Reproducibility Companion Paper: Recommendation of Mix-and-Match Clothing by Modeling Indirect Personal Compatibility

Shuiying Liao
The Hong Kong Polytechnic University
Hong Kong
21122587r@connect.polyu.hk

Yujuan Ding
The Hong Kong Polytechnic University
Hong Kong
dingyujuan385@gmail.com

P. Y. Mok*
The Hong Kong Polytechnic University
Laboratory for Artificial Intelligence in Design
Hong Kong
tracy.mok@polyu.edu.hk

Qiushi Huang
University of Surrey
Guildford, UK
hqsiswiliam@gmail.com

Jialun Cao
Hong Kong University of Science and Technology
Hong Kong
jcaoap@cse.ust.hk

ABSTRACT

This reproducibility companion paper accompanies our original study, "Recommendation of Mix-and-Match Clothing by Modeling Indirect Personal Compatibility," providing a detailed framework for replication and verification of our research results. The primary objective of this document is to enhance the transparency and reproducibility of our findings. We present a comprehensive account of the datasets, software tools, and experiments in the original study. This companion paper serves as a valuable resource for researchers and practitioners who aim to validate, learn from, or build upon our work.

CCS CONCEPTS

• **Information systems** → **Multimedia and multimodal retrieval**.

KEYWORDS

Fashion Recommendation, Personalization, Compatibility, Complementary Recommendation, Multi-modal.

ACM Reference Format:
Shuiying Liao, Yujuan Ding, P. Y. Mok, Qiushi Huang, and Jialun Cao. 2024. Reproducibility Companion Paper: Recommendation of Mix-and-Match Clothing by Modeling Indirect Personal Compatibility. In *Proceedings of the 2024 International Conference on Multimedia Retrieval (ICMR '24), June 10–14, 2024, Phuket, Thailand*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3652583.3658371

*P. Y. Mok is the corresponding author (tracy.mok@polyu.edu.hk).

ICMR '24, June 10–14, 2024, Phuket, Thailand

ACM ISBN 979-8-4007-0619-6/24/06.
https://doi.org/10.1145/3652583.3658371

1 ARTIFACTS DESCRIPTION

1.1 Introduction

To enhance mix-and-match clothing recommendations, we introduced a novel ***N****ormalized* ***i****ndirect* ***P****ersonal* ***C****ompatibility* modeling scheme based on ***B****ayesian* ***P****ersonalized* ***R****anking* (NiPC-BPR) as elaborated in the initial study [4]. Our approach uniquely leverages both direct and indirect user-product interaction data, incorporating multimodal information to capture personal user preferences and item compatibility in a comprehensive manner. The provided materials encompass the IQON3000 dataset, the Polyvore-519 dataset, and the source code for the NiPC-BPR model. In this reproducibility companion document, we provide a replication artifact that encompasses a thorough re-implementation and evaluation of the NiPC-BPR framework, to enhance reproducibility in similar future works.

1.2 Source Code Structure

Access to the source code is facilitated through our GitHub repository, which can be found at https://github.com/asaander719/NiPC-BPR. As shown in Fig. 1, we provide a focused overview of the structure of the artifacts. The critical files required to operate the system in the following:

README.md: This file describes an overview of what the project does, its purpose, and how it works.

requirements.txt: The code is built on Pytorch library, and this file contains important dependencies used in our repository.

Models/BPRs: This directory contains the baselines we used in our original paper for comparison experiments, including our proposed NiPC-BPR written in Python language.

dataset: This directory contains the IQON3000 and Polyvore-519 datasets we used in our experiment, including the train/valid/test samples, and pre-trained visual and textual features, which are all accessible at https://drive.google.com/file/d/1Dg7918zUGcL7tzs_OisNzc_FxYlQMG4E/view?usp=sharing.

config: This folder is used to set up parameters and settings that the software and script in the repository require to run correctly.

