Expressive latent feature modelling for explainable matrix factorisation-based recommender systems

Expressive Latent Feature Modelling for Explainable Matrix Factorisation based Recommender Systems

ABDULLAH ALHEJAILI, Department of Computer Science, Loughborough University, UK and Faculty of Computing and Information Technology, King Abdulaziz University, Saudi Arabia
SHAHEEN FATIMA, Department of Computer Science, Loughborough University, UK

The traditional matrix factorisation (MF) based recommender system methods, despite their success in making the recommendation, lack explainable recommendations as the produced latent features are meaningless and cannot explain the recommendation. This paper introduces an MF-based explainable recommender system framework that utilises the user-item rating data and the available item information to model meaningful user and item latent features. These features are exploited to enhance the rating prediction accuracy and the recommendation explainability. Our proposed feature-based explainable recommender system framework utilises these meaningful user and item latent features to explain the recommendation without relying on private or outer data. The recommendations are explained to the user using text message and bar chart. Our proposed model has been evaluated in terms of the rating prediction accuracy and the reasonableness of the explanation using six real-world benchmark datasets for movies, books, video games and fashion recommendation systems. The results show that the proposed model can produce accurate explainable recommendations.

CCS Concepts: • **Information systems** → **Recommender systems**; **Collaborative filtering**.

Additional Key Words and Phrases: matrix factorisation, explainable recommendation, feature extraction

ACM Reference Format:
Abdullah Alhejaili and Shaheen Fatima. 2022. Expressive Latent Feature Modelling for Explainable Matrix Factorisation based Recommender Systems. *ACM Trans. Interact. Intell. Syst.* 1, 1 (March 2022), 31 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The primary purpose of recommender systems is to provide valuable recommendations to active users. This goal can be achieved by applying traditional recommender system approaches, including collaborative filtering (CF) and content-based (CB). In CF [31], the recommendation is made based on the analysis of the user-item rating history in which user-to-user similarities are derived from their rating history so they can collaborate to recommend unrated items to each other. On the other hand, content-based filtering [18] methods rely on making recommendations to the user based on their rated items. In CB, item-to-item similarities are extracted based on the items' contents, and the user is recommended with the items that are similar to what they liked. These recommender system approaches can be combined into hybrid approaches [4].

Authors' addresses: Abdullah Alhejaili, A.E.S.Alhejaili@lboro.ac.uk, Department of Computer Science, Loughborough University, Epinal Way, Loughborough, Leicestershire, UK, LE11 3TU, AAlhejaili@kau.edu.sa, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia, 21589; Shaheen Fatima, S.S.Fatima@lboro.ac.uk, Department of Computer Science, Loughborough University, Epinal Way, Loughborough, Leicestershire, UK, LE11 3TU.

2160-6455/2022/3-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

The collaborative filtering approach has received much attention in the field of recommender systems, as it can provide a reasonable accuracy with less critical information analysis compared to the content-based approach. However, it is limited to the available user-item interaction information; the more user-item interaction cases there are, the more accurate recommendation can be provided [29]. Therefore, it would be helpful to combine some useful information, such as item side information with the user-item interaction history data. This step would be more crucial when considering explainable recommender systems.

An explainable recommendation is an approach where the user is provided with justifications for the recommendations they get. Thus, this approach gives more attention not only to provide an accurate recommendation to the user but also to justify why the recommendation is made. This would increase the user's satisfaction with the recommender system [10] and help in encouraging the user [32]. However, more information is needed to be involved to help in providing reasonable explanations. For example, matrix factorisation (MF), which is one of the most successful methods used for collaborative filtering [14], is not able to provide explainable recommendation without incorporating more information, such as users' personalities, items' descriptions, users' reviews on items and items' images. Typically, MF relies on using only user-item interaction history to map users and items with latent factors, in which they are used to predict the rating of any item by any user. These data are not enough for providing explanation. Thus, incorporating some additional data will be helpful for providing recommendation explanation.

Due to the spread of big data around us, massive data sources for building RSs have become widely available. However, RSs conceptually deal with three forms of data: data about items (i.e., item side information), data about users (i.e., user side information), and data about transactions (i.e., user-item interaction data) [26]. These data are retrieved from different sources and may be dealt with in different statuses. For example, accessing user side information data, such as age, gender and occupation, requires user permission; thus, they are considered to be private data. At the same time, these data need to be specified by the users themselves. On the other hand, item side information and user-item interaction data are not private. However, the item side information data are less likely to be available in the same RS domain dataset than user-item interaction data. To clarify these differences, in this paper, we define the following vocabularies for describing the data in RS:

- *Local data:* The data that exist in the same domain of the RS that holds the user-item rating data.
- *Outer data:* The data from other sources, i.e., the data that do not exist in the same RS domain.
- *Private data:* The data that are related to users and require their permission to be accessed.

Most of the existing ER models rely on incorporating additional *outer data* beyond what exist in the local dataset, and sometimes *private data*. These data, which include user reviews on items [5, 12] and user social networks data [24, 35], are to produce reasonable recommendation explanation. Despite that this useful information can enhance the ER model, accessing such information may require special access to private information and/or external domains. Therefore, considering user privacy and cross-domains access restrictions, there are unavoidable challenges in collecting and analysing such information [16, 31].

Taking into consideration the above-mentioned challenges, it is worth noting that there is a need for providing a reasonable proposal to an explainable recommender system that overcomes these limitations. Therefore, there is a need to investigate the possibility of modelling an MF-based CF method that is able to provide accurate and explainable recommendation utilising only *local data* that are available in the same domain without incorporating any *outer data* or *private data* (e.g., user personal data). The raw data of RS is usually comprised of user-item rating data, with a possibility of

providing some item side information in some datasets. We consider these data, if they are available, to be *local data* as they exist in the same domain. To this end, we propose an MF-based explainable recommender system model that utilises only the available *local data* for providing accurate and explainable recommendations. Different to the existing ER methods, our proposed method is able to provide accurate and explainable recommendations without relying on outer or private data. To the best of our knowledge, the existing ER methods fails to provide explanations when the only available data are user-item rating data. In contrast, our proposed method can handle datasets with only user-item rating data and is able to provide accurate and explainable recommendations even in the case of the absence of item side information. We introduce WSLER: a latent feature modelling method that provides meaningful latent features for users and items, which enhances the accuracy of the MF model and produce reasonable explanations of the recommendations. In this research, we extend our previous work [2] to incorporate the users and items features for explainable recommendations.

In previous work [2], we introduced WAFE: a Weighted Average based Feature Extraction method that extracts useful users and items features, and SVD-LM (SVD with Local Mean) that incorporates the local means of the users' and items' rating in the prediction model. We also introduced WSL, which utilises WAFE for latent features modelling and SVD-LM for solving the rating prediction problem. In this article, we propose an MF-based explainable recommendation model that extends WSL to utilise user and item features for explaining the recommendation. The main contributions of this paper are to:

- propose a feature-based explainable matrix factorisation based recommender system model that provides accurate and reasonable recommendation explanation,
- introduce an explanation representation method that adopts both text and bar chart to represent the recommendation explanation to the user,
- evaluate our proposed method using six real-world benchmark datasets, and
- compare the performance of our proposed methods with five related works using the relevant dataset.

The remaining of this paper is organised as follow. Section 2, presents the literature review and Section 3 presents the SVD foundation. Our proposed WSLER framework is presented in Section 4 followed by its three components that are explained in Section 5, Section 6 and Section 7. The evaluation of our method is explained in Section 8 and we discuss the results in Section 9. The work is concluded in Section 10.

2 LITERATURE REVIEW

The recognition of the importance of recommendation explanation started in the late 90s. For instance, in 1999, Schafer et al. [28] pointed out that the use of recommender systems should include an explanation of what the user has been recommended, which would increase the user's satisfaction. In addition, in the work of Herlocker et al. (2000) [10], the authors emphasised that the chance of users to accept the recommendation produced by collaborative filtering would be increased with the explanation of the recommendation. In their study, they concluded that the most convincing component is to explain to the user why they have been recommended with the suggested recommendation. However, they only examine the effectiveness of the recommendation explanation in terms of the promotion aspect. Similarly, in 2002, Sinha and Swearingen [30] suggested that one of the main aspects of increasing the recommender system reliability to the user is to explain why the recommendation is performed. Besides, Bilgic and Mooney (2005) [3] investigated

the effectiveness of the recommendation explanation in the users' opinion about the recommended item. Both [30] and [3] considered the users satisfaction aspect to evaluate the recommendation explanation. However, the formal introduction to the term explainable recommendation has been introduced recently [1, 12, 34].

The two main approaches of recommender systems, collaborative filtering (CF) and content-based (CB) both have received remarkable attention by researchers in the early recommender systems methods [25]. While CB approaches, which mainly rely on the item's content information, can straightforwardly produce explanation to justify the recommendation to the user, it would be more time-consuming to collect such information especially if they do not exist in the same domain (*outer data*) [33]. On the other hand, models that adopt the approach of collaborative filtering (CF) [6] seem to overcome this limitation, as they rely on analysing users and items interaction history without involving *outer data*. However, the ability to justify the recommendation to the user seems to be more difficult in CF-based methods compared to CB-based methods [33].

CF-based recommendation methods have shown remarkable success, especially when integrating with latent factor models [13], which increase the success of collaborative filtering methods even more. Latent factor models [27] have been widely adopted in the recommender system application and research. For example, methods such as NMF [17], SVD, SVD++ [13] and PMF [21] have received a great attention in the literature. Nevertheless, latent factors produced by these models are meaningless and would not be useful to explain why a recommendation is performed. The researchers of recommender systems, as a result, have become more interested in Explainable Recommendation Systems (ERS). In ERS methods, the main goal is not only to provide an accurate recommendation to the users but also to explain why an item is recommended to the user. For instance, in the work of [34], to provide an explainable recommendation, they propose an Explicit Factor Model (EFM) that joins the latent dimensions with explicit features.

Vig et al. [32] adopted movie tags as features to generate recommendations and explanations. To explain the recommended movie, the system displays the movie features and tells the user why each feature is relevant to her. One limitation of this method is that as the features extracted based on tags which are produced by users themselves, they may not always describe the item correctly (tag quality). On the other hand, in our model, we extract items' features from the exact domain knowledge, which typically describe them.

Zhao et al. [36] represented products and users in the same demographic feature space, and used the weights of the features learned by a ranking function to explain the results. The limitation of this method is that the user's features are extracted from their profiles at social networks. Item's features are extracted from users reviews and profiles in the social networks. (privacy and cross-domain issues). In our method, we adopt only information in the same domain without adopting any user's personal information. [35] further explored demographic information in the social media environment for product recommendation with feature-based explanations.

Abdollahi and Nasraoui [1] adopted the k-nearest neighbour approach for producing an explainability matrix representing the weight (score) of each rating to be explainable. Entities in the explainability matrix are involved in the rating prediction task using the MF concept. Since the explainability matrix is calculated based on the concept of k-NN, its produced values are limited to the available ratings in the nearest users/items neighbours. In our method, we calculate the explainability score based on the user's rating history without relying on their neighbours.

Hou et al. [12] used radar charts to explain why an item is recommended and why others are not. In their model, they extract aspects from user's text reviews, adopt them as features, and then, link all items and users to these aspects. However, features are extracted from text reviews, which have less availability than ratings, and requires more computational cost for text mining. In our

method, we extract features only form user-item rating data and provide meaningful factors that can be used for explaining the recommendation.

In summary, the existing proposed explainable recommended system approaches mainly adopt information such as users' text reviews, users demographic information and items tags. However, this information's adoption suffers from three limitations: i) user privacy, ii) cross-domain access difficulty, and iii) high analysis cost. In our method, we adopt only data that are not private, exist in the same domain, and easy to collect and analyse.

