NeuTraL: Neural Transfer Learning for Personalized Ranking

Rasaq Otunba

4400 University Drive, Fairfax, Virginia 22030

Abstract

Personalized ranking continues to be an important aspect of many information systems and personalization systems. Neural networks and deep learning continue to gain popularity because of their success in different fields of artificial intelligence such as computer vision and natural language processing. Recently, researchers began to apply deep learning to personalized ranking with success. Most personalization systems exploit historical preference data for users and items in warm-start scenario. A major challenge in personalized ranking occurs in the cold-start scenario which arises when there is little to no historical preference information. Content information is sometimes available and it can be used to alleviate the cold-start problem.

We propose a solution that involves transfer learning from a deep model to a shallow model for both warm-start and cold-start personalized ranking. We corroborate our proposal with experiments on publicly available datasets in comparison with other baseline and state-of-the-art techniques.

Keywords

neural networks; deep learning; recommendations; personalization; cold-start; ranking

1. Introduction

Personalized ranking with adequate historical preference is referred to as *warm-start* while recommendation with inadequate historical preference is referred to as *cold-start*. We subsequently refer to personalized ranking as ranking except otherwise clearly stated. We propose a machine learning solution called **Neu**ral **Tra**nsfer **L**earning for warm-start personalized ranking, otherwise referred to as **NeuTraL**. We then propose a cold-start version of NeuTraL referred to as NeuTraL-C. NeuTraL and NeuTraL-C use neural networks and transfer learning for warm-start and cold-start item ranking respectively. Item cold-start personalized ranking involves ranking cold-start items while user cold-start personalized ranking involves ranking cold-start users. There is also the full cold-start entity personalized ranking problem where both the user and item entities have no historical preference information. Although we focus on cold-start item personalized ranking in this work, we believe the concept is extensible to both user cold-start and full cold-start personalized ranking problems. Entity content information is sometimes used to compensate for the lack of historical preference information by learning from content information and existing preference information. Ranking can be done for implicit or explicit feedback [1]. We focus on implicit feedback in this work due to its more prevalent nature. The contributions made in this work include:

- We propose a unique approach to extracting pre-trained user latent factors from a state-of-the-art (SOTA) personalization model.
- The transfer of the pre-trained user latent factors to a renowned personalization model for warm-start and cold-start ranking respectively.
- We provide thorough evaluation and conduct experiments comparing our proposed solutions with other SOTA and baseline techniques.

The remainder of this paper is organized as follows: in Section 2, we highlight related work. We provide pertinent background and notations for the rest of this work in Section 3. We describe our approach in Section 5. In Section 6, we describe our experiments and discussed the results in section Section 6.3.3. We conclude with potential directions for future work in Section 7.

2. Related Work

Personalized ranking techniques typically belong in one of the following categories: collaborative filtering (CF), content-based or a hybrid of the aforementioned techniques. Different CF techniques ranging from matrix factorization (MF) [2, 3] to k-Nearest Neighbor (kNN) [4] have seen success in personalization systems research. In recent years, deep learning has also been successfully applied for personalization. He et al. replaced the typical dot product of user and item latent features with a deep learning model in their technique referred to as neural collaborative filtering, NCF [5]. NCF performs better than the vanilla MF because the non-linearity of the deep learning model captures complex interactions between users

Published in the Workshop Proceedings of the EDBT/ICDT 2022 Joint Conference (March 29-April 1, 2022), Edinburgh, UK

rotunba@gmu.edu (R. Otunba)

and items better. Deep representation models such as autoencoders and restricted Boltzmann machines (RBM) have been used for personalization [6, 7, 8]. These techniques have been successfully applied and demonstrated on a variety of real world data, but they are known to suffer from the cold-start problem. Content-based techniques are typically used to tackle the cold-start problem by incorporating entity attributes [9, 10]. Entity attributes are sometimes combined with CF to compensate for the weakness in CF [11, 12] for the cold-start scenario. To alleviate the cold-start problem, some deep learning techniques have been developed with the use of content information, e.g., the deep content-based music recommendation work proposed by Oord et al. [13]. Most of the deep learning personalization systems proposed for cold start are hybrid in that they combine historical preference and content information [14, 15, 16, 17, 18, 19]. Some of the cold-start personalization systems [20] adopt active learning. However, there are situations where active feedback from users for the cold start items are unavailable. Transfer learning has also been used in personalization systems research [21, 22].

3. Background & Notations

The set of users and items are denoted by U and I, respectively. A measure of preference is recorded as a positive feedback from some set P or as a negative feedback recorded as 0. When explicitly provided, P could be a set of values e.g., $\{1, 2, ..., 5\}$. When implicitly provided, typically $P = \{0, 1\}$. The matrix of user-item interactions is denoted by:

$$\mathbf{Y} \in (\{0\} \cup P)^{|U| \times |I|}, \tag{1}$$

where an interaction refers to an observable action by a user e.g., the purchase of an item. User vector for user u in Y is denoted as y_u. Conversely, item vector for item i in Y is denoted as y_i^T. The implicit feedback for a user $u \in U$ on an item $i \in I$ is:

$$y_{ui} = \begin{cases} 1, & \text{if } u \text{ interacted with } i; \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

$$I_u^+ = \{\text{set of items interacted with by user } u\}. \tag{3}$$

$$I_u^- = I - I_u^+ \tag{4}$$

U^+, U^-, and U are user sets analogous to the definitions in Equations 3– 4. $\mathbf{A}^U$ and $\mathbf{A}^I$ represent the m-dimensional user-attribute and n-dimensional item-attribute matrices, respectively.

