COLLABORATIVE FILTERING AND CONTENT BASED HYBRID MODELS FOR RECOMMENDING SCIENTIFIC ARTICLES

by

Mine Öğretir

B.S., Computer Engineering, Middle East Technical University, 2002

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering

Boğaziçi University

2018

COLLABORATIVE FILTERING AND CONTENT BASED HYBRID MODELS FOR RECOMMENDING SCIENTIFIC ARTICLES

APPROVED BY:

Prof. Ali Taylan Cemgil (Thesis Supervisor)	.....
Prof. Sadık Fikret Gürgen	.....
Assoc. Prof. Emine Yılmaz	.....

DATE OF APPROVAL: 04.06.2018

ACKNOWLEDGEMENTS

I would like to thank my advisor Prof. Ali Taylan Cemgil for his support and patience during my study. His guidance was highly valuable for me. He was always supportive and spared time for me whenever I needed.

I want to thank Heysem Kaya for his support at the right time. There was a time that I was thinking of dropping out my study. I may have continued because of his guidance at that moment. I also thank Çağıl Sönmez and Ahter Sönmez for their support and their influence on the fun moments during this period. I feel lucky to be able to work with the members of PILAB, especially, Serhan Daniş for his support and Onur Poyraz for his considerable contribution at final phase of this thesis.

Special thanks to Tina Varon, my dear friend. Her moral and accommodation support during my toughest times, at the beginning and at the end of my study, was invaluable. I would like to thank Elif Burgaz Olloqui Mateo and Neslihan Ağırgüden for their belief in me and their moral support.

I wouldn't even start again this study without my parent's moral and financial support. I would like to express my gratitude for the opportunity they provided and their endless moral support.

ABSTRACT

COLLABORATIVE FILTERING AND CONTENT BASED HYBRID MODELS FOR RECOMMENDING SCIENTIFIC ARTICLES

Recommendation systems (RS) are programs that assist users in accessing information in vast amount of data collections. In this thesis, we investigate hybrid models that use both implicit ratings such as tags, bookmarks or impressions, and content information such as user's profile or item properties. In the literature, recommendation approaches that use such information are known as collaborative filtering (CF) and content-based methods, respectively. As computation methodology we investigate and compare two techniques, one is based on matrix decomposition and the other one is based on deep learning. As a matrix decomposition based approach, we investigate Bayesian nonnegative matrix factorization (BNMF), that we enhance using side information, the titles and abstracts of scientific articles, besides the implicit rating matrix. As a deep learning method, we explore collaborative deep learning (CDL), which uses probabilistic matrix factorization as CF method and Bayesian stacked denoising autoencoder (SDAE) as content feature extraction. We apply these techniques in our experiments to a CiteULike dataset with a rating density of 0.22%. Our experimental results show that CDL is more effective than coupled BNMF on this dataset. In our opinion, CDL performs better due to its Bayesian SDAE component which has nonlinear and deep structure.

ÖZET

BİLİMSEL MAKALE ÖNERİSİ İÇİN İŞBİRLİKÇİ FİLTRELEME VE İÇERİK TABANLI HİBRİT MODELLER

Öneri sistemleri (RS), kullanıcıların, çok sayıda veri koleksiyonu içerisinden bilgiye erişmelerine yardımcı olan programlardır. Bu tezde, etiketler, yer imleri veya izlenimler gibi örtük derecelendirmeleri ve kullanıcının profili veya öğe özellikleri gibi içerik bilgilerini kullanan hibrit modelleri araştırıyoruz. Literatürde, bu tür bilgileri kullanan öneri yaklaşımları, sırasıyla işbirlikçi filtreleme (CF) ve içerik tabanlı yöntemler olarak bilinmektedir. Hesaplama metodolojisi olarak biri matris ayrıştırmasına, diğeri ise derin öğrenmeye dayanan iki tekniği karşılatırıyoruz. Matris ayrıştırma temelli yaklaşım olarak, örtük derecelendirme matrisinin yanı sıra, bilimsel makalelerin başlıklarını ve özetlerini yan bilgi olarak kullanan Bayesci negatif olmayan matris faktörizasyonu (BNMF) yöntemini araştırıyoruz. Derin öğrenme metodu kapsamında CF yöntemi olarak olasılıksal matris faktörizasyonunu ve içerik tabanlı özellik çıkarımı olarak istiflenmiş gürültü giderici otokodlayıcılarına (SDAE) Bayesci yaklaşımı kullanan işbirlikçi derin öğrenme (CDL) yöntemini araştırıyoruz. Deneylerimizde bu teknikleri % 0.22'lik derecelendirme yoğunluğuna sahip bir CiteULike veri kümesine uyguluyoruz. Deneysel sonuçlarımız, CDL'nin bu veri setinde bağlaşık BNMF'den daha etkili olduğunu göstermektedir. Bizim görüşümüzce, CDL, doğrusal olmayan ve derin bir yapıya sahip olan Bayesci SDAE bileşeni nedeniyle daha iyi performans göstermektedir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

$\mathbf{b}_l$	Bias vector of lth layer
$\mathcal{B}_{VB}[q]$	Bound of instrumental distribution q in Variational Bayes method
e_{ij}	Error of the estimated value
$f_e(.)$	Encoder function
$f_r(.)$	Decoder function
$H[q]$	Entropy of q
$\mathbf{I}_K$	KxK identity matrix
$\mathbf{M}$	IxJ Masking matrix
m_{ij}	Masking or confidence value of user i for item j
r_{ij}	Rating of user i for item j
r^*_{ij}	Estimation of the rating of user i for item j
$\mathbf{R}$	IxJ Rating matrix
$\mathbf{R}_i$	Row item vector for user i
$\mathbf{R}_j$	Column user vector for item j
$s_{i,k,j}$	k. latent source for r_{ij}
u_i	Feature vector for user i
$\mathbf{U}$	IxK User-feature matrix
v_j	Feature vector for item j
$\mathbf{V}$	KxJ Item-feature matrix
$\mathbf{W}$	Weight matrix
$\mathbf{W}_{l,*n}$	nth column vector of weight matrix for lth layer
$\mathbf{W}^+$	Representation for weights and biases for all layers
$\mathbf{x}^-$	Corrupted version of $\mathbf{x}$
$\mathbf{X}_0$	Corrupted input
$\mathbf{X}_C$	Uncorrupted input
$\mathbf{X}_l$	Input matrix for lth layer
$\mathbf{X}_{l,j*}$	lth row vector of input matrix for lth layer

γ	Learning rate
Θ	Parameter set
$\lambda, \lambda_u, \lambda_v$	Precision values for Gaussian distributions of rating, latent user row vector and latent item column vector
$\sigma(x)$	Logistic function of x: $\sigma(x) = 1/(1 + e^{-1})$
$a^u_{i,k}$, $b^u_{i,k}$, $a^v_{k,j}$, $b^v_{k,j}$	Shape and scale parameters for Gamma distributions
$\{A, B\}$	Vertical concatenation of matrices A and B
$\mathcal{L}_R(\Theta)$	Loglikelihood of R given Θ
$\|\|u_i\|\|_{Fro}$	Frobenius norm of u_i
$\nabla_{\mathbf{W}_l}\mathcal{L}$	Gradient of loglikelihood with respect to $\mathbf{W}_l$
$\nabla_{\mathbf{b}_l}\mathcal{L}$	Gradient of loglikelihood with respect to $\mathbf{b}_l$

LIST OF ACRONYMS/ABBREVIATIONS

ALS	Alternating Least Squares
CDL	Collaborative Deep Learning
CF	Collaborative Filtering
CBNMF	Coupled Bayesian Nonnegative Matrix Factorization
CTR	Collaborative Topic Regression
KL	Kullback-Leibler
LDA	Latent Dirichlet Allocation
mAP	Mean Average Precision
MAP	Maximum A Posteriori
MCMC	Markov Chain Monte Carlo
MF	Matrix Factorization
NMF	Nonnegative Matrix Factorization
PMF	Probabilistic Matrix Factorization
RBM	Restricted Boltzmann Machine
RS	Recommendation System
SDAE	Stacked Denoising Autoencoder
SGC	Stochastic Gradient Descent

1. INTRODUCTION

The growth in accessibility and wide adoptation of Internet services and online markets led to a rapid expansion in the number of products and the amount of information available to users. As a side effect of this expansion, the difficulty of access to the effective products and useful information that address the needs of users grows. The available collections of consumable data and products have reached such levels that it is practically impossible that a user will be able to even browse through a tiny fraction of available items. For example, there are over 35 million songs and over 2 billion playlists in Spotify and over 20.000 new songs are added daily [8,9]. We cannot expect a single user to listen to all new music to pick the ones that she would like. In another case, Netflix has nearly 15,400 titles across the globe [10]. So, users would like to be informed about movies that they may like to watch instead of browsing the whole content. Therefore, the relevant information based on the user's preferences should be acquired according to the task at hand or to the need of the users. Recommendation systems are tools that suggest specific items like academic papers, movies, news, music, books and any other products that users aim to access or users would be interested even they have no previous knowledge about [11]. Recommendation systems (RS) are being developed in order to assist people with relevant and useful information in locating products that they need. With the help of RS, the users benefit from avoiding excessive search time. Also, many times users do not have enough experience in the task they need. So that, they may have difficulties in evaluating the items among many alternatives. These users may also benefit from these systems considerably.

1.1. Related Work

The fundamental structure of an RS can be formalized by a relation R, where $R(i, j)$ denotes the relation between user i and item j. In the collaborative filtering approach, which is explained below, R corresponds to the rating which user i gives to item j. In this approach the task is estimating the values $R^*(i, j)$ which converges to missing values $R(i, j)$ [12]. An example of such a rating matrix is given in Figure 1.1.

