Proceedings of the 52nd Hawaii International Conference on System Sciences | 2019

Divergence Based Non-Negative Matrix Factorization for top-N Recommendations

Md. Enamul Haque, SM Zobaed, Mehmet Engin Tozal, Vijay Raghavan
School of Computing and Informatics
University of Louisiana at Lafayette
Lafayette, LA 70504 USA
{enamul, sm.zobaed1, metozal, raghavan}@louisiana.edu

Abstract

Personalized top-N recommendation algorithms are among the most effective techniques providing customized suggestions in information retrieval applications. Most of the current methods construct personalized recommendations based on various loss functions such as pairwise ranking loss and point-wise recovery loss. In this paper, we propose a personalized top-N recommendation method based on non-negative matrix factorization with divergence as a point-wise ranking loss function. Our method finds the latent factors from the existing data to improve recommendation predictions. We formulate the learning problem with regularized divergence as a constrained non-convex minimization problem and develop a projected gradient descent optimization algorithm to solve the divergence problem. We evaluate our approach using six personal recommendation task related datasets by employing root mean squared error (RMSE) and hit rate (HR). Our experimental results demonstrate improved RMSE and HR for most of the datasets.

1. Introduction

Many companies have both online and traditional brick-and-mortar presence, because it helps to reach a larger and more diverse customer base. These companies use recommendation systems or engines to attract customers who are more likely to purchase their products and services. In other words, recommendation systems are employed to personalize user experience on different media. These systems suggest users what to purchase, where to eat, which movie to watch, which songs to listen or even who to be friends with. Although individual users' preferences vary, we can find preference patterns by analyzing their historical data. Typically, people like things that are similar to their other likings. They also tend to have similar tastes as other people who are "closer" to them. Recommendation systems capture these patterns using different learning algorithms to facilitate user-level item suggestion. E-commerce, social media, video streaming systems, and online news platforms use their own recommendation systems to improve their users' overall experience. Similarly, other industries have been employing recommendation engines to improve their customer services. Intelligent traffic and navigation system for personal use (waze[1]), IBM traffic control system, and airline route navigation for efficient fuel consumption are few examples from the transportation industry using recommendation systems.

Informally, top-N recommendation identifies a set of N items that will be of interest to a certain user. Over the past decade, a variety of recommendation techniques have been introduced for top-N recommender systems [1]. These techniques are categorized into three groups based on user behavior and similarities: collaborative, content-based, and hybrid filtering. Collaborative filtering method includes neighborhood, model, and ranking based methods. Neighborhood based methods identify similarities among user-item pairs [2–5]. For instance, item based k-NN method first identifies a set of similar items with respect to the user purchase history, and then recommends top-N items. Model based methods such as matrix factorization (MF) use latent user-item factors to find unknown relations between user and item pairs. Finally, ranking methods for top-N recommendation such as Bayesian personalized ranking (BPR) criteria [6] are used to differentiate between user-purchased items and the rest of the items. Content-based filtering systems generate recommendations based on item descriptions and user interests on the items [7]. Hybrid recommendation systems combine both collaborative and content-based approaches.

Matrix factorization is a common technique used in top-N recommendation algorithms. Most of these algorithms use different matrix norms to compute the distance between original and reconstructed matrices.

[1] www.waze.com

URI: https://hdl.handle.net/10125/59485
ISBN: 978-0-9981331-2-6

In this paper, we present the effectiveness of K-L divergence for top-N recommendations instead of the traditional distance measures. We formulate the learning problem with regularized divergence as a constrained non-convex minimization problem, and solve it using projected gradient descent algorithm, called as top-N recommendation using Matrix Factorization (TNMF). We apply our approach on six well-known datasets and compare with three state-of-the-art methods. The experimental evaluations demonstrate improved root mean square error (RMSE) in the majority of the datasets. Additionally, we present hit-rate with respect to different latent factors and iterations. In general, we observe improved hit-rate for higher latent factors.

Our proposed technique can be extended in different domains such as e-commerce applications, on-line streaming services, and academic institutions. For instance, TNMF can be used to recommend a student a list of suitable universities to apply for graduate studies based on his profile evaluation (e.g., GPA, GRE, TOEFL, and Publications).

The remainder of this paper is organized as follows. Section 2 presents some state-of-the-art recommendation methods as background. Section 3 defines our proposed problem and solution formulation using K-L divergence based user-item preference loss function and projected stochastic gradient descent optimization. Section 4 presents experimental settings along with dataset descriptions. Section 5 presents evaluation results of our proposed approach and comparisons with three state-of-the-art top-N recommendation methods. Finally, section 6 concludes the paper.

2. Background

Different methods have been developed in the literature for top-N recommendation systems [8]. In recent years, top-N recommendation systems have been used in a number of different applications such as recommending products which a customer will most likely buy, advertisements that a user is likely to click, and identifying webpages of interest [9–12]. Based on user-item interactions, there are three main classes of recommendation systems: collaborative, content-based, and hybrid filtering. Collaborative filtering systems generate recommendations based on crowd-sourced input [13, 14]. They recommend items based on user behavior, and similarities between users. An example is Google PageRank, which recommends similar web pages based on different methods such as web pages' back links. Most of the collaborative filtering based studies generally focus on the two dimensional user-item problem [15]. Koren et al. claimed that Matrix Factorization (MF) techniques perform better than traditional nearest neighbor techniques [16]. Matrix Factorization characterizes both users and items based on a set of latent factors, also denoted as features, to represent the relationships between users and items. These factors are determined from the patterns of different item ratings [17]. Then, two low rank matrices are formed that represent the relationship between users (or items) and the set of features [15].