$$\mathbf{A}^U \in \mathbb{R}^{|U| \times m}, \tag{5}$$

$$\mathbf{A}^I \in \mathbb{R}^{|I| \times n}. \tag{6}$$

Let $\mathbf{a}_u^U$ be the vector of user attributes $1 \dots m$ for user u, and $\mathbf{a}_i^I$ be the vector of item attributes $1 \dots n$ for item i, so that a_{ik}^I is the k-th item attribute value and a_{ik}^U is the k-th user attribute value. $a_{ik}^I = 0$ when the attribute is unavailable. Sets U and I are represented by latent feature matrices $\mathbf{U}$ and $\mathbf{I}$ respectively where

$$\mathbf{U} \in \mathbb{R}^{|U| \times r} \tag{7}$$

$$\mathbf{I} \in \mathbb{R}^{|I| \times r}, \tag{8}$$

where r is the number of latent features. User u and item i are represented by $\mathbf{u}$ and $\mathbf{i}$, respectively. Content data would sometimes contain only user attributes, item attributes or both. User attributes include demographic information such as age and gender, education level, etc. Social network data can also be mined for user attributes. Item attributes include physical attributes, time of production, location, etc. The task of item ranking is to estimate the relative ranking of the items for each user. We denote the predicted ranking of item i for user u as $\hat{y}_{ui}$ from an inference function f:

$$\hat{y}_{ui} = f(\mathbf{u}, \mathbf{a}_u^U, \mathbf{i}, \mathbf{a}_i^I, \theta), \tag{9}$$

where θ denotes the model parameters learned during training. Equation 9 shows $\hat{y}_{ui}$ is a function of the input and learned model parameters. Model parameters are typically learned via optimization such that an objective loss function is minimized or a utility function is maximized. Objective loss function minimization is expressed as:

$$\theta_E = \arg\min_{\theta} \mathcal{L}(\theta; \mathbf{Y}), \tag{10}$$

where θ is learned from observation matrix $\mathbf{Y}$ to optimize the estimate function θ_E that predicts $\hat{y}_{ui}$. Learning is usually done with machine learning techniques such as gradient descent (GD) [23] or its variants e.g., Adaptive Moment Estimation (Adam) [24] on carefully sampled user-item pairs.

4. NeuTraL: Neural Transfer Learning for Personalized Ranking

We provide further background on pertinent information that will aid the understanding of NeuTraL.

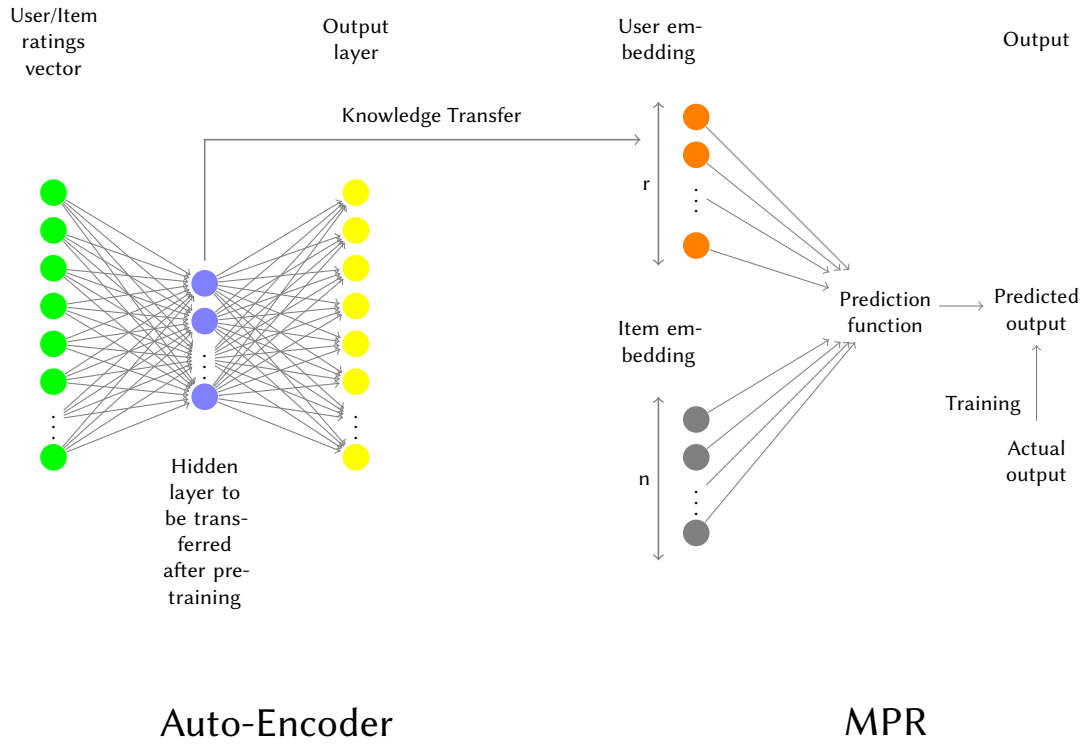

Figure 1: NeuTraL: Left side shows the pre-trained Auto-Encoder with the transfer to MPR

4.1. MPR: Multi-Objective Pairwise Ranking

MPR is of the pairwise ranking function family where the optimization task is with respect to the actual and predicted values for a pair of items by a user. For item ranking, the pairwise prediction function for a user u, a preferred item i and a less preferred item j is expressed as

$$\hat{y}_{u(i,j)} = \hat{y}_{ui} - \hat{y}_{uj}, \tag{11}$$

while the actual value is

$$y_{u(i,j)} = y_{ui} - y_{uj}. \tag{12}$$

Conversely, for user ranking, the pairwise prediction function for an item f preferred by user v but not preferred by user w is expressed as

$$\hat{y}_{f(v,w)} = \hat{y}_{fv} - \hat{y}_{fw}, \tag{13}$$

while the actual value is

$$y_{f(v,w)} = y_{fv} - y_{fw}. \tag{14}$$

MPR combines item ranking and user ranking. The optimization function is expressed as:

$$\sum_{u \in U} \sum_{i \in I_u^+} \sum_{j \in I_u^-} \mathcal{L}(\hat{y}_{u(i,j)}) + \mathcal{L}(\hat{y}_{f(v,w)}), \tag{15}$$

and the objective function $\mathcal{L}$ is the log-sigmoid function:

$$\mathcal{L}(x) = \ln \sigma(x), \tag{16}$$

and

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{17}$$

$\hat{y}_{ui}$ is estimated from a MF model learned with GD. $\hat{y}_{ui}$ is the dot product of the user latent vector u and the item latent vector i.

$$\hat{y}_{ui} = u^T \cdot i \tag{18}$$

Assume

$$u = \{u_1, u_2, \ldots, u_k\} \tag{19}$$

and

$$i = \{i_1, i_2, \ldots, i_k\}. \tag{20}$$

Component u_k of u represents user u's affinity for an item factor k. Component i_k of i represents the concentration of factor k in item i.

$$u^T \cdot i = u_1 * i_1 + u_2 * i_2 \ldots, u_k * i_k \tag{21}$$

.