3 SINGULAR VALUE DECOMPOSITION (SVD) FOUNDATION

Singular Value Decomposition (SVD) is one of the most successful MF methods used in the literature for recommender systems. In MF recommender system models, where the user-item rating data are stored in the rating matrix $R \in \mathbb{R}^{n \times m}$, where n and m represent the number of users and items respectively, joint latent factors of size d are defined, so all users and items are assigned to d latent features. Assuming that U is a set of n users and I is a set of m items, users' latent features are stored in $P \in \mathbb{R}^{n \times d}$ and all items' latent features are stored in $Q \in \mathbb{R}^{m \times d}$. This is practically performed by factorising the high-rank rating matrix R into two lower rank matrices P and Q, in which $R \approx \hat{R} = PQ^T$, where $\hat{R} \in \mathbb{R}^{n \times m}$ is the estimated values of R.

Traditionally, the SVD method relies on defining the dimension of latent factors d and initialise the entries of P and Q using random values. These initial values are updated thought a learning process until achieving their optimal values that can estimate the entries of $\hat{R}$ using Equation 1.

$$\hat{r}_{ui} = \mu + b_u + b_i + p_u^T q_i \tag{1}$$

Where $\hat{r}_{ui}$ is the estimated rating given by user u to the item i, p_u is the vector of user u latent features, q_i is the vector of item i latent features, μ is the overall global average rating, b_u and b_i are the biases of user u and item i respectively. This biased SVD model was proposed by [13].

Typically, SVD method aim to minimise the rating prediction error by solving an optimisation problem in Equation 2:

$$\min_{p^*, q^*, b*} \sum_{ui \in \kappa} \left(r_{iu} - \mu - b_u - b_i - p_u^T \right)^2 + \lambda \left(\|p_u\|^2 + \|q_i\|^2 + b_u^2 + b_i^2 \right) \tag{2}$$

where, κ is the set of the (u, i) pairs for all known r_{ui} and λ is a constant that controls the extent of the regularisation.

Despite that SVD has received much attention in the literature showing a reasonable success, less attention has been given to the starting point of defining the latent factors and initialising user and item feature values. The traditional method randomly defines the number of latent factors d, and initialises the values of user features matrix P and item features matrix Q with random values. However, this results in three limitations:

(1) The random definition of d requires more effort to search its optimal value, as we need to try different values until achieving the best accuracy of the model.
(2) Initialising P and Q with random values may lead to get the search stuck in a local minima.
(3) These latent factors are meaningless, so they cannot be used for explaining the resulting recommendation.

Thus, our proposed WSLER framework resolves these limitations by extending the SVD model to exploit raw data to improve the performance of the SVD model in terms of prediction accuracy, and recommendation explainability.

4 WSLER FRAMEWORK

We introduce a new framework called WSLER, an SVD-based recommendation framework that extends the SVD to provide more accurate and explainable recommendations. It improves the mechanism of defining the latent factors, initialising the user and item features values, and improving the rating prediction model. It also introduces explanation criteria for explaining the recommendation.

Our proposed WSLER framework is comprised of three components, as follows:

(1) **WAFE component:** WAFE is a feature engineering method that utilises raw data to produce meaningful user and item features. The engineered features are used to define the latent factors and items and users feature values (Figure 1 (a)). The inputs of this component are the raw data, and it produces item and user features as outputs. This component is explained in Section 5.
(2) **WSL component:** WSL extends the SVD method to meaningfully define latent factors and initialise the values of user and item features, and improve the rating prediction model (Figure 1 (b)). This component takes the item and user features produced by WAFE as inputs and produces the estimated rating matrix as outputs. We will explain this component in Section 6.
(3) **ER component:** ER is an explainable recommendation model that utilises WSL for producing explainable recommendation (Figure 1(c)). This component takes the outputs of the WAFE component (i.e., the item and user features) and the output of the WSL component (i.e., the estimated rating matrix) as inputs. These inputs are utilised to process the explainable recommendations as outputs. We will explain this component in Section 7.

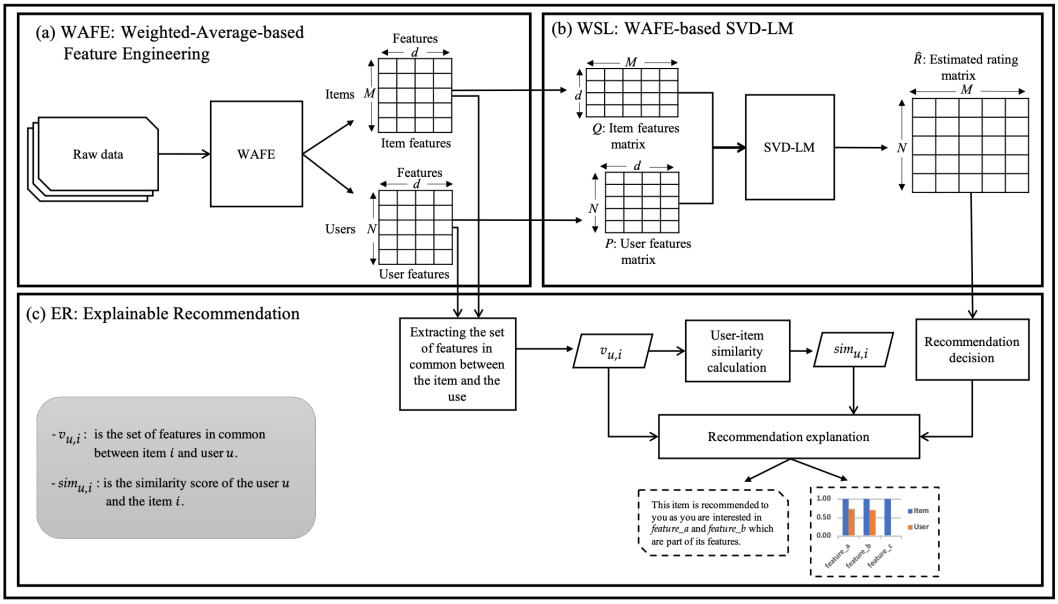

Fig. 1. A general framework of our WSL-based explainable recommendation (WSLER) model.

These three components are explained in the following text.

5 THE WEIGHTED AVERAGE-BASED FEATURE ENGINEERING (WAFE) COMPONENT

In this method, we extract information about users that can reflect their interest in items (user's preferences) by analysing their behaviours in their rating history. We aim to link each user directly with the features of items that they have already rated in order to be able to predict how they will behave with new items that have similar features values. The mechanism of WAFE is based on two stages:

(1) Item feature engineering: manually extract an item features using information available in the raw data. We assume that this step can be done differently, in which all the resulted features values are encoded into categorical values.
(2) User feature Extraction: extracting user features from each item feature. These extracted features will represent the user interest of each item feature (user preferences).

5.1 Item Features Engineering

In some datasets, the items are provided with some side information, such as item name, year of release, and type or category. These data, if they are available will enrich the item features. However, in some domains, this kind of information is not provided. Nevertheless, in our model, we introduce some useful item information that are extracted from the user-item rating data, which are available in any recommender system dataset. In the following text, we are describing how these two kinds of item features are modelled.

Features from side information: We extract this kind of item features straightforwardly from the raw data representing some feature concepts. Each feature concept is represented using some feature values. For example, in a movie recommender system, the year of release is one feature concept. This feature concept has a defined range of values, i.e., the range of movie release years. The values of this range are grouped into periods (e.g., each group represents a period of 10 years). Recall that a movie can be assigned to only one year; this feature concept can be dealt with as a categorical variable, in which it is encoded using representation methods such as one-hot-encoding [15]. In this kind of encoding, if there are g categories for a feature concept, g features are used to represent this feature concept, in which one feature is assigned to 1 representing that the item is corresponding to that feature and the rest will be zeros.

Another feature concept of a movie recommender system is the genre of the movie. Assuming that there are three possible genres that a movie can be categorised with, three values can be used to represent the feature concept of the movie's genre, i.e. $genre_1, genre_2$ and $genre_3$. Though, different from the feature concept "year of release", the movie can be categorised with more than one genre. Hence, this is a special case of the categorical variable with a possibility of a movie to be corresponding to more than one genre. In order to reflect the degree to which a particular movie belongs to a certain genre, we introduced weights, which were spread evenly across all relevant genres. For example, if $movie_1$ corresponds to $genre_1$ and $genre_3$, the values of its genre features $genre_1, genre_2$ and $genre_3$ will be 0.5, 0, 0.5 respectively. Meanwhile, if $movie_2$ corresponds to only $genre_1$, the values of its genre features $genre_1, genre_2$ and $genre_3$ will be 1, 0, 0 respectively.

This information can be extracted directly from the item's description if they are available. However, in some domains, where the side information is not provided, some other information can be derived from the user-item rating data, such as the item number of ratings and the item ratings average. The former can represent the item's popularity, while the latter can represent the item's reputation. This kind of information is represented as continuance values. However, they can be transformed into ordinal variables, e.g., converting them into levels (1 to 10) in which 1 refers to the low level and 10 refers to the high level. Hence, this kind of information can be easily represented using

one-hot-encoding, as explained above, resulting in representing each feature using 10 columns (i.e., $popularity1, popularity2, ..., popularity10$ and $reputation1, reputation2, ..., reputation10$). More information about feature engineering can be found in [15] and [37].

Following this methodology for all available item's information, we will end up with a set of item features V of size K. For each feature $v_j \in V$, the value v_j^i, which is the value of feature v_j for the item i is represented in a value between 0 and 1. This value represents the extent to which the item i belongs to the feature v_j, i.e., 1 means that the item i fully corresponds to the feature v_j and 0 means that the item is not corresponding to the feature.

5.2 User Features Extraction

The extracted features (as described in Section 5.1) can be considered to be the latent (joint) factors. Thus, we define $d = K$ latent factors by extracting all possible item features following the same methodology. Each item feature $v_j \in V$ is considered to be a latent factor, and the set of item features V are the set of latent factors. As the items have already been linked with these factors (by their features values), similarly, user features need to be extracted so they can be linked to the defined factors. The following text is to explain how we extract the user features.

For each defined latent factor $v_j \in V$, we calculate the weighted average of the user u's ratings for all items that belong to the factor v_j (i.e. items with $v_j^i > 0$). Thus, the value of the user latent feature v_j^u can be calculated using Equation 3:

$$v_j^u = \left(\frac{1}{|R_u|} \sum_{r \in R_u^{v_j}} r \right) \cdot w_u \tag{3}$$

where, R_u is the set of all ratings assigned by user u, $R_u^{v_j}$ is the set of ratings assigned by user u to all items with $v_j > 0$ and w_u is the weight of the users rating, which can be calculated using equation 4 as follow:

$$w_u = \frac{|R_u|}{|R|} \tag{4}$$

The resulting user feature values will represent the extent to which a user is interested in each feature. As the user's feature value is calculated based on their weighted average ratings, these values will be in the rating range depending on the domain. Therefore, all users will be linked to each feature with their values, in which a user feature of high value means that the user is interested in the feature and vice versa.

Applying this methodology for all users and items in the R, we can end up with a reasonably defined number of latent factors d along with meaningful values of item features and user features.

6 THE WAFE-BASED SVD-WITH LOCAL MEAN (WSL) COMPONENT

We propose WSL, a method that extends the SVD method in two aspects as follows:

(1) **WAFE-SVD:** Instead of the random setting of the number of latent factors and the random initialisation of the user and item feature values (as it is adopted in SVD), WAFE-SVD uses WAFE for defining meaningful latent factors and assign user and item features with reasonable initial values. This is explained in more detail in Section 6.1.
(2) **SVD-LM:** This extension improves the prediction model of the SVD by introducing two new predictors. This introduction incorporates the local mean of user and item ratings for formalising the bias in addition to the global (overall) rating mean. This is explained in more detail in Section 6.2.

WSL combines these two SVD extension methods to leverage their advantages and enhance the prediction accuracy. The combination mechanism is explained in Section 6.3.