A new algorithm for collaborative filtering is proposed in [18], which explicitly optimizes the Area Under the Curve (AUC) with improved performance on the MovieLens dataset. However, the drawback of this approach is its complexity. The algorithm cannot be promptly applied to large datasets as it requires the optimization of n quadratic problems where each one is defined over m variables.

Two different algorithms, designed for Non Negative Matrix Factorization (NMF), are analyzed in [19]. They vary only slightly in the multiplicative factor used in the update rules. One algorithm is responsible to minimize the traditional least squares error and the other minimizes the generalized Kullback-Leibler divergence. The algorithms can also be interpreted as diagonally rescaled gradient descent, where the rescaling factor is optimally chosen to ensure convergence.

Generally, content-based filtering systems generate recommendations based on items' description and user's interests on the items [7]. Such systems are widely used in several domains such as generic web contents, news articles, movies, and serials. For example, Pandora uses content-based filtering to make its music recommendations. Additioanlly, in [7], the author discusses the importance of data representation for semi structured data. The authors claim that content-based systems do not provide satisfactory recommendation if the content does not have sufficient distinguishable data that help to segregate items liked by a user from his unliked items. Hybrid recommendation systems combine both collaborative and content-based approaches. They help to improve recommendations that are derived from sparse datasets. Netflix is a prime example of a hybrid recommender.

On the other hand, we can categorize the recommender systems into two major groups based on the loss functions used. The first category uses point-wise comparison losses. For example, the sparse linear method (SLIM) was developed to recover the missing recommendation entries [20]. Collaborative filtering methods also fall into this category. Pairwise ranking methods, on the contrary, produces top-N recommendations by optimizing preference structure

consistency between original and reconstructed recommendation matrices [18, 21–24].

Pan et. al. proposed a relaxed assumption of pairwise preferences over item-sets, which defines a user's preference on a set of items instead of on a single item in [23]. Park et. al [24] proposed a large-scale non-convex implementation (AltSVM), which trains a factored form of the matrix via alternating minimization. According to their presented result, this algorithm performs well to large problem settings. Moreover, the proposed solution outperforms Cofirank [21] and Robirank [25].

In this paper, we present a personalized top-N recommendation method based on non-negative matrix factorization with divergence as a point-wise ranking loss function. Our proposed approach falls in the collaborative filtering category, as we use matrix factorization to construct a recommendation score for the users from unseen items.

3. Problem Statement

In this section, we present regularized non-negative matrix factorization (NMF) [19] for top-N recommendation tasks which minimizes the divergence between original and reconstructed recommendation matrices. We assume that the factored matrices represent the latent factors (features) for both users and items that can expose users' behaviors. The basic NMF problem can be represented as follows. Given a non-negative matrix $R \in \mathcal{R}_{+}^{m \times n} (R \geq 0)$ and a number of latent factors λ, find two non-negative matrices $X_{\lambda \times m}$ and $Y_{\lambda \times n}$ such that $R \approx X^{\top} Y$ where $\lambda \leq \min(m, n)$.

For our particular problem, we consider a rating matrix $R^{m \times n}$ with m users and n items where R_{ij} refers to the rating of item j given by user i. Additionally, the matrix R is very sparse as there are only a few ratings for each user with respect to the total items. Our aim is to fill the missing rating values for all items in R and provide top-N items for each user to choose from. We present a generic framework for a top-N recommendation task and our solution formulation for regularized divergence minimization using projected Stochastic Gradient Descent (SGD) method. Note that, the optimization can be solved using alternating least squares (ALS) method as well.

3.1. A Generic Learning Framework for top-N Recommendations

For a given input user-item rating matrix R, a generic framework for learning the reconstructed matrix is performed by minimizing the regularized loss function presented in Equation 1.

$$\arg\min_{\theta} \mathcal{L}(R, \hat{R}(\theta)) + R(\theta) \tag{1}$$

where $\hat{R} = X^{\top} Y$ denotes the reconstructed recommendation matrix with parameter θ, $\mathcal{L}(\cdot)$ denotes a convex reconstruction loss function with $R(\cdot)$ as a regularization function. $\mathcal{L}(\cdot)$ can be replaced by different loss functions with reconstruction parameter θ. An example of such case is the AltSVM [24] that uses non-convex optimization problem formulation for the user-item preference completion presented in Equation 2.

$$\min_{U,V} \sum_{(i,j,k) \in \Omega} \mathcal{L}(Y_{ijk}(X) \cdot U_i (V_j - V_k)^{\top}) + \frac{\beta}{2}(||U||_F^2 + ||V||_F^2) \tag{2}$$

where Y_{ijk} is a function of X such that $Y_{ijk} = 1$ when user i prefers item j over item k in X and $Y_{ijk} = -1$ otherwise.

3.2. Divergence Function

Most matrix factorization algorithms use some form of distance function to measure the difference between original and reconstructed matrices. Standard Euclidean distance ($\mathcal{L}$) as the distance measure between original (R) and reconstructed matrices (X, Y) is presented in Equation 3.

$$\mathcal{L} = \|R - X^{\top} Y\|_F^2 \tag{3}$$

In our problem formulation, we assume randomness in continuous rating distribution for users. In order to compute the differences between users' rating distribution, a commonly used measure with convexity properties is the Kullback-Leibler divergence [26]. Essentially, KL divergence is an information-based measure of disparity. Therefore, we use Generalized Kulback-Leibler divergence and user-item regularization to find the divergence between the matrices using the minimization function in Equation 4.

$$\arg\min_{X,Y} \sum_{(i,j) \in \Upsilon} \mathcal{L}\left(R_{ij} \log \frac{R_{ij}}{(X^{\top} Y)_{ij}} - R_{ij} + (X^{\top} Y)_{ij} \right) + \frac{\beta}{2}\left(\|X\|_F^2 + \|Y\|_F^2 \right) \tag{4}$$

where $\mathcal{L}$ is considered as max-margin hinge loss, R denotes the original matrix, X denotes the user matrix, Y denotes the item matrix, β denotes the regularization parameter, and Υ refers to the set of user-items that has non-negative ranks assigned in R.

