} - \boldsymbol{\beta} \odot \boldsymbol{\mu}^-_{u_t} \tag{9}$$

$$\boldsymbol{\mu}^+_{u_t} = \frac{1}{\big|\{i|y_{u,i} = 1\}\big|} \sum_{\{i|y_{u,i}=1\}} \mathbf{x}_i \tag{10}$$

$$\boldsymbol{\mu}^-_{u_t} = \frac{1}{\big|\{i|y_{u,i} = 0\}\big|} \sum_{\{i|y_{u,i}=0\}} \mathbf{x}_i \tag{11}$$

where $\mathbf{x}_i \in \mathbb{R}^d$ is the embedding of item i, $\boldsymbol{\mu}^+_{u_t} \in \mathbb{R}^{d\times 1}$ is the mean of the user u's clicked items' embeddings at time t (user positive centroid) and $\boldsymbol{\mu}^-_{u_t} \in \mathbb{R}^{d\times 1}$ is the mean of the user u's non-clicked

items' embeddings at time t (user negative centroid). $\boldsymbol{\beta} \in \mathbb{R}^{d\times 1}$ is parameter that is learned with other parameters, and it scales the influence of the user negative centroid. The notation of time t is dropped when it is clear from the context. $\odot$ denotes element-wise product.

5.1.2 Relevance Scoring Function. The relevance scoring of an item i for a user u is calculated from a simple bilinear form: $f(u, i|W) = \mathbf{x}_i^{\mathsf{T}}\, \mathbf{W}\, \mathbf{x}_u$ by using a diagonal weight matrix $\mathbf{W} \in \mathbb{R}^{d\times d}$. This amounts to consider a generalized dot product between the user's and the item's representation, where the scaling factor (or weight) for each dimension is learned from the data.

5.1.3 Adaptive Mixing Function. Here again, we adopt a relatively simple model for our $g(i, j|\theta_g)$ adaptive mixing function by using a bilinear form:

$$g(i, j|\theta_g) = \frac{\exp(\mathbf{x}_i^{\mathsf{T}}\, \mathbf{W}_g\, \mathbf{x}_j)}{\sum_{(i',j')\in\mathcal{B}} \exp(\mathbf{x}_{i'}^{\mathsf{T}}\, \mathbf{W}_g\, \mathbf{x}_{j'})} \tag{12}$$

where $\mathbf{W_g} \in \mathbb{R}^{d\times d}$ is a diagonal weight matrix, and $\mathcal{B}$ corresponds to the minibatch. In other words, it is defined by the softmax of a generalised dot product over the pairs included in the minibatch. In Equation 12, we expect the aligned $\mathbf{x}_i$ and $\mathbf{x}_j$ lead to higher $g(i, j|\theta_g)$, and lower otherwise.

5.2 Datasets

For our experiments, we use two personalized news recommendation datasets: one (proprietary) dataset consists of click logs of one of the main news portal in Korea (here after designated as INR), and a public Outbrain news recommendation dataset from the Outbrain click prediction Kaggle Challenge. Each dataset contains implicit feedback in the form of click/no-click information for each impressed item in given recommendation lists.

The Outbrain click prediction dataset consists of a sample of users' page views and clicks, as observed on multiple publisher sites for fourteen days. The length of the recommendation lists, which user viewed/interacted with, is on average five articles, and it can go up to twelve.

INR news dataset consists of two subsets, corresponding to two different genres of news: entertainment and sports; each subset includes logs of users' news reading activity for seven days. Users' reading activities are composed of news article impressions and clicks on recommended news. Each recommended news article list is composed of 28 news articles[2]. For Outbrain click prediction dataset, each news article is represented as a concatenation of the category, topic and entity features (total dimension of 24K). For the experimental setup we restricted the users' activities to the impressions with at least one click.

INR news article texts were lemmatized by using the KoNLPy toolkit [21], and lemmas appearing less than 3 times were discarded. After preprocessing, the resulting vocabulary size is 40K for entertainment news and 30K for sports news. A news article was represented by the bag of words of the news' content[3].

