MCRecKit: An Open-Source Library for Multi-Criteria Recommendations

Yong Zheng
Center for Decision Making & Optimization
Illinois Institute of Technology
Chicago, Illinois, 60616, USA
yzheng66@iit.edu

David Xuejun Wang
Morningstar, Inc.
22 W Washington St #7
Chicago, Illinois, 60602, USA
David.wang@morningstar.com

Qin Ruan
School of Computer Science
University College Dublin
Dublin, D04 V1W8, Ireland
qin.ruan@ucdconnect.ie

***Abstract*—Recommender systems (RSs) are designed to help users navigate through large amounts of information by providing personalized suggestions tailored to their preferences. Multi-criteria recommender systems (MCRSs) extend this concept by utilizing users' ratings on multiple aspects of items (i.e., multi-criteria ratings) to predict their overall preferences. Currently, there are several open-source libraries released for RSs. However, none of these existing libraries can handle multi-criteria recommendations due to the special challenges in MCRSs. In this paper, we introduce a Multi-Criteria Recommendation Kit (MCRecKit) which is a Python-based open-source library for multi-criteria recommendations. MCRecKit fills the gap by providing flexible tools and algorithms specifically designed to address the complexity of MCRSs. The library offers a variety of methods for processing, modeling, and evaluating multi-criteria data, enabling researchers and developers to experiment with novel approaches.**

***Index Terms*—multi-criteria, recommender system, collaborative filtering, open-source, recommendation library, MCRecKit**

I. Introduction

A recommender system (RS) is a well-known application that provides personalized suggestions or recommendations to users tailored to user preferences. It has been successfully applied to multiple domains, e.g., video suggestions at Youtube.com, item recommendations at Amazon.com or social suggestions at Facebook, to alleviate the issue of information overloads and assist users' decision-makings.

After decades of the developments, there are several special types of recommender systems (RSs) proposed, such as context-aware RSs [1], [2] which can take contextual information (e.g., time, location) into consideration, multi-objective RSs [3], [4] which can optimize multiple objectives simultaneously, group recommendations which can produce item recommendations for a group of users rather than a single user [5], and so forth. In addition, several open-source recommendation libraries were developed for evaluating and extending recommendation algorithms, such as Surprise [6] for general recommendations, TorchRec [7] for large-scale recommendations, CARSKit [8] and DeepCARSKit [9] for context-aware recommendations, and RecBole [10] which can handle different types of recommendations, etc.

Multi-criteria recommender systems (MCRSs) [11], [12] have also emerged as a special type of RSs, where MCRS can produce more accurate personalized recommendations by taking user preferences in multiple criteria (i.e., different aspects of the items) into account. Take hotel booking at TripAdvisor.com shown by Figure 1 for example, MCRS can better predict user preferences on a hotel by taking users' tastes in room size and cleanliness, hotel location, check-in services, value of the money into considerations. Similar examples can also be found in dining services (e.g., the restaurant reservations at OpenTable.com [13]) and movie ratings (e.g., Yahoo!Movies).

Fig. 1. An Example: Hotel Ratings at TripAdvisor.com

Although there are several existing recommendation libraries, there are no tools built specifically for MCRSs. It can likely be attributed to at least two reasons. First, there are multiple ratings to be loaded into the library for an entry of a user and an item, while traditional RSs usually have only one rating to be loaded for each user-item pair. Moreover, there are a number of MCRS algorithms which require a multi-stage pipeline to run the recommendation algorithms For example, most MCRS algorithms need to predict a user's ratings on an item from the perspective of multiple criteria first, and then aggregate these predicted multi-criteria ratings into a single one as the estimated overall preference.

To overcome the two challenges above, we developed and released the Multi-Criteria Recommendation Kit (MCRecKit), an open-source Python-based library tailored for multi-criteria recommendation systems. This library provides a range of methods for processing, modeling, and evaluating multi-criteria data, facilitating experimentation with innovative approaches for both researchers and developers.