6.1 WAFE-based SVD (WAFE-SVD)

WAFE-SVD extends the SVD model to utilise WAFE to define meaningful latent factors and initialise user and item feature values reasonably. In contrast to the SVD model, instead of randomly setting the number of latent factors d, in WAFE-SVD, d is assigned to the number of the item/user features produced by WAFE. Besides, WAFE-SVD assigns the initial values of user feature matrix P and item features matrix Q with the values of user and item features that extracted by WAFE instead of initialising them randomly. This step improves the rating prediction accuracy and decreases the required number of iterations to achieve the optimal values of P and Q.

Users and items latent factors are defined to represent the joint intersection between items and users. For example, in a movie recommendation dataset, suppose that one latent factor represents action feature. A user and an item are linked with this factor by a feature value (i.e., a real number) representing the extent to which this movie belongs to action movies and how much the user likes action movies. Based on the same concept, WAFE is utilised for modelling latent (joint) features for items and users.

As explained in Section 5, WAFE produces the same number of features for both items and users, which is K. The values of the user and item features represent the extent to which each user and item is linked to each feature. These features are considered to be the latent factors, in which the number of the extracted features K is assigned to be the number of latent factors d and the initial values of P and Q are assigned with the user and item feature values. As a result, the latent factors here are assigned with meaningful information that describes the factor focus feature (e.g., action). Applying this methodology, we will end up with meaningfully defined latent factors and reasonably initialised values of item features Q, and user features P. This initialisation reduces the initial prediction error, which will enhance the performance of the model.

6.2 SVD with Local Mean (SVD-LM)

The prediction model in Equation 1 can achieve a reasonable prediction accuracy; however, it only adopts the overall global rating mean and ignores the local rating mean for each user and item. The user rating means can reflect the user rating behaviour, while the mean of the ratings received by each item can reflect the item reputation. Thus, we incorporate these two values in our prediction model as it is shown in Equation 5:

$$\hat{r}_{ui} = \frac{1}{3}\Big(\Big(\mu + b_u + b_i + p_u^T q_i\Big) + \mu_u + \mu_i \Big) \tag{5}$$

where μ_u and μ_i are the local rating meat of user u and item i respectively. The factor $(\frac{1}{3})$ is applied to average the resulting value of adding $\big(\mu + b_u + b_i + p_u^T q_i\big)$ and μ_u and μ_i. The estimated value produced by the SVD prediction model (i.e., $\mu + b_u + b_i + p_u^T q_i$) is supposed to be a value in the rating range (i.e., a value between the maximum and the minimum rating). However, an out of range value may sometimes be generated, especially in the early iterations. On the other hand, the values of μ_u and μ_i are always in the rating range. Thus, we adopt μ_u and μ_i as a kind of bounding the estimated rating value to be in the rating range by averaging these three values (the estimated value produced by the SVD prediction model, μ_u and μ_i) by dividing them by the constant 3. As a result, if the estimated value produced by the SVD prediction model is higher or lower than the rating range, the adoption of μ_u and μ_i will decrease or increase this value to be in the rating range.

These newly introduced terms (i.e., μ_u and μ_i) differ from the two biases terms (i.e., b_u and b_i) as they refer to the actual rating average values of the user and the item, while b_u and b_i are initialised randomly and updated during the learning process.

The SVD-LM model gets the initial values of P and Q as inputs and applies the standard SVD methodology to learn the optimal values of P and Q. However, the main difference here is that SVD-LM incorporates the local mean of the users' and items' ratings to enhance the rating prediction. The user ratings local mean μ_u and the item ratings local mean μ_i can be extracted directly from the user-item rating matrix R.

6.3 The WSL Algorithm

WSL algorithm combines WAFE-SVD and SVD-LM, which extend the SVD model differently, to leverage their advantages and enhance the prediction accuracy. The main advantage of WAFE-SVD is to define meaningful latent factors and initialise users and items features with meaningful values representing the extent to which items and users are involved in each factor. Besides, SVD-LM is advantaged of improving the prediction model by introducing two new predictors (i.e., the local mean of user ratings and the local mean of item ratings). Thus, these two extension methods (i.e., WAFE-SVD and SVD-LM) are combined reasonably in WSL to exploit their advantages. Algorithm 1 describes how WSL fit the two components WAFE-SVD and SVD-LM together.

WSL starts with utilising WAFE to extract item and user features and assign P and Q to user and item features values. On the other hand, users' biases (Bu) and items' biases (Bi) are initialised with random values. Then, the local mean of users (Mu) and items (Mi) ratings are calculated along with the global (overall) rating average (mu). Then, using all known ratings in R, the estimated ratings are calculated using Equation 5 to get the initial RMSE. Then, the estimation step is repeated until we get the optimal RMSE taking into consideration the predefined number of iteration ($epochs$). Then, P, Q, Bu and Bi are assigned their optimal values at the iteration with the lowest RMSE. Finally, all entities in $\hat{R}$ are estimated using Equation 5 taking into consideration the optimal values of P, Q, Bu and Bi.

In the next section, we describe our proposed explainable recommendation model.

7 THE EXPLAINABLE RECOMMENDATION (ER) COMPONENT

In this model, the explanation of each user's provided recommendations is generated based on the relationship between the user and the recommended item. More precisely, ER relies on analysing the correlation between the user and each feature of the recommended item's features. Then, a recommendation is justified by extracting the extent to which the user is interested in the item's features. In other words, ER explains the recommendation based on the intersection of the item's features and the user interesting features. This will convince the user of the recommended item. Recall from Section 6, WSL produces meaningful users and items features that are used to define the latent factors and represent user and item features values P and Q. These meaningful factors and features values are helpful for making the recommendation results interpretable. This motivates us to use these meaningful features to explain the recommendation.

The produced feature values by WSL represent the extent to which the item and the user are involved in each latent factor. For example, in a movie recommender system, in which one of the defined latent factors is *action*, the user u and the item i are assigned with a value that represents the extent to which u is interested in *action* movies and the extent to which i is belonging to be an *action* movie. Suppose that i is recommended to u. If the value of u's *action* feature and the value of i's *action* feature are high (i.e., closer to the maximum feature value than the minimum feature value), this recommendation is justified as follows: "*This movie has been recommended to you because*

Algorithm 1 WSL

Input: Rating matrix R, item features, learning rate γ, regularisation constant λ and number of iteration *epochs*.
Output: The estimated rating matrix $\hat{R}$

Extract item and user features using WAFE
$d \leftarrow K$
$P \leftarrow$ user features
$Q \leftarrow$ item features
initialise Bu, Bi
$\mu \leftarrow$ overall rating average
$Mu \leftarrow$ users' rating averages vector
$Mi \leftarrow$ items' rating averages vector
for all known r_{ui} **do**
  calculate $\hat{r}_{ui}$ using Equation 5
end for
calculate RMSE, using Equation 7
$RMSE^{min} \leftarrow$ RMSE
for *epochs* **do**
  for all known r_{ui} **do**
    calculate $\hat{r}_{ui}$ using Equation 5
    $e_{ui} = r_{ui} - \hat{r}_{ui}$
    $b_u \leftarrow b_u + \gamma \cdot (e_{ui} - \lambda \cdot b_u)$
    $b_i \leftarrow b_i + \gamma \cdot (e_{ui} - \lambda \cdot b_i)$
    $p_u \leftarrow p_u + \gamma \cdot (e_{ui} \cdot q_i - \lambda \cdot p_u)$
    $q_i \leftarrow q_i + \gamma \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i)$
  end for
  calculate RMSE, using Equation 7
  $RMSE^{current} \leftarrow$ RMSE
  if $RMSE^{current} < RMSE^{min}$ **then**
    $RMSE^{min} = RMSE^{current}$
    $Q^{opt}, P^{opt}, Bu^{opt}, Bi^{opt} \leftarrow P, Q, Bu, Bi$
  end if
end for
calculate $\hat{R}$ for $Q^{opt}, P^{opt}, Bu^{opt}, Bi^{opt}$

we found that you are interested in action movies, which it is one of this movie's features". In addition, ER presents the recommendation justification using a bar chart to show the intersection between the user and the item features. The following text is to explain in detail how this explanation is performed.

First, we introduce the following thresholds:

- ts^{rec}: the recommendation threshold for assigning whether an item is recommended to a user or not.
- ts^{int}: user's interest threshold, for assigning whether a user is interested in a feature or not.
- ts^{sim}: the similarity threshold for assigning whether a user and an item are similar or not.

Then, we calculate the similarity $sim_{u,i}$ between u and i. To do so, the following notation is defined:

- v_i^{pos}: the set of features that the item i is assigned with a positive value. We refer to this as item positive features.
- v_u^{int}: the set of features that user u is assigned with a value $\geq ts^{int}$. We refer to this as user interesting features.
- $v_{u,i}$: the intersection features between item i positive features and user u interesting features all features (i.e. $v_{u,i} = v_i^{pos} \cap v_u^{int}$).

The similarity $sim_{u,i}$ between user u and item i is calculated using Equation 6.

$$sim_{u,i} = \frac{1}{2}\left(\frac{|v_{u,i}|}{|v_i^{pos}|} + \frac{\hat{r}_{u,i}}{r^{max}}\right) \tag{6}$$

We applied the factor $\frac{1}{2}$ to average the resulting value of adding $\frac{|v_{u,i}|}{|v_i^{pos}|}$ and $\frac{\hat{r}_{u,i}}{r^{max}}$. The resulting value of the expression $\frac{|v_{u,i}|}{|v_i^{pos}|}$ is a value in the range (0,1) as $|v_{u,i}| \leq |v_i^{pos}|$. Similarly, the expression $\frac{\hat{r}_{u,i}}{r^{max}}$ will result in a value in the rang (0,1), as $\hat{r}_{u,i} \leq r^{max}$. Therefore, since we are aiming to calculate the degree of how the user and the item are similar, we applied the factor $\frac{1}{2}$ to average these two values for representing the similarity score. Considering the introduced similarity threshold ts^{sim}, user u and item i are considered to be similar if their similarity score $sim_{u,i} \geq ts^{sim}$. This reflects how strong the relationship between u and i is.

Therefore, for each recommendation case in $\hat{R}$ that is assigned with $\hat{r}_{ui} \geq ts^{rec}$, this recommendation is justified based on the values of $sim_{u,i}$ and $v_{u,i}$. If the user u and the item i are similar (i.e., $sim_{u,i} \geq ts^{sim}$), then we examine whether $v_{u,i} > 0$?, which reflects whether there are intersection features between u and i or not. If so, the recommendation is assigned to be explainable and it is explained by representing the intersection features between u and i. As a result, an explanation statement is generated showing that the user u is interested in some or all positive features of the item i depending on the result of the condition $|v_i^{pos}| = |v_{u,i}|$? If $|v_i^{pos}| = |v_{u,i}|$, this means that the user u is interested in all the item i's features, otherwise u is interested in some of i's features.

In addition, a bar chart is generated for the user representing the intersection between the user interesting features with the positive item features besides the explanation statement. The graph presents all item positive feature values and the values of the user interesting features of these positive item features. This will show how many of the item positive features are of the user interest.

8 EVALUATION

The evaluation of recommender systems can be performed using either an online or an offline experiment. In the online evaluation approach [32, 34], the participants are provided with a list of recommendations and/or explanations; then, they give some feedback on what they have been recommended with (e.g., by answering some questions or taking a reaction on the recommendation). Then, their feedback is analysed to evaluate the model. On the other hand, the offline evaluation approach [1, 12] is done on the basis of rating predictions. Online evaluation requires more resources than offline evaluation, e.g., accessing existing e-commerce websites and interacting with some of its active users. It also may require a long time for achieving a convincing result depending on the activity levels of those websites. In contrast, in the offline evaluation approach, the available data is split into training and testing sets. The evaluation is performed based on the results of running the trained model to predict the results of the test data. In this study, we use the offline method on six benchmark real-world datasets. The evaluation criteria are described in the following text.