Table 1: Introduction of Optional Parameters for Customized Training

Parameter	Description	Default Value	
		Polyvore	IQON3000
#epochs	the maximum number of training epochs	80	80
#hidden_dim	the vector dimension of hidden latent representation after projection	512	512
#batch_size	the batch size	1024	1024
#base_lr	the learning rate	0.001	0.001
#wd	the weight decay	0.0001	0.00001
#num_his	the number (N) of user historical preferred given item	2	2
#iPC_w	the weight (η) of proposed iPC branch	2	3
#with_visual	using visual modality	True	True
#with_text	using textual modality	True	True
#with_Nor	using feature scaling	True	True
#f_test	Evaluation under different product interaction frequencies (f)	True	True
#iPC	using proposed iPC branch	True	True

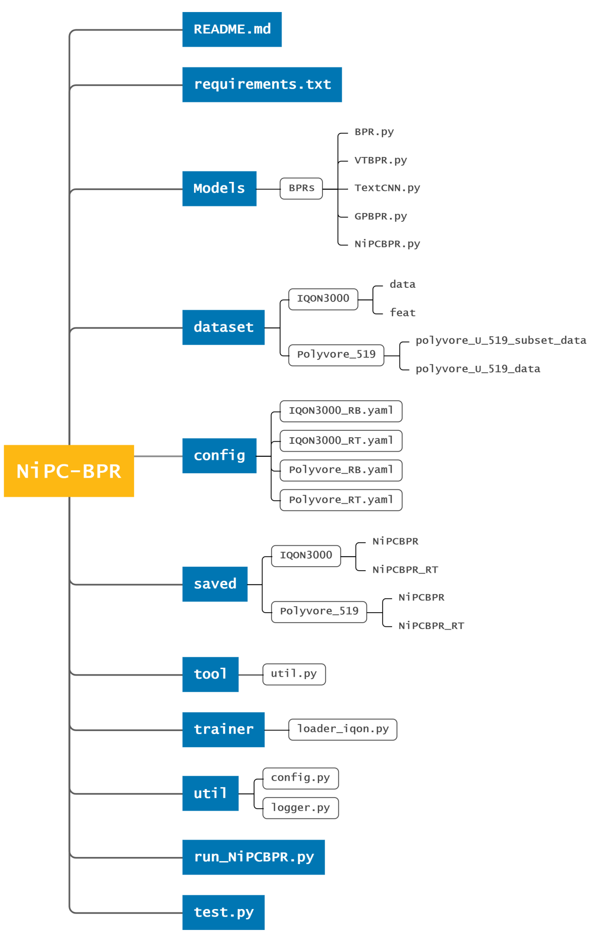

Figure 1: The structure of NiPC-BPR artifact.

The important artificially modifiable parameters and corresponding descriptions can be found in Table 1. The implementation for each recommendation setting is contained in the files as follows:

- **IQON3000_RB.yaml**: Given Top and **R**ecommend **B**ottom for IQON3000 dataset.
- **IQON3000_RT.yaml:** Given Bottom and **R**ecommend **T**op for IQON3000 dataset.
- **Polyvore_RB.yaml:** Given Top and **R**ecommend **B**ottom for Polyvore-519 dataset.
- **Polyvore_RT.yaml:** Given Bottom and **R**ecommend **T**op for Polyvore-519 dataset.

saved: This folder contains the well-trained NiPC-BPR models under the settings of Given Top and Given Bottom in two datasets, which are also available at https://drive.google.com/file/d/1Dg7918zUGcL7tzs_OisNzc_FxYlQMG4E/view?usp=sharing. The results and event log will also be recorded in this folder.

trainer/loader_iqon.py: Loading data for training and validation.

run_NiPCBPR.py: This script contains the steps and preparation needed to train our proposed model.

test.py: This script is used to evaluate the well-trained models saved in the "saved" folder with default format (Given Top and Recommend Bottom in the Polyvore-519 dataset).