Since users cannot review even a tiny proportion of all items, R mostly tends to be sparse.

Figure 1.1. An example of rating matrix representation

RS emerged around the middle of 1990's as a separate research task. The research in the subject gained momentum with Netflix's awarded competition and with conferences and workshops dedicated to this subject [12]. In the literature, RS are categorized according to the knowledge they utilize [12, 13].

(i) Collaborative filtering (CF) methods use explicit information such as direct ratings of users for the items [1, 14]. The main idea behind this approach is that if a user has similar preferences to another user, it means that they have similar taste and recommendations should be made accordingly.

(ii) Neighborhood methods aim to find the nearest neighbors according to a dedicated similarity measure between the items or the users or both. Some of the latent factor techniques in RS can be viewed under this method [12].

(iii) Content-based methods use implicit information like user profiles and product descriptions [12, 15]. These methods focus on the similarity of items and make suggestions according to the user's previous preferences and suggest similar items.

(iv) Demographic RS use user's demographic profile, like language, country, age or any other related information.

(v) Knowledge-based RS approach is based on the domain knowledge, and they are case-based, that is they utilize the specific item features. They check which user preferences are met with the items.

(vi) In community-based systems, the suggestions are made according to the user's friends' preferences assuming close friends share similar taste.

(vii) On the other hand, hybrid models merge different techniques to get the advantage of each method [6, 7, 16].

Various machine learning methods are used in recommendation systems. Some approaches use restricted Boltzmann machines or recurrent neural networks instead of latent factor analysis [3, 17, 18]. They do not use content information. Some hybrid studies for music recommendation use recurrent neural networks and deep belief networks with bag-of-words for content representation [19, 20]. These models are deterministic and do not state noise factor, therefore they are not very robust.

There seems to be no consensus about the best approach for recommendation systems in general. The context, density, quality and characteristics of the gathered data and the goal or the task of the system can affect the best approach to the particular dataset [21]. In addition to that, the evaluation of the system should be made according to the task at hand. Evaluation metrics used to evaluate the performance of the system should differ depending on the objectives.

1.2. Data Types

There are various data types utilized in recommendation systems. One kind of data is explicit feedback such as user-item ratings, i.e. direct opinion of the user on the item. The other kind of data is implicit feedback such as the items that purchased by the user, the clicks on a specific web page or songs listened. Implicit feedback does not explicitly provide direct satisfaction ratings from the users [12]. Generally implicit feedback data may be much more dense; however, it provides indirect and low-quality

information. On the other hand, explicit feedback provides more relevant information about the preferences of the users. However, it is much more difficult to obtain, so that it is mostly very sparse.

1.3. Evaluation

Evaluating the quality and value of the system is not a straightforward task in the context of RS. Hence, evaluation methodologies used for this purpose is another research topic widely studied within this area. These systems' outputs are not suitable for comparing ground truths with the predicted outcomes most of the time. How relevant a prediction to the aim of the system may differ significantly. Evaluating a system for novelty of recommendations would require using different evaluation mechanisms compared to evaluating the same system for its utility to an end user. Choosing the relevant evaluation metric is crucial for performance measurement. Generally, evaluation metrics are classified in [22] as (i) prediction metrics, e.g. Mean Absolute Error, Root of Mean Square Error, Normalized Mean Average Error; (ii) set recommendation metrics, e.g. Precision, Recall and Receiver Operating Characteristic; (iii) rank recommendation metrics, e.g. the half-life and the discounted cumulative gain; and (iv) diversity metrics, e.g. the diversity and the novelty of the recommended items. We used recall metric in our study, because the density of the bookmarked articles is very low. The details are explained in Chapter 4.

1.4. Challenges

There are various challenges in recommendation systems. Often the data acquired for the recommendation system is very sparse. That is there are very few explicit data with respect to the user and item number. A commercial recommendation system may have millions of customers and millions of items but explicit ratings from users may only be available for a tiny fraction of the items. Therefore, the proposed models should handle large but sparse data. The other problem is the rapid addition of new users or new items to the existing system. So, scalability is another challenge in recommendation systems, where the proposed methodology should handle rapidly growing

and sparse data [23].

One of the main challenges in RS is the recommendation of a new added item or recommendation for a new user entered the system. This is the so-called 'cold-start' problem. Since the new item in the system does not yet have any ratings, it will not be recommended and will be out of reach. One possible solution is using additional data related to the items. The other possible solution is to use some motivated users to rate each new item. The new user problem is a more challenging problem since the system does not have any preference information about the user [22]. Hybrid models are often used to overcome the cold-start problem by merging different techniques, i.e. different source of information. For example, using item content information beside rating values is a hybrid method as we stated in this thesis.

Collaborative filtering methods are the most popular techniques in RS. However, they are sensitive to the sparsity level and the balance of a dataset. CF methods cannot make effective recommendations in sparse settings [22]. In addition, they suffer deeply when a new item or a new user is added to the system because of the dependency on explicit data. On the other hand, content-based methods do not rely on explicit ratings for recommendation.

1.5. Contributions of Our Work

In this study, we focused on a subset of hybrid recommendation models. Coupled Bayesian nonnegative matrix factorization (CBNMF) is a Bayesian nonnegative matrix factorization technique which utilizes both rating and content data as a coupled matrix. Collaborative topic regression (CTR) is a hybrid model, which consists of probabilistic matrix factorization on the CF approach, and Latent Dirichlet Allocation (LDA) for topic modelling (i.e. inferring topical categories of the content). Collaborative deep learning (CDL) is similar to CTR but uses stacked denoising autoencoder for content modeling instead of LDA. Some details of our study are as follows:

(i) Our main contribution is to apply Bayesian nonnegative matrix factorization method as a hybrid recommendation system. Collaborative filtering and content-based approach are applied as coupling the rating matrix and the content matrix. We compared our proposed method with a state-of-art hybrid method, collaborative deep learning.

(ii) We used implicit user-item preferences. We only know whether a user has chosen a specific item. The lack of information may indicate a seen but not preferred item or not-seen item. The rating matrix R is a binary matrix, where 1 indicates a user's preference. 0 stands for either a seen, but not preferred item or not seen item.

(iii) For content representation, we used the bag-of-words representation with normalized term frequency-inverse document frequency (TF-IDF) values. TF-IDF is a measure used in natural language processing for indicating how a word is important or relevant to a document in a corpus [4]. It is the product of the word's frequency in the document and inverse logarithm of the ratio between the number of documents in the corpus and the number of documents the word appears.

(iv) We compared the methods, coupled Bayesian nonnegative matrix factorization and collaborative deep learning, at two different settings which are sparse and dense settings. In the sparse setting, the training set contains one item for each user. In the dense setting, the training set contains ten items for each user. The test set is all the remaining ratings of the users. For each setting five different training-test sets are used.

(v) We used Recall@300 values of the recommended items as the evaluation metric. It is the ratio between successful recommendations of the top 300 recommendation of the system and all the items that users have chosen in the test set.

The rest of this thesis is organized as follows: In Chapter 2, we introduce and provide an overview of the related subjects under the scope of this thesis. The subjects include knowledge characteristics of this problem, notation for the rest of this document, related collaborative filtering methods, approaches related to the content representation and content models, and an overview of the approaches used for evaluat-

ing RS. In Chapter 3, we explain three hybrid models which are based on collaborative filtering and content-based methods. These are coupled Bayesian nonnegative matrix factorization, collaborative topic regression and collaborative deep learning methods. In this thesis, we mainly focus on coupled Bayesian nonnegative matrix factorization and collaborative deep learning as to introduce CBNMF and compare it with a state of the art method, CDL. Collaborative topic regression is the main model that collaborative deep learning method is based on. Chapter 4 provides the details and the results of our experiments. Finally, in Chapter 5, we discuss and conclude our work and offer future directions for our study.

2. BACKGROUND

In this section, we outlined some concepts and methodologies that are addressed in our work. Feedback types and the notation used in our work are explained as well as some commonly used methods. As we introduce each method, we explain both the method itself and commonly used inference techniques for that method in the same subsection. We briefly reviewed some evaluation approaches in recommendation systems at the last section.

2.1. Explicit vs Implicit Feedback in Recommendation Systems

The data used in recommendation systems are the interaction of the users with the system provided. The data can be any kind of information related to a user and a particular item, such as given ratings, purchases, mouse clicks, repeated usage. In general, the data is categorized as explicit feedback and implicit feedback according to the quality of the information [12].

Explicit data reflect the direct preferences of users on a particular item [12]. A recommendation system inquires the user about the preference on the specified item. The rating may be in the discrete scale of [1-5] as in former Netflix rating system, or of [1-10] as in IMDB on a movie in these cases, or may be binary as in Netflix new rating system or as in [24] on news stories. The binary rating indicates whether the user likes the item or not, or the item is relevant or not.

Implicit feedback is user's behavior on the recommendation system. Implicit feedback is the actions of the user in the system like mouse clicks, bookmarking, saving, watching a whole video, movie rental history, search history, and any other activity. User's opinion is not directly available, however, user's preference can be inferred according to her/his actions [12].

Explicit feedback allows to infer more reliable information on the users' preferences, however, it can be considered as costly. It requires encouraging users to rate items in the recommendation system. On the other hand, implicit feedback is available by navigating and using the system. Users do not have to do additional task other than using the system. Some recommendation systems use explicit feedback as a primary source and implicit feedback as supportive data [12]. Our work focuses on the implicit feedback of the users in a scientific article platform, which we will give the details in Section 4.1.