3.3. Optimization Algorithm

Equation 4 presents a non-convex optimization function for our top-N recommendation using matrix Factorization (TNMF) approach. The function is convex with respect to either the entries of the matrix X or the matrix Y, but not both [19]. Hence, finding the global minimum is not feasible for this function. Nevertheless, there exists several numerical optimization methods that can be applied to find local minimum. Gradient descent is the simplest among them with the cost of slow convergence. Other faster techniques such as conjugate gradient methods are complex to implement compared to gradient descent. We use projected Stochastic Gradient Descent (SGD) algorithm to solve the problem. As the function is non-convex, we fix each of the variables once and take partial derivatives to formulate standard gradient descent optimization. Later, we project the updated user/item variables onto the feasible set. The overall learning algorithm using projected SGD is given in Algorithm 1.

The partial derivative of Equation 4 is set to zero to solve for the optimal user/item vector for the loss or divergence function. The general formula for updating variable X_i with respect to the loss function $\mathcal{L}$ in the SGD for convergence is given in Equation 5.

$$X_i^{new} \leftarrow X_i^{old} - \eta \frac{\partial \mathcal{L}}{\partial X_i} \tag{5}$$

where X_i^{new}, X_i^{old}, and η refer to current user component, old user component, and learning rate, respectively.

In our specific problem formulation, we take partial derivative of our divergence function (Equation 4) with respect to user component X_u and item component Y_v, respectively.

$$\frac{\partial \mathcal{L}}{\partial X_u} = \sum_{i=1}^{n} y_{ui} \frac{R_{ui}}{(X^\top Y)_{ui}} - \sum_{i=1}^{n} Y_{ui} + \beta X_u^\top \tag{6}$$

$$\frac{\partial \mathcal{L}}{\partial Y_v} = \sum_{i=1}^{m} x_{vi} \frac{R_{vi}}{(X^\top Y)_{vi}} - \sum_{i=1}^{m} X_{vi} + \beta Y_v^\top \tag{7}$$

Finally, we get our desired gradient update rules for each iteration using Equations 8 and 9 .

$${X_u}^{new} \leftarrow {X_u}^{old} - \eta \left(\sum_{i=1}^{n} y_{ui} \frac{R_{ui}}{(X^\top Y)_{ui}} + \sum_{i=1}^{n} Y_{ui} - \beta X_u^\top \right) \tag{8}$$

$${Y_v}^{new} \leftarrow {Y_v}^{old} - \eta \left(\sum_{i=1}^{m} x_{vi} \frac{R_{vi}}{(X^\top Y)_{vi}} + \sum_{i=1}^{m} X_{vi} - \beta Y_v^\top \right) \tag{9}$$

The learning rate η in the SGD update is typically set to a small fixed number. We use $\eta = 1 \times 10^{-6}$ and $\beta = 1 \times 10^{-9}$ in our experiments. The overall learning algorithm using projected SGD is given in Algorithm 1.

Algorithm 1: Projected Stochastic Gradient Descent

Input: $\eta > 0, \beta > 0$, initialize $X_{\lambda \times m} = \mathbf{0}$ and $Y_{\lambda \times n} = \mathbf{0}$
Output: X,Y
Set: latent factors array, λ
repeat
1. Randomly select i, j
2. Update X_u by using (8)
3. Project the updated X_u onto the feasible set: $X_u = \max(0, X_u)$
4. Update Y_v by using (9)
5. Project the updated Y_v onto the feasible set: $Y_v = \max(0, Y_v)$

until *converge*;
return X,Y

4. Experimental Settings

In this section, we present experimental evaluation of our proposed approach using root mean squared error (RMSE) and hit rate (HR). We set N as 10 for top-N recommendations in all experimental results reported in this paper We use datasets from different domains to establish the effectiveness of our approach.

4.1. Datasets

We evaluate the performance of our method using six different real datasets: *Yahoo!* music rating v1.0, *Yahoo!* movie rating v1.0, *Movielens-100k (ml-100k)* [27], *Netflix*, *Book crossing*, and *Jester* datasets. The dataset statistics are presented in Table 1. We choose different datasets for building our model with a wide variety of ratings from different domains. *Netflix*, *Yahoo! Movie*, and *Movielens* are related to the movie ratings. *Book crossing* is related to book ratings and *Jester* is a jokes review datasets. We also consider selecting datasets based on user interest. *Netflix* and *Yahoo! Music* are the most popular among the selected. In addition, *Yahoo! Movie* and *Jester* are not commonly used. Next, we briefly discuss each of the dataset properties.

Yahoo! music dataset contains ratings for music from users during normal interactions with *Yahoo!* music services. This dataset represents a snapshot of the Yahoo! Music community's preferences for various musical artists. The ratings range from 0 to 5 in this dataset. We can interpret the lowest rating as user dislike. The dataset also contains missing entries which we replace with 0 at the beginning. Note that Equation 4 is defined on user-item pairs that has actual ratings in the dataset. For example, one user x might dislike song a, but user y might like that song.