Each component product $u_k * i_k$ represents user u's affinity for factor k in item i. We subsequently refer to this component product as latent vector product (LVP) for ease of reference.

4.2. Transfer Learning

Transfer learning [25] is premised on the idea that a related pre-trained model can serve as an initializer for a main model. This initialization can be beneficial by speeding up learning and/or improving accuracy on the main task as seen in Figure 4. Transfer learning is similar to multi-task learning (MTL) with the main difference being the sequential versus simultaneous nature of the two techniques, respectively. Transfer learning has been successful in image processing [26] and natural language processing [27] among other areas of machine learning.

4.3. Auto-Encoders & Personalization

Auto-encoders have been successfully applied in personalization systems [7, 6]. Auto-encoders derive their name from the ability to encode input data with un-supervised learning. The utility of auto-encoders include dimensionality reduction of input while ignoring noise in the input optimally. For the purpose of personalization, entity vector data is passed as input with missing entries. The goal is to recover the original input in the output including the missing entries. To the best of our knowledge, the pioneer research work in this area is AutoRec [7]. User vectors y_u or item vectors y_i can serve as input where each vector component is the actual preference value or a missing entry. However, the authors of AutoRec stated that user vector inputs performed better than item vector inputs, and we observed the same in our experiments. Perhaps this is due to the peculiar characteristics of the datasets used, e.g., number of users and items, ratings per item and ratings per user. Wu et al. presented a more sophisticated auto-encoder personalization technique, Collaborative Denoising Auto-Encoders (CDAE) [6] which incorporates denoising with dropout [28] and an extra identifier input. Dropout can be seen as a form of noise introduction [29].

Deep learning techniques have the advantage of being able to model linear and non-linear complex interactions between users and items. Auto-encoders for personalization are depicted in Figure 1. We denote the nodes in the input layer as $\hat{y}_u^0$, hidden layer as $\hat{y}_u^1$ and the output layer as $\hat{y}_u^2$ where

$$\hat{y_u^0} = f_0(y_u, u), \tag{22}$$

and f_0 is a concatenation function. The nodes vector in the hidden layer are:

$$\hat{y_u^1} = f_1(W_1^T \cdot \hat{y}_u^0 + b_1). \tag{23}$$

W_1 is the g x h weight matrix between the input and hidden layers. g and h are the number of nodes in the input and hidden layers respectively. b_1 is the bias for the hidden layer. f_1 is an activation function.

$$\hat{y_u^2} = f_2(W_2^T \cdot \hat{y}_u^1). \tag{24}$$

W_2 is the h x g weight matrix between the hidden and output layers. f_2 is an activation function. We use sigmoid activation functions since they produced optimal results. W_1, W_2 and b_1 are model parameters. There are also hyper-parameters such as learning rate, batch size and objective function that should be tuned during training with validation. We use the binary cross-entropy cost function.

$$-\hat{y}_{u(i,j)} \ln y_{uij} - (1 - \hat{y}_{u(i,j)}) \ln(1 - y_{uij}). \tag{25}$$

and backpropagation to update the model parameters.

4.4. NeuTraL Algorithm

The development of NeuTraL as depicted in fig:NeuTraL begins with the supposition that a more representative user embedding model could improve performance in the MF for personalized ranking. A pre-trained neural network model may be appropriate since we are aware of the success of deep learning models in personalization systems. It has also been shown that neural networks are better at modelling complex non-linearity in user-item interactions than MF models [5]. We chose CDAE as our pre-training model based on its proven improvement over AutoRec. User latent features in MF can be considered a form of dimensionality reduction for the user preference vector in Y. A close look at both CDAE and MF reveals that the hidden layer nodes of CDAE are analogous to user latent features as smaller dimension versions of the original user vectors in Y. This analogy implies we can use a pre-trained $|U|$ x k matrix C of hidden layer node values as the user latent feature matrix model which forms the basis for our contribution. We subsequently refer to C as the transfer matrix. In other words, we transfer user vector c_u from C as the latent vector for user u. We leave out the algorithm for NeuTraL since it is essentially the same as the MPR algorithm with the use of the pre-trained user embedding from CDAE.