5.3 Experimental Setup

We split the dataset into train, validation and test sets by temporal order, which is the setting that corresponds most closely to real conditions. For the INR dataset, the first three days form the training set, the fourth day forms the validation set, and the last three days form the test set. For Outbrain dataset, the first seven days form training, the following three days form validation and the last four days form the test set. For each dataset, the experiments were repeated for each model five times, and each run is randomly initialized with a predefined seed specific to this run. For each run, 1000 users for the INR dataset are selected at random, 5000 users for the Outbrain. Furthermore, each user has at least one click in the training, validation and test sets.

Our adaptive pointwise-pairwise (adaptive pp) ranking approach is compared with the following baselines: the "pure" pointwise, pairwise (BPR) [24] and listwise ranking approaches, namely listNET (cross-entropy) [4], λRank [1], and listwise Average Precision (listAP) [3, 25][4]; as well as with methods utilizing both pointwise and pairwise losses: alternating pointwise-pairwise (alternating pp) method described in [17] and joint pointwise-pairwise (joint pp) method proposed in [27]. Here, the same architecture (Section 4 and Section 5.1) is trained by using different surrogate losses. All models are implemented[5] in PyTorch [22]. The model parameters are optimized by Adam optimizer [16], and with the binary cross entropy loss on the output (see Equation 3, 6, 8).

The mini-batch size is chosen as 128. For regularization, L_1 norm and batch normalization [13] are applied. For hyperparameter tuning, we used a grid search over the learning rate values of $\{10^{-2}, 10^{-3}, 10^{-4}\}$, and the regularization coefficient values of $\{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$. The chosen hyperparameter setting is the one which leads to the best average normalized Discounted Cumulative Gain (NDCG) over five runs on the validation set.

In Section 6, we further assess the ranking performance of the proposed and baseline ranking approaches in terms of Hit Ratio at rank 1 (HR@1), normalized Discounted Cumulative Gain at rank K[6] (NDCG@K), and mean reciprocal rank (MRR@K). We report the average score on the test set over five runs; the standard deviation is also indicated between brackets. Furthermore, the statistical significance of the results is assessed by using Wilcoxon signed-rank test with Bonferroni correction.

6 RESULTS

Table 1 illustrates the personalized news recommendation performances of different ranking approaches on Outbrain, INR Entertainment and INR Sports datasets. On all three datasets, our adaptive pointwise-pairwise ranking significantly improves over solely pointwise, pairwise, listwise (listNet, λRank, and listAP), but also

[2] Actually, the layout interface only displays 8 news items, and the the user has to scroll down the list in order to get access to another subset of 8 news. This motivates to use 8 as intermediate threshold when assessing common retrieval metrics such as NDCG.

[3] We investigated the use of different unsupervised (non-sequential) word embeddings for this task, but found no significant improvement over a simple appropriately weighted bag-of-words representation.

[4] The loss function is taken from https://github.com/almazan/deep-image-retrieval.

[5] https://github.com/ygcinar/pointwise-pairwise-recommendation.

[6] K is determined depending on the recommendation list size for each dataset.

Table 1: Results averaged over 5 folds on Outbrain, INR Entertainment, and INR Sports. The standard deviation is given in paranthesis. The best results are in bold. "$\downarrow$" indicates a model significantly worse than the best one according to a Wilcoxon signed-rank test using Bonferroni correction at 5% and "$\Downarrow$" at 1%.

		HR@1	NDCG@12	MRR@12
Outbrain	pointwise	$0.2776^{\Downarrow}$(0.0056)	$0.6408^{\Downarrow}$(0.0035)	$0.5241^{\Downarrow}$(0.0047)
	BPR	$0.2895^{\downarrow}$(0.0090)	$0.6474^{\downarrow}$(0.0050)	$0.5328^{\downarrow}$(0.0066)
	listNET	$0.2692^{\Downarrow}$(0.0047)	$0.6335^{\Downarrow}$(0.0025)	$0.5148^{\Downarrow}$(0.0033)
	listAP	$0.2717^{\Downarrow}$(0.0018)	$0.6349^{\Downarrow}$(0.0013)	$0.5166^{\Downarrow}$(0.0017)
	λRank	$0.2791^{\Downarrow}$(0.0036)	$0.6394^{\Downarrow}$(0.0023)	$0.5224^{\Downarrow}$(0.0030)
	alternating pp	$0.2861^{\Downarrow}$(0.0046)	$0.6453^{\Downarrow}$(0.0023)	$0.5301^{\Downarrow}$(0.0031)
	joint pp	$0.2868^{\Downarrow}$(0.0065)	$0.6461^{\Downarrow}$(0.0034)	$0.5310^{\Downarrow}$(0.0045)
	adaptive pp (ours)	**0.2948(0.0040)**	**0.6511(0.0024)**	**0.5376(0.0031)**