Yahoo! movie contains movie ratings from users. This dataset contains a small sample of the Yahoo! Movies community's preferences for various movies. We created a small arbitrary sample of ratings for 2309 users and 2380 movies to test our proposed approach. The ratings range from 0 to 5 in this dataset as well.

Table 1: Dataset statistics

Dataset	User	Item	Sparsity	Transact
Yahoo! music	268	4641	1.03%	12791
Yahoo! movie	2309	2380	0.18%	10136
ml-100k	943	1682	6.30%	100000
Netflix	83539	22	5.44%	100000
Book crossing	2582	24009	0.02%	12102
Jester	3000	100	71.01%	213037

ml-100k data sets are collected by the GroupLens Research Project at the University of Minnesota. The dataset consists of ratings for movies from different users. As this dataset is quite smaller compared to the other movie ratings data used in our experiment, we do not take a smaller sample for the experiments.

Netflix data contain more than 100 millions of ratings from 480 thousand arbitrarily selected viewers over 17 thousand movies. The dataset comprises of movie ratings provided by users where the ratings range from 1 to 5 and missing entries represent unseen movies by users. We consider the missing entries as zero which can be treated as the user has not seen the movie. Therefore, we want to predict whether a user should be recommended that movie for future watch list or not.

Book crossing contains book rating information from various users where the ratings are between 1 and 10. As the dataset is extremely sparse, we remove the book titles with fewer than 10 ratings. Additionally, users with at least 10 ratings are kept for the sampling. Again in this scenario, we took a sample ratings of 2582 users and 24009 books.

Jester dataset contains joke recommendation information for over 4.1 million continuous ratings of 100 jokes from 73,421 users. The major difference between this dataset and the others is that it contains ratings between -10 to +10. Although we transformed the ratings into 1 to 5 scale, the continuous property was preserved to see how our method performs.

Table 1 shows detailed information of the datasets used in our experimental evaluation. We included two additional information: sparsity, and transact for each of the datasets that explain how sparse the data set is, and the number of nonzero entries in the matrix, respectively.

4.2. Evaluation Methodology

We use root mean squared error (RMSE) and hit rate to evaluate our top-N recommendation system with matrix factorization (TNMF). As our matrix factorization (MF) and divergence based error function computes the differences between rating distributions for each user and movie interchangeably, we use root mean squared error to evaluate our method and compare with recent works. On the other hand, we use hit-rate to explain the performance of our method as well.

Root mean squared error of two rating matrices, $\hat{R}$ and R is defined as

$$RMSE = \sqrt{\frac{1}{r}\sum_{(u,i)}(\hat{R}_{ui} - R_{ui})} \tag{10}$$

where $\hat{R}_{ui}$ and R_{ui} refers to predicted and actual ratings of user-item pairs, respectively. r refers to the size of the test set.

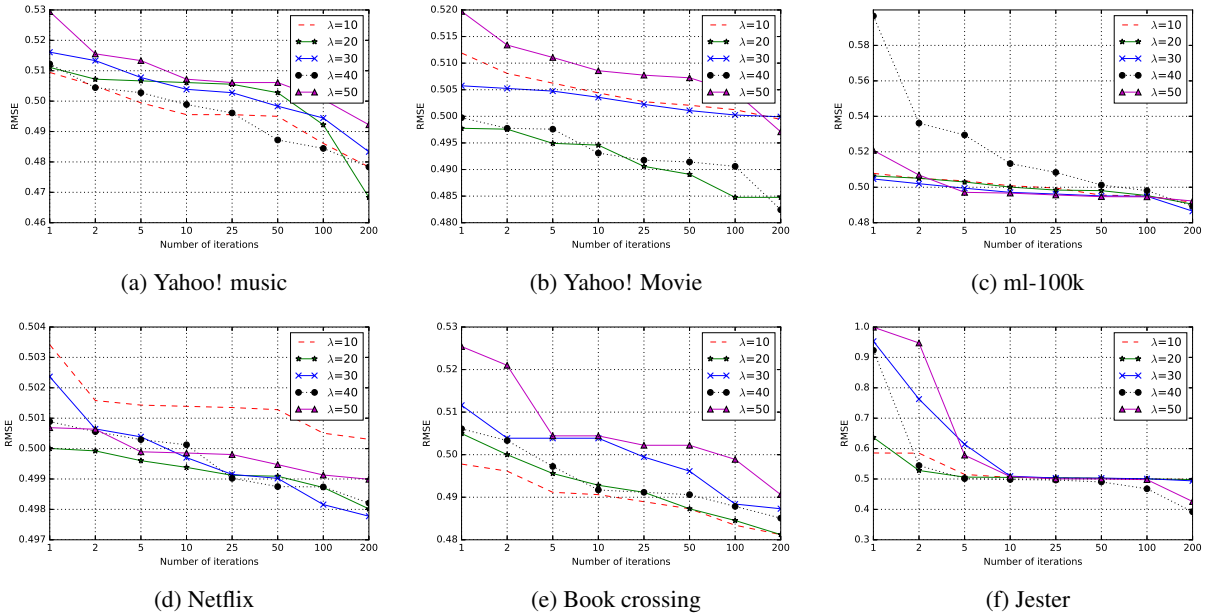
The recommendation quality is measured by the hit-rate [2]. The metric directly quantifies the performance of the model based on user feedback. Ning et. al. mentions that this is one of the most meaningful metrics for top-N recommendation task evaluations [20]. Hit rate (HR) is defined as

$$HR = \frac{\#hits}{\#users} \tag{11}$$

where $\#hits$ refer to the number of users whose item in the testing set is contained (i.e., hit) in the size-N recommendation list, and $\#users$ is the total number of users [28]. An HR value of 1.0 means that the algorithm is able to always recommend the proper items correctly, whereas an HR value of 0.0 indicates that the algorithm is not able to recommend any of the proper items.

