ECTI Transactions on Computer and Information Technology
Journal homepage: https://ph01.tci-thaijo.org/index.php/ecticit/
Published by the ECTI Association, Thailand, ISSN: 2286-9131

Multiple Latent Space Ensemble for Matrix Factorization Based Collaborative Filtering

Vasin Jinopong[1], Dussadee Praserttitipong[2] and Jakramate Bootkrajang[3]

ABSTRACT

Matrix factorization is arguably one of the most widely employed collaborative filtering techniques for recommender systems. The recommendation is obtained based on the user-item relationships discovered within some lower-dimensional latent space. The optimal latent space is generally data-dependent and often needs to be selected using the time-consuming cross-validation scheme. In this paper, we propose to leverage the power of the ensemble method not only to facilitate the hyper-parameter selection but also to improve the predictive performance of the system. Specifically, we studied ways to combine predictions from multiple Singular Value Decomposition models, each operates in its own latent space. Experimental results based on MovieLen100K, MovieLen1M, Bookcrossing and Filmtrust datasets demonstrated that the ensembles outperformed a tuned single model in terms of RMSE and MAE while requiring no additional model selection step. Ensemble sizes experiment have shown that the 21 submodel of the ensemble models produce better results than the 14, 8 and standalone model. However, it takes longer to complete. We also found that an ensemble that pays more attention to lower-dimensional latent spaces tends to generalize better.

Article information:

Keywords: Collaborative Filtering, Ensemble, Matrix Factorization, Singular Values Decomposition

Article history:
Received: March 12, 2022
Revised: May 27, 2022
Accepted: June 2, 2022
Published: November 12, 2022 (Online)

DOI: 10.37936/ecti-cit.2022164.247873

1. INTRODUCTION

Recommender systems play an essential role in modern business strategy. They can be found in both online shopping and movie businesses such as Amazon and Netflix, to name a few. The goal is to increase the ability to recommend interesting products to the users. As a consequence, several product recommendation systems have been developed continuously. In general, there are two types of recommendation approaches: Content-based (CB) [1] and Collaborative Filtering (CF) [4]. The basic idea of the content-based approach is to use properties of an item to predict the user's interest in it. Most CBs advise users with an inferential model trained from the characteristics of both users' and items' profiles. On the other hand, the CF approach generates personalized recommendations according to the calculated similarity of historical data among collaborative users. Various techniques have been developed to build recommendation solutions based on the CF concept. However, CF has attracted much attention in the past decade, resulting in significant progress. It was adopted by several successful commercial systems, including Amazon, TiVo, CDNow.com, and Netflix. This is because CF models can be more flexible with regard to the types of rating [4]. The historical database represents relationships between users and items with the corresponding rating scores which are collected in the form of a users-items rating matrix. The pre-constructed rating prediction model, which represents the behavior of the users assembled from the users-items rating matrix, is established using offline learning algorithms.

One of the notable techniques for rating prediction in CF is Singular Value Decomposition (SVD). The SVD-based technique factorizes the users-items rating matrix into latent factor matrices and other learning parameters. The overall size of these learning parameters is smaller than the entire users-items rating matrix. Therefore, when recommendations are

[1] The author is with Graduate Program in Data Science,Data Science Consortium, Faculty of Engineering, Chiang Mai University, Chiang Mai, Thailand 50200, E-mail: vasinearth14@gmail.com
[2,3] The authors are with Data Science Research Center, Department of Computer Science, Faculty of Science, Chiang Mai University, ChiangMai, Thailand 50200, E-mail: dussadee.p@cmu.ac.th and jakramate.b@cmu.ac.th

called for, the recommendation results are returned based on the re-composition of the learning parameters, which encompass latent factor matrices and other additional learning parameters [2]. In many research efforts, a recommendation system model based on SVD shows sufficient efficiency in predicting the rating. Many researchers try to improve SVD for better prediction and reduction of errors between the actual rating and the predicted rating [3]. However, there are still challenges associated with using SVD in recommendation systems.

The crucial factor that affects the model's accuracy is the dimensionality of the latent space, e.g., the number of latent factors. Interestingly, the hyper-parameter is usually data dependent [5]. The optimal value for one dataset might not necessarily be optimal for other datasets. This is because each dataset may have different distribution characteristics, such as the distribution of the data or the sparseness of the users and items. Furthermore, the size of the users-items rating matrix grows explosively due to the enormous number of people accessing the service while new items are being added to the system every day. Thus, another essential concern is how to efficiently re-train the SVD-based model in order to adapt to new information. Clearly, the traditional brute-force method for determining the appropriate hyper-parameter value might be impractical for the fast-changing nature of data. To mitigate the difficulties mentioned above, this paper proposes to employ ensemble of a constant set of multiple SVD-based models to facilitate hyper-parameter tuning and improve predictive performance. The predictions from the ensemble which the combination of multiple SVD-based models' predictions.

By considering SVD-based techniques, the computationally intensive brute-force method for finding the best k-latent factors can be replaced with ensemble of a constant set of multiple SVD-based rating predictions models. The latter is much faster, as can be seen from the experimental results. Moreover, the combined multiple SVD-based models as an ensemble tend to outperform the tuned single SVD-based model.

The rest of this paper is organized as follows. Section 2 presents the background on matrix factorization and SVD as well as notable related work. The proposed ensemble learning methodologies are presented in Section 3. Empirical evaluation of the proposed method is given in Section 4, Section 5 concludes the study and outlines future research directions.