		HR@1	NDCG@8	NDCG@28	MRR@28
INR Entertainment	pointwise	$0.1596^{\Downarrow}$(0.0043)	$0.3059^{\Downarrow}$(0.0053)	$0.4502^{\Downarrow}$(0.0029)	$0.3151^{\Downarrow}$(0.0042)
	BPR	$0.1568^{\Downarrow}$(0.0075)	$0.2968^{\Downarrow}$(0.0037)	$0.4447^{\Downarrow}$(0.0032)	$0.3090^{\Downarrow}$(0.0058)
	listNET	$0.1469^{\Downarrow}$(0.0026)	$0.2897^{\Downarrow}$(0.0035)	$0.4393^{\Downarrow}$(0.0013)	$0.3013^{\Downarrow}$(0.0025)
	listAP	$0.1638^{\Downarrow}$(0.0022)	$0.2999^{\Downarrow}$(0.0037)	$0.4474^{\Downarrow}$(0.0017)	$0.3141^{\Downarrow}$(0.0026)
	λRank	$0.1513^{\Downarrow}$(0.0170)	$0.3032^{\Downarrow}$(0.0088)	$0.4466^{\Downarrow}$(0.0075)	$0.3089^{\Downarrow}$(0.0126)
	alternating pp	$0.1699^{\Downarrow}$(0.0020)	$0.3034^{\Downarrow}$(0.0056)	$0.4498^{\Downarrow}$(0.0032)	$0.3189^{\Downarrow}$(0.0039)
	joint pp	$0.1592^{\Downarrow}$(0.0054)	$0.2963^{\Downarrow}$(0.0014)	$0.4450^{\Downarrow}$(0.0018)	$0.3098^{\Downarrow}$(0.0034)
	adaptive pp (ours)	**0.1767(0.0047)**	**0.3242(0.0035)**	**0.4624(0.0023)**	**0.3324(0.0035)**
INR Sports	pointwise	$0.1897^{\downarrow}$(0.0061)	$0.3539^{\Downarrow}$(0.0034)	$0.4757^{\Downarrow}$(0.0032)	$0.3456^{\Downarrow}$(0.0048)
	BPR	$0.1847^{\Downarrow}$(0.0049)	$0.3540^{\Downarrow}$(0.0041)	$0.4746^{\Downarrow}$(0.0032)	$0.3436^{\Downarrow}$(0.0046)
	listNET	$0.1514^{\Downarrow}$(0.0051)	$0.3274^{\Downarrow}$(0.0047)	$0.4536^{\Downarrow}$(0.0041)	$0.3150^{\Downarrow}$(0.0054)
	listAP	$0.1838^{\Downarrow}$(0.0067)	$0.3430^{\Downarrow}$(0.0078)	$0.4688^{\Downarrow}$(0.0052)	$0.3381^{\Downarrow}$(0.0065)
	λRank	$0.1699^{\Downarrow}$(0.0392)	$0.3360^{\Downarrow}$(0.0325)	$0.4618^{\Downarrow}$(0.0260)	$0.3266^{\Downarrow}$(0.0382)
	alternating pp	$0.1890^{\Downarrow}$(0.0049)	$0.3495^{\Downarrow}$(0.0043)	$0.4726^{\Downarrow}$(0.0035)	$0.3425^{\Downarrow}$(0.0051)
	joint pp	$0.1863^{\Downarrow}$(0.0086)	$0.3520^{\Downarrow}$(0.0036)	$0.4736^{\Downarrow}$(0.0037)	$0.3433^{\Downarrow}$(0.0061)
	adaptive pp (ours)	**0.1964(0.0083)**	**0.3742(0.0052)**	**0.4874(0.0045)**	**0.3596(0.0063)**

over the combined (multi-objective) pointwise-pairwise baselines, namely, alternating pp and joint pp ranking approaches.