8.1 Dataset

Six real-world datasets have been used as follows: three datasets form GroupLens[1] [7, 38] namely, MovieLens-1M (ML1M), MovieLens-10M (ML10M) and Book-Crossing (BC), and three datasets from Amazon [2] [8, 20] namely, Amazon Video Games(AVG) Amazon Instant Video (AIV) and Amazon Fashion (AF). MovieLens-1M and MovieLens-10M are for movie recommendations, in which the ratings are given in a 5-star rating scale (with half-star increments for ML10M). Demographic user information is also provided in Ml1M, such as age, gender, occupation, postcode, in addition to movies' side information, such as title, genre and year of release. In ML10M, side information about movies is provided; however, no user information is given apart from user id. Book-Crossing is a dataset for book recommendations, in which the ratings are given on a 10-star scale rating. Book-Crossing provides some item side information such as year of release. Amazon datasets, i.e., AVG, AIV and AF, provide user ratings and reviews on products. The rating is given on a 5-star scale rating, and no side information is provided in Amazon datasets. As the datasets ML1M, ML10M and BC provides some item side information, these are local data as they are available in the same domain. Thus, we utilised these data for modelling item features along with some useful features that are extracted from the user-item rating data. However, as the datasets AVG, AIV and AF do not provide any side information, we only adopt the features that are extracted from user-item rating data for modelling item features. The statistics of all datasets are shown in Table 1.

Table 1. Statistic of datasets.

Dataset	#Users	#Items	#Ratings	Density
ML1M	6,040	3,900	1,000,209	4.2461%
ML10M	71,567	10,681	10,000,054	1.3082%
BC	278,858	271,379	355,250	1.5488%
AVG	24,303	10,672	231,780	0.0894%
AIV	426,922	23,965	583,933	0.0057%
AF	186,189	749,233	883,636	0.0006%

8.2 Baselines

We compare our method with a number of related methods in terms of rating prediction accuracy as follows:

- **SVD** [22]: Singular Value Decomposition is a standard matrix factorisation method that adopts only user-item rating data.
- **SVD++** [13]: as an extension of SVD, SVD++ incorporate implicit feedback of users and items.
- **PMF** [21]: Probabilistic Matrix Factorization is a matrix factorisation method that improves the performance of the traditional matrix factorisation using only user-item rating data.
- **NMF** [17]: Non-negative Matrix Factorisation is a matrix factorisation method that factorises the rating matrix into two non-negative matrices.

[1]https://grouplens.org/datasets/
[2]https://jmcauley.ucsd.edu/data/amazon/

- **HFT** [19]: Hidden Factors as Topics is a method that models reviews and ratings jointly. It uses an exponential transformation function to perform a combination between latent factors vectors and topic description.
- **NeuMF** [9]: Neural Matrix factorisation method that utilises multiple perception neural network architecture into matrix factorisation model to learn latent factors.
- **PrefCRBM** [23]: Preference Relation-based Conditional Restricted Boltzmann Machine is a CF method that integrates the preference relationship and side information of items to produce recommendations using RBM.
- **EFM** [34]: Explicit Factor Model is a method that adopts phrase-level sentiment analysis to model latent factors by integrating features of explicit and implicit feedback.
- **AMF** [12]: Aspect-based Matrix Factorisation is a method that analyses user's text reviews to derive aspects as features and link users and items through these aspects.

8.3 Evaluation Metrics

Two groups of evaluation metrics have been used in this study to evaluate our model in terms of i) rating prediction and ii) explanation performance.

8.3.1 Rating Prediction. For rating prediction evaluation, we adopted root mean squared error (RMSE) and mean absolute error (MAE), which represent the prediction error of user ratings on unrated items. RMSE and MAE are calculated using Equation 7 and Equation 8.

$$RMSE = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(\hat{r}_t - r_t)^2} \tag{7}$$

$$MAE = \frac{1}{T}\sum_{t=1}^{T}|\hat{r}_t - r_t| \tag{8}$$

where T is the number of examples in the test set, $\hat{r}_t$ is the predicted rating of the t^{th} test example, and r_t is the actual rating of the t^{th} test example.

8.3.2 Top-n Recommendation. A further evaluation method for recommender systems is to perform a top-n recommendation. In this evaluation method, the system produce a list of recommendations for each user and select the top n of them to be presented to the user. The evaluation of this method is performed using some relevant metrics. Mean Average Precision (MAP) is a widely adopted metric in this manner [23]. In our evaluation experiment, we adopted MAP@n, which is calculated using Equation 9 as follows:

$$MAP@n = \frac{1}{|T^{user}|}\sum_{u \in T^{user}}\sum_{c=1}^{n}\frac{pre_u@c}{min(l_u, c)} \tag{9}$$

where, $|T^{user}|$ is the number of users in the test data, l_u is the number of relevant items to the user u in the test data, and $pre_u@c$ is the precision of the user u at the position c, which is calculated using Equation 10 as follows:

$$pre_u@c = \frac{tp@c}{c} \tag{10}$$

where $tp@c$ is the number of true positive cases at position c, i.e., the number of relevant items appear in the top-c recommendation list. We conducted the experiment of this evaluation metric and compared the results to those reported by [23].

8.3.3 Explanation Performance. For explanation performance analysis, the evaluation is performed from two aspects, i.e., i) overall satisfaction and ii) user case studies, as follows:

(a) **Overall Satisfaction**: for evaluating the overall user satisfaction, we adopt some classification evaluation metrics including accuracy, error rate, precision, recall and F-measure to evaluate the performance of our explanation model as follows. In the test data, each test case is assigned whether to be a recommended case or not, based on the predicted rating considering the recommendation threshold (ts^{rec}). The decision to be recommended is based on the predicted rating, which would be already defined in the previous step (rating prediction). Hence, we could convert each case in the test data to be either 1 for recommended case or 0 for unrecommended case. Besides, in our explanation method, the user and the item for each test case are assigned to be either similar or not, as explained in Section 7. This assignment is performed using our similarity calculation method, which can be used to examine whether the recommendation can be reasonably explained or not, i.e., 1 for similar user and item case and 0 otherwise. Hence, we can assign each case in the test data to one of the following:
- true positive case (tp): a recommended case (1) with similar user and item (1).
- false positive case (fp): not recommended case (1) with similar user and item (0).
- false negative case (fn): a recommended case (0) with non-similar user and item (1).
- true negative case (tn): not recommended case (0) with non-similar user and item (0).

The confusion matrix of the explainable recommendation cases conducted by our method are illustrated in Table 2.

Table 2. Confusion matrix of our defined true positive, true negative, false positive and false negative cases.

Factor	Recommended (1)	Not recommended (0)
Similar user and item (1)	tp	fp
Non similar user and item (0)	fn	tn

Thus, metrics such as, accuracy, error rate, precision, recall and F-measure can be used to evaluate the method. The evaluation metrics, accuracy (acc), error rate (err) precision (pre), recall (rec), F-measure (fme) are illustrated in Table 3.

We also define another evaluation metric called Positive Satisfaction Ratio (PSR) [12] as an evaluation metric for explanation evaluation. For each user u in the test data, we randomly select two items i_1, i_2 in which one has high predicted rating i.e., greater than or equal to the recommendation threshold ($\hat{r}_{u,i_1} \leq ts^{rec}$) and the other has low predicted rating i.e., less than the recommendation threshold ($\hat{r}_{u,i_2} < ts^{rec}$). Then, we calculate the similarity score for each case using equation 6. Then, we examine whether or not the case with high rating prediction was assigned a higher similarity score than the case with low rating prediction. If so, then the case is assigned to be a positive satisfaction case. Applying this for all users in the test data, we then, calculate the ratio of positive satisfaction cases over all test cases using the formula in Equation 11:

$$PSR = \frac{\#positive\_cases}{\#total\_cases} \tag{11}$$

Table 3. Evaluation metrics for binary classification, as concluded by [11].

Metric	Formula	Evaluation Focus
Accuracy (acc)	$acc = \frac{tp+tn}{tp+fp+tn+fn}$	The ratio of correct predictions over the total number of evaluation examples.
Error rate (err)	$err = \frac{fp+fn}{tp+fp+tn+fn}$	The ratio of incorrect predictions over the total number of evaluation examples.
Precision (pre)	$pre = \frac{tp}{tp+fp}$	The ratio of the correctly predicted positive cases over the total positive cases.
Recall (rec)	$rec = \frac{tp}{tp+tn}$	The ratio of the correctly predicted positive cases over the positive predicted cases.
F-measure (fme)	$fme = \frac{2 \cdot pre \cdot rec}{pre+rec}$	The harmonic mean value between the values of precision and recall.

where $\#positive\_cases$ and $\#total\_cases$ are the number of positive satisfaction cases and the number of users in the test data respectively.

(b) **User Case Studies**: Similar to [12], we introduce user case studies to evaluate another aspect of the explanation performance for each dataset as follows. For each dataset, we randomly select a user u from the test data and study their relationship with the two items (i.e., i_1 and i_2) that have been selected in the previous step (i.e., PSR calculation step). We aim to investigate whether the relationship between the user u and the item i_1 (the item with high predicted rating) is stronger than the relationship between the user u and the item i_2 (the item with low predicted rating). To do so, we introduce a user-item intersection score ($intSec_{u,i}$) which is calculated using Equation 12:

$$intSec_{u,i} = \frac{|v_{u,i}|}{|v_i^{pos}|} \tag{12}$$

where an intersection score of 1 means that the user is interested in all of the item's features, and 0 means that the user is not interested in any of the item's features. We calculate the intersection score for u and i_1 ($intSec_{u,i_1}$), and for u and i_2 ($intSec_{u,i_2}$). Then, we compare the values of $intSec_{u,i_1}$ and $intSec_{u,i_2}$ to examine whether the u is interested in i_1's features more than i_2's features. If so, then, we conclude that our recommendation for this case study is reasonably explained. We use bar chart for showing the features in common between each user and item. We also present a summary table of the intersection scores for each user and item in each case study. This will clearly illustrate how the user is related to each item in each case study.

8.4 Parameters Setting

The parameters of this experiment include d -the number of latent factors, λ – the regularisation extent, γ – the step size (learning rate) and the number of epochs (iterations).

In an MF-based model, a well-known technique for defining the best value of d is to examine the method's performance when verifying the value of d. However, in WSL, since the users and items features will be already defined, the number of latent factors d is assigned the number of the extracted features. The number of the extracted features for all datasets are presented in Table 4. The evaluation was done for different values of the remaining parameters as follow. $\lambda = [0.01, 0.02, ..., 0.15]$ and $\gamma = [0.001, 0.002, ..., 0.01]$, and assigned them to their optimal values as illustrated in Table 4. The number of epochs is assigned to 100 with early stopping; we picked the best RMSE and MAE when achieved in an earlier iteration.

Table 4. Threshold values and parameter settings for all datasets.

Dataset	ts^{rec}	ts^{int}	ts^{sim}	γ	λ	#Features
ML1M	3	3	0.5	0.006	0.04	46
ML10M	3	3	0.5	0.006	0.04	47
BC	6	6	0.5	0.004	0.13	30
AVG	3	3	0.5	0.005	0.05	19
AIV	3	3	0.5	0.005	0.05	18
AF	3	3	0.5	0.005	0.05	19

9 RESULTS AND ANALYSIS

We evaluate our proposed model in two aspects, i.e., 1) the rating prediction performance, and 2) the explanation performance. For the former, we evaluate the performance of the recommendation by comparing the accuracy of the rating prediction of our method with the baselines. For the latter, similar to [12], we evaluate our explanation performance by calculating the ratio of positive satisfaction for all users in the test data. Then, we compare the performance of our explanation model with the work of [12]. More details are given in Section 9.3.