2. BACKGROUND AND RELATED WORK

Recommender Systems implement a long-established technique that has been applied in many industrial domains [12]. For example, they have been used in music [13], e-commercial [14], and the film industry [15], especially Netflix. CF is the substantial part of recommender systems that provide practical recommendations via behavior correlation between users and items. In recent years, matrix factorization such as the SVD and the SVD++ models has become popular by providing good predictive accuracy and flexibility for real-life scenario modeling [16]. Here we briefly describe the working of the two models which will be used as the base models in the proposed ensemble methods.

2.1 SVD

Singular Values Decomposition (SVD) is one approach to matrix factorization which is used to create an efficient CF recommendation system. The SVD technique factorizes the user-rating matrix into a user matrix and an item matrix both correlated with each other. The conventional SVD technique [8] maps users and items in the correlated directions. The associated rating matrix $R \in \Re^{m\times n}$ is decomposed into user latent matrix $p_u \in \Re^k$ and item latent matrix $q_i \in \Re^k$, where k is the number of latent factors. By the characteristic of matrix factorization, both decomposed matrices can be used to recover to the original user-rating matrix and approximate the rating of the user-item pair for an unknown rating. A baseline estimation for predicted the rating score of user u over item i, denoted as $\hat{r}_{ui}$ of this conventional SVD technique, is formulated as in Equation (1).

$$\hat{r}_{ui} \approx p_u q_i^T \tag{1}$$

Normally, the user-item rating matrix exhibits systematic tendencies for some users to give higher ratings than others, and for some items to receive higher ratings than others. These tendencies are known as biases or intercepts. An improved regularized SVD [9] was proposed for handling the biases and data variations. This technique extends the prediction accuracy of this conventional SVD technique by adding some bias parameters to its model. A baseline estimation for predicted rating score of user u over item i, ($\hat{r}_{ui}$) of this technique is given in Equation (2).

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u \tag{2}$$

μ is a global average rating. p_u and q_i are k-dimensional latent feature vectors of user u and item i, respectively. The parameters b_u and b_i are the observed bias of both user u and item i, respectively. In addition, p_u and q_i are k-dimensional latent feature vectors of user u and item i, respectively.

In 2017, Xin Guan [17] presented an enhanced SVD model to improve the accuracy of the recommender system. A series of user and item models were implemented and validated the performance of collaborative filtering recommendations. Transforming models into different formats can make a significant difference in model performance.

2.2 SVD++

Recommendation systems often suffer from data sparsity and cold start issues [9, 11]. Limited explicit information about user-item rating interactions does not provide efficient recommendation performance. Thus, SVD++ [10] adds other concerning information to its model, named implicit feedback, which is comprised of the Boolean implicit feedback values that indicate whether the concerning item has been rated by users or not. A baseline estimation for predicted the rating score of user u over item i, ($\hat{r}_{ui}$) of SVD++ is established in Equation (3).

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T(p_u + |I_u|^{(-1/2)} \sum_{j \in I_u} y_j) \quad (3)$$

μ is the global mean. p_u and q_i are k-dimensional latent feature vectors of user u and item i, respectively. The bias term b_u is referred to as the bias of the user, and b_i denotes the bias of the item. Additionally, $|I_u|$ is a set of items rated by user u, and y_j is k-dimensional vector that collects the implicit influence of items rated by user u. The term $|I_u|^{-1/2} \sum_{j\in\ I_u} y_j$ represents the eigenvector of the user u on the implicit feedback.

An interesting meta-learning approach to improve the performance of a single SVD or SVD++ model is the ensemble method. Generally, an ensemble with sufficiently diverse members often yields better results than each of its members. Ariel Bar [18] studied the improvement of collaborative filtering models using two ensemble methods: the bagging and the boosting techniques. The results showed that the ensemble outperformed one single weak model as measured by the Root Mean Square Error. Motivated by the above study, we set out to explore other model ensemble strategies hopes of finding a method with improved rating prediction accuracy. We also want to leverage the collective power of weak models to bypass the need to search for a single optimal model. The idea is similar to that used in multiple kernel learning.

3. PROPOSED ENSEMBLE METHODS

In this section we will describe three ensemble approaches for combining matrix factorization based on collaborative filtering models. We also provide a time complexity analysis which highlights the efficiency of each of the proposed ensemble methods. Without loss of generality, we shall be using the Singular Value Decomposition (SVD) model as our base model.

3.1 Ensemble with simple averaging

The ensemble method with simple averaging averages the prediction results of individual base models. The final prediction is an equally weighted prediction from all of the base models. Each base model is an SVD model employing a unique number of latent factors. The working of the ensemble with constant weighted average is illustrated in Fig 1. Apparently, this is the most straight-forward way to combine models. However, as we shall see in the experiment, this simple approach can outperform the more sophisticated approaches. The reason for that will be explained later.

3.2 Ensemble with non-negative linear combination

In contrast to the simple averaging ensemble, the non-negative linear combination assigns a non-negative weight to the prediction from each member of the ensemble. The weights are the coefficients of a linear regression model fitted to the predictions from the base models to explain the ground truth ratings. The gradient descent technique is applied to optimize the coefficient or weight of each ensemble instance. The predicted rating from the SVD ensemble model with non-negative linear combination is given in Equation 4.

$$\hat{r}_{ui} = c + \sum_{j=1}^{E} w^e r^e_{ui}. \quad (4)$$

r^e_{ui} is the predicted rating score on an item i for user u by the e-th based model from the total of E models. w^e is the coefficient for the e-th based model from the linear regression analysis. The term c is the intercept of the linear regression model. It is worth noting that the value of the coefficients w^e cannot be negative because the prediction of one base model might cancel out those from the other base models.