2 EXPERIMENT

2.1 Data

In our original paper, the proposed NiPC-BPR model underwent evaluation using two established fashion datasets: IQON3000 [6] and Polyvore-519 [5]. In our mix-and-match clothing recommendation experiment, each training/valid/test sample consists of [user ID, given item ID, positive recommended matching item ID, negative recommended matching item ID]. Each item ID is associated with pre-trained visual and textual features. In the IQON3000 dataset, we use the same sample distribution, visual and textual feature sets as our baseline GP-BPR [6] dose. As for the Polyvore-519 dataset, we generated samples from the user-outfit interaction records, selectively maintaining pairs that featured one top and one bottom clothing item from each outfit for each user, while omitting other types of items.

Pre-trained content features in the IQON3000 dataset provided by [6] include 2048-dimensional (2048-D) visual features extracted via a pre-trained ResNet50 model, along with 400-dimensional (400-D) textual features obtained through TextCNN module in

"TextCNN.py" file from the "Models" folder. Regarding the Polyvore-519 dataset, we utilized the original 2400-dimensional (2400-D) textual features, which were derived using a pre-trained AlexNet [3]. To capture the visual features, we employed a Resnet152 model [1], pre-trained on the ImageNet database, to extract a 2048-D visual feature vector for each fashion product. As mentioned in the first section, all the samples and pre-trained features are available in the "dataset" folder.

2.2 System Environment

The necessary details to recreate the computational environment used for the original experiments are provided below.

- **Hardware Specifications**: Intel(R) Xeon(R) Gold 5220R CPU @ 2.20GHz CPU, and NVIDIA GeForce RTX 3080. 8GB GPU memory is needed for batch size 1024 during the training and testing.
- **Operating System**: Ubuntu 22.04.2 LTS.
- **Programming Language**: Python language with a version larger than 3.10.5.
- **CUDA Toolkit**: Tested with the version 12.0.
- **cuDNN**: Tested with the version 11.1.
- **Dependency Management**: The important Python packages used in our experiment are listed in the "requirements.txt" file.

2.3 Experiment

2.3.1 Experiment Settings: To demonstrate the effectiveness of the proposed NiPC-BPR, we conduct our experiment under two different settings, which are also detailed in the "config" folder: Given Top and Recommend Bottom, and Given Bottom and Recommend Top for each user. To facilitate the training of our NiPC-BPR model utilizing both visual and textual features, reproducers can initiate the process by adjusting the configuration argument within the "run_NiPCBPR.py" script. This is achieved by setting the *config* parameter in the *get_parser* function to the appropriate path that corresponds to one of the four *yaml* configuration files located in the "config" directory.

2.3.2 Evaluation Metric: The area under the ROC curve (AUC) [7] was adopted as the metric to evaluate the performance of our method and other advanced baselines for comparison purposes. The AUC metric quantifies the proportion that the model will correctly identify a compatible garment pairing as opposed to an incompatible one across the entire dataset.

2.3.3 Implementation details: The adaptive moment estimation method (Adam) [2] optimizer was used to train our model, with the maximum training epoch being set to 80, but an early stopping strategy was applied with the patient parameter being set to 5 epochs.

Compatibility Module and *Personalization Module* evaluate inter-product compatibility between the given and matching products, and user-product preference between the user and matching products. As for the *iPC* branch, set the given top and recommend bottom in the Polyvore-519 dataset as an example. Initially, we collate the tops that have previously interacted with each user, excluding the top under current consideration. These are then sorted based on their feature congruence with the given top, balancing the emphasis equally between visual and textual attributes, with a 50 percent weight assigned to each feature type. For modeling the *Indirect Personal Compatibility*, we retain only the top-ranked Top garments as representative of each user's historical preference for a given item, indicated by the parameter *num_his* (N) in Table 1. In cases where a user has no interaction history with a given top, we employ the most frequently chosen tops among users (referred to as popular tops) as fillers. For those historical interactions with less than the specified length, we choose the most similar top to padding. Parameter *iPC_w* (η) is used to customize the contribution of the *iPC* branch, as refers to Table 1. Adjusting the N and η parameters generates the line graph associated with the ablation study presented in the original paper. Changing the *iPC* parameter to *False* allows us to observe the outcomes of our method devoid of the integrated **iPC** branch.