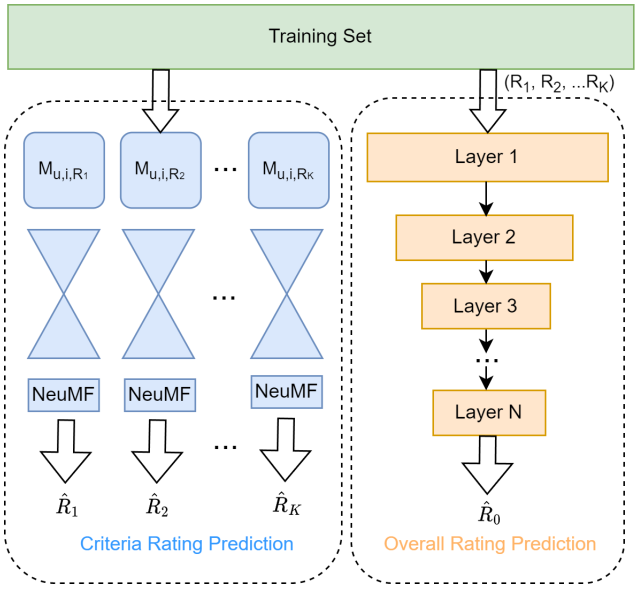

Fig. 2. The IndNeuMF Model [14]

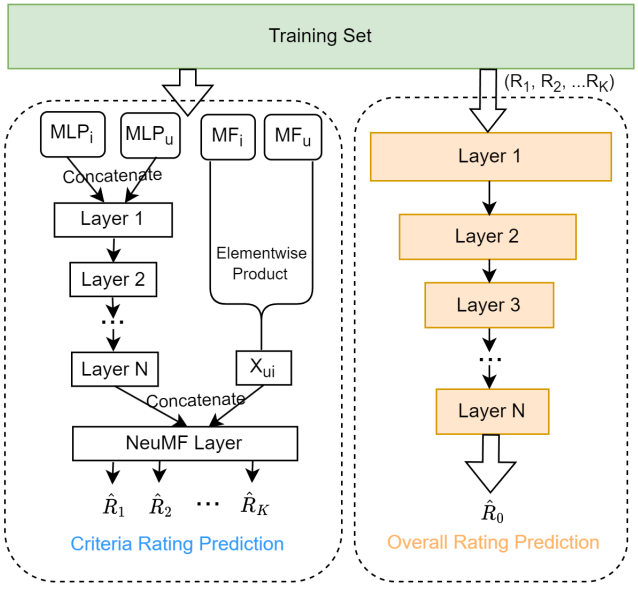

Fig. 3. The MONeuMF Model [15]

II. Related Work

This section reviews the existing research related to our studies, including multi-criteria recommendations, and existing open-source recommendation libraries.

A. Multi-Criteria Recommender Systems

In comparison with traditional RSs, MCRSs can additionally take user preferences in multiple criteria to better predict a user's rating on an item [11], [12]. An example of the multi-criteria rating data is shown by Table I. The *rating* in the table is equivalent to users' overall rating on the items. We also have users' ratings on multiple criteria, such as ratings from the perspective of room cleanliness, check-in and service. The rating-prediction problem in MCRSs, therefore, can be summarized as how to additionally take multi-criteria ratings into account to better predict how a user may like an item. Namely, given a user U_3 and an item T_1 shown in Table I, we would like to predict U_3's overall rating on T_1. Note that we also do not know U_3's multi-criteria ratings on T_1. The predicted overall rating can also be utilized to produce top-N item recommendations.

TABLE I
Example of Rating Data from TripAdvisor

User	Item	Rating	Room	Check-in	Service
U_1	T_1	3	3	4	3
U_2	T_2	4	4	4	5
U_3	T_1	?	?	?	?

1) Multi-Stage Predictions: Existing multi-criteria recommendation models can be summarized into two categories – *multi-stage predictions* and *one-stage predictions* [12]. The approaches by *multi-stage predictions* are more straightforward and also more popular in MCRS research. First of all, predictive models are applied to predict a users' ratings on an item from the perspectives of multiple criteria, e.g., we can predict U_3's multi-criteria ratings on T_1 in Table I in the first step. Afterwards, we can either aggregate these predicted multi-criteria ratings to estimate the user's overall rating on the item by using different *aggregation functions*, or utilize these predicted multi-criteria ratings to produce top-N recommendations directly by adopting *multi-criteria rankings* [14]. It is worth mentioning that each stage in the multi-stage predictions may require an individual process of model training.