3.3 Ensemble with regularized non-negative linear combination

It is possible that in the case where there are not enough ratings to learn from, the non-negative linear combination approach might overfit the training ratings and will not generalize well. To reduce such unwanted behavior, regularization terms can be added to the linear regression model to mitigate the problem. Here, instead of fitting a non-negative linear regression model directly to the predicted ratings by the base models, we employ lasso [6] and elastic net [7] to regularize the linear regression model. The regularized losses for lasso and elastic net are given in Equations 5 and 6, respectively.

$$L_{lasso}(\mathbf{w}) = \sum_{u \in U, i \in I} (r_{ui} - \hat{r}_{ui})^2 + \lambda_1 \sum_{e=1}^{E} |w^e| \quad (5)$$

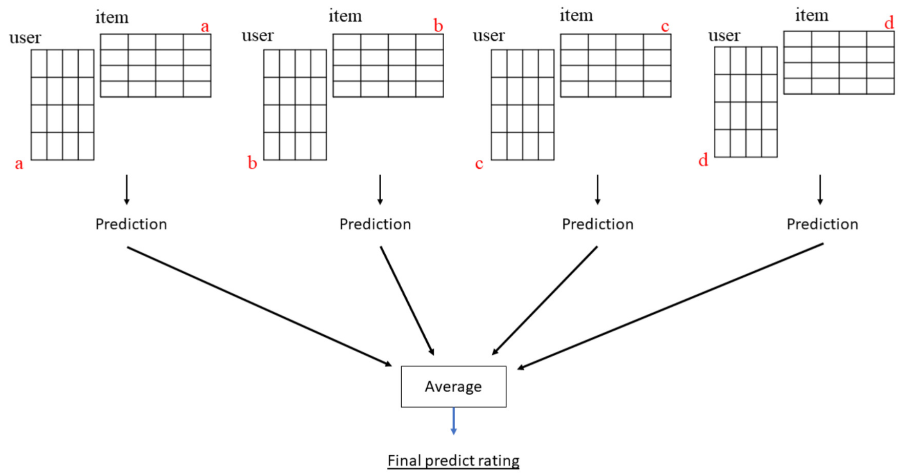

Fig.1: *Constant Weighted Average Ensemble*

$$L_{elastic}(\mathbf{w}) = \sum_{u\in U, i\in I} (r_{ui} - \hat{r}_{ui})^2 + \lambda_2(\alpha \sum_{e=1}^{E} |w^e| + \frac{1-\alpha}{2} \sum_{e=1}^{E} (w^e)^2) \quad (6)$$

After obtaining the weight vector, the prediction of the ensemble can be computed using Equation 4.

3.4 Time complexity analysis

We shall now show that performing ensemble learning and bypassing the need to select the optimal number of latent factors for a single SVD via, for example, cross validation, is more computationally efficient than single run.

3.4.1 Time complexity analysis for SVD

To analyse the time complexity, let us first recall the learning process of an SVD model with the objective function in Equation 7.

$$\min \sum_{u\in U, i\in I} (r_{ui} - \hat{r}_{ui})^2 + \lambda(b_u^2 + b_i^2 + ||p_u||^2 + ||q_i||^2) \quad (7)$$

Typically, estimating the learning parameters is carried out using the gradient descent technique with some regularization to minimize the sum of squared errors.

The generalized algorithm for the learning process to calculate all model parameters for SVD with k-latent factors tuning is summarized as Algorithm 1.

Algorithm 1 SVD with k-latent factors tuning

for $K = 1, 2, \ldots max(m, n)$ **do**
    ▷ *single SVD learning*
    Random b_u, b_i , p_u and q_i with small values.
    while terminal conditions do not met **do**
        for $u = 1, 2, \ldots, m$ **do**
            for $i = 1, 2, \ldots, n$ **do**
                $\epsilon = r_{ui} - \hat{r}_{ui}$
                for $k = 1, 2, \ldots, K$ **do**
                    $b_{uk} += \gamma(\epsilon - \lambda b_{uk})$
                    $b_{ik} += \gamma(\epsilon - \lambda b_{ik})$
                end for
                for $k = 1, 2, \ldots, z$ **do**
                    $p_{uk} += \gamma(\epsilon q_{ik} - \lambda p_{uk})$
                    $q_{ik} += \gamma(\epsilon p_{uk} - \lambda q_{ik})$
                end for
            end for
        end for
        $epoch = epoch + 1$
    end while
end for

Here, λ is regularization parameter, and γ represents of learning rate.

From the above, the time complexity of learning a single SVD model is $O(mnk)$, where m, n, and k denote the number of users, number of items, and the size latent factors, respectively. The time complexity can be rewritten as $O(mn^2)$ when $n < m$ and $k = n$, or $O(m^2n)$ when $m < n$ and $k = m$. Thus, the time complexity of the SVD learning process is cubic. In other words, the polynomial time complexity is of degree 3.