2.2. Notation

Even though we give details of the notation in each subsection, the following notation is common throughout this thesis.

The IxJ matrix $\mathbf{R}$ denotes the rating matrix of I users and J items. That is r_{ij} is the rating of user i for the item j. The IxK matrix $\mathbf{U}$ states for the user latent feature matrix and KxJ matrix $\mathbf{V}$ states the item latent feature matrix. The row vector u_i and column vector v_j represent latent feature vectors of user i and item j throughout the dataset respectively. IxS matrix, $\mathbf{X}$, denotes content representation. $\mathbf{I}_.$ are identity matrices with dimensions stated in subscripts, such as $\mathbf{I}_J$, $\mathbf{I}_K$, which are JxJ and KxK identity matrices.

The hyperparameters symbolized with λ, like λ_u, λ_v etc, denote precision parameters for Gaussian distributions. The matrix, $\mathbf{W}$, and the vector, $\mathbf{b}$, are weight matrix and bias vector respectively. The parameter denoted by C_{ij}, is the confidence parameter for user i and item j, where $\mathbf{C}_i$ is JxJ diagonal matrix, where diagonal values are item rating confidence values for user i.

2.3. Collaborative Filtering for Recommendation Systems (CF)

Collaborative filtering methods mostly use explicit feedback which is the direct opinion of the user on an item [12]. The main idea behind this approach is that users

that have a similar preference will prefer alike items. Similarity can be measured according to their preference history [23]. A user will give a similar rating to an item with another user if they gave similar ratings for some other items in the recommendation system [12].

Su and Khoshgoftaar categorize collaborative filtering methods in two categories in their survey on CF. These are memory based and model based CF methods. The most common method in memory based CF methods is neighbor-based collaborative filtering method. Neighborhood approach is interested in the relationships between items or users. It is based on the idea that there are users with similar preferences or items in similar categories. The similarity measure is the main component in memory based CF methods. Two commonly used similarity measures are the correlation-based similarity and vector cosine based similarity [23]. Selecting top-N recommendations according to k most similar users or items are two tasks in this approach. Since our focus is more on model-based CF methods, we will not go into detail on memory-based methods.

Usage of Bayesian belief nets, clustering, latent semantic, sparse factor analysis, dimensionality reduction techniques such as singular value decomposition and principal component analysis in collaborative filtering are examples of model-based CF techniques [23].

Matrix factorization for recommendation systems is categorized as a latent factor model in collaborative filtering methods [11]. Latent factor models extract same latent factors from users and items, which can explain the observed explicit feedback, that is ratings [11].

Explicit feedback in collaborative filtering methods is a high-quality information on preferences, though it generally provides sparse information. Nevertheless, implicit feedback or temporal data can also be used besides the explicit data [12].

2.3.1. Matrix Factorization

As mentioned above, matrix factorization is an efficient and reliable method for recommendation systems with explicit feedback [11]. In this method, the rating matrix **R** is factorized as the product of user-feature matrix **U** and item-feature matrix **V** as shown in Figure 2.1. The rating of user i for the item j is the product of the K-length row vector u_i and column vector v_j, which are the K feature weights corresponding to user i and item j respectively. This model is highly related with singular value decomposition (SVD), a well-known matrix factorization method. However, conventional SVD is defined on the complete matrices. The rating matrices are generally highly sparse matrices. Therefore, while computing the feature matrices, only the known entries are taken into consideration [12]. The objective is to minimize the regularized squared error given in Equation 2.1. Regularization is required to avoid overfitting. The error is computed over the observed ratings. α is the regularization term and determined via cross training.

$$\min_{u,v} \sum_{(u,v)\in\mathcal{K}} (r_{i,j} - u_i v_j)^2 + \alpha(\|u_i\|^2 - \|v_j\|^2) \tag{2.1}$$

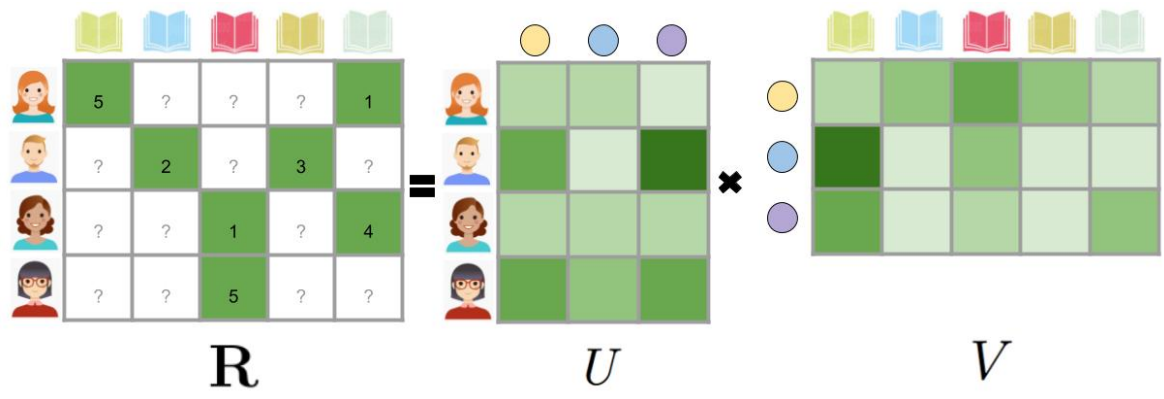

Figure 2.1. Matrix factorization: Coloured circles represents the extracted features that may explain the observed ratings.

2.3.2. Learning

Stochastic gradient descend and alternating least squares methods are two frequent used methods for learning the latent factors [11].

2.3.2.1. Stochastic Gradient Descent(SGD). In stochastic gradient descent learning method, the gradient of the objective function as stated in equation 2.1 is used to modify the values of the parameters in iteration. The parameters are updated in the opposite direction of the gradient. The update is done for each rating in the training set [11]. The error and update equations are given in equations 2.2 where γ is the learning rate. Learning rate determines how the gradient will effect the update amount.

$$\begin{aligned} e_{ij} &= r_{ij} - u_i v_j \\ u_i^{t+1} &= u_i^t + \gamma(e_{ij} v_j - \alpha u_i) \\ v_j^{t+1} &= v_j^t + \gamma(e_{ij} u_i - \alpha v_j) \end{aligned} \tag{2.2}$$

2.3.2.2. Alternating Least Squares(ALS). Another method to compute the parameters is to fix $\mathbf{U}$ and solve the system for $\mathbf{V}$, and then fix $\mathbf{V}$ and solve for $\mathbf{U}$. Iterating between $\mathbf{U}$ and $\mathbf{V}$ allows objective to converge. Alternating least squares is generally favorable when parallel computing is possible and when implicit data is used. When the data is not sparse SGD may be not practical. The update equations for ALS are as follows:

$$\begin{aligned} \mathbf{V} &= (\mathbf{U}^T\mathbf{U})^{-1}\mathbf{U}^T\mathbf{R} \\ \mathbf{U} &= \mathbf{R}\mathbf{V}^T(\mathbf{V}^T\mathbf{V})^{-1} \end{aligned}$$

2.3.3. Probabilistic Matrix Factorization (PMF)

Probabilistic matrix factorization (PMF) is a latent factor collaborative filtering method [1]. The aim is to handle large and imbalanced datasets effectively. The model adds Gaussian observation noise to matrix factorization model.

The conditional distribution is defined as follows:

$$p(R|U,V,\lambda) = \prod_{i=1}^{I}\prod_{j=1}^{J}\left[\mathcal{N}(r_{ij}|u_i v_j, \lambda^{-1})\right]^{m_{ij}}$$

User and item feature vectors have zero mean Gaussian priors. The generative process of PMF is as follows:

(i) Draw K-length user latent row vector u_i for user i:
$u_i \sim \mathcal{N}(0, \lambda_u^{-1}\mathbf{I}_K)$

(ii) Draw K-length item latent column vector v_j for item j:
$v_j \sim \mathcal{N}(0, \lambda_v^{-1}\mathbf{I}_K)$

(iii) Draw rating values of user i and item j pair:
$r_{ij} \sim \mathcal{N}(u_i v_j, \lambda^{-1})$

The matrix denoted by $\mathbf{I}_K$ is KxK identity matrix. The precision hyperparameters are symbolized with λ_u, λ_v and λ. The Gaussian probability distribution function, with mean μ and precision λ is depicted as $\mathcal{N}(x;\mu,\lambda^{-1})$. The graphical model of PMF is shown in Figure 2.2.

The logarithm of the posterior distribution is as follows:

$$\begin{aligned}\ln p(U,V|R,\lambda,\lambda_u,\lambda_v) = &-\frac{\lambda}{2}\sum_{i=1}^{I}\sum_{j=1}^{J} m_{ij}(r_{ij}-u_i v_j)^2 - \frac{\lambda_u}{2}\sum_{i=1}^{I} u_i^T u_i - \frac{\lambda_v}{2}\sum_{j=1}^{J} v_j v_j^T \\ &+\frac{1}{2}\left(\left(\sum_{i=1}^{I}\sum_{j=1}^{J} m_{ij}\right)\ln\lambda + IK\ln\lambda_u + JK\ln\lambda_v\right) + C\end{aligned} \tag{2.3}$$

The precision hyperparameters are assumed to be fixed. Maximizing the log-posterior is equivalent to minimizing the sum of squared-errors objective function with regularization terms, as given:

$$E = \frac{\lambda}{2}\sum_{i=1}^{I}\sum_{j=1}^{J} m_{ij}(r_{ij} - u_i v_j)^2 + \frac{\lambda_u}{2}\sum_{i=1}^{I}\|u_i\|_{Fro}^2 + \frac{\lambda_v}{2}\sum_{j=1}^{J}\|v_j\|_{Fro}^2$$

Maximizing the log posterior is equivalent to minimizing the given objective function, and this can be done by gradient descent.