9.1 Rating Prediction Performance

The evaluation of the rating prediction has been performed in this study to examine the effectiveness of the components of out proposed method. The evaluation of performance of rating prediction is done in two aspects: 1) The effusiveness of WSL on the SVD method, and 2) Comparison with baseline methods, using RMSE and MAE as evaluation metrics.

9.1.1 The effusiveness of WSL on the SVD method. WSL extend the SVD method in two main aspects: 1) adopting WAFE for defining latent factors and initialising user and item feature values, and 2) introducing two new predictors to be incorporated in the prediction model (user rating local mean μ_u and item rating local mean μ_i), we run an ablation study to show the effectiveness these two extensions by splitting WSL into the following sub-methods:

(1) **SU**: This sub-method is extend the SVD by adding only user rating local mean μ_u to the prediction model. In SU, the prediction model is defined as in Equation 13:

$$\hat{r}_{ui} = \frac{1}{2}\Big(\Big(\mu + b_u + b_i + p_u^T q_i\Big) + \mu_u \Big) \tag{13}$$

(2) **SI**: This sub-method is extend the SVD by adding only item rating local mean μ_i to the prediction model. In SI, the prediction model is defined as in Equation 14:

$$\hat{r}_{ui} = \frac{1}{2}\Big(\Big(\mu + b_u + b_i + p_u^T q_i\Big) + \mu_i\Big) \tag{14}$$

(3) **SVD-LM**: This sub-method is extend the SVD by adding both user rating local mean μ_u and item rating local mean μ_i to the prediction model. In SVD-LM, the prediction model is defined as in Equation 5.

(4) **WAFE-SVD**: This sub-method is extends the SVD by utilising WAFE for defining latent factors and initialising user and item feature values. WAFE-SVD utilises the prediction model defined in Equation 1.

(5) **WSU**: This sub-method is extends the SVD by incorporating only user rating local mean and utilising WAFE for defining latent factors and initialising user and item feature values. WSU utilises the prediction model defined in Equation 13.

(6) **WSI**: This sub-method is extends the SVD by incorporating only item rating local mean and utilising WAFE for defining latent factors and initialising user and item feature values. WSI utilises the prediction model defined in Equation 14.

These sub-methods have been implemented in addition to our main method WSL, which combines SVD-LM and WAFE-SVD. WSL extends the SVD by incorporating both user rating local mean and item rating local mean, and utilising WAFE for defining latent factors and initialising user and item feature values. WSL utilises the prediction model defined in Equation 5.

The simulation for this evaluation is done using two datasets (i.e., ML-1M and ML-10M) running each method three times and take the average of the achieved RMSE and MAE and the number of epochs needed for achieving the best RMSE and MAE. The results are shown in Table 5.

Table 5. The reduction of the RMSE and MAE achieved by our proposed methods WAFE-SVD, SVD-LM and WSL compared with the SVD for ML-1M and ML-10M datasets.

Method	ML-1M				ML-10M			
	RMSE	Epochs	MAE	Epochs	RMSE	Epochs	MAE	Epochs
SVD	0.8669	85	0.677	100	0.8063	81	0.6198	87
SU	0.8578	57	0.6729	72	0.7934	57	0.6092	66
SI	0.8573	58	0.6728	72	0.7932	56	0.6092	65
SVD-LM	0.8569	55	0.6726	70	0.7929	55	0.6092	62
WAFE-SVD	0.8456	17	0.6636	19	0.7809	22	0.5976	24
WSU	0.8441	19	0.6626	20	0.7784	24	0.5962	25
WSI	0.8433	19	0.6625	20	0.7772	23	0.5988	25
WSL	**0.8386**	18	**0.6549**	20	**0.7689**	24	**0.5861**	25

The results in Table 5 present the performance of the all defined sub-methods along with WSL and SVD by reporting the best RMSE and MAE achieved by each method and the number of epochs

needed for achieving them. These results illustrate the effectiveness of our proposed SVD extension as follows:

(1) **The effectiveness of incorporating the local mean of user and item rating:** Recall that our three methods SU, SI and SVD-LM, examine the effectiveness of incorporating the local mean of user and item rating, it is clear from the results that this incorporation has improved the rating prediction accuracy and reduced the required number of epochs for achieving the best accuracy. For the ML1M dataset, the RMSE achieved by SVD-LM (which combines SU and SI) is 0.8569 comparing to 0.8669 for SVD. Similarly, for the same dataset, the MAE achieved by SVD-LM is 0.6726 comparing to 0.677 for SVD, while the number of epochs needed for achieving the best RMSE was reduced to 55 for SVD-LM, comparing to 85 for SVD. In addition, for the ML10M dataset, SVD-LM achieved RMSE and MAE of 0.7929 and 0.6092, respectively, compared to 0.8063 and 0.6198 for the SVD method. Besides, the required epochs was reduced to 55 for achieving the best RMSE comparing to 81 for the SVD, and 62 for achieving the best MAE comparing to 87 for the SVD.

(2) **The effectiveness of utilising WAFE for defining the latent factors and initialising the user and item features values:** The methods WAFE-SVD, WSU and WSI were proposed to examine the impact of utilising WAFE for defining the latent factors and initialising the user and item features values. The results for this examination show a greater improvement in the performance of the SVD comparing to the previous examined methods (i.e., SU, SI and SVD-LM). It is clear from the results that the three WAFE-based methods have outperformed the other methods. Amongst all six sub-methods (i.e., SU, SI, SVD-LM, WAFE-SVD, WSU and WSI), the best RMSE and MAE were achieved by WSI. For the ML1M dataset, WSI conducted 0.8433 RMSE and 0.6625 MAE, while the number of the required epochs for achieving the best RMSE and MAE were reduced to 19 and 20. For the ML10M dataset, WSI conducted 0.7772 RMSE and 0.5988 MAE, which were achieved at epoch 24 for RMSE and epoch 25 for MAE.

(3) **The effectiveness of WSL:** Our main method WSL, which combines SVD-LM and WAFE-SVD sub-methods, has achieved the best performance amongst all compared methods. WSL achieved 0.8386 RMSE and 0.6549 MAE for ML1M dataset comparing to 0.8669 RMSE and 0.677 MAE for the SVD. It also significantly reduced the number of epochs needed for achieving the best accuracy with only 18 epochs for attaining the best RMSE and 20 epochs for attaining the best MAE comparing with 85 and 100 epochs needed for achieving the best RMSE and MAE, respectively, in SVD. Besides, for the ML10M dataset, WSL achieved RMSE of 0.7689 and MAE of 0.5861 comparing to RMSE of 0.8063 and MAE of 0.6198 in SVD. WSL also reduced the required epochs for achieving the best RMSE and MAE to 24 and 25, respectively, comparing to more that 80 for both metrics in SVD.

9.1.2 Comparison with baseline methods. We compared the prediction accuracy performance achieved by our proposed method WSL with the baseline methods using two MovieLens datasets (ML1M and ML10M) and Amazon Video Games (AVG) dataset. The two datasets ML1M and ML10M were adopted for the comparison with SVD, SVD++ and PMF baseline methods, while the AVG dataset was used for the comparison with the methods NMF, HFT, EFM and AMF. The results and the analysis of this experiment are presented in the following two bullet points.

(1) **MovieLens datasets (ML1M and ML10M):** Using these two datasets, we run the simulation for our proposed method WSL in addition to three baseline methods SVD, SVD++ and PMF. We evaluated the performance of our method (WSL) by comparing its best RMSE and MAE with those achieved by SVD, SVD++ and PMF. In addition, we compared the number of epochs needed for achieving the best accuracy for all methods. Table 6 present the comparison of

the RMSE and MAE achieved by all methods, and Figure 2 illustrates the number of epoch needed to achieve the best RMSE and MAE by all methods.

Table 6. The RMSE and MAE obtained by our proposed method WSL compared with SVD, SVD++ and PMF for ML-1M and ML-10M datasets.

Method	ML-1M		ML-10M	
	RMSE	MAE	RMSE	MAE
SVD	0.8669	0.677	0.8063	0.6198
SVD++	0.8617	0.6708	0.7961	0.6096
PMF	0.8583	0.6656	0.7905	0.608
WSL	**0.8386**	**0.6549**	**0.7689**	**0.5861**

The results in Table 6 show that our method WSL outperformed all the compared methods for both datasets (ML1M and ML10M). The results also show that WSL has significantly reduced the required epoch for achieving the best accuracy. WSL achieved the best RMSE and MAE in the first 25 epochs for both datasets, while all other methods needed at least 40 epochs to accomplish the best accuracy. For example, for the ML1M dataset, SVD needed 85 epoch to accomplish the best RMSE and 100 epochs to achieve the best MAE. Figure 2 illustrates a comparison of the best RMSE and MAE obtained by each method considering the number of epochs needed when running the experiment for ML1M and ML10M datasets.

(2) **Amazon Video Games dataset:** For the AVG dataset, we evaluate the performance of WSL by running the simulation using different data split criteria to compare its prediction accuracy with four baseline methods from the literature that are reported in [12] namely NMF, HFT, EFM and AMF. Table 7 shows the results for the RMSE and MAE for the four baseline methods against the WSL method when verifying the size of the test data for the AVG dataset.

Table 7. The overall RMSE and MAE of the rating prediction achieved by the baseline methods NMF, HFT, EFM and AMF compared to our WSL method when verifying the size of test data for AVG dataset.

Method	RMSE					MAE				
	10%	20%	30%	40%	50%	10%	20%	30%	40%	50%
NMF	1.156	1.175	1.186	1.208	1.264	0.886	0.904	0.914	0.933	0.983
HFT	1.138	1.154	1.193	1.223	1.230	0.869	0.881	0.892	0.921	0.947
EFM	1.119	1.129	1.152	1.183	1.224	0.829	0.835	0.847	0.863	0.891
AMF	1.102	1.107	1.112	1.121	1.148	0.817	0.821	0.823	0.826	0.841
WSL	**1.038**	**1.052**	**1.062**	**1.072**	**1.082**	**0.774**	**0.782**	**0.790**	**0.795**	**0.802**

It is clear from the results in Table 7 that our method WSL outperforms all baseline methods in all cases. Figure 3 shows a comparison of the RMSE and MAE for all compared methods