For SVD learning with the k-latent factors tuning process, the additional outer loops are taken into account for iterating to find the most appropriate size of k-latent factors. Thus, the additional outer iteration rounds for the learning process, denoted as $\mathcal{K}$ with $\mathcal{K} = \min(m, n)$, are required. Therefore, the overall time complexity of SVD learning with k-latent factors tuning process is quartic, having a polynomial time complexity of degree 4.

3.4.2 Time complexity analysis for SVD++

Next let us analyze the time complexity of SVD++, which is another possible base model. Again let us recall the objective function of the SVD++ model, as shown in Equation 8.

$$min \sum_{r_{ui} \in \hat{r}_{ui}} (r_{ui} - \hat{r}_{ui})^2 + \lambda(b_u^2 + b_i^2 + ||p_u||^2 + ||q_i||^2 + ||y_j||^2) \quad (8)$$

From the SVD++ learning process, the learning parameters are assigned by moving in the opposite direction of the gradient with a magnitude proportional to a constant γ. The generalized algorithm for the learning process to calculate all model parameters for SVD++ with k-latent factors tuning is summarized as Algorithm 2.

γ_2 denotes a learning rate of y_j. The additional learning parameters y_j, which collect the implicit influence of items rated by user u, require extra time for the learning.

Accordingly, the time complexity of a single SVD++ model is $O(mn^2k)$, which has a polynomial having time complexity of degree 4, or quartic time complexity. Additionally, the overall time complexity of SVD++ learning with a k-latent factors tuning process has quintic time complexity, which means the polynomial time complexity is of degree 5.

3.4.3 Time complexity analysis for the ensemble methods

In contrast to the learning with k-latent factors tuning process, which increases the degree of single model's time complexity to a polynomial of higher degree, the proposed ensemble approaches preserve the time complexity and also return more accurate rating predictions.

The overall time complexities for our proposed simple averaging ensembles for SVD and SVD++ are $O(mnkE)$ and $O(mn^2kE)$, respectively. E denotes the constant number of ensemble models and $E \ll \mathcal{K}$. Thus, the time complexity of the SVD and SVD ++ ensemble method with simple averaging have polynomial time complexities of degree 3 and 4, respectively.

In addition, the time complexity of the SVD and SVD++ ensemble methods with other regression models are $O(mnkE + E^2mn)$ and $O(mn^2kE + E^2mn)$, respectively. Since E is constant, the time complexity of the SVD and SVD++ ensemble methods with other regression models also have polynomial time complexities of degree 3 and 4, respectively.

From this theoretical point of view, the ensemble models' time complexity is classified into the same class of time complexity for both the primitive SVD and SVD++ learning processes. This infers that the SVD-based ensemble models will spend less time learning than the SVD-based model with a k-latent factors tuning process. The concrete results of this study are demonstrated by the empirical results shown in the next section.

4. EXPERIMENTS

4.1 Experimental protocol

The experiments are designed to investigate the following research questions: 1) Can the proposed ensemble methods effectively facilitate the task of latent factor tuning?, 2) Which of the proposed approaches is more appropriate for the task?, and 3) Can the ensemble methods improve upon the single SVD or SVD++ models in terms of running time?

To answer the first two questions we validated the proposed ensemble methods against an SVD and an SVD++ model on publicly available recommendation system testbeds. The predictive performance should highlight the benefit of the ensemble methods. We should also be able to study the comparative performance between the three ensemble approaches. In each repetition of the experiment, we tuned all hyperparameters, such as the number of latent factors and k ranging from 20 to 200, of SVD and SVD++ and λ_1, $\lambda_2 \in (0, 1)$ for the regularized linear combination ensemble via 5-fold cross validation. In the ensemble methods, the base SVD and SVD++ models require no k-factor tuning, because each member uses its own predefined k-factors. The predefined latent factors are chosen so that they cover the whole search space from 20 to 200. A diagram of ensemble learning is shown in Fig. 2.

We performed 5 repetitions of the experiment using 80/20 random train/test splitting in order to obtain the performance statistics.

4.2 Datasets

In this study, we use the MovieLens 100K [19], MovieLens 1m [19], Book crossing [20], and Filmtrust [21] datasets for benchmarking. The statistics of the datasets for the experiment are presented in Table 1.

Table 1: *Statistics of the datasets used in the experiment*

	#Users	#Items	#Ratings	Range
MovieLens-100k	943	1,682	100,000	1-5
MovieLens-1m	6,040	3,900	1,000,000	1-5
Filmtrust	1,508	2,071	35,497	0.5-4
Book Crossing	1,295	14,684	62,657	1-10

4.3 Evaluation Metrics

To evaluate the model's performance, the metrics used in this experiment were the root mean square error (RMSE) and the mean absolute error (MAE) defined in Equations (9) and (10), respectively. Residual error from both indicators is a measure of how far from the regression line data point is. Smaller values show the model's predictive capability is better.