4.3. Comparison Algorithms

In this part, we consider those approaches that perform better on sparse and imbalanced datasets



(a) Yahoo! music (b) Yahoo! Movie (c) ml-100k

(d) Netflix (e) Book crossing (f) Jester

Figure 1: RMSE scores for Yahoo! music, movies, ml-100k, Netflix, book crossings, and Jester datasets, respectively.

such as *Netflix* dataset and use matrix factorization as a recommendation technique. Therefore, we compare the performance of the proposed method with three state-of-the-art top-N recommendation algorithms, including probabilistic matrix factorization (PMF) [29], Bayesian probabilistic matrix factorization (BPMF) [30], and Large-scale parallel collaborative filtering for the Netflix prize (ALS-WR) [31].

PMF. The goal of this method is to present probabilistic algorithms that scale linearly with the number of observations. The method is extended to include an adaptive prior on the model parameters and show how the model capacity can be controlled. Additionally, a constrained version of PMF model is introduced in the paper that assumes the users who have rated similar sets of movies are likely to have similar preferences. This method works well on very large datasets and deals with users who have very few ratings.

BPMF. A fully Bayesian treatment of Probabilistic Matrix Factorization is presented by placing hyper-priors over the hyper-parameters and using Markov Chain Monte Carlo (MCMC) methods to perform approximate inference. This algorithm also demonstrates that Bayesian PMF (BPMF) models achieve better recommendation accuracy compared to the MAP-trained PMF models.

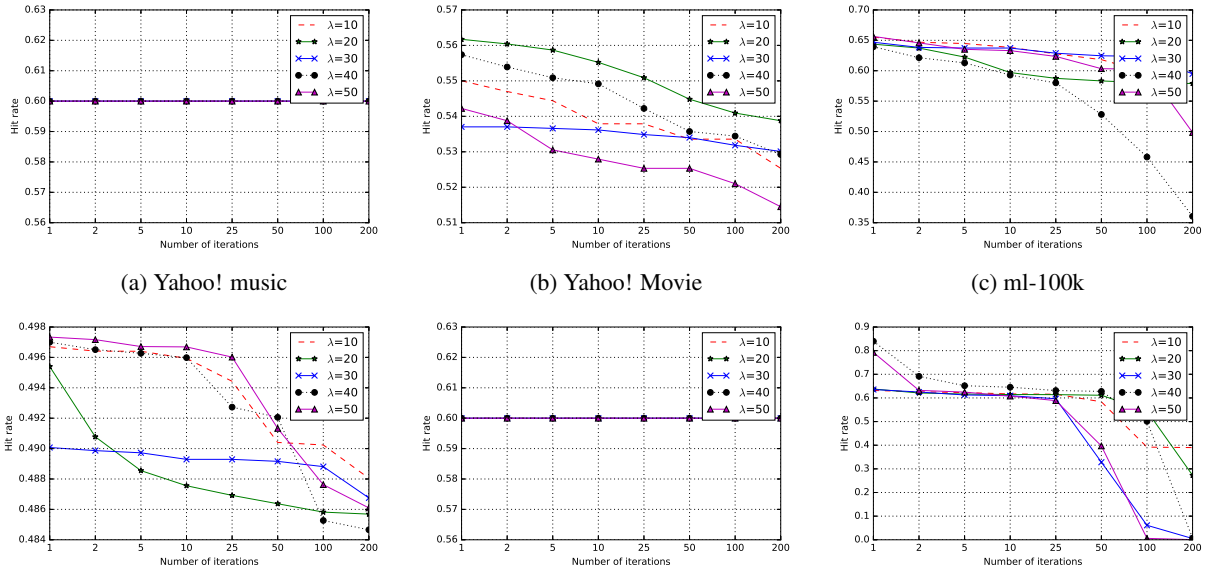
ALS-WR. Most of the collaborative filtering based recommender systems face two problems related to scalability and sparseness of the user profiles. ALS-WR (Alternating Least Squares with Weighted-λ-Regularization) method is designed as a parallel algorithm for Netflix Prize challenge. It is reported that the performance of ALS-WR monotonically increases with respect to the number of features and the number of ALS iterations.

5. Experimental Results

In this section, we present experimental results for our proposed approach and three state-of-the-art approaches. Then, we compare and contrast our method (TNMF) with those three approaches.

While performing TNMF on six datasets we used latent factor parameter, λ as 10,20,30,40, and 50. Although the increased factorization creates opportunity for better performance, the model complexity becomes higher. To make a trade-off between these two performance aspects, we suggest using λ between 20 and 40 for different data domains. In all our experiments, we empirically choose η and β as very small numbers, 1×10^{-6} and 1×10^{-9}, respectively. Note that the parameters can be efficiently optimized using naïve or brute-force grid-search method. We envision to use those in our future improvement.

For PMF method, we use default parameters mentioned in the original implementation where the number of features is 10, ϵ=1, λ=0.1, and momentum=0.8. Similarly, we kept the suggested parameters for BPMF and ALS-WR while running the comparison part in our experiments. In the following, we first demonstrate the



(a) Yahoo! music (b) Yahoo! Movie (c) ml-100k

(d) Netflix (e) Book crossing (f) Jester

Figure 2: Hit rate for Yahoo! music, movies, and ml-100k, Netflix, book crossings, and Jester datasets, respectively.

performance of TNMF. Next, we compare TNMF with other approaches.