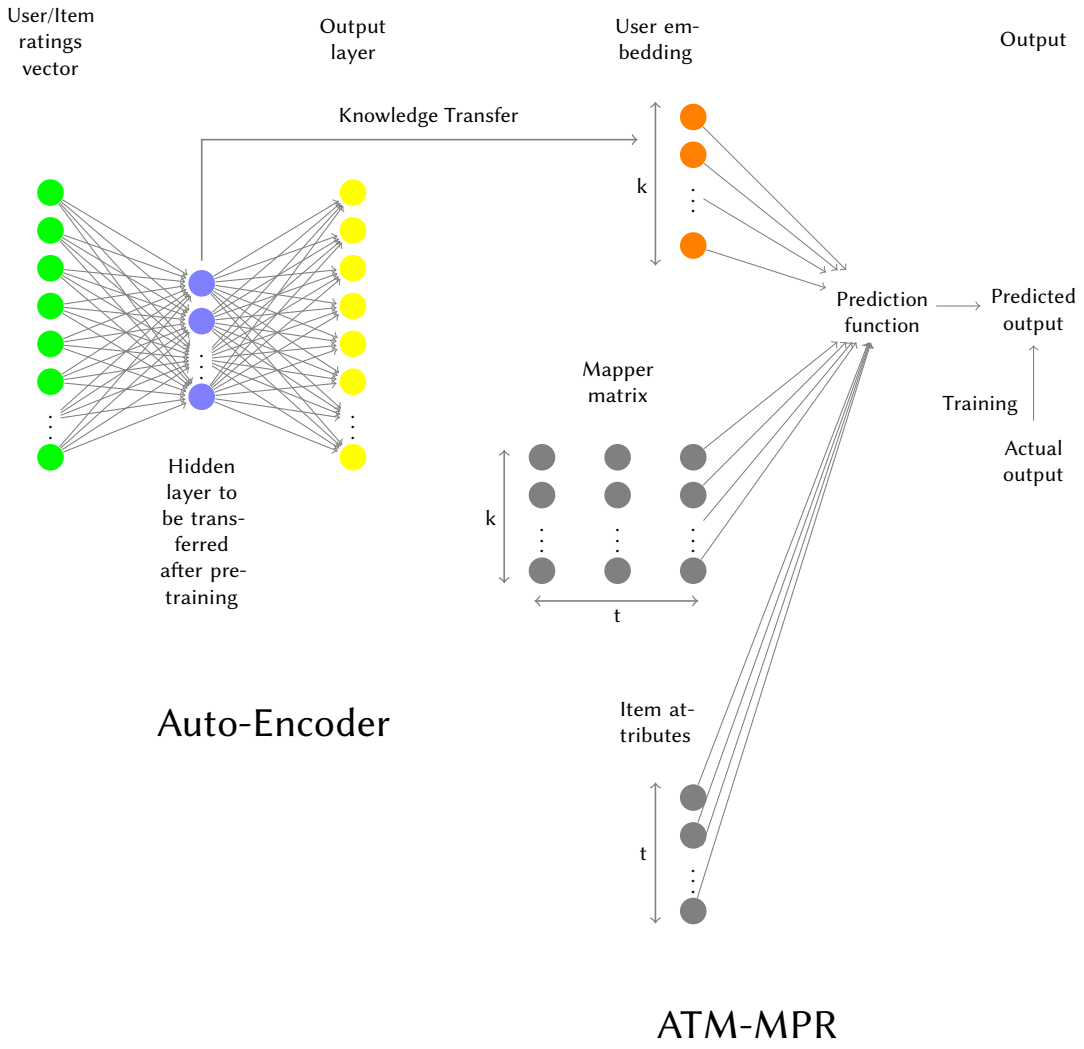

Figure 2: NeuTraL-C: Left side shows the pre-trained Auto-Encoder with the transfer to ATM-MPR

5. NeuTraL-C: Neural Transfer Learning for Cold-Start Personalized Ranking

We provide further background on pertinent information that will aid the understanding of NeuTraL-C as depicted in fig:NeuTraL-C.

5.1. Item Attribute-to-Feature Mappings

Cold-start items have little to no historical preference information to exploit for personalized ranking. Hence recommending cold-start items pose a different challenge. However, both warm-start and cold-start items have item attributes that can be exploited for recommendations. An item Attribute-to-Feature Mapping (ATM) as a framework capable of providing item latent features from item attributes i.e., a function that accepts item attributes as input and produces item latent features as output. The output can then be used in conjunction with user latent features for prediction. We consider the ATM technique presented by Gantner et al [12] referred to as ATM-BPR in this work. ATM-MPR is an extension of the ATM-BPR technique for cold-start personalization.

5.1.1. ATM-MPR

ATM-MPR adds cold-start capability to MPR by learning a shallow linear model of latent features and attributes.

The main differences between MPR and ATM-MPR is the derivation of the item latent vector i where

$$i = \mathcal{M}(a_i^I), \tag{26}$$

and $\mathcal{M}$ is a *mapping function.*

$$\mathcal{M}(a_i^I) = M \cdot a_i^I, \tag{27}$$

where M is a mapper matrix to be learned similar to how U and I are learned in MPR with GD. MPR optimizes the NeuTraL-C optimization criterion which is the same as neutral-opt.
However, the respective prediction functions for user ranking and item ranking in NeuTraL-C are different. We subsequently describe the item ranking prediction function but the user ranking prediction function is analogous. The item ranking prediction function is expressed as:

$$\hat{y}_{u(i,j)} = (u^T \cdot M \cdot a_i^I) - (u^T \cdot M \cdot a_j^I). \tag{28}$$

With transfer learning, the prediction function becomes:

$$\hat{y}_{u(i,j)} = (c_u^T \cdot M \cdot a_i^I) - (c_u^T \cdot M \cdot a_j^I). \tag{29}$$

$$\hat{y}_{u(i,j)} m_i = c_u^T (a_i^I - a_j^I). \tag{30}$$

Hence, M is updated in GD with the following expression:

$$M = M + \alpha \left(Neu\text{-C-}OPTM \right), \tag{31}$$

$$M = M + \alpha \left(\frac{\partial \mathcal{L}(\hat{y}_{u(i,j)})}{\partial \hat{y}_{u(i,j)}} \cdot \frac{\partial \hat{y}_{u(i,j)}}{\partial M} - \lambda_M \cdot M \right), \tag{32}$$

and λ_M is a regularization hyper-parameter.

5.2. NeuTraL-C Algorithm

The NeuTraL-C algorithm is listed in alg-neutral-c

6. Experiments

We proceed to address the following research questions:

- how does NeuTraL compare with other SOTA warm-start item personalization systems.
- how does NeuTraL-C compare with other SOTA cold-start item personalization systems.

We begin by describing our experiment setup. We subsequently describe our experiments on warm-start personalized ranking followed by cold-start.

Algorithm 1 NeuTraL-C(U, M, A)

Output: Optimized matrices **U** and **M**
initialize **U** with the extracted hidden layer matrix C from CDAE
initialize α, η and **M**
repeat
  draw u, i, j from U, I_u^+, I_u^- uniformly
  $u \leftarrow u - \eta * Neu\text{-C-}OPTu$
  $M \leftarrow M - \eta * Neu\text{-C-}OPTM$ wrt a_i^I and a_j^I
  draw f, v, w from I, U_k^+, U_k^- uniformly
  $M \leftarrow M - \eta * Neu\text{-C-}OPTM$ wrt v and w
  $v \leftarrow v - \eta * Neu\text{-C-}OPTv$
  $w \leftarrow w - \eta * Neu\text{-C-}OPTw$
until convergence or maximum number of iterations
return **U**, **M**

6.1. Experimental Repeatability

Experiment Artifacts (software, datasets, etc.) for this work are available on demand. These artifacts will be made publicly available with publication. All of the techniques use GD and/or Adam for training as is the case in NeuTraL where we use Adam for pretraining CDAE but use GD for actual training in the ATM-BPR framework. The benchmarks will converge differently during training based on hyperparameters but 1 factor that affects the space and time requirements during each epoch is the size of model parameters. Avoidance of bias forms the basis for model design and other hyperparameter selections throughout our experiments. We use one hidden layer in the deep models. We use 100 factors in the MF models. We also have the number of nodes in the deep learning model amount to 100. We used the tower architecture for the deep learning models. We used learning rates between $0.00001 - 0.01$ and batch sizes of 10000. We tuned model hyperparameters and stopped training early with validation.