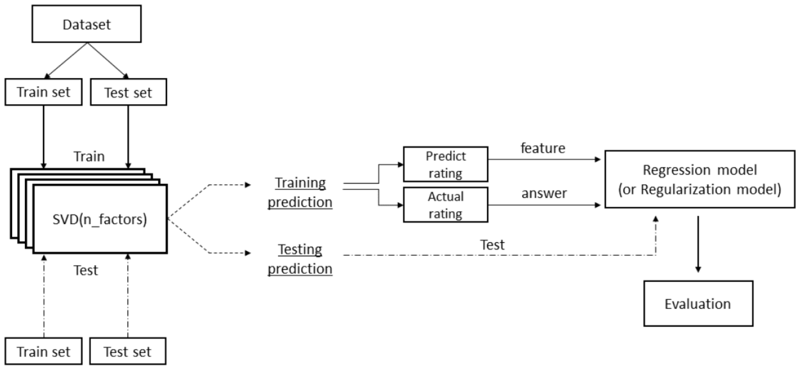

Fig.2: *The schematic diagram of the proposed ensemble methods.*

Algorithm 2 SVD++ with k-latent factors tuning

for $K = 1, 2, \ldots max(m, n)$ **do**
$\quad\quad\quad\quad\quad\quad \triangleright$ *single SVD++ learning*
$\quad$ Random b_u, b_i, p_u, q_i and y_i with small values.
$\quad$ **while** terminal conditions do not met **do**
$\quad\quad$ **for** $u = 1, 2, \ldots, m$ **do**
$\quad\quad\quad$ $\beta = \sum_{j \in I_u} y_j$
$\quad\quad\quad$ **for** $i = 1, 2, \ldots, n$ **do**
$\quad\quad\quad\quad$ $\epsilon = r_{ui} - \hat{r}_{ui}$
$\quad\quad\quad\quad$ **for** $k = 1, 2, \ldots, K$ **do**
$\quad\quad\quad\quad\quad$ $b_{uk} += \gamma(\epsilon - \lambda b_{uk})$
$\quad\quad\quad\quad\quad$ $b_{ik} += \gamma(\epsilon - \lambda b_{ik})$
$\quad\quad\quad\quad\quad$ $p_{uk} += \gamma(\epsilon q_{ik} - \lambda p_{uk})$
$\quad\quad\quad\quad\quad$ $q_{ik} += \gamma(\epsilon p_{uk} - \lambda p_{uk})$
$\quad\quad\quad\quad\quad$ **for** $j \in I_u$ **do**
$\quad\quad\quad\quad\quad\quad$ $y_j += \gamma_2(\epsilon \, |I_u|^{-\frac{1}{2}} \, q_{ik} - \gamma_2 y_j)$
$\quad\quad\quad\quad\quad$ **end for**
$\quad\quad\quad\quad$ **end for**
$\quad\quad\quad$ **end for**
$\quad\quad$ **end for**
$\quad\quad$ $epoch = epoch + 1$
$\quad$ **end while**
end for

Algorithm 3 Ensemble method

for $z = 1, 2, \ldots \mathcal{E}$ **do**
$\quad$ ...
$\quad\quad \triangleright$ *Perform single SVD or SVD ++learning*
$\quad$ ...
end for
$\quad\quad\quad\quad\quad \triangleright$ *Perform simple ensemble method*
for $u = 1, 2, \ldots, m$ **do**
$\quad$ **for** $i = 1, 2, \ldots, n$ **do** $\hat{r}_{ui} = 0$
$\quad\quad$ **for** $z = 1, 2, \ldots \mathcal{E}$ **do**
$\quad\quad\quad$ $\hat{r}_{ui} += \hat{r}_{ui}$ from z-latent factor
$\quad\quad$ **end for**
$\quad\quad$ $\hat{r}_{ui} = \hat{r}_{ui} / \mathcal{E}$
$\quad$ **end for**
end for

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (\hat{r}_{ui} - r_{ui})^2} \tag{9}$$

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |\hat{r}_{ui} - r_{ui}| \tag{10}$$

4.4 Results

In this experiment we have compared the results produced in a number of ways, such as with a simple averaging ensemble, with a non-negative linear combination ensemble, and with a regularized linear combination model. Simple Average Ensemble is an ensemble system in which voting weights are distributed equally according to the number of base models available in the system. This allows all the base models to contribute equally. On the other hand, the linear combination ensemble has a process to learn the weight variables to create conditional decision-making and give preference to a more capable base model. Regularization eliminates the problem of model overfitting, and employs both Lasso regularization and Elastic Net regularization. We used 21 models for both SVD and SVD++ variants in this experiment.

Tables 2 and 4 summarize the RMSE and MAE indicators of each of the proposed models compared to the standard single SVD and SVD++ model on all datasets. As can be observed in Table 2, averaging the predictions of multiple SVD models each operating in its own latent space yields the lowest RMSE. The elastic net ensemble of SVDs provided the lowest MAE for all the datasets. We can also see from Table 4 that averaging predictions from base SVD++ models provided the best results on both RMSE and MAE for all datasets except Book Crossing and Filmtrust, where the MAE value was inferior to the elastic net ensemble model. However, it can be seen that all of the proposed ensemble methods were superior to either of the single models, SVD or SVD++. This proves the suitability of using ensemble methods for

Table 2: *The RMSE and the MAE measurements of the SVD ensembles compared to a single SVD model.*

Dataset	MovieLen100K		MovieLen1M		Book Crossing		Filmtrust	
Model	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
SVD	0.9384	0.7392	0.8743	0.6865	1.4951	1.1335	0.7989	0.6187
SVD Average Ensemble	**0.9208**	0.7266	**0.8504**	0.6698	**1.4832**	1.1253	**0.7887**	0.6110
SVD Linear Ensemble	0.9434	0.7340	0.8682	0.6748	1.5156	1.1346	0.8135	0.6211
SVD Lasso Ensemble	0.9269	0.7316	0.8564	0.6701	1.4911	1.1290	0.7941	0.6118
SVD Elastic Net Ensemble	0.9214	**0.7260**	0.8515	**0.6673**	1.4843	**1.1240**	0.7894	**0.6093**