5.1. TNMF Performance

In Figure 1, we discuss RMSE for our proposed method, TNMF, with respect to the numbers of iterations and latent factors. In all cases, we notice significant improvement over RMSE with respect to increased iterations. In case of Yahoo! music dataset, the latent factor 10 performs better compared to the other factors. TNMF creates low dimensional matrices from Yahoo! music data. As a result, the processing complexity is quite lower compared to other factors. In case of Yahoo! movie dataset, λ=30 performs better compared to other factors. ml-100k dataset exploits consistent RMSE throughout the iterations for all the factors except 40. On the other hand, Netflix data shows that higher latent factors perform better. Book crossing and Jester datasets exploit lower factors to be better for producing low RMSE values because of the transformed rating values.

Next, we explain the hit rate for our proposed approach for the same six datasets using Figure 2. We expect a consistent hit-rate with respect to different iterations from a good recommendation method. Yahoo! music dataset exploits uniform hit rate for all the factors used. On the contrary, Yahoo! movie performs better in terms of hit-rate when latent factor is 30. In this case, we notice a consistent hit rate for almost all iterations. ml-100k, on the other hand, produces acceptable hit rate for the first half of the iterations for all factors. Similar outcome is noticed for Netflix dataset using factor parameter 30. For Book crossing and Jester data, hit-rate exploits reverse effect from one another. Firstly, Book crossing shows flat hit rate for all factors. Jester shows inconsistent hit rate for all factors. In all cases, we intentionally removed 10 ratings from each users ratings profile and matched them after the prediction step. Note that the matching is not performed using exact similarity, instead, we define a threshold scheme that enables a soft assignment and comparison between element-wise rating entry for the previously removed ratings.

5.2. Comparison

In the following, we first present RMSE for probabilistic matrix factorization approach using Figure 3. PMF is proposed to scale linearly with the increase in user and item observations. We noticed in the earlier results that our method does not work well on Netflix data. This is due to the highly sparse structure of the dataset. PMF performs very well on large, sparse, and imbalanced data as well. In all datasets, PMF performs a smooth transition by decreasing RMSE with the increase in iterations. However, the Netflix dataset shows an interesting property, where RMSE does not decrease linearly, instead, it remains steady. RMSE for Yahoo! music and Jester converges very quickly compared to

Table 2: RMSE with standard deviation for different datasets and methods.

Methods	ml-100k	yahoo! music	yahoo! movies	Netflix	Jester	Book X
TNMF	**0.522±0.032**	0.496±0.011	0.493±0.005	0.500±0.001	**0.539±0.151**	**0.494±0.007**
PMF	1.013±0.071	0.424±0.001	0.142±0.001	0.825±0.010	1.270±0.223	1.098±0.001
ALS-WR	0.703±0.022	**0.292±0.003**	0.143±0.001	**0.186±0.034**	0.898±0.021	0.898±0.021
BPMF	0.782±0.040	0.299±0.011	**0.137±0.002**	0.323±0.025	0.913±0.023	0.913±0.023

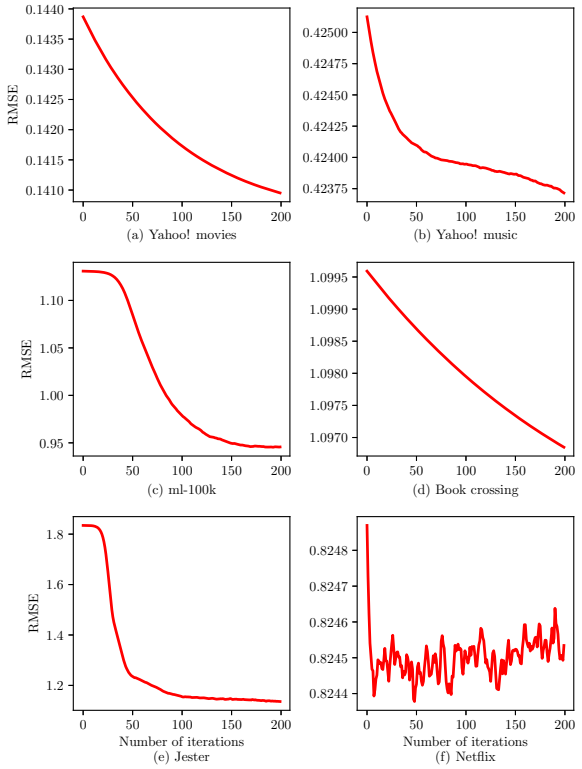


Figure 3: RMSE for Yahoo! movies, music, ml-100k, book crossings, Jester, and Netflix datasets, respectively for PMF method.

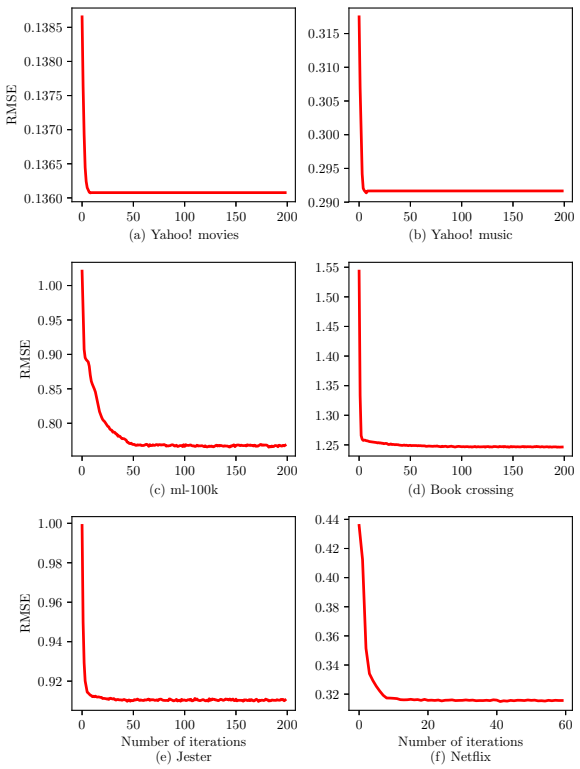


Figure 4: RMSE for Yahoo! movies, music, ml-100k, book crossings, Jester, and Netflix datasets, respectively for BPMF method.

other datasets as well for PMF. On the other hand, RMSE for Book crossing data takes longer to converge due to high sparsity.