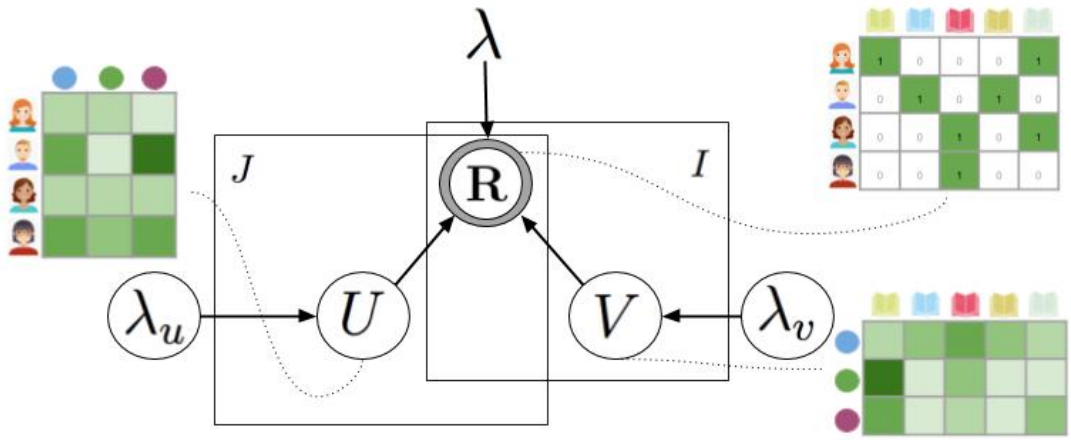

Figure 2.2. Graphical model of PMF [1] and representation of the matrices for CiteUlike dataset. The colored circles on the U and V matrices represent for the extracted features.

Salakhutdinov and Mnih also proposed Bayesian treatment for PMF [25]. Since PMF requires regularization parameter tuning, full Bayesian approach allows to integrate over all model parameters and hyperparameters. They also showed that training can be done efficiently by MCMC methods on Netflix dataset.

2.3.4. Bayesian Nonnegative Matrix Factorization (BNMF)

In real life, most data are nonnegative. Factorizing a nonnegative matrix with the nonnegativity constraint in the product matrices is called nonnegative matrix fac-

torization [2, 26]. In addition to SGD and ALS techniques in matrix factorization, multiplicative update algorithm can be used to obtain the product matrices. When the matrices are initialized with positive values, the updates do not violate positivity [26].

In this case, Bayesian nonnegative matrix factorization is a Bayesian approach to nonnegative matrix factorization [2]. The hierarchical model is as follows:

$$\mathbf{U} \sim p(\mathbf{U}|\Theta^u), \qquad \mathbf{V} \sim p(\mathbf{V}|\Theta^v),$$
$$s_{i,k,j} \sim \mathcal{PO}(s_{i,k,j}; u_{i,k}v_{k,j}), \qquad r_{i,j} = \sum_k s_{i,k,j}.$$

A schematic representation and the graphical model is given in Figures 2.3 and 2.4.

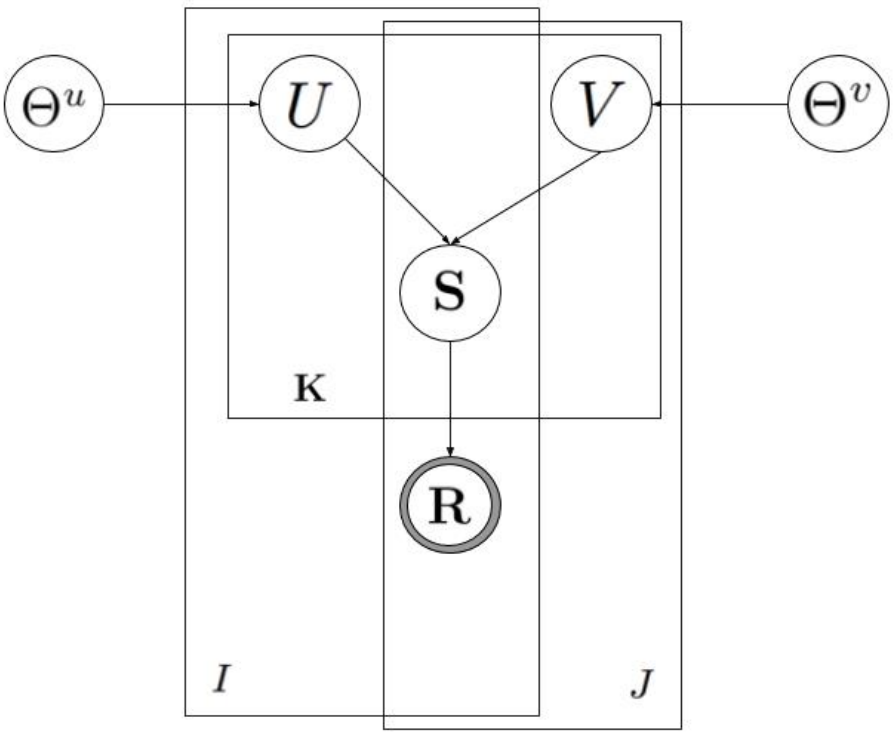

Figure 2.3. Graphical model of Bayesian nonnegative matrix factorization [2]

Cemgil showed that a fully Bayesian treatment can be implemented efficiently by variational Bayes or MCMC [2]. Computation of marginal likelihoods allows model selection

in full Bayesian approach. The priors are chosen as Gamma distribution from the conjugate family of Poisson intensity.

$$u_{i,k} \sim \mathcal{G}(u_{i,k}; a^u_{i,k}, \frac{b^u_{i,k}}{a^u_{i,k}}), \qquad v_{k,j} \sim \mathcal{G}(v_{k,j}; a^v_{k,j}, \frac{b^v_{k,j}}{a^v_{k,j}}),$$

$a^u_{i,k}$, $b^u_{i,k}$, $a^v_{k,j}$ and $b^v_{k,j}$ are shape and scale parameters for the Gamma distributions. The objective is to minimize the Kullback-Leibler (KL) distance of $\mathbf{R}$ and $\mathbf{UV}$.

$$(\mathbf{U}, \mathbf{V})^* = \arg\min_{U,V>0} D(\mathbf{R}||\mathbf{UV})$$

Missing values can also be taken into consideration via masking.

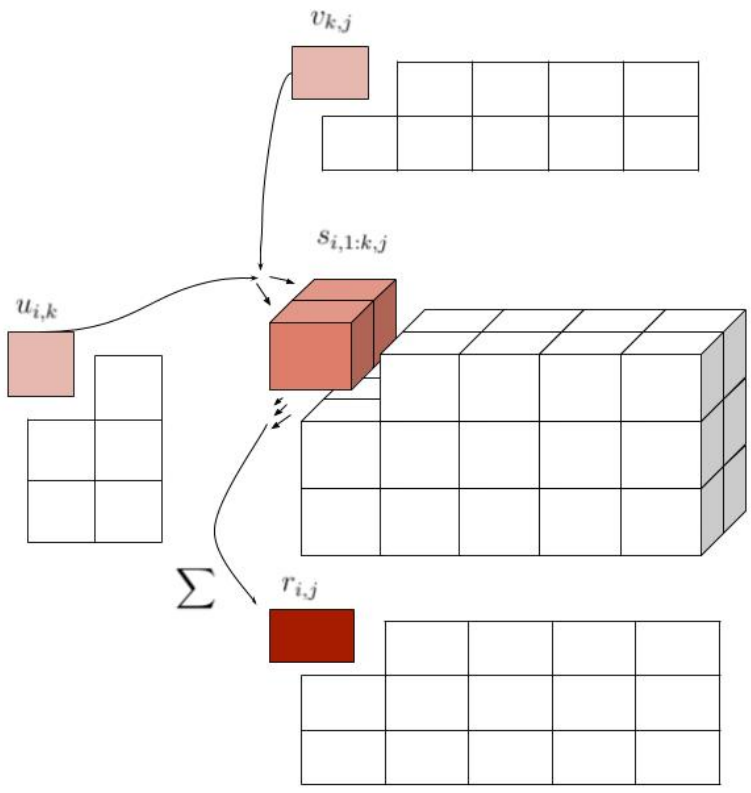

Figure 2.4. A schematic representation of Bayesian nonnegative matrix factorization [2]

2.3.4.1. Variational Bayes. The bound of the marginal log-likelihood for Variational Bayes method is as follows: [2]

$$\mathcal{L}_R(\Theta) \equiv \log p(R|\Theta) \geqslant \sum_S \int d(U,V) q \log \frac{p(R,S,U,V|\Theta)}{q}$$
$$= \langle \log p(R,S,U,V|\Theta) \rangle_q + H[q] \equiv \mathcal{B}_{VB}[q]$$

q is the instrumental distribution. It is defined as a factorised form in order to make it tractable:

$$q(S,U,V) = q(S)q(U)q(V)$$
$$= \left(\prod_{i,j} q(s_{i,1:K,j}) \right) \left(\prod_{i,k} q(u_{i,k}) \right) \left(\prod_{k,j} q(v_{k,j}) \right) \equiv \prod_{\boldsymbol{\alpha} \in \mathcal{C}} q_{\boldsymbol{\alpha}}$$

$\boldsymbol{\alpha} \in \mathcal{C} = \{\{S\}, \{U\}, \{V\}\}$ is the set of disjoint clusters. As stated in [2], although a perfect q distribution is not available in closed form, a local optimum can be reached by the following fixed point iterations:

$$q(S)^{(n+1)} \propto \exp(\langle \log p(R,S,U,V|\Theta) \rangle_{q(U)^n q(V)^n})$$
$$q(U)^{(n+1)} \propto \exp(\langle \log p(R,S,U,V|\Theta) \rangle_{q(V)^n q(S)^n})$$
$$q(V)^{(n+1)} \propto \exp(\langle \log p(R,S,U,V|\Theta) \rangle_{q(U)^n q(S)^n})$$

For each iteration, $\mathcal{B}[q_{(n)}] \leq \mathcal{B}[q_{(n+1)}]$ for $n = 1, 2, ...$ given $q^{(0)}$ as initialization.