6.2. Evaluation metrics

Evaluation is done with 5-fold cross validation. We use 3 popular information retrieval metrics: MRR, NDCG and AUC which are described further in subsequent subsections. We evaluate the techniques on their ability to rank items relative to 9 and 99 other items. The ranking metrics relative to 9 other items are denoted as @10 e.g., MRR@10 measures MRR score for a technique when ranking 1 of 10 items for a user.

Table 1
Datasets

Dataset	#Users	#Items	#Ratings
Movielens 1M	6,040	3,706	1,000,209
Eachmovie	72,916	1,628	2,811,983
Pinterest	55,187	9,916	1,500,809
Goodreads	10,000	5,000	647,458

6.3. Experiments for warm-start ranking

6.3.1. Datasets

We performed experiments on four publicly available datasets. A summary of these datasets is provided in Table 1. The datasets contain explicit ratings for users on items but we convert the ratings to implicit feedback by treating ratings greater than 0 as positive feedback. Our focus in this work is implicit feedback but we believe NeuTraL is applicable to explicit feedback.

- Movielens 1M: Movielens dataset of different datasets [30] are made publicly available by the GroupLens Research lab at the University of Minnesota. We use the Movielens 1M dataset. The data is extracted from the Movielens website which is a free website that provides personalized movie recommendation to users.
- Eachmovies dataset: This dataset [31] is made available by the Digital Equipment Corporation (DEC) Systems Research Center at Compaq. The research center ran a CF service for experimental purposes and made the data available for research.
- Goodreads dataset: This dataset [32] was collected from goodreads.com, a book social network and recommendation website.
- Pinterest Dataset: This is a dataset of implicit feedback representing whether a user pinned an image on their board on the pinterest platform at https://www.pinterest.com.

6.3.2. Benchmarks

We compare our NeuTraL technique with 3 SOTA cold-start personalization systems and a baseline item popularity (IPop) technique. IPop recommends items based on popularity. The benchmarks will converge differently during training based on hyperparameters but 1 factor that affects the space and time requirements during each epoch is the size of model parameters. We select model parameters to avoid bias throughout our experiments. The SOTA benchmarks used are described below:

- BPR: we described BPR in bpr.

Table 2
Movielens results on warm-start items

Metrics	IPop	NCF	BPR	MPR	NeuTraL
MRR@10	0.246	0.409	0.400	0.421	**0.437**
NDCG@10	0.310	0.485	0.480	0.497	**0.515**
MRR	0.270	0.424	0.415	0.435	**0.451**
NDCG	0.417	0.548	0.542	0.557	**0.570**
AUC	0.853	0.921	0.923	**0.924**	**0.929**

Table 3
Pinterest results on warm-start items

Metrics	IPop	NCF	BPR	MPR	NeuTraL
MRR@10	0.111	0.475	0.465	0.487	**0.492**
NDCG@10	0.151	0.566	0.559	0.578	**0.584**
MRR	0.138	0.483	0.475	0.496	**0.501**
NDCG	0.298	0.600	0.595	**0.611**	**0.615**
AUC	0.724	0.947	0.955	**0.958**	**0.960**

Table 4
Books results on warm-start items

Metrics	IPop	NCF	BPR	MPR	NeuTraL
MRR@10	0.087	0.170	0.167	0.239	**0.245**
NDCG@10	0.114	0.224	0.217	0.302	**0.309**
MRR	0.112	0.197	0.193	0.262	**0.268**
NDCG	0.266	0.353	0.348	0.410	**0.415**
AUC	0.590	0.793	0.770	0.829	**0.834**

- Multi-objective pairwise ranking (MPR) [33]: MPR is a MTL technique that combines item ranking and user ranking tasks. MTL learns from historical preference data from item and user ranking perspectives. MTL was demonstrated to able to improve item ranking accuracy by learning from both perspectives.
- Neural Collaborative Filtering (NCF) [5]: NCF is an ensemble recommender that combines MF and deep learning. NCF was demonstrated to achieve superior performance compared to other SOTA techniques.

Table 5
Eachmovies results on warm-start items

Metrics	IPop	NCF	BPR	MPR	NeuTraL
MRR@10	0.123	0.284	0.261	0.275	**0.293**
NDCG@10	0.159	0.357	0.329	0.349	**0.368**
MRR	0.149	0.305	0.284	0.296	**0.313**
NDCG	0.303	0.449	0.430	0.442	**0.456**
AUC	0.646	0.861	0.841	0.857	**0.862**

6.3.3. Results

We record the best average results observed during experiments for each dataset and depict them in movielens-table,eachmovie-table. NeuTraL significantly out-performs the other techniques based on a Wilcoxon signed-rank test with a p-value < 0.01. The winning algorithm per metric is emboldened in each row of all tables. We assume a margin of error of 0.005, hence the winning algorithm has to be greater than the next winner by at least a margin of 0.005. All techniques are emboldened in the case of a tie on a metric. Techniques within the margin of error of the highest score are also emboldened.

6.4. Experiments for cold-start ranking

6.5. Datasets

We performed experiments on 3 of the 4 publicly available datasets used for warm-start experiments in section warm-start-datasets. We used the datasets with item attributes, hence their suitability for our experiments. A summary of these datasets is provided in warm-datasetstable. The 3 datasets used for cold-start personalization experiments are highlighted below:

- Movielens 1M: Item attributes in the dataset include release year and genre. The genre attribute is one-hot encoded into 18 dimensions because we have 18 possible genres. The year is an additional dimension.
- Eachmovies dataset: The items/movies in this dataset are a subset of the items in the Movielens dataset, hence we are able to us the same attribute feature engineering as described for Movielens.
- Goodreads dataset: We use the genres as book attributes for cold-start personalization. The genre attribute is one-hot encoded into 10 dimensions because we have 18 possible genres.