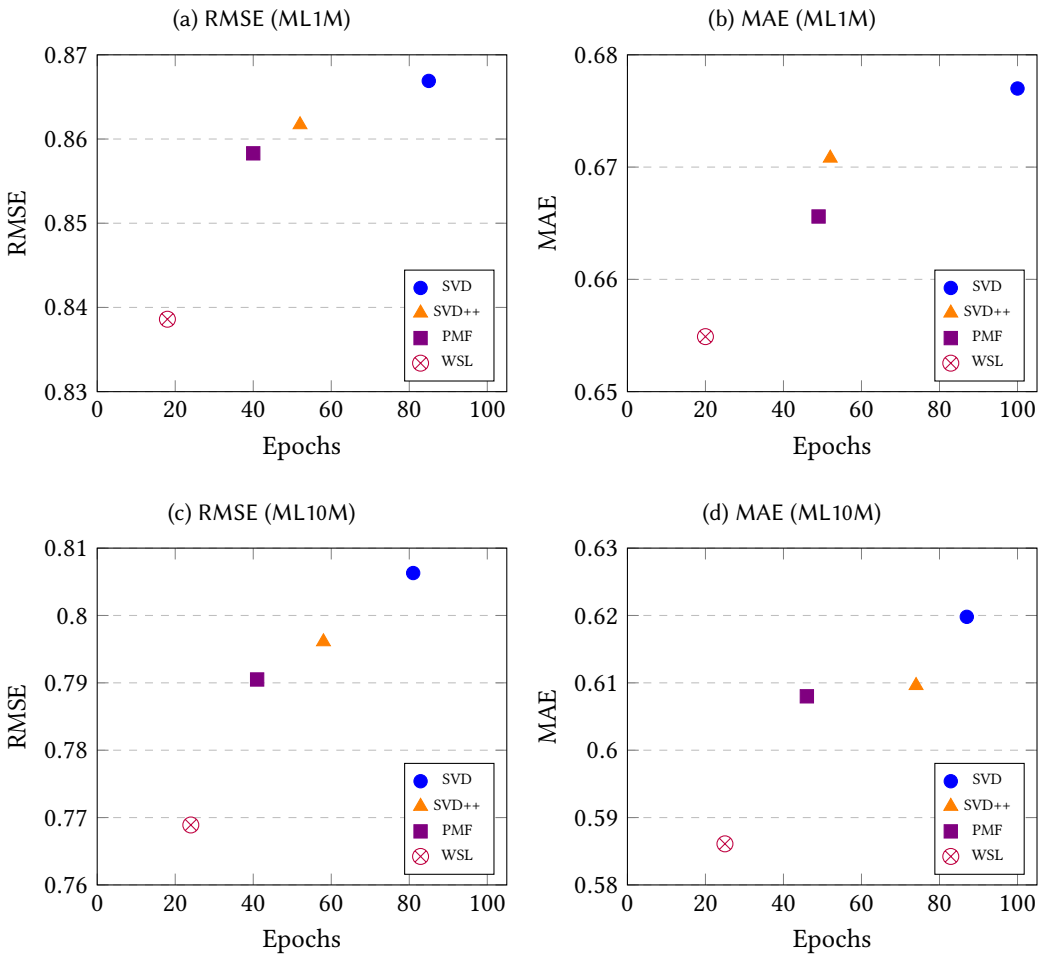

Fig. 2. A comparison of the performance of the methods SVD, SVD++, PMF and WSL considering the number of epochs needed to achieve the best RMSE and MAE for ML1M and ML10M datasets.

when splitting the data into 80% for training and 20% for testing. It shows that our method WSL has achieved the lowest RMSE and MAE compared with the baseline methods.

In addition, we conducted a significance test using paired t-test for our method WSL and each of the baseline methods (i.e., NMF, HFT, EFM and AMF) resulting in four paired t-test cases. Each t-test for WSL and one of the baseline methods. In this experiment, we consider a null hypothesis and an alternative hypothesis as follows:

- *Null hypothesis*: The performance of WSL and the considered baseline method are at the same level.
- *Alternative hypothesis*: The performance of WSL is better than the considered baseline method.

We examined these two hypothesises by calculating the p-value for each t-test. The resulting p-value for each t-test are reported in Table 8.

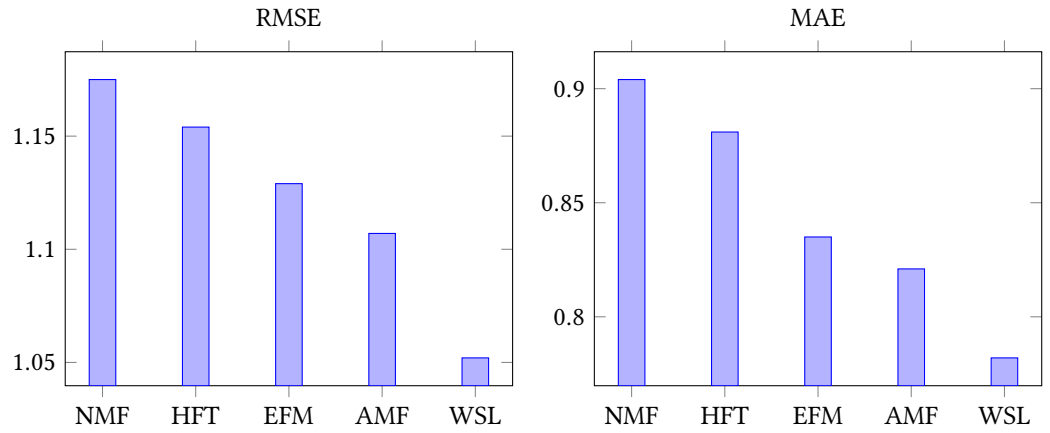

Fig. 3. A comparison of the performance of all methods in terms of rating prediction error for AVG dataset when splitting the data into 80% for training and 20% for testing.

Table 8. The p-value resulted by conducting t-test for our method WSL and each of the baseline methods (i.e., NMF, HFT, EFM and AMF) using the AVG dataset.

Method	p-value
NMF	0.0001375188
HTF	0.0002128798
EFM	0.0012524121
AMF	0.0009465036

The results of the paired t-test conducted for WSL and each of the baseline methods show that the achieved p-value for each case is within the range of rejecting the null hypothesis (i.e., p-value < 0.05). Thus, this evidence leads us to accept the alternative hypothesis and conclude that the improvement achieved by WSL is statistically significant.

9.2 Top-n Recommendation

We conducted this experiment to evaluate the WSL performance in terms of top-n recommendation. As explained in Section 8.3.2, we followed the experimental setting of [23] in this experiment for the purpose of comparison. We adopted the evaluation metric MAP@n and split the data into 70% for training and 30% for testing. We conducted the experiment for the ML-1M dataset. The results are reported in Table 9 . Similar to [23], the n is assigned to 10.

The results in Table 9 represents the performance of all compared methods in terms of MAP@10, i.e., the mean average precision at the top 10 recommendation list generated for all users. Table 9 reports the MAP@10 in different data sparsity as we verify the percentage of the data taken for training. The results show that our method, WSL outperformed all baseline methods in all cases. Despite the fact that increasing the test size can enlarge the size of the related items in the test data for each user, it decreases the rating prediction accuracy. Recall that the items in the top-n recommendation are listed in descending order based on the predicted ratings for them with the user. Thus, the more accurate rating prediction the method achieves, the more relative items can

Table 9. The MAP@10 achieved by WSL compared to the baseline methods, SVD, NueMF and PrefCRBM.

Method	Training data		
	70%	40%	20%
SVD	0.1254	0.1164	0.1109
NueMF	0.4025	0.3727	0.3497
PrefCRBM	0.4099	0.3809	0.3657
WSL	**0.5405**	**0.4048**	**0.3737**

be listed in the top-n recommendations. It is clear from the results that the prediction accuracy has a significant impact on the MAP@n measure for top-n recommendation accuracy.

In addition, we examined the effectiveness of the number of items in the top recommendation list, i.e., the value of n. In this examination, we conducted our simulation for calculating the MAP@n metric when verifying the value of n in the range (10, 20, ..., 50). This experiment was performed for the SVD and WSL methods using the ML1M dataset, splitting the data into 70% for training and 30% for testing. The results are presented in Table 10.

Table 10. The change of the MAP@n score when verifying the value of n for the SVD and WSL methods using the ML1M dataset.

Method	MAP@n				
	n=10	n=20	n=30	n=40	n=50
SVD	0.1255	0.1423	0.1584	0.1700	0.1791
WSL	0.5405	0.6447	0.7025	0.7424	0.7729

The results in Table 10 show that the MAP@n score is improving with higher value of n. This improvement is illustrated as expanding the size of the list of the top recommendations is more likely to increase the chance of including more relative items to the user. At the same time, the rating prediction accuracy remains at the same level as we do not decrease the size of the training data. It should also be noted that the improvement of the MAP@n score decreases after the point of n=20; as the difference between the achieved score and the previous score becomes smaller starting from the point of n=30. This result is reasonable as the number of uncovered relative items in the top-n list for each user is becoming smaller while increasing the value of n.

9.3 Explainable Recommendation

As explained in Section 8.3.3, we evaluated the explanation of our recommendations model in two ways, i.e. overall satisfaction and user case study.

9.3.1 Overall Satisfaction. The overall explanation performance is evaluated in terms of the *accuracy*, *errorrate*, *precision*, *recall*, and *F-measure* metrics. Table 11 shows the results of these metrics running our experiment for the six datasets when splitting the data into 80% for training and 20% for testing and setting the parameters as specified in Table 4.

As shown in Table 11, the proposed method WSLER performs very well for the explainable recommendation for all datasets. The results show that the heights accuracy and the lowest error

Table 11. The performance of the SWLER model for all datasets in terms of accuracy, error rate, precision, recall and F-measure.

Evaluation metric	Dataset					
	ML1M	ML10M	BC	AVG	AIV	AF
Accuracy	0.88	0.86	0.94	0.92	0.81	0.91
Error rate	0.12	0.14	0.06	0.08	0.19	0.09
Precision	0.98	0.98	0.95	0.94	0.81	0.94
Recall	0.91	0.90	0.97	0.96	0.96	0.93
F-measure	0.94	0.94	0.96	0.95	0.88	0.93

rate were achieved by WSLER for the BC dataset with 0.94 accuracy and 0.06 error rate. Besides, the highest F-measure score, which somehow concludes the precision and recall, was achieved by WSLER for the BC dataset with 0.96. It is also clear that the lowest accuracy was achieved is 0.81, and the highest scored error rate is 0.19 for the AIV dataset. In addition, the lowest F-measure score is 0.88, which was accomplished for the AIV dataset.

In addition, we evaluated our explainable recommendation model WSLER by calculating the positive satisfaction ratio (PSR) for all users in the test data as explained in Section 8.3.3. We applied this evaluation criteria for all datasets, and the results showed a good performance with a minimum positive satisfaction ratio (PSR) of 94% for the AIV dataset and 98% for the AF dataset as the highest PSR. Results are shown in Figure 4.

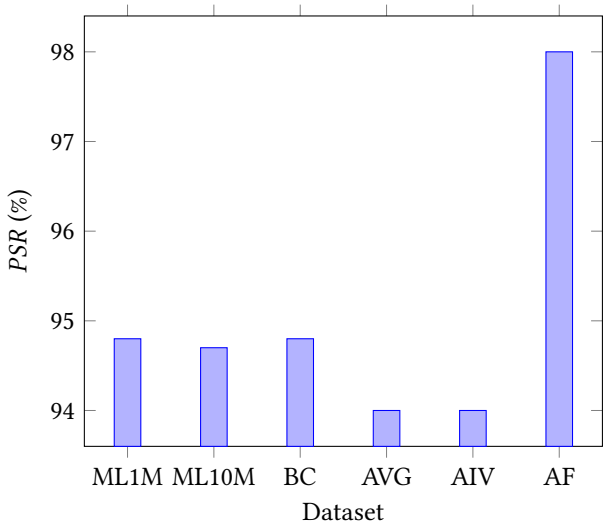

Fig. 4. The Positive Satisfaction Ratio (*PSR*) achieved by our method WSLER for all datasets.

Moreover, we compared the performance of our explainable recommendation model WSLER with the AMF model proposed by Hou et al. (2019) [12] in terms of positive satisfaction ratio (PSR). For the purpose of the comparison, we selected AVG dataset as it is the only dataset adopted by [12] among all datasets that we adopt in our study. Therefore, we set the recommendation threshold

(ts^{rec}) to 4 instead of 3, as similar to [12]. Figure 5 shows a comparison of the PSR achieved by our model WSLER with the AMF model [12].

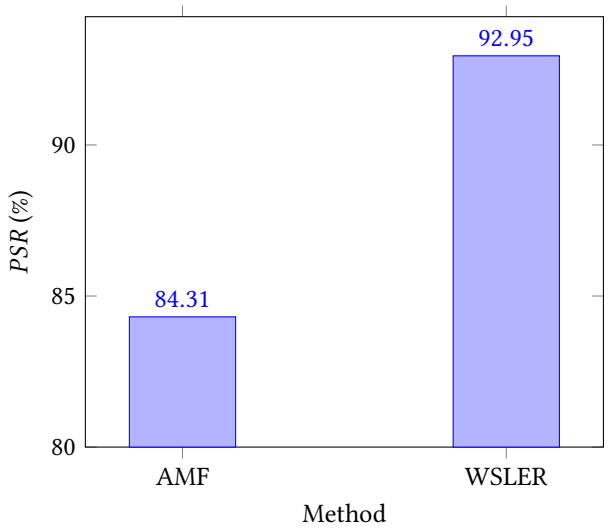

Fig. 5. A comparison the Positive Satisfaction Ratio (*PSR*) achieved by our method WSLER and AMF [12] for AVG dataset (when assigning $ts^{rec} = 4$ similar to [12]).