The update equations with sufficient statistics are given below. The details are given in [2].

$$q(s_{i,1:K,j}) \propto \mathcal{M}(s_{i,1,j}, ..., s_{i,k,j}, ..., s_{i,K,j}; r_{i,j}, p_{i,1,j}, ..., p_{i,k,j}, ..., p_{i,K,j})$$
$$p_{i,k,j} = \frac{\exp(\langle \log u_{i,k} \rangle + \langle \log v_{k,j} \rangle)}{\sum_k \exp(\langle \log u_{i,k} \rangle + \langle \log v_{k,j} \rangle)},$$
$$\langle s_{i,k,j} \rangle = r_{i,j} p_{i,k,j}$$

$$q(u_{i,k}) \propto \mathcal{G}(u_{i,k}; \alpha^u_{i,k}, \beta^u_{i,k}),$$
$$\alpha^u_{i,k} \equiv a^u_{i,k} + \sum_j \langle s_{i,k,j} \rangle, \qquad \beta^u_{i,k} \equiv \left(\frac{a^u_{i,k}}{b^u_{i,k}} + \sum_j \langle v_{k,j} \rangle \right)^{-1},$$
$$\exp(\langle \log u_{i,k} \rangle) = \exp(\Psi(\alpha^u_{i,k}))\beta^u_{i,k},$$
$$\langle u_{i,k} \rangle = \alpha^u_{i,k}\beta^u_{i,k}$$

$$q(v_{k,j}) \propto \mathcal{G}(v_{k,j}; \alpha^v_{k,j}, \beta^v_{k,j}),$$
$$\alpha^v_{k,j} \equiv a^v_{k,j} + \sum_i \langle s_{i,k,j} \rangle, \qquad \beta^v_{k,j} \equiv \left(\frac{a^v_{k,j}}{b^v_{k,j}} + \sum_i \langle u_{i,k} \rangle \right)^{-1},$$
$$\exp(\langle \log v_{k,j} \rangle) = \exp(\Psi(\alpha^v_{k,j}))\beta^v_{k,j},$$
$$\langle v_{k,j} \rangle = \alpha^v_{k,j}\beta^v_{k,j}$$

The matrices with u and v in the subscript are $\mathbb{R}_+^{I \times K}$ and $\mathbb{R}_+^{K \times J}$, respectively. Elementwise matrix multiplication and division are represented as $.*$ and $./$, respectively. The algorithm derived for Variational Bayes for nonnegative matrix factorization is given in Figure 2.5. The notation is as follows:

$$E_u = \{\langle u_{i,k} \rangle\} \qquad L_u = \{\exp(\langle \log u_{i,k} \rangle)\},$$
$$\Sigma_u = \left\{ \sum_j \langle s_{i,k,j} \rangle \right\},$$
$$A_u = \{a^u_{i,k}\} \qquad B_u = \{b^u_{i,k}\},$$
$$\boldsymbol{\alpha}_u = \{\alpha^u_{i,k}\}, \qquad \boldsymbol{\beta}_u = \{\beta^u_{i,k}\}$$
$$E_v = \{\langle v_{k,j} \rangle\} \qquad L_v = \{\exp(\langle \log v_{k,j} \rangle)\},$$
$$\Sigma_v = \left\{ \sum_i \langle s_{i,k,j} \rangle \right\},$$
$$A_v = \{a^v_{k,j}\} \qquad B_v = \{b^v_{k,j}\},$$
$$\boldsymbol{\alpha}_v = \{\alpha^v_{k,j}\}, \qquad \boldsymbol{\beta}_v = \{\beta^v_{k,j}\}$$

Initialise:

$$L_u^{(0)} = E_u^{(0)} \sim \mathcal{G}(.; A_u, b_u/A_u), \qquad L_v^{(0)} = E_v^{(0)} \sim \mathcal{G}(.; A_v, B_v/A_v)$$

for n=1... **to** MAXITER **do**

Source sufficent statistics

$$\Sigma_u^{(n)} := L_u^{(n-1)} .* (((X. * M)./(L_u^{(n-1)} L_v^{(n-1)}))L_v^{(n-1)^T})$$

$$\Sigma_v^{(n)} := L_v^{(n-1)} .* (L_u^{(n-1)^T}((X. * M)./(L_u^{(n-1)} L_v^{(n-1)})))$$

Means

$$E_u^{(n)} := \boldsymbol{\alpha}_u^{(n)} .* \boldsymbol{\beta}_u^{(n)} \qquad \boldsymbol{\alpha}_u^{(n)} = A_u + \Sigma_u^{(n)} \qquad \boldsymbol{\beta}_u^{(n)} = 1./(A_u./B_u + M E_v^{(n-1)^T})$$

$$E_v^{(n)} := \boldsymbol{\alpha}_v^{(n)} . * \boldsymbol{\beta}_v^{(n)} \qquad \boldsymbol{\alpha}_v^{(n)} = A_v + \Sigma_v^{(n)} \qquad \boldsymbol{\beta}_v^{(n)} = 1./(A_v./B_v + E_u^{(n)^T} M)$$

Compute Bound (Optional)

Means of Logs

$$L_u^{(n)} = \exp(\Psi(\boldsymbol{\alpha}_u^{(n)})) . * \boldsymbol{\beta}_u^{(n)} L_v^{(n)} = \exp(\Psi(\boldsymbol{\alpha}_v^{(n)})) . * \boldsymbol{\beta}_v^{(n)}$$

Update Hyperparameters (Optional)

end for

Figure 2.5. Variational Nonnegative Matrix Factorization Algorithm [2].

2.3.5. Restricted Boltzmann Machines for Collaborative Filtering (RBM)

Salakhutdinov, Mnih, and Hinton presented a model to use RBMs, a two-layer undirected graphical model, as a tool for collaborative filtering, and they showed efficient learning and inference procedures for related models. They demonstrated this model on Netflix data, where movies are rated in the range of 1-5 [3].

The graphical representation of the model is shown in Figure 2.6. In this model, each user has an RBM with the movies they rated. The number of latent units is fixed for all users, and the weights and biases are shared, that is all RBMs use the same weights and biases between hidden units and the visible movie softmax unit. The binary latent units and softmax values related to the rated movies for each user differ for each RBM.

The conditional multinomial and conditional Bernoulli distribution of the observed rating matrix and latent hidden units respectively are given in Equations 2.4

and 2.5. $\mathbf{V}$ is a K x m binary indicator matrix, where K is the scale of the rating and m is the number of movies the user rated. v_i^k is 1, if the user rated movie i with a rating of k, 0 otherwise. $\mathbf{h}$ is binary latent feature vector, W_{ij}^k is the weight for movie i, feature j and rating k, b_i^k is the bias for movie i and rating k. F is the number of hidden units.

$$p(v_i^k|\mathbf{h}) = \frac{\exp(b_i^k + \sum_{j=1}^{F} h_j W_{ij}^k)}{\sum_{l=1}^{K} \exp(b_i^l + \sum_{j=1}^{F} h_j W_{ij}^l)} \tag{2.4}$$

$$p(h_j = 1|\mathbf{V}) = \sigma(b_j + \sum_{i=1}^{m}\sum_{k=1}^{K} v_i^k W_{ij}^k) \tag{2.5}$$

The marginal distribution over $\mathbf{V}$ is as follows:

$$p(\mathbf{V}) = \sum_{h} \frac{\exp(-E(\mathbf{V}, \mathbf{h}))}{\sum_{\mathbf{V}', \mathbf{h}'} \exp(-E(\mathbf{V}', \mathbf{h}'))} \tag{2.6}$$

$E(\mathbf{V}, \mathbf{h})$ is the energy term:

$$E(\mathbf{V}, \mathbf{h}) = -\sum_{i=1}^{m}\sum_{j=1}^{F}\sum_{k=1}^{K} W_{ij}^k h_j v_i^k + \sum_{i=1}^{m} \log Z_i - \sum_{i=1}^{m}\sum_{k=1}^{K} v_i^k b_i^k - \sum_{j=1}^{F} h_j b_j$$

Here, $Z_i = \sum_{l=1}^{K} \exp(b_i^l + \sum_j h_j W_{ij}^l)$ is the normalization term, which restricts the condition $\sum_{l=1}^{K} p(v_i^l = 1|\mathbf{h}) = 1$.