Table 3: *Overall improvement of SVD ensemble model over single SVD model.*

Dataset	MovieLen100K		MovieLen1M		Book Crossing		Filmtrust	
Model	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
SVD Average Ensemble	**1.88%**	1.70%	**2.73%**	2.43%	**0.80%**	0.72%	**1.28%**	1.24%
SVD Linear Ensemble	-0.53%	0.70%	0.70%	1.70%	-1.37%	-0.10%	-1.83%	-0.39%
SVD Lasso Ensemble	1.23%	1.03%	2.05%	2.39%	0.27%	0.40%	0.60%	1.12%
SVD Elastic Net Ensemble	1.81%	**1.79%**	2.61%	**2.8%**	0.72%	**0.84%**	1.19%	**1.52%**

Table 4: *The RMSE and the MAE measurements of the SVD++ ensembles compared to a single SVD++ model.*

Dataset	MovieLen100K		MovieLen1M		Book Crossing		Filmtrust	
Model	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
SVD++	0.9200	0.7221	0.8626	0.6731	1.5074	1.1368	0.7970	0.6143
SVD++ Average Ensemble	**0.9028**	**0.7113**	**0.8407**	**0.6588**	**1.4806**	1.1246	**0.7818**	0.6032
SVD++ Linear Ensemble	0.9221	0.7180	0.8624	0.6694	1.4905	1.1245	0.7995	0.6088
SVD++ Lasso Ensemble	0.9127	0.7179	0.8512	0.6646	1.4905	1.1250	0.7886	0.6058
SVD++ Elastic Net Ensemble	0.9055	0.7122	0.8446	0.6646	1.4814	**1.1214**	0.7835	**0.6025**

Table 5: *Overall improvement of SVD++ ensemble model over single SVD++ model.*

Dataset	MovieLen100K		MovieLen1M		Book Crossing		Filmtrust	
Model	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
SVD++ Average Ensemble	**1.87%**	**1.5%**	**2.54%**	**2.12%**	**1.78%**	1.07%	**1.91%**	1.81%
SVD++ Linear Ensemble	-0.23%	0.57%	0.02%	0.55%	1.12%	1.08%	-0.31%	0.90%
SVD++ Lasso Ensemble	0.79%	0.58%	1.32%	1.26%	1.12%	1.04%	1.05%	1.38%
SVD++ Elastic Net Ensemble	1.58%	1.37%	2.09%	1.26%	1.72%	**1.35%**	1.69%	**1.92%**

facilitating the tuning of a single matrix factorization model in collaborative filtering tasks.

The ensemble of multiple members, each operating in its own data space, can be seen as employing different views on the data for prediction. Although each view was not optimally tuned for the task, collectively they were able to provide sufficient information to accurately predict the ratings. This might explain why the ensemble yields better predictive results than using a single model.

When considering the linear combination and the regularized ensemble models, the linear combination one is different from the simple averaging ensemble model in the sense that the weights are adaptive and learned from the data. Consequently, the base model will immediately lose its vote when its weight approaches zero. That means that not every base model in the ensemble contributes to the final prediction but only the base models which explain the training rating well will be preferred. This results in overfitting in our experience. Therefore, we designed the regularized linear combination ensemble models to counteract this unwanted consequence. The regularizations were able to lessen the overfitting effect. Increased generalization of performance can be observed in the regularized ensemble models. From Tables 2 and 4, the lasso ensembles and elastic net ensembles show excellent results in comparison with the linear ensemble. This was due to the reduction of overfitting of the training model, resulting in better predictive results.

Tables 3 and 5 show the performance relative to a single model in percent. The positive values indicate that the respective ensemble model has better predictive performance than the standard single model.

On the other hand, if the ensemble's performance was worse, the value will be negative.

4.5 Study on the ensemble size

In the previous experiments on SVD and SVD++ ensembles, we fixed the ensemble size to 21. It is then interesting to examine the impact of the ensemble size. In this experiment, we shall study the performance of an ensemble compared to a single model as the number of base models varies. As the number of base models within an ensemble system increases, the weight of each base model should be reduced proportionally, but the diversity of the ensemble will increase due to the fact that each member uses its own k-factor. Hopefully, as diversity increases, we could see improved predictive performance.

To see this, we plotted the RMSE of SVD and SVD++ based ensembles utilizing 3, 8, 14 and 21 base models respectively. The number of latent factors for the base models were chosen uniformly in the range of 20 to 200. For example, an ensemble with 3 members consists of base models working in 20, 110, and 200 dimensional latent spaces. Also included for reference is the performance of a single model.