2.3.4 Experiment Customization: A grid search method is applied to determine the optimal hyperparameter configuration for our training process. Specifically, the optional parameters for experiment customization are listed in Table 1, for example, we varied the mini-batch size (*batch_size*) among [64, 128, 256, 512, 1024], the weight decay (*wd*) was explored within [0.001, 0.0001, 0.00001, 0.000001, 0.0000001], the hidden layer dimension (*hidden_dim*) was tested at [256, 512], and we adjusted the learning rate (*base_lr*) across [0.01, 0.001, 0.0001]. During the training process, we implement feature scaling techniques as described by the original paper's "*NMPP*". By setting the "*with_Nor*" parameter to "False," we are able to assess the model's performance in the absence of feature scaling. By toggling the parameters "*with_visual*" for the visual modality and "*with_text*" for the textual modality to "False," one can independently evaluate the contribution of each modality, visual or textual, to the overall performance. Other hyperparameters can also be fine-tuned for each dataset based on empirical evidence, detailed in *yaml* files. The well-trained model would automatically save in the "saved" folder under the corresponding subfolder, for further evaluation.

2.3.5 Test on the well-trained model: To test the result of the well-trained model provided in the "saved" folder, the "test.py" file can be performed with customization of changing the *yaml* file in *get_parser* function. To test the model from customized training, additionally, update the *model_path* parameter in the appropriate *yaml* configuration file located within the "config" directory before proceeding to execute the "test.py" script.

In the experiment titled " Performance comparison on two datasets in terms of AUC, under different product interaction frequency f", execution is enable by setting the "f_test" parameter to "True" in the *yaml* configuration files. In this study, we initially classify the target recommended matching products in the training dataset into four categories based on their interaction frequency. Following this classification, we evaluated the performance of these targeted products in the testing dataset using the AUC metric. The results demonstrate that our model substantially enhances performance, particularly for products with high interaction frequencies.

3 CONCLUSION

In this reproducibility companion paper, we have endeavored to present a thorough and detailed account of the data, and experiment that underpin the findings of our original research. Our reproduction efforts have yielded results that are consistent with those reported in our paper, which reinforces the reliability and validity of the initial findings. This paper will serve as a valuable resource for researchers who wish to build upon our work, as well as a template for those who seek to create their own reproducibility companion papers.

ACKNOWLEDGMENTS

The work described in this paper is supported, in part, by grants from the Innovation and Technology Commission of Hong Kong under grant ITP/028/21TP, the Laboratory for Artificial Intelligence in Design (Project Code: RP1-1) under the InnoHK Research Clusters, Hong Kong Special Administrative Region Government, National Natural Science Foundation of China (Grant No. 62202241), and Jiangsu Province Natural Science Foundation for Young Scholars (Grant No. BK20210586).

REFERENCES

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[2] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2017. Imagenet classification with deep convolutional neural networks. *Commun. ACM* 60, 6 (2017), 84–90.

[4] Shuiying Liao, Yujuan Ding, and PY Mok. 2023. Recommendation of Mix-and-Match Clothing by Modeling Indirect Personal Compatibility. In *Proceedings of the 2023 ACM International Conference on Multimedia Retrieval*. 560–564.

[5] Zhi Lu, Yang Hu, Yunchao Jiang, Yan Chen, and Bing Zeng. 2019. Learning binary code for personalized fashion recommendation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10562–10570.

[6] Xuemeng Song, Xianjing Han, Yunkai Li, Jingyuan Chen, Xin-Shun Xu, and Liqiang Nie. 2019. GP-BPR: Personalized compatibility modeling for clothing matching. In *Proceedings of the 27th ACM International Conference on Multimedia*. 320–328.

[7] Hanwang Zhang, Zheng-Jun Zha, Yang Yang, Shuicheng Yan, Yue Gao, and Tat-Seng Chua. 2013. Attribute-augmented semantic hierarchy: towards bridging semantic gap and intention gap in image retrieval. In *Proceedings of the 21st ACM international conference on Multimedia*. 33–42.