Next, we present RMSE for Bayesian probabilistic matrix factorization (BPMF) approach using Figure 4. BPMF is a variant of PMF where Markov Chain Monte Carlo method is used to train the model. This method is very efficient in terms of convergence. Although the convergence is faster, the RMSE is not always better than our approach. However, if we want quick convergence and ignore the RMSE parameter, this method turns out to be very powerful and efficient.

In Figure 5, we present the performance of ALS-WR for the six datasets where alternating least squares with weighted regularization scheme is used to solve the optimization problem. In ALS-WR, the user and item components are fixed alternatively at each iteration to minimize the least squares objective function. The experimental results show a similar behavior in RMSE compared with BPMF. All datasets except Yahoo! movie presents early convergence in RMSE. In our proposed approach we used SGD (Algorithm 1) as an optimization approach to minimize the loss function. In case of ALS-WR, the alternating least squares is used as the optimization approach where the update-indices are selected randomly at every iteration until the

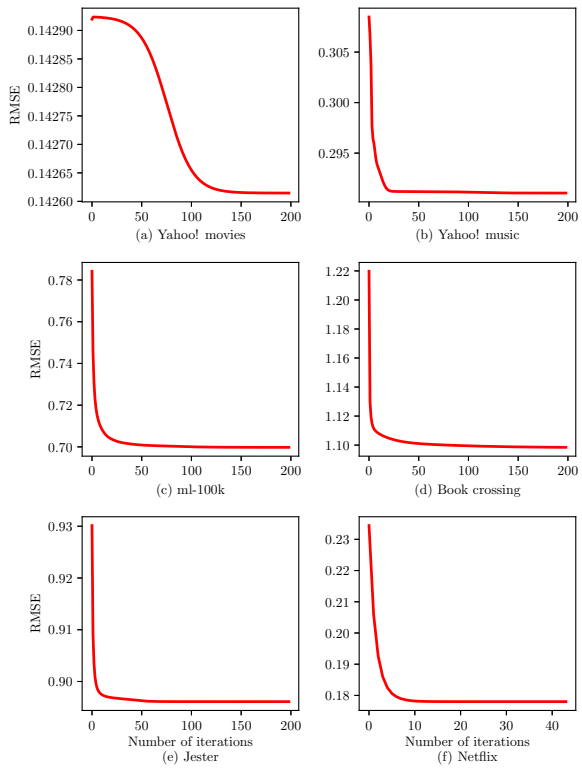


Figure 5: RMSE for Yahoo! music, movies, ml-100k, Netflix, book crossings, and Jester datasets, respectively for ALS-WR method.

convergence is ensured.

In the following, we present a direct comparison of our approach with the three state-of-the-art approaches using mean RMSE with standard deviation in Table 2. Our method (TNMF) performs better for ml-100k, Book crossings, and Jester data. ALS-WR performs better in both Yahoo! music and Netflix. On the other hand, BPMF performs better in Yahoo! movies. We include standard deviation with the mean RMSE to show that the recommendation performance has a very little or insignificant variance. The mean RMSE is computed using 200 iterations for every method. The major reason for TNMF to perform poorly compared to other approaches on Yahoo! music, movie, and Netflix datasets is dimensionality (represented by latent factor parameter) reduction by matrix factorization and individual dataset structure such as different rating scales (*e.g.,* 0-5, 1-5, 1-10) and missing entries. Note that it is not possible to perform matrix multiplication with incomplete or missing entries. Thus, we replace those entries with zero values. As a result, both original and replaced zeros are interpreted as missing entries that causes biased learning during SGD optimization. Another dimension that we investigated is the correlation between matrix sparsity and rating range. Our results do not provide any conclusive arguments on the impact of correlation between matrix sparsity and rating range over the mean square error. In our experiments, the latent factors are considered between 10 and 50. The ranking performance of our approach varies for different datasets because matrix factorization is a lossy transformation and it's efficiency depends on particular task and the number of latent factors (usually between 5 and 100).

6. Conclusions

In this paper, we propose a personalized top-N recommendation method based on non-negative matrix factorization with divergence as a point-wise ranking loss. Our proposed method finds the latent factors from the existing data to improve recommendation predictions. We formulate the learning problem with regularized divergence as a constrained non-convex minimization problem. We develop projected gradient descent optimization algorithm to solve the divergence problem. We evaluate our proposed approach on a set of personal recommendation tasks related data using root mean squared error (RMSE) and hit rate. Although we first envisioned our approach to perform better on all datasets, our approach (TNMF) has room for improvement for few of them, e.g., *Yahoo!* music and movies. We plan to include pair-wise ranking scheme in our optimization function to alleviate the aforementioned concern. In addition, our preliminary analysis show that estimation of missing recommendations increases mean square error due to the replacement of missing values with zeros. Therefore, our approach demonstrates limited applicability for datasets having large number of missing values. Finally, we made our implementation and data publicly available at `https://github.com/enamul-haque/TNMF` to support reproducibility in future studies.