There are several aggregation functions for *aggregation-based multi-stage predictions*, including linear regressions [16] and non-linear aggregation, e.g., support vector regressions [17], [18] or predictions through the multilayer perceptron (MLP) [19] methods. The IndNeuMF [14] model shown in Figure 2 and the MONeuMF [15] model shown in Figure 3 are these examples. In these two models, the MLP (models shown on the right in Figure 2 and 3 for the purpose of overall rating prediction) is used to aggregate the predicted criteria ratings to estimate the overall rating. The models on the left in Figures 2 and 3 are the ones utilized to predict multi-criteria ratings. The model in MONeuMF shown in Figure 3 has multiple outputs from a single neural matrix factorization (NeuMF) model, so that the dependency among criteria is considered. By contrast, the model in IndNeuMF just predicts a user's rating on each criterion independently.

Alternatively, the predicted multi-criteria ratings can be utilized to produce top-N recommendations directly through *multi-criteria rankings* [14]. Researchers could utilize theories or methodologies in multi-criteria decision making [12], preference ordering [20] or multi-objective optimization [3] to derive the approaches for the purpose of multi-criteria rankings. Our previous research has examined the effectiveness

of using Pareto ranking [14], relaxed Pareto ranking [21], multi-criteria rankings with subsorting [22] and other different preference ordering approaches [20].

2) One-Stage Predictions: By contrast, there are no fixed mechanisms in *one-stage predictions*. For example, user preferences on multiple criteria can be utilized to find better user neighborhoods which can improve the traditional user-based collaborative filtering algorithms [16]. By this way, it is not necessary to predict multi-criteria ratings for each entry of users and items in the evaluation data sets. Hong et al. proposed to utilize tensor factorization [23], where criteria are considered as context conditions in tensor.

B. Existing Recommendation Libraries

Researchers in RSs have developed numerous open-source libraries to support recommendation tasks. In the early stages, when collaborative filtering algorithms, such as matrix factorization, gained popularity, researchers began creating open-source libraries for benchmark evaluations. Notable examples include the C#-based MyMediaLite [24] and Java-based libraries such as LensKit [25], LibRec [26], and CARSKit [8]. As Python became widely adopted in data science and artificial intelligence, the development of Python-based recommendation libraries increased. For instance, Surprise [6] is a popular Python-based benchmark library, the original LensKit was also updated to a Python version [27], and DeepCARSKit [9], an advanced version of CARSKit, was also developed by using Python. Python-based open-source libraries have since become dominant, especially with the popularity of using neural networks and deep learning in recommendation models, as exemplified by RecBole [10], DeepCARSKit [9] and TorchRec [7].

At the beginning, these open-source libraries primarily focused on the implementation of collaborative filtering-based algorithms, such as those in Surprise [6] and the Java-based LensKit [25]. As the field progressed, researchers began incorporating more advanced, state-of-the-art algorithms into their libraries, exemplified by the development of LibRec [26]. This shift allowed the contributors of the libraries to quickly implement new algorithms as soon as they were introduced. With the growing popularity of deep learning-based recommendation algorithms, the RecBole library [10] has gained particular prominence due to the breadth and diversity of the algorithms it supports.

Regarding the functionalities of these libraries, the majority were initially designed for traditional RSs, exemplified by Surprise, LensKit and LibRec, which facilitate rating prediction and top-N recommendation tasks in standard RSs. However, these libraries lack support for specialized RSs, such as context-aware RSs, group recommendations and cross-domain recommendations, etc. In response, some researchers have created individual libraries tailored for specific applications; for instance, CARSKit [8] and DeepCARSKit [9] were specifically developed for context-aware recommendations. Subsequently, there has been an effort to develop a unified library capable of addressing multiple tasks. An example of this is RecBole [10], which offers support for traditional RSs, social RSs, context-aware RSs, sequential RSs, among others.