6.5.1. Benchmarks

We compare our NeuTraL technique with 4 state-of-the-art cold-start personalization systems. NeuTraL-C, DropouNet and ATM-BPR require pre-training. The benchmarks used are described below:

- Multi-layer perceptron (MLP): The MLP baseline used here predicts output from interactions between user embedding and item attributes with deep learning. The first hidden layer is the input combination layer that combines user embedding input and item attributes. The combination model is the piece-wise product since this has been demonstrated to outperform concatenation or a dot product [34]. The dot product also doesn't allow us assign different weights to the combined nodes. The output from this combination layer are propagated through extra hidden layers. More hidden layers can be added as needed before the final output.
- ATM-BPR The ATM-BPR technique used a baseline here is described in atm-bpr except the pre-trained user embedding is extracted from BPR instead of an CDAE recommender which is used in NeuTraL-C.
- DropoutNet: Addressing Cold Start in Recommender Systems DropoutNet [22] is a state-of-the-art deep learning based personalization system. DropoutNet is analogous to NeuTraL and ATM-BPR. DropoutNet adopts a different transfer learning procedure compared to NeuTraL. DropoutNet transfers a pre-trained shallow model to a deep model while NeuTraL transfers a pre-trained deep model to a shallow model. We use the MLP model described here as the deep learning model. DropoutNet allows the use of different pre-trained models but we use pre-trained user latent features from CDAE similar to NeuTraL-C i.e. the DropoutNet implementation used here is a combination of the extracted user latent factors from CDAE and MLP. Although DropoutNet is primarily a cold start recommender but it is expected to perform relatively well on warm start recommendations with the appropriate dropout rate. We use a maximum input dropout rate of 1.00 for our experiments with DropoutNet to maximize performance on cold-start because that is the focus of this research work. DropoutNet also allows inference transform but we do not apply it in our experiments because we do not consider the case of incremental item preference data collection as described in their work. We refer to DropoutNet as D-Net to conserve space in the results tables.
- W&D: Wide & Deep Learning for Recommender Systems W&D [19] combines generalization and

Table 6
Movielens results on cold-start items

Metrics	W&D	MLP	ATM-BPR	D-Net	NeuTraL-C
MRR@10	0.043	0.050	0.070	0.053	**0.083**
NDCG@10	0.053	0.059	0.100	0.063	**0.117**
MRR	0.083	0.089	0.097	0.093	**0.109**
NDCG	0.244	0.249	0.257	0.252	**0.269**
AUC	0.604	0.610	0.629	0.617	**0.656**

Table 7
Goodreads results on cold-start items

Metrics	W&D	MLP	ATM-BPR	D-Net	NeuTraL-C
MRR@10	0.030	0.036	0.057	0.054	**0.077**
NDCG@10	0.037	0.045	0.088	0.067	**0.114**
MRR	0.067	0.076	0.083	0.101	**0.107**
NDCG	0.228	0.238	0.245	0.264	**0.271**
AUC	0.570	0.603	0.588	0.672	**0.689**

memorization capabilities of recommender systems for more robust personalization. They used deep learning for its demonstrated superior generalization capability. However, deep learning tends to over-generalize when the input is too sparse and high-rank. On the other hand, generalized linear models are highly capable of memorization of feature interactions through cross product feature transformations. Hence, the combination of a deep learning and a cross product model (wide) in W&D for personalization.

6.5.2. Evaluation metrics for cold-start

We measured how well a recommender system is able to rank a preferred cold-start item relative to other items. The evaluation is similar to the evaluation for warm-start items. The main difference is the absence of test items in the training dataset for cold-start personalized ranking.

6.5.3. Results

We record the best results observed during experiments for each dataset and depict them in movielens-table-cold-start,eachmovies-table-cold-start. NeuTraL-C performs best overall and we subsequently discuss the results further. The winning algorithm per metric is emboldened in each row of all tables. We assume a margin of error of 0.005, hence the winning algorithm has to be greater than the next winner by at least a margin of 0.005. All techniques are emboldened in the case of a tie on a metric. Techniques within the margin of error of the highest score are also emboldened.

6.6. Discussion

We begin our discussion with the results of the warm-start experiments. We stated that NeuTraL performed best overall because of its highest number of wins which corresponds to the number of times a technique has the highest score per dataset. We also validated this observation with a significance test. IPop has the worst performance overall. This is not surprising since it is merely a baseline technique that ranks items based on popularity. The ranking produced by IPop is not personalized as it does not take personal attributes, context or historical preference into account. We expect a decent personalized ranking technique to out-perform IPop. This is the case as least performing personalized ranking technique is BPR but it ourperforms IPop. NCF performs better than BPR. This was already demonstrated by the creators of NCF in their research work [5]. NCF combines both deep learning (MLP) and piecewise product of interactions between user and item embeddings in a generalized

Table 8
Eachmovie results on cold-start items

Metrics	W&D	MLP	ATM-BPR	D-Net	NeuTraL-C
MRR@10	0.031	0.032	**0.052**	0.032	**0.055**
NDCG@10	0.037	0.038	**0.072**	0.038	**0.068**
MRR	0.065	0.065	**0.076**	0.065	**0.075**
NDCG	0.221	0.222	0.232	0.221	**0.237**
AUC	0.490	0.492	0.507	0.481	**0.525**

matrix factorization (GMF). BPR uses a dot product of user and item embeddings to represent the interactions. Dot product assigns equal weights to the LVPs as described in dot-product while the GMF component of NCF learns different weights for the LVPs with a neural network. The MLP component of NCF also learns different weights for user and item embedding combinations. This results in more complex representation of interactions between users and items and better performance. MPR out-performs NCF. The MTL nature of MPR gives it an advantage. NeuTraL's superior performance butrresses the effectiveness of transfer learning since it is essentially MPR combined with transfer learning but it outperforms MPR. We surmise that transfer learning improved the performance of NeuTraL. We also believe that the type of pre-trained model that is transferred is significant. Our experiment here reveals that the extraction mechanism from an autoencoder based model like CDAE is effective.