Among the baselines on Outbrain, BPR is the leading baseline, but pointwise is the leading one on the INR sports dataset, demonstrating that the optimal choice between a pure point- or pairwise approach is dataset dependent. Pointwise (in terms of NDCG@8 and NDCG@28) and alternating pointwise-pairwise (in terms of HR@1 and MRR@28) are the leading approaches among the baselines on INR entertainment dataset. On the INR sports and Outbrain datasets, pointwise-pairwise baselines, *i.e.*, alternating pp and joint pp ranking, performs in between or poorer than solely pointwise and pairwise approaches. The ListAP and λRank losses achieves better ranking scores than ListNET. However, their performance is still lower than our *adaptive pp* approach, probably due to their sensitivity to label noise and bias. The results show the effectiveness of our adaptive pointwise-pairwise approach.

7 CONCLUSION

In this paper, we proposed a new learning-to-rank loss that forms a continuum between the pointwise and pairwise losses It has the ability to learn to combine these two basic losses into a single one optimally and adaptively for each pair or triplet. Our recommendation results on several personalized news recommendation datasets demonstrate the effectiveness of this new learning-to-rank loss. In the future, we plan to extend this adaptive mixing approach to integrate also the listwise approaches.

REFERENCES

[1] Christopher J. Burges, Robert Ragno, and Quoc V. Le. 2007. Learning to Rank with Nonsmooth Cost Functions. In *Advances in Neural Information Processing Systems 19*, B. Schölkopf, J. C. Platt, and T. Hoffman (Eds.). MIT Press, 193–200.

[2] Christopher J. C. Burges. 2010. *From RankNet to LambdaRank to LambdaMART: An Overview*. Technical Report. Microsoft Research. http://research.microsoft.com/en-us/um/people/cburges/tech_reports/MSR-TR-2010-82.pdf

[3] Fatih Cakir, Kun He, Xide Xia, Brian Kulis, and Stan Sclaroff. 2019. Deep Metric Learning to Rank. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[4] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to Rank: From Pairwise Approach to Listwise Approach. In *Proceedings of the 24th International Conference on Machine Learning* (Corvalis, Oregon, USA) *(ICML '07)*. ACM, New York, NY, USA, 129–136.

[5] Vitor R Carvalho, Jonathan L Elsas, William W Cohen, and Jaime G Carbonell. 2008. A meta-learning approach for robust rank learning. In *SIGIR 2008 workshop on learning to rank for information retrieval*, Vol. 1.

[6] Koby Crammer and Yoram Singer. 2002. Pranking with Ranking. In *Advances in Neural Information Processing Systems 14*, T. G. Dietterich, S. Becker, and Z. Ghahramani (Eds.). MIT Press, 641–647.

[7] Jingtao Ding, Guanghui Yu, Xiangnan He, Yong Li, and Depeng Jin. 2018. Sampler Design for Bayesian Personalized Ranking by Leveraging View Data. *ArXiv* abs/1809.08162 (2018).

[8] Wenkui Ding, Xiubo Geng, and Xu-Dong Zhang. 2015. Learning to Rank from Noisy Data. *ACM Trans. Intell. Syst. Technol.* 7, 1, Article 1 (Oct. 2015), 21 pages.

[9] Yoav Freund, Raj Iyer, Robert E. Schapire, and Yoram Singer. 2003. An Efficient Boosting Algorithm for Combining Preferences. *J. Mach. Learn. Res.* 4 (Dec. 2003), 933–969.

[10] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 580–587.

[11] Xiangnan He, Jinhui Tang, Xiaoyu Du, Richang Hong, Tongwei Ren, and Tat-Seng Chua. 2019. Fast Matrix Factorization With Nonuniform Weights on Missing Data. *IEEE Transactions on Neural Networks and Learning Systems* (2019), 1–14.