The results show that our proposed WSLER model significantly outperformed AMF model. WSLER have achieved 92.95% PSR compared to 84.31% achieved by AMF method as reported in their paper [12].

9.3.2 User Case Studies. In this section, we present the result of the user case studies analysis for each dataset. As we adopted six datasets in this study, i.e., ML1M, ML10M, BC, AVG, AIV and AF, we performed six case studies; one for each dataset. For each case study, we randomly selected one user with two items of their rated items, in which one is assigned a high predicted rating (recommended) and the other is given a low predicted rating (unrecommended). Then, we examine whether the user is more interested in the recommended item's features than the unrecommended item's features. We calculated the intersection score $intSec_{u,i}$ for the user and the recommended item, and for the user and the unrecommended item. Figure 6 presents the features in common between each user and item for each case study, and Table 12 show the comparison of the intersection score for the user and both items in each case study. The results are discussed in the following text.

- Case study 1 (ML1M): $user22$, $item383$ and $item570$ were randomly selected, in which $item383$ is assigned a high predicted rating while $item570$ is assigned a low predicted rating by the user $user22$. We found that $user22$ is interested in 4 out of 5 features of the recommended item $item383$, while user $user22$ is interested in only 1 feature out of 4 of the features of the unrecommended item $item570$ (see Figure 6a). We also calculated the intersection score for $user22$ with both $item383$ and $item570$, and we found that the intersection score for $user22$ and $item383$ is 0.8 compared to only 0.25 for $user22$ and $item570$ (see Table 12). The recommended item $item383$ corresponds to 5 features, i.e. *drama*, *thriller*, *reputation9*, *popularity4* and *year4*. Out of these 5 features, the user is interested in the features *drama*, *thriller*, *reputation9*, and *year4*. This means that the user is interested in most of the item's features, so we can confidently recommend it. This recommendation can be simply justified

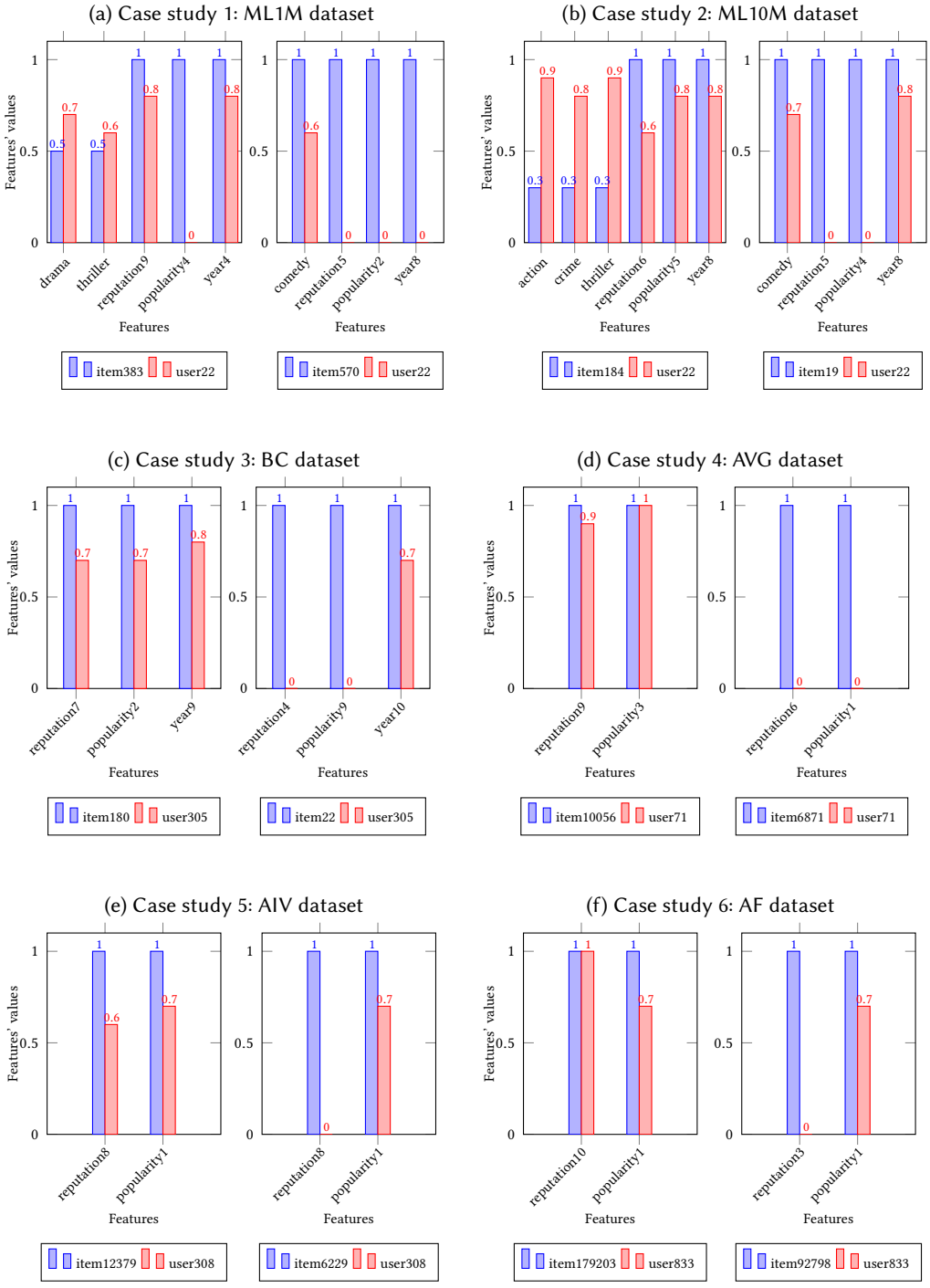

Fig. 6. The values of the features in common between the user and both items for each case study, where the listed features in each bar chart are the features of the concerned item.

Table 12. The intersection score for the user (u) and both items (i_1 and i_2) for each case study, where i_1 is the recommended item and i_2 is the unrecommended item for the user u in each case study.

Case study	Intersection between u and i_1			Intersection between u and i_2		
	$\lvert v_{i_1}^{pos} \rvert$	$\lvert v_{u,i_1} \rvert$	$intSec_{u,i_1}$	$\lvert v_{i_2}^{pos} \rvert$	$\lvert v_{u,i_2} \rvert$	$intSec_{u,i_2}$
ML1M	5	4	0.8	4	1	0.25
ML10M	6	6	1	4	2	0.5
BC	3	3	1	3	1	0.33
AVG	2	2	1	4	0	0
AIV	2	2	1	4	1	0.5
AF	2	2	1	4	1	0.5

as "*We recommend this item to you as it has the features (drama, thriller, reputation9, and year4) which all are some of your interest*". In contrast, the unrecommended item $item570$ corresponds to 4 features, i.e., *comedy*, *reputation5*, *popularity2* and *year8*. Out of these 4 features, the user $user22$ is interested in only 1 feature, i.e., *comedy*.

- Case study 2 (ML10M): $user22$ was randomly selected along with $item184$ and $item19$, in which $item184$ has been assigned with a higher predicted rating than $item19$. The recommended item $item184$ is categorised with 6 features, i.e., *action*, *crime*, *thriller*, *reputation6*, *popularity5* and *year8*. We found that the user $user22$ is interested in all $item184$'s features. This means that the recommendation can be reasonably explained as the intersection between the user's interesting features and the item's features is very high. An explanation of this recommendation could be "*We found this movie which has the features (action, crime, thriller, reputation6, popularity5 and year8) in which we think that you are interested in movies with these features*". On the other hand, $user22$ is interested in only 2 features out of 4 features of the item $item19$. Figure 6b illustrates the values of $user22$'s features against the features of $item184$ and $item19$.

- Case study 3 (BC): $user305$ was selected randomly and $item180$ was assigned a high predicted rating while $item22$ was assigned a low one. In this case study, the recommended item $item180$ has 3 features, i.e., *reputation7*, *popularity2* and *year9*. We found that the user $user305$ is interested in all these three features. This reflects a strong relationship between the user and the item, as their intersection features are 3 out of 3 item's features. This recommendation could be explained as "*This item is recommended to you as we found that you are interested in all its features, which are (reputation7, popularity2 and year)* ". On the other hand, we found that the user is interested in only 1 of the features of the unrecommended item $item22$. Thus, the relationship between the user and the recommended item is stronger than the relationship between the user and the unrecommended item. This can be seen clearly from the intersection features between the user and the two items. Figure 6c shows the intersections between $user305$'s interesting features with the features of $item180$ and $item22$.

- Case study 4 (AVG): $user71$ was randomly selected along with $item10056$ and $item6871$, and $item10056$ was assigned a higher predicted rating than $item6871$. In AVG dataset, user and item features are categorised into two feature concepts, which means that the maximum

possible number of positive item features is 2. We found that the user *user*71 is interested in both features of the recommended item *item*10056, which are *reputation9 and popularity3*. On the other hand, the user is not interested in any of the features of the unrecommended item *item*6871. Therefore, it is reasonable to recommend *item*10056 to the user *user*71 which can be justified as follows: "*This item is recommended to you as we found that you are interested in items with features (reputation9 and popularity3) which are the features of this item*". A summary of features intersections between the user and the two items are presented in Figure 6d.

- Case study 5 (AIV): *user*308 was randomly selected along with *item*12379 and *item*6229, and *item*12379 was assigned with a higher predicted rating than *item*6229. All features of the users and items in AIV dataset represent two feature concepts, in which the maximum possible number of positive item features is 2. The results show that *user*308 is interested in all features of the recommended item *item*12379', while he/she is not interested in only one of the features of the unrecommended *item*6229'. Thus, it is reasonable to recommend *item*12379 to the user *user*308, in which the recommendation can be justified as: "*This item is recommended to you as we found that you like the items with features (reputation8 and popularity1) which both are the features of this item*". The values of the intersection features between *user*308 and both *item*12379 and *item*6229 are illustrated in Figure 6e.

- Case study 6 (AF): *user*833 was randomly selected along with *item*179203, which was assigned a high predicted rating, and *item*92798, which was assigned a low predicted rating. Similar to AVG and AIV, all features of the users and items in AF dataset represent two feature concepts, which means that the maximum possible number of positive item features is 2. In this case study, we found that the user is interested in both features of the recommended item *item*179203, which are (*reputation10 and popularity1*). However, only 1 feature of the features of the unrecommended item *item*92798 is of the user's interest. This can explain the reason of recommending *item*179203 to *user*833 as "*You are interested in items with features (reputation10 and popularity1), and we found this item which has both features*". The intersection features between the user *user*833 and both items *item*179203 and *item*92798 are shown in Figure 6f.

In all case studies, the results show that WSLER is able to provide a reasonable explanation for the recommendations suggested to the users. The relationship between the user and the recommended item (i.e., the item with a high predicted rating) is stronger than the relationship between the user and the item with the low predicted rating in all case studies. In fact, the preformed case studies have examined how much the user is interested in the item features taking into consideration the number of features that describe the item. For example, in case study 1, the user is interested in 4 out of 5 features of the recommended item features compared to only 1 out of 4 of the unrecommended item's features. Similarly, in case study 2, the user is interested in 6 features out of 6 features that describe the item compared to only 2 out of 4 of the unrecommended item features. Moreover, case study 3 shows that all features of the recommended item are interesting features to the user, while only one feature out of 3 of the unrecommended item's features is interesting to the user. In case studies 4, 5 and 6, in which the items are described using two features, the results show that all recommended item features are interesting features to the users in all three cases.