Even so, there are no recommendation libraries that can offer support for MCRSs, due to at least two challenges in MCRS – First, multiple ratings need to be loaded into the library, such as the overall rating and the multi-criteria ratings shown in Table I. Furthermore, there are a number of MCRS algorithms which require a multi-stage pipeline to run the recommendation algorithms, e.g., the algorithms need to predict a user's ratings on an item from the perspective of multiple criteria first, and then aggregate these predicted multi-criteria ratings into the estimated overall preference. Most of the recommendation algorithms in other libraries require a single process of training-testing, rather than these multi-stage processing.

III. The MCRecKit Library

Our MCRecKit library[1], depicted by Figure 4, is a specialized toolkit designed for multi-criteria recommendations. We introduce its design, features and implemented algorithms in this section.

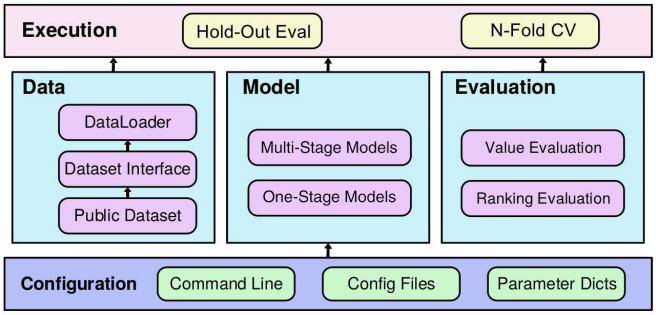

Fig. 4. The MCRecKit Library

A. Development and Design

MCRecKit was built upon the Recbole library v1.0.1 [10] by using Python and Pytorch. As mentioned previously, the RecBole library implements state-of-the-art recommendation algorithms for different types of RSs, but not for MCRSs. We implemented MCRecKit on top of the RecBole library for two reasons – the evacuation pipelines in RecBole were well defined, and we can directly reuse them without significant changes. Furthermore, several algorithms in RecBole can be reused in some components in MCRSs, e.g., for predicting multi-criteria ratings.

Figure 4 depicted the system design of our MCRecKit library. The library supports both hold-out evaluation and N-fold cross validations. The library has easy-to-use data loaders, where a list of multi-criteria rating data sets, such as the ITM-Rec [28], OpenTable [13] and Yahoo!Movie[2] data sets are also provided in the library. The library can run MCRS algorithms

[1]https://github.com/irecsys/MCRecKit

[2]https://github.com/an888ha/multi_criteria_recommender_system/blob/master/data_movies.txt

```
gpu_id: 0
use_gpu: True
show_progress: False
print_loss: False
seed: 2024

# data settings
dataset: itm
field_separator: ","
USER_ID_FIELD: user_id
ITEM_ID_FIELD: item_id
LABEL_FIELD: rating
MULTI_LABEL_FIELD: ['app', 'data', 'ease']
RATING_RANGE: [1, 5]
# rating threshold to determine the relevance of an item
threshold: {'rating': 0}

load_col: ~  # using ~ to load all columns in data
neg_sampling: ~

# model and evaluation setting
model: CombRP

learner: adam
eval_args:
  # split: {'RS':[0.8,0.1,0.1]}
  split: {'CV':5}
  group_by: user
  order: RO
  mode: labeled

# [Metrics for the rating prediction task]
# metrics: ['RMSE', 'MAE', 'AUC']
# valid_metric: RMSE
```

```
# [Metrics for the Top-N recommendation task]
topk: [10, 20, 30]
metrics: ['Recall', 'Precision', 'NDCG']
valid_metric: Recall@10

# specific settings for training models to predict multi-criteria ratings
# using a traditional recommender to predict criteria ratings independently
criteria_model: {'model': 'CriteriaRP',
                 'GENERAL_MODEL': 'NeuMF',
                 'mf_embedding_size': 512,
                 'mlp_embedding_size': 512,
                 'mlp_hidden_size': [128, 64, 32, 16, 8, 4],
                 'learning_rate': 0.001,
                 'dropout_prob': 0.1,
                 'epochs': 50,
                 'train_batch_size': 4000,
                 'mf_train': True,
                 'mlp_train': True,
                 'use_pretrain': False,
                 'parallel': False,
                 'metrics': [ 'RMSE', 'MAE', 'AUC' ],
                 'valid_metric': RMSE,
                }

# specific settings for training models to predict the overall rating
# using predicted criteria ratings to estimate the overall rating via MLP
overall_model: {'model': 'OverallRP',
                'mlp_embedding_size': 0,
                'mlp_hidden_size': [512,256,128,64,32],
                'learning_rate': 0.001,
                'dropout_prob': 0.001,
                'epochs':50,
                'train_batch_size': 6000,
                'metrics': [ 'RMSE', 'MAE', 'AUC' ],
                'valid_metric': RMSE,
                }
```