[12] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent Neural Networks with Top-k Gains for Session-based Recommendations. *Proceedings of the 27th ACM International Conference on Information and Knowledge Management* (Oct 2018). https://doi.org/10.1145/3269206.3271761

[13] Sergey Ioffe and Christian Szegedy. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *CoRR* abs/1502.03167

(2015). arXiv:1502.03167 http://arxiv.org/abs/1502.03167

[14] Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. 1991. Adaptive Mixtures of Local Experts. *Neural Comput.* 3, 1 (March 1991), 79–87.

[15] Thorsten Joachims. 2002. Optimizing Search Engines Using Clickthrough Data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Edmonton, Alberta, Canada) *(KDD '02)*. ACM, New York, NY, USA, 133–142.

[16] Diederik Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations* (12 2014).

[17] Yu Lei, Wenjie Li, Ziyu Lu, and Miao Zhao. 2017. Alternating Pointwise-Pairwise Learning for Personalized Item Ranking. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management* (Singapore, Singapore) *(CIKM '17)*. ACM, New York, NY, USA, 2155–2158.

[18] Tie-Yan Liu. 2011. *Learning to rank for information retrieval*. Springer Science & Business Media.

[19] Taesup Moon, Alex Smola, Yi Chang, and Zhaohui Zheng. 2010. IntervalRank: Isotonic Regression with Listwise and Pairwise Constraints. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining* (New York, New York, USA) *(WSDM '10)*. Association for Computing Machinery, New York, NY, USA, 151–160.

[20] Rong Pan, Yunhong Zhou, Bin Cao, Nathan Nan Liu, Rajan M. Lukose, Martin Scholz, and Qiang Yang. 2008. One-Class Collaborative Filtering. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*. 502–511.

[21] Eunjeong L. Park and Sungzoon Cho. 2014. KoNLPy: Korean natural language processing in Python. In *Proceedings of the 26th Annual Conference on Human & Cognitive Language Technology*. Chuncheon, Korea.

[22] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic Differentiation in PyTorch. In *NIPS Autodiff Workshop*.

[23] Steffen Rendle and Christoph Freudenthaler. 2014. Improving Pairwise Learning for Item Recommendation from Implicit Feedback. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining* (New York, New York, USA) *(WSDM '14)*. ACM, New York, NY, USA, 273–282.

[24] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence* (Montreal, Quebec, Canada) *(UAI 2009)*. AUAI Press, Arlington, Virginia, United States, 452–461.

[25] Jérôme Revaud, Jon Almazán, Rafael Sampaio de Rezende, and César Roberto de Souza. 2019. Learning with Average Precision: Training Image Retrieval with a Listwise Loss. *CoRR* abs/1906.07589 (2019). arXiv:1906.07589 http://arxiv.org/abs/1906.07589

[26] Suzan Verberne, Hans Halteren, D. Theijssen, Stephan Raaijmakers, and Lou Boves. 2011. Learning to rank for why-question answering. *Information Retrieval* 14 (04 2011), 107–132.

[27] Yilin Wang, Suhang Wang, Jiliang Tang, Huan Liu, and Baoxin Li. 2016. PPP: Joint Pointwise and Pairwise Image Label Prediction. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. 6005–6013.

[28] Ga Wu, Maksims Volkovs, Chee Loong Soon, Scott Sanner, and Himanshu Rai. 2019. Noise Contrastive Estimation for One-Class Collaborative Filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*. 135–144.

[29] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. 2008. Listwise Approach to Learning to Rank: Theory and Algorithm. In *Proceedings of the 25th International Conference on Machine Learning* (Helsinki, Finland) *(ICML '08)*. Association for Computing Machinery, New York, NY, USA, 1192–1199.

[30] Jun Xu and Hang Li. 2007. AdaRank: A Boosting Algorithm for Information Retrieval. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Amsterdam, The Netherlands) *(SIGIR '07)*. ACM, New York, NY, USA, 391–398.

[31] Nengjun Zhu and Jian Cao. 2019. CPL: A Combined Framework of Pointwise Prediction and Learning to Rank for top-N Recommendations with Implicit Feedback. In *WISE*.