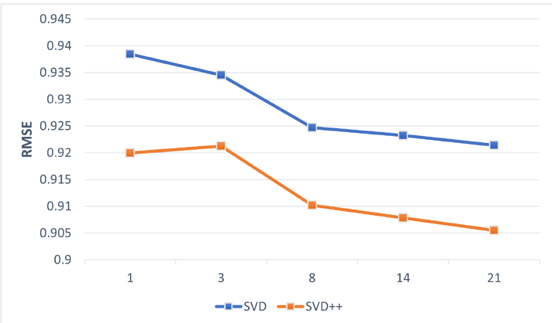

Fig.3: *MovieLen100k.*

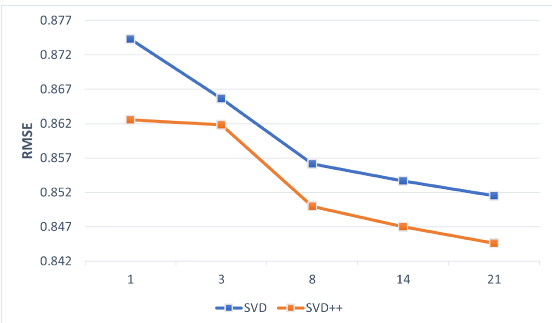

Fig.4: *MovieLen1m.*

The comparative performance is shown in Fig. 3 to 6 for MovieLen100K, MovieLen1M, Book Crossing, and Filmtrust, respectively. In each data set, it can be seen that increasing the number of base models in the ensemble system decreases the RMSE value despite the fact that the k-factor values are not necessarily the optimal ones. An ensemble with 21 base

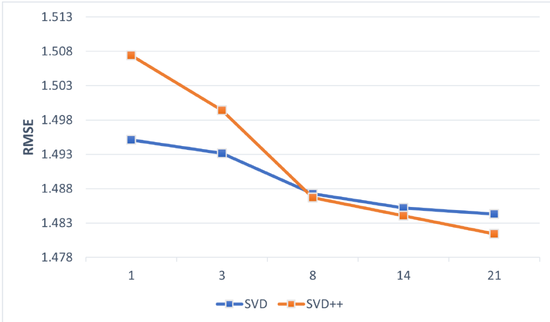

Fig.5: *Book crossing.*

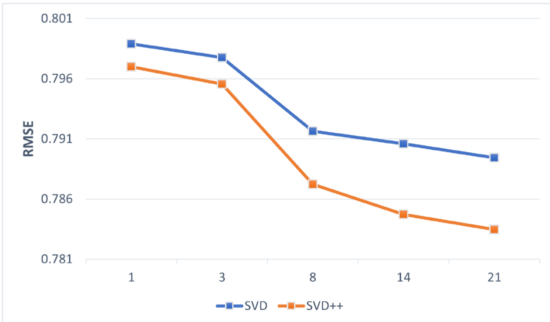

Fig.6: *Filmtrust.*

models has a lower RMSE than an ensemble system with 14 base models. The behavior occurs for every dataset verifying the assumption that having more base models with different k-factors will create diversity and hence the benefit of ensemble learning can be expected. This behavior can be observed when using SVD or SVD++ as a base learner.

4.6 Comparison with other state-of-the-art methods.

To see how well the proposed methods perform compared to the existing state-of-the-art collaborative filtering techniques, we present the predictive performances of k-NN [22], SlopeOne [23] and Non-negative Matrix Factorization (NMF) [24] together with the best performing configuration of the proposed ensemble method in Table 6. From the result, we can see that the proposed ensemble methods outperform the existing techniques in all the testbeds. The finding confirms the effectiveness of using the ensemble approach to improve the predictive performance of SVD and SVD++ models.

5. CONCLUSION

This article offers techniques for taking advantage of the power of the ensemble method. It not only facilitates the selection of hyperparameters but also improves the prediction performance of the system. Several SVD-based techniques using learning with brute-force methods for finding the best k-latent factors and the ensemble of a constant set of multiple SVD-based

Table 6: *The RMSE and MAE of the existing state-of-art techniques compared with the best results from the proposed ensemble methods.*

Dataset	**MovieLen100K**		**MovieLen1M**		**Book Crossing**		**Filmtrust**	
Model	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**
k-NN	0.9772	0.7710	0.9229	0.7275	1.8776	1.4164	0.8705	0.6564
SlopeOne	0.9460	0.7430	0.9070	0.7150	1.7537	1.3108	0.8506	0.6389
NMF	0.9630	0.7580	0.9160	0.7240	2.6519	2.2720	0.8620	0.6525
SVD Average Ensemble	**0.9208**	0.7266	**0.8504**	0.6698	**1.4832**	1.1253	**0.7887**	0.6110
SVD Elastic Net Ensemble	0.9214	**0.7260**	0.8515	**0.6673**	1.4843	**1.1240**	0.7894	**0.6093**
SVD++ Average Ensemble	**0.9028**	**0.7113**	**0.8407**	**0.6588**	**1.4806**	1.1246	**0.7818**	0.6032
SVD++ Elastic Net Ensemble	0.9055	0.7122	0.8446	0.6646	1.4814	**1.1214**	0.7835	**0.6025**

rating prediction models were experimented with to determine the performance in terms of prediction accuracy and learning time was compared.

The experimental results showed that the ensemble of a constant set of multiple SVD-based rating prediction models outperformed a brute-force method for tuning the best k-latent factors in a single SVD-based model. Thus, it can be concluded that despite the explosive growth in size of users-items rating matrix, the learning time for the ensemble of a constant set of multiple SVD-based rating prediction models proposed in this paper was increasing linearly.

The design of more appropriate regularization for the ensemble is an interesting future research direction.

References

[1] B. Walek and P. Spackova, "Content-Based Recommender System for Online Stores Using Expert System," *2018 IEEE First International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, pp. 164–165, 2018.

[2] S. Wang, G. Sun, and Y. Li, "SVD++ recommendation algorithm based on backtracking," *information*, vol. 11, no. 7, 2020.

[3] W. Shi, L. Wang, and J. Qin, "User embedding for rating prediction in SVD++-based collaborative filtering," *Symmetry*, vol. 12, no. 1, 2020.