Fig. 5. An Example of the configuration file for IndNeuMF

for both the rating prediction (i.e., value evaluation in Figure 4) and the top-N recommendation (i.e., ranking evaluation in Figure 4) tasks. Users can easily change the configuration file to tune up the settings and hyper-parameters. The library can be run on both GPU and CPU in command line. The running results and the corresponding configuration file can be automatically saved in the output folder.

B. Features and Algorithms

The features in MCRecKit can be summarized as follows:

- Easy data preparation and loading. Users can easily prepare their data sets by following the instructions in RecBole[3]. The MCRecKit added additional settings to load multi-criteria ratings, such as the multi-label fields shown in the configuration file depicted by Figure 5.
- A unified configure file. Similar to mainstream recommendation libraries, we provide a single and unified configure file, where an example can be shown by Figure 5. Users can customize data options, evaluation strategies (e.g., hold-out and N-fold cross validation), evaluation metrics and hyper-parameters (e.g., embedding size, number of layers, number of neurons on each layer, loss functions, etc.) for different MCRS algorithms.
- A flexible multi-stage pipeline was implemented so that several multi-stage MCRS algorithms can be implemented or easily extended. Take the IndNeuMF shown in Figure 2 and its configuration in Figure 5 for example, we first utilized NeuMF as the recommendation model to predict a user's rating on an item for each criterion independently (e.g., the 'criteria_model' in Figure 5), and then aggregate these predicted multi-criteria ratings into the estimation of the overall rating (e.g., the 'overall_model' in Figure 5). The integrated multi-stage pipeline enables developers to effortlessly adapt the MCRS algorithms into various variants. For example, we can simply change the configuration file to assign another recommendation algorithm, e.g., the auto-encoder model or the factorization machines in RecBole, to replace the NeuMF model in 'criteria_model', in order to produce predicted multi-criteria ratings by another algorithm.
- Autosaved results and log files. The results of each run can be automatically saved in the designated output folder. In addition to the final recommendation outcomes, the running logs and configuration files are also autosaved. This allows users to easily access their execution history through the output folder, ensuring efficient tracking and management of past runs.
- GPU support. The library relies on Pytorch to run deep learning based recommendation algorithms. Users can run the library on both CPU and GPU. For special algorithms, such as the MCRS algorithms based on Pareto rankings [14], [20]–[22], we implemented parallel computing by using CUDA programming, which can accelerate execution and reduce running time significantly.

The implemented MCRS algorithms can be organized by

[3]https://recbole.io/docs/user_guide/usage/running_new_dataset.html

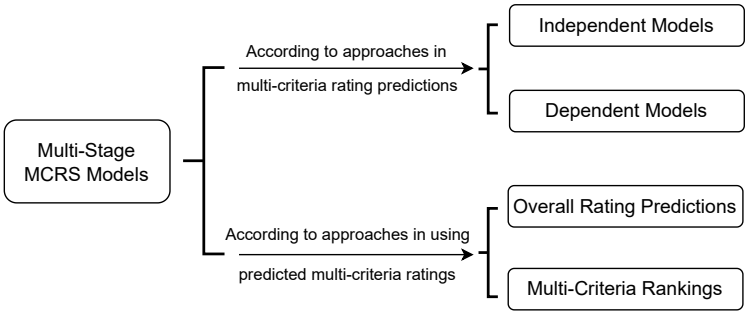

Fig. 6. Multi-Stage MCRS Models

two categories as shown in Figure 4 – *multi-stage models* where multiple training processes were involved, and *one-stage models* where only one training process was included. At the current stage, we implemented several multi-stage models by using IndNeuMF and MONeuMF as the algorithm basis. In addition, we also provide a hybrid strategy to combine multiple models by using a weighted linear aggregation. We will implement one-stage models in our future work.