The proposed learning algorithm of this model is derived by gradient descent in the log-likelihood of the marginal distribution on $\mathbf{V}$ to update weights and biases. Gradient ascent parameter update can be derived from the marginal distribution given in Equation 2.6. The derivation lead to the following update rule [3]:

$$\begin{aligned} \nabla W_{ij}^k &= \epsilon \frac{\partial \log p(\mathbf{V})}{\partial W_{ij}^k} \\ &= \epsilon(\langle v_i^k h_j \rangle_{data} - \langle v_i^k h_j \rangle_{model}) \end{aligned}$$

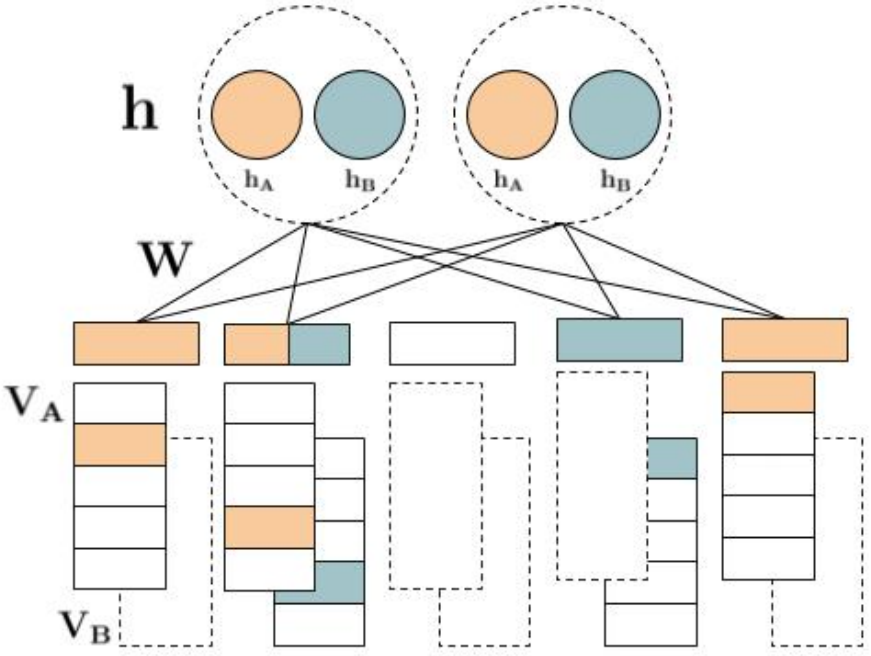

Figure 2.6. RBM model for two users and five movies [3] The number of binary latent nodes are the same but values are distinct for each user. The weights are the same between the latent nodes and the corresponding movie. If a movie is rated for a user, that weight is used. Different colors indicate different users.

$\langle v_i^k h_j \rangle_{model}$ is the expectation according to the model distribution, model statistics, and it cannot be calculated analytically less than exponential time. So, they used an objective function called "Contrastive Divergence" [27]. CD is the difference between two Kullback-Leibler divergences with one missing term in the objective function [28].

$$\nabla W_{ij}^k = \epsilon(\langle v_i^k h_j \rangle_{data} - \langle v_i^k h_j \rangle_T)$$

$\langle v_i^k h_j \rangle_T$ is the Tth sample from Gibbs sampling, started from a data point. Model samples drawn with Gibbs sampling are also called negative samples, where data points are called positive samples. Instead of calculating expectation, a point estimate is performed. The gradient of the log-likehood of the marginal distribution on $\mathbf{V}$ turns out to be the difference between the actual data and the generated data from the

model. In practice, it turns out that only one sample from Gibbs sampling is sufficient for getting reasonable evaluations [3].

Variations of this model are also proposed. RBM's with Gaussian hidden units, where latent units are h are Gaussian variables. In Netflix dataset, the information of the complete set of movies which the user rated is given in advance. The training data contains the rating values of a subset of the rated movies, but not the rates of the remaining movies. In conditional RBM, the authors make use of this advantage and they define a joint distribution over $\mathbf{V}$ and $\mathbf{h}$ conditional on $\mathbf{r}$. This is the usage of conditional RBM's on Netflix dataset. Conditional factored RBM's are factorization of $\mathbf{W}$, in two low-rank matrices in order to reduce the number of free parameters. This allows the model to converge much faster than the unfactorized conditional RBM.

2.4. Latent Dirichlet Allocation (LDA)

Latent Dirichlet allocation is a topic modeling technique where the topics of the documents are learnt in an unsupervised approach. It is a generative model, mostly used on text corpora for corpus exploration, document classification, and information retrieval [4, 7]. It is a mixed-membership model for a collection of discrete data. A document in the corpus is considered as related to different topics. Each topic has a probability distribution on the vocabulary, and each word in a document comes from a topic and considered as drawn from the vocabulary according to the topic's distribution on the vocabulary.

The generative model of LDA is as follows:
For each item j, draw words of the context as follows:

(i) Draw topic proportions for item j:

$$\theta_j \sim Dirichlet(\alpha);$$

(ii) For each word w_{js}:

(a) Draw topic assignment for the related word:

$$z_{js} \sim Mult(\theta_j)$$

(b) Draw the word:

$$w_{js} \sim Mult(\beta_{z_{js}})$$

LDA is a hierarchical model, as shown in Figure: 2.7. In LDA, learned topic proportion vector allows representing a document as a low dimensional topic vector. Topic distribution stands for latent factors of the document.

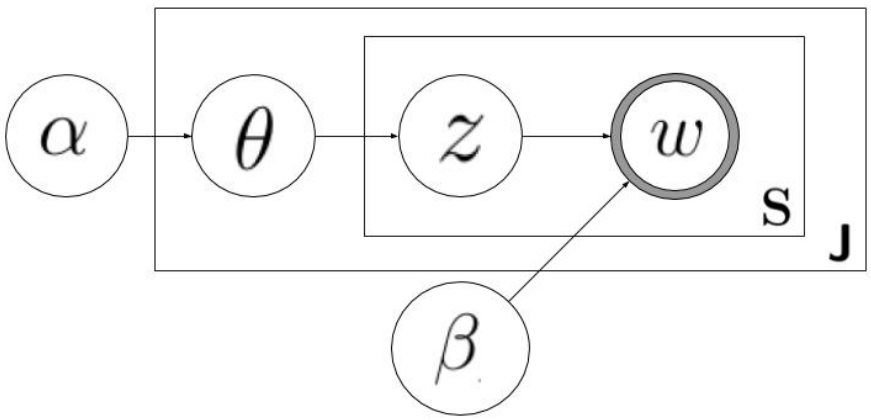

Figure 2.7. Graphical model of LDA [4]

Model parameters can be estimated by variational EM as described in [4].

2.5. Autoencoders

Autoencoders are unsupervised learning approaches used to extract useful information from data. For example, they can be used for pretraining data for a classification task to increase performance. Traditionally, an encoder is a deterministic function,

$\mathbf{y} = f(\mathbf{x})$, that maps $\mathbf{x}$ to $\mathbf{y}$. An decoder is a deterministic function, $\mathbf{z} = g(\mathbf{y})$, that maps $\mathbf{y}$ to $\mathbf{z}$, where $\mathbf{z}$ is typically the mean of distribution that is considered to generate $\mathbf{x}$ in high probability. [5]

If the dimension of $\mathbf{y}$, d', is less than the dimension of $\mathbf{x}$, d, the autoencoder is called an under-complete autoencoder. On the other hand, if d' is greater than d, it is an overcomplete autoencoder. In most cases, traditionally, an under-complete autoencoder is used for dimension reduction [29].

Recently, encoders and decoders are generalized to stochastic mappings $p_{enc}(\mathbf{y}|\mathbf{x})$ and $p_{dec}(\mathbf{x}|\mathbf{y})$ rather than deterministic functions. The aim is to prevent learning identity function and learn useful features for the task in hand [29].

The general form of an autoencoder is using an affirm transformation on the data and adding nonlinearity. The reconstruction form is similar, applying an affine transformation and adding nonlinearity.

$$\mathbf{y} = f(\mathbf{x}) = s(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{2.7}$$

$$\mathbf{z} = g(\mathbf{y}) = s(\mathbf{W}'\mathbf{y} + \mathbf{b}') \tag{2.8}$$

$s(.)$ is a nonlinear function like a sigmoid or tanh. If the nonlinearity is not introduced and loss function is squared loss error, the model would be principal component analysis (PCA). [5]

2.5.1. Denoising Autoencoders (DAE)

Denoising autoencoders aim to reconstruct the original data from the noisy version of the data. The idea is to extract useful features while reconstructing the input from its corrupted version. A schematic representation is given in Figure 2.8. $\mathbf{x}^-$ denotes the corrupted version of $\mathbf{x}$ [5].

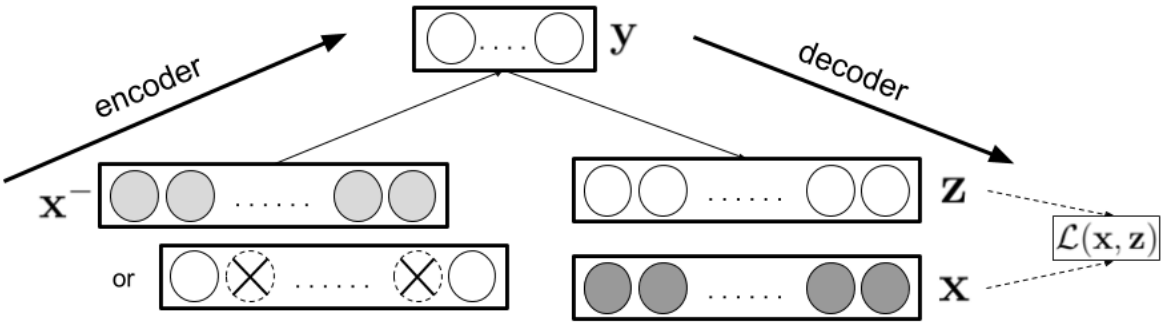

Figure 2.8. Denoising Autoencoder [5]- $\mathcal{L}(\mathbf{x}, \mathbf{z})$ is the associated loss function, which is taken as proportional to the negative log-likelihood of the input, given the representation, $p(\mathbf{x}|\mathbf{z})$.

$$\mathbf{y} = f(\mathbf{x}^-) : f \rightarrow \text{encoder function}$$
$$\mathbf{z} = g(\mathbf{y}) : g \rightarrow \text{decoder function}$$
$$\mathcal{L}(\mathbf{x}, \mathbf{z}) \sim -\log p(\mathbf{x}|\mathbf{z})$$

In [5], three types of noise are introduced for input corruption. An isotropic Gaussian noise can be added, $\mathbf{x}^-|\mathbf{x} \sim \mathcal{N}(\mathbf{x}, \sigma^2\mathbf{I})$. Second alternative, masking noise, is to set randomly chosen elements of $\mathbf{x}$ to 0 with a given fraction γ. Lastly, salt-and-pepper noise can be used. In this case, randomly chosen elements of $\mathbf{x}$, again with a fraction of γ, is set to maximum or minimum value according to a fair coin. For example, 0 or 1 [5].