It is also clear from the results in Table 12 that for all case studies, the intersection score between the user and the recommended item is 0.8 or higher, compared to 0.5 for the unrecommended item.

Thus, this means that the users in all cases are interested in all or most of the recommended item features. At the same time, they are interested in only a few or none of the unrecommended item features. These results can explain why such recommendations are performed, i.e., an item can

be recommended to a user if there is a strong relationship between the user and the item. This is because our method, WSLER, different from other MF-based methods, adopts useful knowledge for modelling the user and item latent factors, making these factors interpretable. More precisely, the defined meaningful latent factors and the initial values of user and item features can explain how much each user or item is related to each latent factor. We found that these relationships are useful for explaining the recommendations provided by WSLER. In fact, the relationships between user and item that are derived from the initial values of their latent features are compatible with the predicted ratings performed by WSLER, which is the base of the recommendation.

10 CONCLUSION AND FUTURE WORK

Explainable recommender systems provide users with reasonable justification of what they have been recommended. This explanation plays a vital role in increasing the system reliability. This explanation can be performed using different approaches, such as feature-based explainable recommendations. In this paper, we introduced WSLER, an explainable recommender system framework. Inside WSLER, we proposed WSL: an MF-based recommender system model, and ER: an explainable recommendation model. WSL extends the well-performing MF model (i.e., SVD) to produce meaningful latent factors and useful features values and improve the prediction accuracy. At the same time, ER utilises the useful features produced by WSL for explaining the recommendation. In contrast to the existing SVD methods, WSLER exploits only local data without relying on private or outer data for providing accurate and explainable recommendations. We evaluated our proposed method using six real-world benchmark datasets in terms of rating prediction and explainability performance. Our experimental simulation results show that our method outperformed very well in both evaluation aspects.

One limitation of this study is that the experimental evaluation was performed using offline criteria. However, it would be worthy to perform the evaluation using an online evaluation method to investigate the effectiveness of the proposed method on the users purchasing behaviours. Consequently, the evaluation criteria for the top-n recommendations can be performed to genuinely reflect the users' reaction beyond the top recommendation list. In addition, this study mainly focused on the rating prediction evaluation. However, one of the most important aspects of recommender system research is to investigate how the recommendations are presented to the user. This topic needs further research. It is also essential to address the question of how to best present recommendations to the users.

REFERENCES

[1] Behnoush Abdollahi and Olfa Nasraoui. 2017. Using Explainability for Constrained Matrix Factorization. In *Proceedings of the Eleventh ACM Conference on Recommender Systems* (Como, Italy) *(RecSys '17)*. Association for Computing Machinery, New York, NY, USA, 79–83. https://doi.org/10.1145/3109859.3109913

[2] A. Alhejaili and S. Fatima. 2020. Latent Feature Modelling for Recommender Systems. In *2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI)* (Las Vegas, NV, USA). IEEE Computer Society, Los Alamitos, CA, USA, 349–356. https://doi.org/10.1109/IRI49571.2020.00057

[3] Mustafa Bilgic and Raymond J Mooney. 2005. Explaining recommendations: Satisfaction vs. promotion. In *Beyond Personalization 2005* (San Diego, California). IUI 2005, San Diego, California, 13–18. https://grouplens.org/beyond2005/bp2005.pdf

[4] Robin Burke. 2002. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction* 12, 4 (01 Nov 2002), 331–370.

[5] Xu Chen, Yongfeng Zhang, and Zheng Qin. 2019. Dynamic Explainable Recommendation Based on Neural Attentive Models. *Proceedings of the AAAI Conference on Artificial Intelligence* 33, 01 (Jul. 2019), 53–60. https://doi.org/10.1609/aaai.v33i01.330153

[6] Michael D. Ekstrand, John T. Riedl, and Joseph A. Konstan. 2011. Collaborative Filtering Recommender Systems. *Found. Trends Hum.-Comput. Interact.* 4, 2 (Feb. 2011), 81–173. https://doi.org/10.1561/1100000009

[7] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.* 5, 4, Article 19 (Dec. 2015), 19 pages. https://doi.org/10.1145/2827872

[8] Ruining He and Julian McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *Proceedings of the 25th International Conference on World Wide Web* (Montréal, Québec, Canada) *(WWW '16)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 507–517. https://doi.org/10.1145/2872427.2883037

[9] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web* (Perth, Australia) *(WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 173–182. https://doi.org/10.1145/3038912.3052569

[10] Jonathan L. Herlocker, Joseph A. Konstan, and John Riedl. 2000. Explaining Collaborative Filtering Recommendations. In *Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work* (Philadelphia, Pennsylvania, USA) *(CSCW '00)*. Association for Computing Machinery, New York, NY, USA, 241–250. https://doi.org/10.1145/358916.358995

[11] Mohammad Hossin and MN Sulaiman. 2015. A review on evaluation metrics for data classification evaluations. *International Journal of Data Mining & Knowledge Management Process* 5, 2 (2015), 1.

[12] Yunfeng Hou, Ning Yang, Yi Wu, and Philip S Yu. 2019. Explainable recommendation with fusion of aspect information. *World Wide Web* 22, 1 (2019), 221–240. https://doi.org/10.1007/s11280-018-0558-1

[13] Yehuda Koren. 2008. Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Las Vegas, Nevada, USA) *(KDD '08)*. Association for Computing Machinery, New York, NY, USA, 426–434.

[14] Y. Koren, R. Bell, and C. Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (2009), 30–37.

[15] Max Kuhn and Kjell Johnson. 2019. *Feature engineering and selection: A practical approach for predictive models*. CRC Press, 6000 Broken Sound Pkwy NW, suite 300.

[16] Shyong K. "Tony" Lam, Dan Frankowski, and John Riedl. 2006. Do You Trust Your Recommendations? An Exploration of Security and Privacy Issues in Recommender Systems. In *Emerging Trends in Information and Communication Security*, Günter Müller (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 14–29.

[17] Daniel Lee and H. Sebastian Seung. 2001. Algorithms for Non-negative Matrix Factorization. In *Advances in Neural Information Processing Systems*, T. Leen, T. Dietterich, and V. Tresp (Eds.), Vol. 13. MIT Press, Cambridge, MA, USA, 556–562. https://proceedings.neurips.cc/paper/2000/file/f9d1152547c0bde01830b7e8bd60024c-Paper.pdf

[18] Pasquale Lops, Marco de Gemmis, and Giovanni Semeraro. 2011. Content-based Recommender Systems: State of the Art and Trends. In *Recommender Systems Handbook*, Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor (Eds.). Springer US, Boston, MA, 73–105.

[19] Julian McAuley and Jure Leskovec. 2013. Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text. In *Proceedings of the 7th ACM Conference on Recommender Systems* (Hong Kong, China) *(RecSys '13)*. Association for Computing Machinery, New York, NY, USA, 165–172. https://doi.org/10.1145/2507157.2507163

[20] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-Based Recommendations on Styles and Substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Santiago, Chile) *(SIGIR '15)*. Association for Computing Machinery, New York, NY, USA, 43–52. https://doi.org/10.1145/2766462.2767755

[21] Andriy Mnih and Russ R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*. Curran Associates Inc., Red Hook, NY, USA, 1257–1264.

[22] Arkadiusz Paterek. 2007. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD cup and workshop*, Vol. 2007. Association for Computing Machinery, New York, NY, USA, 5–8.

[23] Abinash Pujahari and Dilip Singh Sisodia. 2019. Modeling side information in preference relation based restricted boltzmann machine for recommender systems. *Information Sciences* 490 (2019), 126–145.

[24] Zhaochun Ren, Shangsong Liang, Piji Li, Shuaiqiang Wang, and Maarten de Rijke. 2017. Social Collaborative Viewpoint Regression with Explainable Recommendations. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining* (Cambridge, United Kingdom) *(WSDM '17)*. Association for Computing Machinery, New York, NY, USA, 485–494. https://doi.org/10.1145/3018661.3018686

[25] Francesco Ricci, Lior Rokach, and Bracha Shapira. 2011. Introduction to recommender systems handbook. In *Recommender systems handbook*, Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor (Eds.). Springer US, Boston, MA, 1–35.

[26] Francesco Ricci, Lior Rokach, and Bracha Shapira. 2015. *Recommender Systems: Introduction and Challenges*. Springer US, Boston, MA, 1–34. https://doi.org/10.1007/978-1-4899-7637-6_1

[27] Claude Sammut and Geoffrey I. Webb (Eds.). 2010. *Latent Factor Models and Matrix Factorizations*. Springer US, Boston, MA, 571–571. https://doi.org/10.1007/978-0-387-30164-8_887

[28] J. Ben Schafer, Joseph Konstan, and John Riedl. 1999. Recommender systems in e-commerce. In *Proceedings of the 1st ACM Conference on Electronic Commerce, EC 1999 (ACM International Conference Proceeding Series)*. Association for Computing Machinery, New York, NY, USA, 158–166. https://doi.org/10.1145/336992.337035 1st ACM Conference on Electronic Commerce, EC 1999 ; Conference date: 03-11-1999 Through 05-11-1999.

[29] J. Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. 2007. *Collaborative Filtering Recommender Systems*. Springer-Verlag, Berlin, Heidelberg, 291–324.

[30] Rashmi Sinha and Kirsten Swearingen. 2002. The Role of Transparency in Recommender Systems. In *CHI '02 Extended Abstracts on Human Factors in Computing Systems* (Minneapolis, Minnesota, USA) *(CHI EA '02)*. Association for Computing Machinery, New York, NY, USA, 830–831. https://doi.org/10.1145/506443.506619

[31] Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence* 2009 (2009), 1–19.

[32] Jesse Vig, Shilad Sen, and John Riedl. 2009. Tagsplanations: Explaining Recommendations Using Tags. In *Proceedings of the 14th International Conference on Intelligent User Interfaces* (Sanibel Island, Florida, USA) *(IUI '09)*. Association for Computing Machinery, New York, NY, USA, 47–56. https://doi.org/10.1145/1502650.1502661

[33] Yongfeng Zhang and Xu Chen. 2020. Explainable Recommendation: A Survey and New Perspectives. *Foundations and Trends® in Information Retrieval* 14, 1 (2020), 1–101. https://doi.org/10.1561/1500000066

[34] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit Factor Models for Explainable Recommendation Based on Phrase-Level Sentiment Analysis. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval* (Gold Coast, Queensland, Australia) *(SIGIR '14)*. Association for Computing Machinery, New York, NY, USA, 83–92. https://doi.org/10.1145/2600428.2609579

[35] Wayne Xin Zhao, Sui Li, Yulan He, Liwei Wang, Ji-Rong Wen, and Xiaoming Li. 2016. Exploring demographic information in social media for product recommendation. *Knowledge and Information Systems* 49, 1 (2016), 61–89.

[36] Xin Wayne Zhao, Yanwei Guo, Yulan He, Han Jiang, Yuexin Wu, and Xiaoming Li. 2014. We Know What You Want to Buy: A Demographic-Based System for Product Recommendation on Microblogs. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (New York, New York, USA) *(KDD '14)*. Association for Computing Machinery, New York, NY, USA, 1935–1944. https://doi.org/10.1145/2623330.2623351

[37] Alice Zheng and Amanda Casari. 2018. *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists* (1st ed.). O'Reilly Media, Inc., 1005 Gravenstein Highway North Sebastopol, CA 95472.

[38] Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, and Georg Lausen. 2005. Improving Recommendation Lists through Topic Diversification. In *Proceedings of the 14th International Conference on World Wide Web* (Chiba, Japan) *(WWW '05)*. Association for Computing Machinery, New York, NY, USA, 22–32. https://doi.org/10.1145/1060745.1060754