The multi-stage models implemented in the MCRecKit library can be grouped into different subcategories, as shown by Figure 6. Recall that the IndNeuMF and MONeuMF are two examples of the multi-stage models, as shown in Figure 2 and 3. There are two stages involved – one is the process of predicting multi-criteria ratings, and another one is the process of using these predicted multi-criteria ratings for the purpose of either overall rating predictions or multi-criteria rankings [14].

According to the approaches in predicting multi-criteria ratings, these multi-stage models can be put into the following two subcategories:

- *Independent Models*. Users' rating on an item from the perspective of each criterion can be predicted independently. Figure 2 presents the example of IndNeuMF, where each criterion rating was predicted by using NeuMF independent. Users can also easily update the 'criteria_model' in the configuration file as shown by Figure 5, to utilize other recommendation models in RecBole to predict these criteria ratings.
- *Dependent Models*. By contrast, the dependent models can take dependency among multiple criteria into consideration. One example is the MONeuMF shown in Figure 3, where multi-criteria ratings can be predicted from the same NeuMF layer through a process of joint optimization. Another example is the criteria chain approach [18], where the multi-criteria ratings can be estimated one by one in a sequence by following a pre-defined chain.

Alternatively, these multi-stage models can also be grouped into the following two subcategories, according to the ways of using the predicted multi-criteria ratings:

- *Models for overall rating predictions*. The predicted multi-criteria ratings can be utilized to estimate the overall ratings for the rating prediction task, and the estimated overall ratings can also be further utilized to rank items to produce top-N recommendations. As mentioned previously, both linear regression and non-linear aggregation functions can be adopted in this scenario. As shown by Figure 2 and 3, the original IndNeuMF and MONeuMF utilized the MLP component to estimate the overall ratings from the predicted multi-criteria ratings. These models can be executed by utilizing the 'CombRP' pipeline defined in our MCRecKit library.
- *Models for multi-criteria rankings*. Users can also use the predicted multi-criteria ratings to produce top-N recommendations directly through *multi-criteria rankings* [14]. The MCRecKit library implements 'CriteriaSort' pipeline to run Pareto ranking [14], relaxed Pareto ranking [21], multi-criteria rankings with subsorting [22] or other different preference ordering approaches [20] for the purpose of multi-criteria rankings.

In addition, Our library also implements a hybrid approach, combining multiple models through weighted linear aggregation. Figure 7 illustrates an example in the configuration file. In this setup, we generate a ranking score by aggregating scores from two models: the first model produces a ranking score using Pareto ranking through CriteriaSort, and the second model uses MONeuMF. Notably, MONeuMF provides overall rating estimations that may have a different scale than the ranking scores from CriteriaSort. These predicted ratings and ranking scores can be easily normalized and then aggregated to generate a final ranking score, with linear weights of 0.6 and 0.4 (as shown in Figure 7) applied to the two models, respectively.

IV. Conclusions and Future Work

In this paper, we introduce and release MCRecKit which is an open-source library for multi-criteria recommendations. In our future work, we will implement more MCRS algorithms, including both multi-stage and one-stage MCRS models, in order to enhance its functionalities and the popularity in the area of MCRS.