The objective is to maximize the likelihood of reconstructing the input, given the representation. The likelihood $p(\mathbf{x}|\mathbf{z})$, can be chosen according to the input: [5]

- If $\mathbf{x} \in \mathcal{R}^d$, that is $X|\mathbf{z} \sim \mathcal{N}(\mathbf{z}, \sigma^2\mathbf{I})$, $\mathcal{L}(\mathbf{x}, \mathbf{z})$:
  Objective function to minimize is $\mathcal{L}(\mathbf{x}, \mathbf{z}) \sim \| \mathbf{x} - \mathbf{z} \|^2$, which is the squared error objective.
- If $\mathbf{x}$ is binary, $\mathbf{x} \in \{0, 1\}^d$, that is $X|\mathbf{z} \sim \mathcal{B}(\mathbf{z})$:
  $\mathbf{z}$ should also be in $[0, 1]^d$, so using a sigmoid function for activation is appropriate. Cross-entrophy loss is the objective to minimize.

$$\mathcal{L}(\mathbf{x}, \mathbf{z}) = -\sum_j [\mathbf{x}_j \log(\mathbf{z}_j) + (1 - \mathbf{x}_j) \log(1 - \mathbf{z}_j)]$$

Other settings can also be used in autoencoders [5].

Geometric interpretation of DAEs is learning a lower dimensional nonlinear manifold [5]. The corrupted values would be farther to the manifold and the stochastic operator will learn to map them on the manifold.

2.5.2. Stacked Denoising Autoencoders (SDAE)

Stacked denoising autoencoders are deep structures, where denoising autoencoders are stacked one on another. The procedure is similar to other similar deep structures like stacking RBMs in Deep Belief Networks [30]. The corrupted input is used for training the first layer. For the second layer's training, the input is given without corruption and the corresponding representation for the first layer is corrupted for the second layer training. And this is applied to as many layers as the structure is designed. After each layer is trained, a finetuning is applied to whole network [5].

2.5.3. Probabilistic Stacked Denosing Autoencoder (PSDAE)

In probabilistic version of SDAE, the transformation is stochastic [6]. The transformations given on equations 2.7 and 2.8 basically becomes as follows:

$$\mathbf{y}|\mathbf{x}^- \sim \mathcal{N}(\mathbf{W}\mathbf{x}^- + \mathbf{b}, \lambda_w^{-1}\mathbf{I}_k)$$
$$\mathbf{z}|\mathbf{y} \sim \mathcal{N}(\mathbf{W}'\mathbf{y} + \mathbf{b}', \lambda_w^{-1}\mathbf{I}_d)$$

$\mathbf{I}_k$ and $\mathbf{I}_d$ are identity matrices with dimensions of $\mathbf{y}$ and $\mathbf{x}$, respectively. λ_w is a known precision for the Gaussian distribution. The model can be defined by a generative process as follows: [6]

(i) Draw the weight, bias and node values for each layer l of the SDAE,

(a) Draw for each $n.$ column of the weight matrix, $\mathbf{W}_l$, and bias of the layer,

$$\mathbf{W}_{l,*n} \sim \mathcal{N}(\mathbf{0}, \lambda_w^{-1}\mathbf{I}_{K_l})$$
$$\mathbf{b}_l \sim \mathcal{N}(\mathbf{0}, \lambda_w^{-1}\mathbf{I}_{K_l})$$

(b) Draw each row of $\mathbf{X}_l$,

$$\mathbf{X}_{l,j*} \sim \mathcal{N}(\sigma(\mathbf{X}_{l-1,j*}\mathbf{W}_l + \mathbf{b}_l), \lambda_s^{-1}\mathbf{I}_{K_l})$$

(ii) For each item, j,

(a) Draw clean input,

$\mathbf{X}_{c,j*} \sim \mathcal{N}(\mathbf{X}_{L,j*}, \lambda_n^{-1}\mathbf{I}_S)$

As shown in Figure: 2.10, the middle layer is the bottleneck layer to extract useful features as a dimension reduction technique. Namely, for a L layered SDAE, $\mathbf{X}_0$ is the corrupted input, $\mathbf{X}_{L/2}$ is the bottleneck layer, and $\mathbf{X}_L$ is the reconstructed input. The precision values, λ_w, λ_s, and λ_n are hyperparameters. The graphical model of the SDAE is shown in Figure: 2.9.

In the model, all parameters can be considered as random variables. Joint log-likelihood of the parameters with given hyperparameters can be used to maximize the posterior distribution [6, 7].

$$\begin{aligned}\mathcal{L} &= p(\{\mathbf{X}_l\}, \mathbf{X}_c, \{\mathbf{W}_l\}, \{\mathbf{b}_l\}|\lambda_s, \lambda_w, \lambda_n) \\ &= -\frac{\lambda_w}{2}\sum_l(\|\mathbf{W}_l\|_F^2 + \|\mathbf{b}_l\|_2^2) \\ &\quad -\frac{\lambda_n}{2}\sum_j\|\mathbf{X}_{L,j*} - \mathbf{X}_{c,j*}\|_2^2 \\ &\quad -\frac{\lambda_s}{2}\sum_l\sum_j\|\sigma(\mathbf{X}_{l-1,j*}\mathbf{W}_l + \mathbf{b}_l) - \mathbf{X}_{l,j*}\|_2^2\end{aligned}$$

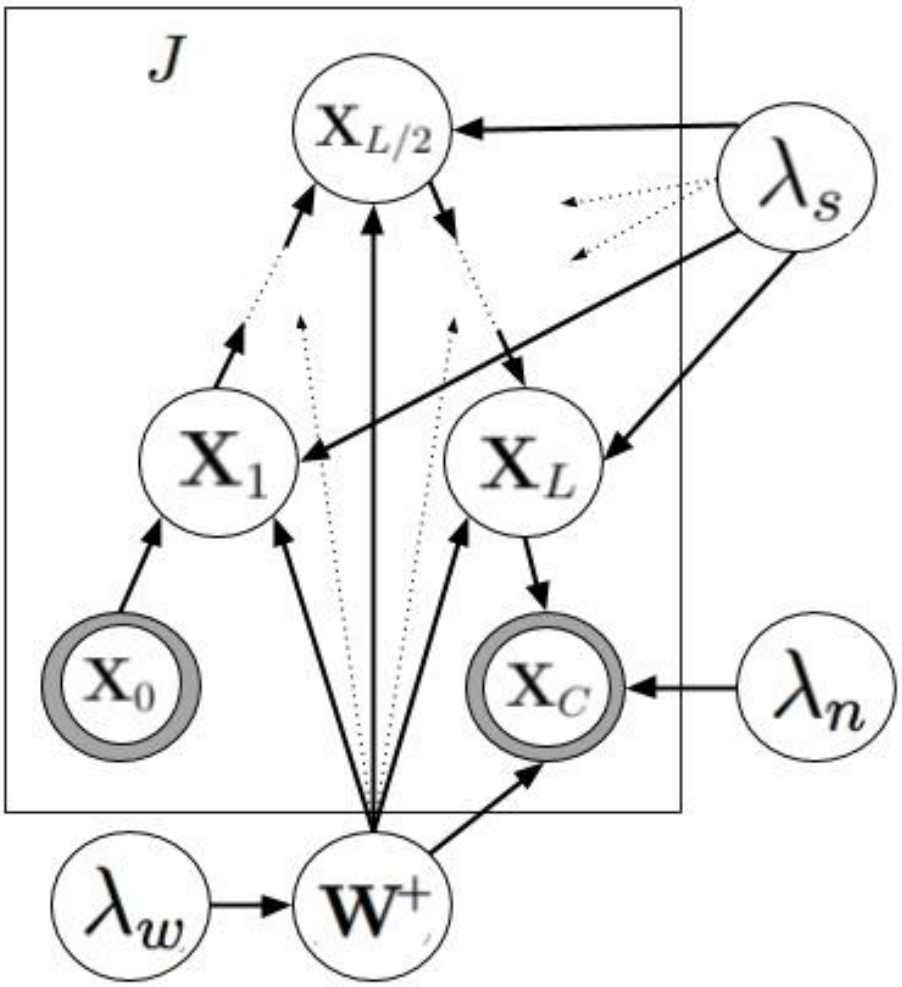

Figure 2.9. Graphical model of SDAE [6]

The weights and biases can be learned by back propagation with respect to $\mathbf{W}_l$ and $\mathbf{b}_l$.