References

[1] X. Ning, C. Desrosiers, and G. Karypis, "A comprehensive survey of neighborhood-based recommendation methods," in *Recommender systems handbook*, pp. 37–76, Springer, 2015.

[2] M. Deshpande and G. Karypis, "Item-based top-n recommendation algorithms," *ACM Transactions on Information Systems (TOIS)*, vol. 22, no. 1, pp. 143–177, 2004.

[3] J. Chen, H. Zhang, X. He, L. Nie, W. Liu, and T.-S. Chua, "Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention," in *Proceedings of the 40th International ACM*

SIGIR conference on Research and Development in Information Retrieval, pp. 335–344, ACM, 2017.

[4] X. Yang, C. Liang, M. Zhao, H. Wang, H. Ding, Y. Liu, Y. Li, and J. Zhang, "Collaborative filtering-based recommendation of online social voting," *IEEE Transactions on Computational Social Systems*, vol. 4, no. 1, pp. 1–13, 2017.

[5] J. A. Pereira, P. Matuszyk, S. Krieter, M. Spiliopoulou, and G. Saake, "A feature-based personalized recommender system for product-line configuration," *ACM SIGPLAN Notices*, vol. 52, no. 3, pp. 120–131, 2017.

[6] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, pp. 452–461, AUAI Press, 2009.

[7] M. J. Pazzani and D. Billsus, "Content-based recommendation systems," in *The adaptive web*, pp. 325–341, Springer, 2007.

[8] F. Ricci, L. Rokach, and B. Shapira, "Introduction to recommender systems handbook," in *Recommender systems handbook*, pp. 1–35, Springer, 2011.

[9] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Analysis of recommendation algorithms for e-commerce," in *Proceedings of the 2nd ACM conference on Electronic commerce*, pp. 158–167, ACM, 2000.

[10] C.-N. Ziegler, G. Lausen, and L. Schmidt-Thieme, "Taxonomy-driven computation of product recommendations," in *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pp. 406–415, ACM, 2004.

[11] M. E. Haque, M. E. Tozal, and A. Islam, "Helpfulness prediction of online product reviews," *Proceedings of the 18th ACM symposium on Document engineering*, 2018.

[12] B. Liu, Y. Fu, Z. Yao, and H. Xiong, "Learning geographical preferences for point-of-interest recommendation," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1043–1051, ACM, 2013.

[13] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, "Collaborative filtering recommender systems," in *The adaptive web*, pp. 291–324, Springer, 2007.

[14] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Advances in artificial intelligence*, vol. 2009, p. 4, 2009.

[15] M. H. Abdi, G. O. Okeyo, and R. W. Mwangi, "Matrix factorization techniques for context-aware collaborative filtering recommender systems: A survey," *Computer and Information Science*, vol. 11, no. 2, p. 1, 2018.

[16] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, 2009.

[17] Y. Koren and R. Bell, "Advances in collaborative filtering," in *Recommender systems handbook*, pp. 77–118, Springer, 2015.

[18] F. Aiolli, "Convex auc optimization for top-n recommendation with implicit feedback," in *Proceedings of the 8th ACM Conference on Recommender Systems*, pp. 293–296, ACM, 2014.

[19] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in neural information processing systems*, pp. 556–562, 2001.

[20] X. Ning and G. Karypis, "Slim: Sparse linear methods for top-n recommender systems," in *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pp. 497–506, IEEE, 2011.

[21] M. Weimer, A. Karatzoglou, Q. V. Le, and A. J. Smola, "Cofi rank-maximum margin matrix factorization for collaborative ranking," in *Advances in neural information processing systems*, pp. 1593–1600, 2008.

[22] H. Steck, "Training and testing of recommender systems on data missing not at random," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 713–722, ACM, 2010.

[23] W. Pan and L. Chen, "Cofiset: Collaborative filtering via learning pairwise preferences over item-sets," in *Proceedings of the 2013 SIAM International Conference on Data Mining*, pp. 180–188, SIAM, 2013.

[24] D. Park, J. Neeman, J. Zhang, S. Sanghavi, and I. Dhillon, "Preference completion: Large-scale collaborative ranking from pairwise comparisons," in *International Conference on Machine Learning*, pp. 1907–1916, 2015.

[25] H. Yun, P. Raman, and S. Vishwanathan, "Ranking via robust binary classification," in *Advances in Neural Information Processing Systems*, pp. 2582–2590, 2014.

[26] J. M. Joyce, "Kullback-leibler divergence," in *International Encyclopedia of Statistical Science*, pp. 720–722, Springer, 2011.

[27] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 5, no. 4, p. 19, 2016.

[28] Z. Kang, C. Peng, and Q. Cheng, "Top-n recommender system via matrix completion.," in *AAAI*, pp. 179–185, 2016.

[29] A. Mnih and R. R. Salakhutdinov, "Probabilistic matrix factorization," in *Advances in neural information processing systems*, pp. 1257–1264, 2008.

[30] R. Salakhutdinov and A. Mnih, "Bayesian probabilistic matrix factorization using markov chain monte carlo," in *Proceedings of the 25th international conference on Machine learning*, pp. 880–887, ACM, 2008.

[31] Y. Zhou, D. Wilkinson, R. Schreiber, and R. Pan, "Large-scale parallel collaborative filtering for the netflix prize," *Lecture Notes in Computer Science*, vol. 5034, pp. 337–348, 2008.