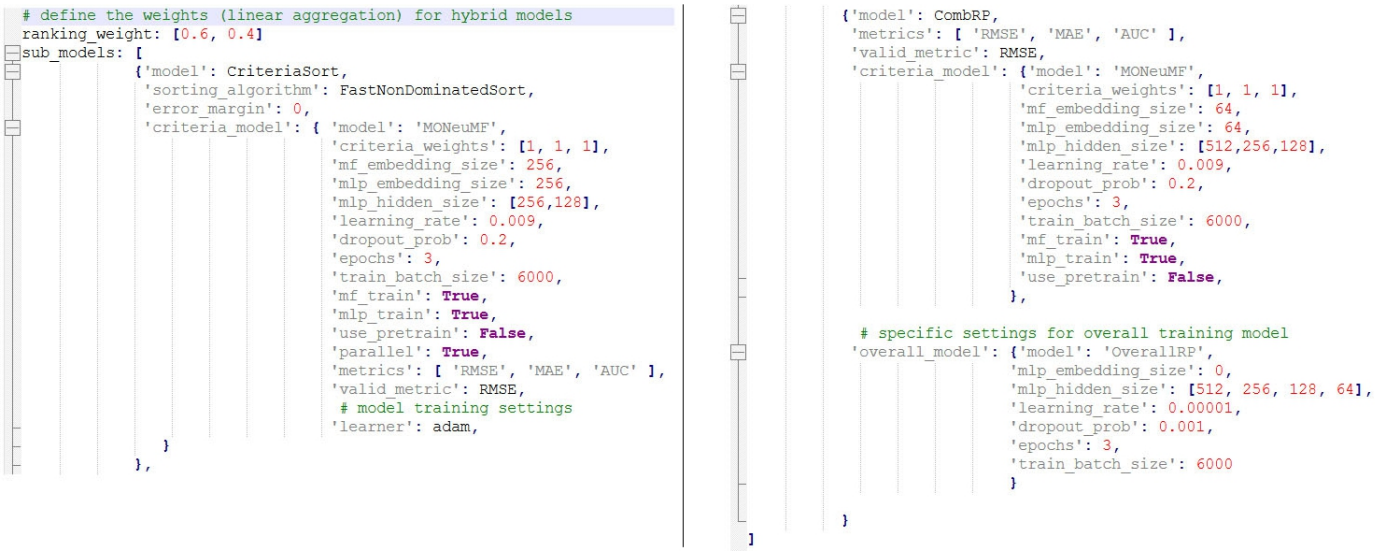

Fig. 7. An Example of Hybrid Models

REFERENCES

[1] Gediminas Adomavicius and Alexander Tuzhilin, "Context-aware recommender systems," in *Recommender systems handbook*, pp. 217–253. Springer, 2010.

[2] Yong Zheng and Bamshad Mobasher, *Context-Aware Recommendations*, pp. 173–202, World Scientific Publishing Co Pte Ltd, 2018.

[3] Yong Zheng and David Xuejun Wang, "A survey of recommender systems with multi-objective optimization," *Neurocomputing*, vol. 474, pp. 141–153, 2022.

[4] Yong Zheng et al., "Multi-objective recommendations: A tutorial," *arXiv preprint arXiv:2108.06367*, 2021.

[5] Michal Kompan and Maria Bielikova, "Group recommendations: Survey and perspectives," *Computing and Informatics*, vol. 33, no. 2, pp. 446–476, 2014.

[6] Nicolas Hug, "Surprise: A python library for recommender systems," *Journal of Open Source Software*, vol. 5, no. 52, pp. 2174, 2020.

[7] Dmytro Ivchenko, Dennis Van Der Staay, Colin Taylor, Xing Liu, Will Feng, Rahul Kindi, Anirudh Sudarshan, and Shahin Sefati, "Torchrec: a pytorch domain library for recommendation systems," in *Proceedings of the 16th ACM Conference on Recommender Systems*, 2022, pp. 482–483.

[8] Yong Zheng, Bamshad Mobasher, and Robin Burke, "CARSKit: A java-based context-aware recommendation engine," in *Proceedings of the IEEE International Conference on Data Mining Workshops*. 2015, IEEE.

[9] Yong Zheng, "Deepcarskit: A deep learning based context-aware recommendation library," *Software Impacts*, vol. 13, pp. 100292, 2022.