We subsequently discuss the results of our experiments on cold-start personalization. We stated that NeuTraL-C performed best overall because of its highest number of wins which corresponds to the number of times a technique has the highest score per dataset. We also validated this observation with a significance test. ATM-BPR is the next best performing technique. Both ATM-BPR and NeuTraL-C adopt transfer learning. However, NeuTraL-C uses a different pre-trained model. NeuTraL-C uses a pre-trained model extracted from CDAE as described in section:NeuTraL while ATM-BPR uses pre-trained user embedding from BPR. This shows that it is not enough to just apply transfer learning but the meticulousness of implementation is as important. The type of pre-trained model is pertinent in such design. NeuTraL-C and ATM-BPR also differ in how they learn the "mapping function". NeuTraL-C uses MPR while ATM-BPR uses BPR. DropoutNet performs next best to ATM-BPR. DropoutNet also uses transfer learning. We used user embedding from CDAE in DropoutNet. However, it uses deep learning to learn the interaction between the transferred embedding and item attributes. The complex nature of DropoutNet deteriorated performance somewhat. For instance, the transferred user embedding is propagated through hidden layers before combination with the item attributes. The output of the hidden layers is a tainted version of the user embedding. The mapping learned by DropoutNet is between this tainted version and the item attributes. We believe this is the reason for a poorer performance compared to ATM-BPR and NeuTraL-C. It is not too surprising that MLP performed less than DropoutNet since it is DropoutNet without transfer learning. Once again, this shows the effectivenes of transfer learning. WD performed the least of all cold-start personalization systems. It does not use transfer learning and we believe the complexity of deep learning in WD deteriorated performance due to overfitting.

A common theme throughout or experiments is the benefit of our neural transfer learning approach. We believe that the transferred user embedding is more representative of the users as latent factors compared to the user embedding in the other models. We show a chart of loss minimization in NeuTraL with and without transfer learning in 3 on the Movielens data. 3 shows the speed-up achieved with transfer learning in the form of lower initial loss. 3 also shows the overall lower loss with training. We know that ATM-BPR and DropoutNet adopt transfer learning as well but are outperformed by NeuTraL. As stated earlier in section:cdae, dropout is a vital component of CDAE, hence we investigated the effect of dropout when pre-training on the final results. The results show that dropout slightly enhances the effect of the transferred user embedding in NeuTraL.

7. Conclusion

We presented a novel personalization system based on transfer learning from a state-of-the-art deep personalization system to a linear cold-start personalization model. This system is applicable to warm-start and cold-start items and users. The results of our experiments show the effectiveness of our proposed method and we discussed the results. Although the results are promising, there is

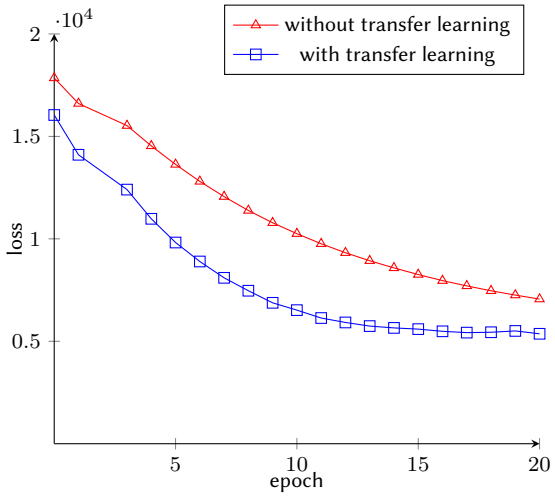

Figure 3: Effect of Transfer Learning with NeuTraL-C on Movielens dataset.

room for future work and improvements. Potential future research work include the extension of our techniques to user cold-start, full cold-start and warm-start ranking. Other potential future work includes investigation of additional attributes and optimum fusion strategy of those attributes. We believe experimentation with more datasets and context attributes such as time and location would also be worthwhile.

References

[1] Y. Hu, Y. Koren, C. Volinsky, Collaborative filtering for implicit feedback datasets, in: 2008 Eighth IEEE International Conference on Data Mining, 2008, pp. 263–272. doi:10.1109/ICDM.2008.22.

[2] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (2009) 30–37. URL: http://dx.doi.org/10.1109/MC.2009.263. doi:10.1109/MC.2009.263.

[3] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, Bpr: Bayesian personalized ranking from implicit feedback, in: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI '09, AUAI Press, Arlington, Virginia, United States, 2009, pp. 452–461. URL: http://dl.acm.org/citation.cfm?id=1795114.1795167.

[4] Y. Koren, Factorization meets the neighborhood: A multifaceted collaborative filtering model, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '08, ACM, New York, NY, USA, 2008, pp. 426–434. URL: http://doi.acm.org/10.1145/1401890.1401944. doi:10.1145/1401890.1401944.

[5] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, T.-S. Chua, Neural collaborative filtering, in: Proceedings of the 26th International Conference on World Wide Web, WWW '17, International World Wide Web Conferences Steering Committee, 2017, pp. 173–182. URL: https://doi.org/10.1145/3038912.3052569. doi:10.1145/3038912.3052569.

[6] Y. Wu, C. DuBois, A. X. Zheng, M. Ester, Collaborative denoising auto-encoders for top-n recommender systems, in: Proceedings of the Ninth ACM International Conference on Web Search and Data Mining, WSDM '16, ACM, New York, NY, USA, 2016, pp. 153–162. URL: http://doi.acm.org/10.1145/2835776.2835837. doi:10.1145/2835776.2835837.

[7] S. Sedhain, A. K. Menon, S. Sanner, L. Xie, Autorec: Autoencoders meet collaborative filtering, in: Proceedings of the 24th International Conference on World Wide Web, WWW '15 Companion, ACM, New York, NY, USA, 2015, pp. 111–112. URL: http://doi.acm.org/10.1145/2740908.2742726. doi:10.1145/2740908.2742726.

[8] Y. Zheng, B. Tang, W. Ding, H. Zhou, A neural autoregressive approach to collaborative filtering, in: Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48, ICML'16, JMLR.org, 2016, pp. 764–773. URL: http://dl.acm.org/citation.cfm?id=3045390.3045472.

[9] M. Bianchi, F. Cesaro, F. Ciceri, M. Dagrada, A. Gasparin, D. Grattarola, I. Inajjar, A. M. Metelli, L. Cella, Content-based approaches for cold-start job recommendations, in: Proceedings of the Recommender Systems Challenge 2017, RecSys Challenge '17, ACM, New York, NY, USA, 2017, pp. 6:1–6:5. URL: http://doi.acm.org/10.1145/3124791.3124793. doi:10.1145/3124791.3124793.