[4] X. Zhou, J. He, G. Huang, and Y. Zhang, "SVD-based incremental approaches for recommender systems," *Journal of Computer and System Sciences*, vol. 81, no. 4, pp. 717–733, 2015.

[5] S. Zheng, C. Ding, and F. Nie, "Regularized Singular Value Decomposition and Application to Recommender System," no. 3, 2018, [Online]. Available: `http://arxiv.org/abs/1804.05090.`

[6] J. C. Huang et al., "Predictive modeling of blood pressure during hemodialysis: a comparison of linear model, random forest, support vector regression, XGBoost, LASSO regression and ensemble method," *Computer Methods and Programs in Biomedicine*, vol. 195, p. 105536, 2020.

[7] R. Harimurti, Y. Yamasari, Ekohariadi, Munoto, and B. I. G. P. Asto, "Predicting student's psychomotor domain on the vocational senior high school using linear regression," *2018 International Conference on Information and Communications Technology (ICOIACT)*, pp. 448–453, 2018.

[8] S. Funk, "Netflix Update: Try This at Home, "2006, [Online]. Available: `http://sifter.org/simon/journal/20061211.html.`

[9] A. Paterek, "Improving Regularized Singular Value Decomposition for Collaborative Filtering," *Proceedings of KDD cup and workshop*, pp. 39-42, 2007.

[10] Y. Koren, "Collaborative filtering with temporal dynamics," *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 447-456, 2009.

[11] D. Prasertttitipong and P. Sophatsathit, "An agent model for information filtering using revolutionary RSVD technique," *Chiang Mai Journal of Science*, Vol.41 No.5/2, pp. 1429-1438, 2014.

[12] P. Clerkin, C. Hayes, P. Cunningham, H. Grant, and A. Macdowell, "Automated Case Generation for Recommender Systems Using Knowledge Discovery Techniques," *Trinity College Dublin, Department of Computer Science*, pp. 7, 2002.

[13] L. Barrington, R. Oda, and G. Lanckriet, "Smarter than genius? Human evaluation of music recommender systems," *10th International Society for Music Information Retrieval Conference (ISMIR 2009)*, pp. 357–362, 2009.

[14] A. O. Afolabi and P. Toivanen, "Recommender Systems in Healthcare: Towards Practical implementation of Real-Time Recommendations to Meet the Needs of Modern Caregiving," in *Handbook of Research on Emerging Perspectives on Healthcare Information Systems and Informatics*, 2018, ch. 14, pp. 323–346.

[15] A. Azaria, A. Hassidim, S. Kraus, A. Eshkol, O. Weintraub, and I. Netanely, "Movie recommender system for profit maximization," *Pro-*

ceedings of the 7th ACM conference on Recommender systems, pp. 121-128, 2013.
[16] J. E. McDade, "A hypothesis for non-oral reading: Argument, experiment, and results," *The Journal of Educational Research*, vol. 30, no. 7, pp. 489–503, 1937.
[17] X. Guan, C. T. Li, and Y. Guan, "Matrix Factorization with Rating Completion: An Enhanced SVD Model for Collaborative Filtering Recommender Systems," *IEEE Access*, vol. 5, pp. 27668–27678, 2017.
[18] P. Classification, *Using Ensemble Methods*, p. 3, 2013.
[19] F. M. Harper and J. A. Konstan, "The MovieLens Datasets: History and Context," *ACM Transactions on Interactive Intelligent Systems*, vol. 5, no. 4, pp. 1–19, 2016.
[20] F. Zhang, "Improving recommendation lists through neighbor diversification," *2009 IEEE International Conference on Intelligent Computing and Intelligent Systems*, vol. 3, pp. 222–225, 2009.
[21] G. Guo, J. Zhang, and N. Yorke-Smith, "A novel bayesian similarity measure for recommender systems," *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, pp. 2619–2625, 2013.
[22] R. Ahuja, A. Solanki, and A. Nayyar , "Movie recommender system using k-means clustering and k-nearest neighbor," *2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pp. 263–268, 2019.
[23] D. Lemire, and A. Maclachlan, "Slope one predictors for online rating-based collaborative filtering," *Proceedings of the 2005 SIAM International Conference on Data Mining*, pp. 471–475, 2005.
[24] X. Luo, M. Zhou, Y. Xia, and Q. Zhu, "An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1273–1284, 2014.

Vasin Jinopong was born on April 29, 1997 in Chiang Mai, Thailand. He graduated from the Chiang Mai University, with a B.Sc degree (Chemistry Science) in 2019 and M.Sc (Data Science) in 2022.

Dussadee Praserttitipong is an Assistant Professor at the Department of Computer Science, Faculty of Science, Chiang Mai University. She received his B.Sc. (1995) and M.Sc. (1999) degrees in Computer Science from Chiang Mai University, Thailand and her Ph.D. (2012) degree in Computer Science from the Chulalongkorn University, Thailand. Her current research focus on recommender system and educational data mining.

Jakramate Bootkrajang is an Assistant Professor at the Department of Computer Science, Faculty of Science, Chiang Mai University. He received his BSc (2007) and MSc (2009) degrees in Computer Science from Seoul National University, Republic of Korea and his PhD (2013) degree in Computer Science from the University of Birmingham. His interests concern statistical machine learning and probabilistic modelling of data. His current research focuses on supervised learning from unreliable annotated data.