[10] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Yushuo Chen, Xingyu Pan, Kaiyuan Li, Yujie Lu, Hui Wang, Changxin Tian, et al., "Recbole: Towards a unified, comprehensive and efficient framework for recommendation algorithms," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021, pp. 4653–4664.

[11] Gediminas Adomavicius, Nikos Manouselis, and YoungOk Kwon, "Multi-criteria recommender systems," in *Recommender systems handbook*, pp. 769–803. Springer, 2010.

[12] Yong Zheng and David Wang, "Multi-criteria decision making and recommender systems," in *Companion Proceedings of the 28th International Conference on Intelligent User Interfaces*, 2023, pp. 181–184.

[13] Yong Zheng, "Opentable data with multi-criteria ratings," in IEEE Dataport, 2024.

[14] Yong Zheng and David Wang, "Multi-criteria ranking: Next generation of multi-criteria recommendation framework," *IEEE Access*, vol. 10, pp. 90715–90725, 2022.

[15] Nour Nassar, Assef Jafar, and Yasser Rahhal, "Multi-criteria collaborative filtering recommender by fusing deep neural network and matrix factorization," *Journal of Big Data*, vol. 7, no. 1, pp. 1–12, 2020.

[16] Gediminas Adomavicius and YoungOk Kwon, "New recommendation techniques for multicriteria rating systems," *IEEE Intelligent Systems*, vol. 22, no. 3, pp. 48–55, 2007.

[17] Jun Fan and Linli Xu, "A robust multi-criteria recommendation approach with preference-based similarity and support vector machine," in *Advances in Neural Networks–ISNN 2013: 10th International Symposium on Neural Networks, Dalian, China, July 4-6, 2013, Proceedings, Part II 10*. Springer, 2013, pp. 385–394.

[18] Yong Zheng, "Criteria chains: a novel multi-criteria recommendation approach," in *Proceedings of the 22nd International Conference on Intelligent User Interfaces*, 2017, pp. 29–33.

[19] Nour Nassar, Assef Jafar, and Yasser Rahhal, "A novel deep multi-criteria collaborative filtering model for recommendation system," *Knowledge-Based Systems*, vol. 187, pp. 104811, 2020.

[20] Yong Zheng and David Xuejun Wang, "A comparative study of preference ordering methods for multi-criteria ranking," in *2023 10th IEEE Swiss Conference on Data Science (SDS)*. IEEE, 2023, pp. 108–111.

[21] Yong Zheng and David Xuejun Wang, "Multi-criteria ranking by using relaxed pareto ranking methods," in *Adjunct Proceedings of the 31st ACM Conference on User Modeling, Adaptation and Personalization*, 2023, pp. 81–85.

[22] Yong Zheng and David Xuejun Wang, "Hybrid multi-criteria preference ranking by subsorting," *arXiv preprint arXiv:2306.11233*, 2023.

[23] Minsung Hong and Jason J Jung, "Multi-criteria tensor model for tourism recommender systems," *Expert Systems with Applications*, vol. 170, pp. 114537, 2021.

[24] Zeno Gantner, Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme, "Mymedialite: A free recommender system library," in *Proceedings of the fifth ACM conference on Recommender systems*, 2011, pp. 305–308.

[25] Michael D Ekstrand, Michael Ludwig, Jack Kolb, and John T Riedl, "Lenskit: a modular recommender framework," in *Proceedings of the fifth ACM conference on Recommender systems*, 2011, pp. 349–350.

[26] Guibing Guo, Jie Zhang, Zhu Sun, and Neil Yorke-Smith, "Librec: A java library for recommender systems," in *Posters, Demos, Late-breaking Results and Workshop Proceedings of the 23rd International Conference on User Modeling, Adaptation and Personalization*, 2015.

[27] Michael D Ekstrand, "Lenskit for python: Next-generation software for recommender systems experiments," in *Proceedings of the 29th ACM international conference on information & knowledge management*, 2020, pp. 2999–3006.

[28] Yong Zheng, "ITM-Rec: An open data set for educational recommender systems," in *Companion Proceedings of the 13th International Conference on Learning Analytics & Knowledge (LAK)*, 2023.