[10] A. I. Schein, A. Popescul, L. H. Ungar, D. M. Pennock, Methods and metrics for cold-start recommendations, in: SIGIR '02, 2002.

[11] A. Arampatzis, G. Kalamatianos, Suggesting points-of-interest via content-based, collaborative, and hybrid fusion methods in mobile devices, ACM Trans. Inf. Syst. 36 (2017) 23:1–23:28. URL: http://doi.acm.org/10.1145/3125620. doi:10.1145/3125620.

[12] Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, L. Schmidt-Thieme, Learning attribute-to-feature mappings for cold-start recommendations, in: 2010 IEEE International Conference on Data Mining, 2010, pp. 176–185. doi:10.1109/ICDM.2010.129.

[13] A. v. d. Oord, S. Dieleman, B. Schrauwen, Deep content-based music recommendation, in: Proceedings of the 26th International Conference on

Neural Information Processing Systems - Volume 2, NIPS'13, Curran Associates Inc., USA, 2013, pp. 2643–2651. URL: http://dl.acm.org/citation.cfm?id=2999792.2999907.

[14] P. Covington, J. Adams, E. Sargin, Deep neural networks for youtube recommendations, in: Proceedings of the 10th ACM Conference on Recommender Systems, New York, NY, USA, 2016.

[15] T. T. Nguyen, H. W. Lauw, Collaborative topic regression with denoising autoencoder for content and community co-representation, in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM '17, ACM, New York, NY, USA, 2017, pp. 2231–2234. URL: http://doi.acm.org/10.1145/3132847.3133128. doi:10.1145/3132847.3133128.

[16] H. Wang, N. Wang, D.-Y. Yeung, Collaborative deep learning for recommender systems, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '15, ACM, New York, NY, USA, 2015, pp. 1235–1244. URL: http://doi.acm.org/10.1145/2783258.2783273. doi:10.1145/2783258.2783273.

[17] G. Sottocornola, F. Stella, M. Zanker, F. Canonaco, Towards a deep learning model for hybrid recommendation, in: Proceedings of the International Conference on Web Intelligence, WI '17, ACM, New York, NY, USA, 2017, pp. 1260–1264. URL: http://doi.acm.org/10.1145/3106426.3110321. doi:10.1145/3106426.3110321.

[18] W. Niu, J. Caverlee, H. Lu, Neural personalized ranking for image recommendation, in: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM '18, Association for Computing Machinery, New York, NY, USA, 2018, p. 423–431. URL: https://doi.org/10.1145/3159652.3159728. doi:10.1145/3159652.3159728.

[19] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, et al., Wide deep learning for recommender systems, in: Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, DLRS 2016, Association for Computing Machinery, New York, NY, USA, 2016, p. 7–10. URL: https://doi.org/10.1145/2988450.2988454. doi:10.1145/2988450.2988454.

[20] Y. Zhu, J. Lin, S. He, B. Wang, Z. Guan, H. Liu, D. Cai, Addressing the item cold-start problem by attribute-driven active learning, IEEE Transactions on Knowledge and Data Engineering 32 (2020) 631–644.

[21] Ming Yan, Jitao Sang, Tao Mei, Changsheng Xu, Friend transfer: Cold-start friend recommendation with cross-platform transfer learning of social knowledge, in: 2013 IEEE International Conference on Multimedia and Expo (ICME), 2013, pp. 1–6.

[22] M. Volkovs, G. Yu, T. Poutanen, Dropoutnet: Addressing cold start in recommender systems, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30, Curran Associates, Inc., 2017, pp. 4957–4966.

[23] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, G. Hullender, Learning to rank using gradient descent, in: Proceedings of the 22Nd International Conference on Machine Learning, ICML '05, ACM, New York, NY, USA, 2005, pp. 89–96. URL: http://doi.acm.org/10.1145/1102351.1102363. doi:10.1145/1102351.1102363.

[24] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization., CoRR abs/1412.6980 (2014). URL: http://dblp.uni-trier.de/db/journals/corr/corr1412.html#KingmaB14.

[25] L. Torrey, J. Shavlik, Transfer learning, 2009.

[26] A. Quattoni, Transfer learning algorithms for image classification, Ph.D. thesis, Citeseer, 2009.

[27] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, S. Gelly, Parameter-efficient transfer learning for nlp, arXiv preprint arXiv:1902.00751 (2019).

[28] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (2014) 1929–1958. URL: http://dl.acm.org/citation.cfm?id=2627435.2670313.

[29] C. M. Bishop, Training with noise is equivalent to tikhonov regularization, Neural computation 7 (1995) 108–116.

[30] F. M. Harper, J. A. Konstan, The movielens datasets: History and context, ACM Trans. Interact. Intell. Syst. 5 (2015) 19:1–19:19. URL: http://doi.acm.org/10.1145/2827872. doi:10.1145/2827872.

[31] P. McJones, Eachmovie Collaborative Filtering Dataset, DEC Systems Research Center,http://www.research.compaq.com/src/eachmovie/, 1997.

[32] M. Wan, J. J. McAuley, Item recommendation on monotonic behavior chains, in: S. Pera, M. D. Ekstrand, X. Amatriain, J. O'Donovan (Eds.), Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018, ACM, 2018, pp. 86–94. URL: https://doi.org/10.1145/3240323.3240369. doi:10.1145/3240323.3240369.

[33] R. Otunba, R. A. Rufai, J. Lin, Mpr: Multi-objective pairwise ranking, in: Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys '17, Association for Computing Machinery, New York, NY, USA, 2017, p. 170–178. URL: https:

//doi.org/10.1145/3109859.3109903. doi:10.1145/3109859.3109903.

[34] R. Otunba, R. A. Rufai, J. Lin, Deep stacked ensemble recommender, in: Proceedings of the 31st International Conference on Scientific and Statistical Database Management, SSDBM '19, Association for Computing Machinery, New York, NY, USA, 2019, p. 197–201. URL: https://doi.org/10.1145/3335783.3335809. doi:10.1145/3335783.3335809.