As explained in Section 3.3, PSDAE can be used to reduce the dimensionality of the content information of the item.

2.6. Evaluation of Recommendation Systems

In order to compare different systems precise metrics should be set, however, in this field, there are different approaches regarding how to evaluate an RS. In this

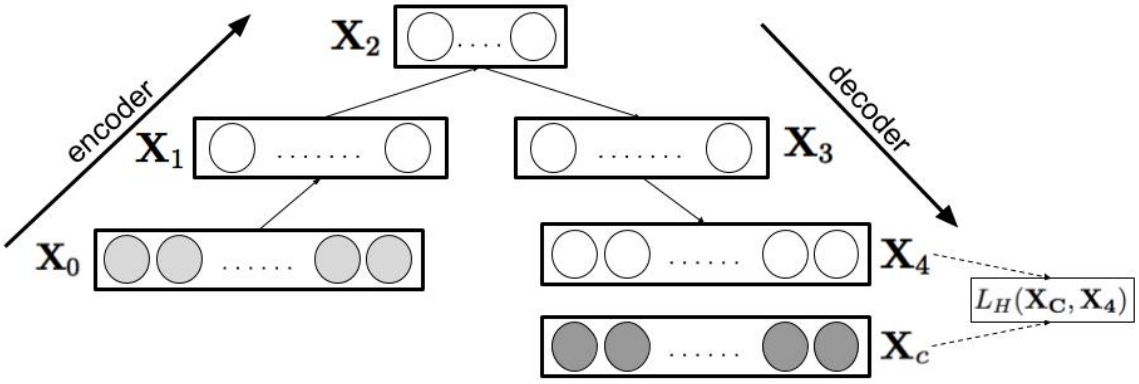

Figure 2.10. A representation of SDAE [6]

section, we will briefly outline the evaluation of recommendation systems.

In the early stages of the recommendation system researches, most of the evaluation paradigms were based on the accuracy of the predicted ratings [31]. Accuracy metrics basically measure how close are the system's predictions to actual user ratings. They can be categorized into thre classes [31]. Prediction Accuracy metrics can evaluate the recommendation system when it tries to predict the user's exact rating. Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and their derivations such as normalized MAE, normalized RMSE, average MAE, average RMSE are predictive accuracy metrics [12, 31].

RMSE and MAE are defined as follows:

$$\text{RMSE} = \sqrt{\frac{1}{\parallel \mathcal{T} \parallel} \sum_{(i,j)\in\mathcal{T}} (r^*_{ij} - r_{ij})^2}$$

$$\text{MAE} = \sqrt{\frac{1}{\parallel \mathcal{T} \parallel} \sum_{(i,j)\in\mathcal{T}} \parallel r^*_{ij} - r_{ij} \parallel}$$

Classification Accuracy metrics, or usage prediction metrics [12] classify the items as relevant or irrelevant, that is evaluating the prediction of the items that the user is

Table 2.1. Notation of the related item properties for classification accuracy

	Recommended	Not Recommended	Total
User Interested	N_{ri}	N_{ni}	N_i
User Not Interested	N_{ru}	N_{nu}	N_u
Total	N_r	N_u	N

interested or not [12, 31]. Three main measures are Precision, Recall and Receiver Operating Characteristics (ROC) Curves.

$$\text{Recall} = \frac{N_{ri}}{N_i}$$

$$\text{Precision} = \frac{N_{ri}}{N_r}$$

The notation for possible rating characteristics and for totals is shown in Table 2.1. Recall is the ratio between the number of relevant recommended items and user's all relevant items. Precision is the ratio between the number of relevant recommended items and all recommended items. ROC Curve measures true positive and false positive rates. After ordering the recommended items from most relevant predictions to less relevant ones, the curve is drawn from the origin. If the recommended item is relevant the curve is drawn vertical if it is not relevant it is drawn horizontal to right. If the relevance is not known, no action is taken. ROC curves are more suitable when a false recommendation is costly [12, 31]. If the order of the recommendation is important Rank Accuracy Metrics give more effective measures.

In recommendation systems, accuracy is not the only metric to measure the effectiveness of recommendations. Quality and usefulness of the predictions are also important. Coverage, confidence, novelty, diversity, utility, scalability, adaptivity are some alternative evaluation views other than accuracy [12].

(i) Coverage can be considered in three ways, item space coverage, userspace coverage or cold start. Item space coverage measure favors recommendations which cover a high proportion of the catalog, where userspace coverage favors recommendations which cover a wide range of users. Cold start problem occurs when a new item or user is introduced to the system as mentioned in Chapter 1.

(ii) Confidence is the measure of a system where how confident it is on the recommendation it made. For example, the system may predict the top rating for two items, the order of the items may be decided according to the confidence level of the system.

(iii) Novelty is for measuring how the recommended items are novel for the users. It can be measured by user questionnaire, or the data can be organized in time and some previously rated items can be hidden and the system would get a rewarding feedback for new recommendations but will be punished for the recommendations that are previously rated but hidden to the system.

(iv) Diversity metric is to evaluate the system in terms of diversity of items. The accuracy is also important, but recommending diverse items rather than similar items may become very valuable in some domains. The item-item similarity metric is taken into consideration besides the accuracy.

(v) Utility of the recommendations may be important in some systems. For example, it is more profitable to match recommendations with high profit and lead to cross-selling in e-commerce platforms.

(vi) Scalability of the system is another crucial point of a recommendation system. This metric is to evaluate the system's performance as the data grows.

(vii) Adaptivity is the sensitivity to the changes in the item catalog or user preferences. For example, when there is a mass shooting incident, the older news on the previous mass shooting incidents will be preferable again, while they would not be in other cases.

In this chapter, we outlined some basic concepts like feedback types and evaluation methodologies in recommendation systems for completeness of our work and gave mathematical background and explained the components of the models that we are

going to discuss in the next chapter.

In the following chapter, we explained the two models that we compared in our work, Coupled Bayesian Nonnegative Matrix Factorization and Collaborative Deep Learning. We also described Collaborative Topic Regression model which is the basis model of Collaborative Deep Learning.

3. HYBRID MODELS

In Chapter 1, different kinds of RS have been mentioned briefly. Each one of them addresses a different approach to the problem. For example, collaborative filtering has been found very accurate when there are sufficient rating data on the items. However, it has the cold start problem. On the other hand, demographic-based systems may get better recommendations when there is high-quality information on the users. The motivation behind the hybrid models in RS is to integrate different approaches in one system so that the advantage of one approach can overcome the setback of the other one and vice versa [12].

There are different ways to combine different approaches, as categorized in [13]. Different recommendation systems can be combined in a (i) weighted score; the system may (ii) switch between recommendation systems; a (iii) mixed set of recommendations can be presented to users; features may be extracted from different methods and (iv) combined features are used in the actual recommendation component; one recommender may be used to extract features and (v) augmented features are fed to the actual recommender; in (vi) cascade systems different techniques are used for the same data and each has a given priority and in case of a tie in high priority systems, low priority techniques determine the results; in (vii) meta-level hybrid systems one recommender technique's output is given as input to another technique [13].

The hybrid approaches we concentrated on can be categorized as feature augmentation technique, though they are tightly coupled models [6,13]. In tightly coupled methods, features extracted from one method affect the other method's features and vice versa. That is, the information of one method is shared and affect the other one. The knowledge transfer is bidirectional. On the other hand, if just one method extracts the features and makes use of the other one, it is called loosely coupled methods [6].

We concentrated on tightly coupled methods in our work. Three different methods are explained in the following sections.

3.1. Coupled Bayesian Nonnegative Matrix Factorization (CBNMF)

In this approach, we used BNMF technique explained in Section 2.3.4 by coupling the rating matrix with the content matrix. Schematic representation of CBNMF is shown in Figure 3.1.

U, **V** and **R** are the user feature matrix, item feature matrix, and rating matrix respectively, as defined in Section 2.2. **X** is the bag-of-words SxJ matrix representation of the content. The most informative top S words are taken into account. We define $\mathbf{Z} = \{\mathbf{R}, \mathbf{X}\}$ as the concatenation of rating matrix and content matrix, and $\mathbf{T} = \{\mathbf{U}, \mathbf{Y}\}$ as the concatenation of user feature matrix and content feature matrix. The generative model of CBNMF becomes as follows:

$$t_{p,k} \sim \mathcal{G}(t_{p,k}; a^t_{p,k}, \frac{b^t_{p,k}}{a^t_{p,k}}), \qquad v_{k,j} \sim \mathcal{G}(v_{k,j}; a^v_{k,j}, \frac{b^v_{k,j}}{a^v_{k,j}}), \tag{3.1}$$

$$s_{p,k,j} \sim \mathcal{PO}(s_{p,k,j}; t_{p,k}v_{k,j}), \qquad z_{p,j} = \sum_k s_{p,k,j}. \tag{3.2}$$

In [2], Cemgil defines mask matrix, in order to omit the missing values. On the other hand, in our data, we do not know whether 0 entries are missing or they are not preferred, and we have a complete content matrix which we do not want to take into consideration as much as rating data, we used a fixed weight matrix W instead of a mask matrix. It functions like confidence parameters as in [7]. In the algorithm stated in Figure 2.5, **M** was the mask matrix. If r_{ij} is not missing m_{ij} is 1, otherwise it is 0. In CBNMF, weight matrix **W** has three different values, as follows:

$$w_{p,j} = \begin{cases} a & p <= I \text{ and } r_{pj} = 1 \\ b & p <= I \text{ and } r_{pj} = 0 \\ c & \text{otherwise, that is } p > I \end{cases}$$

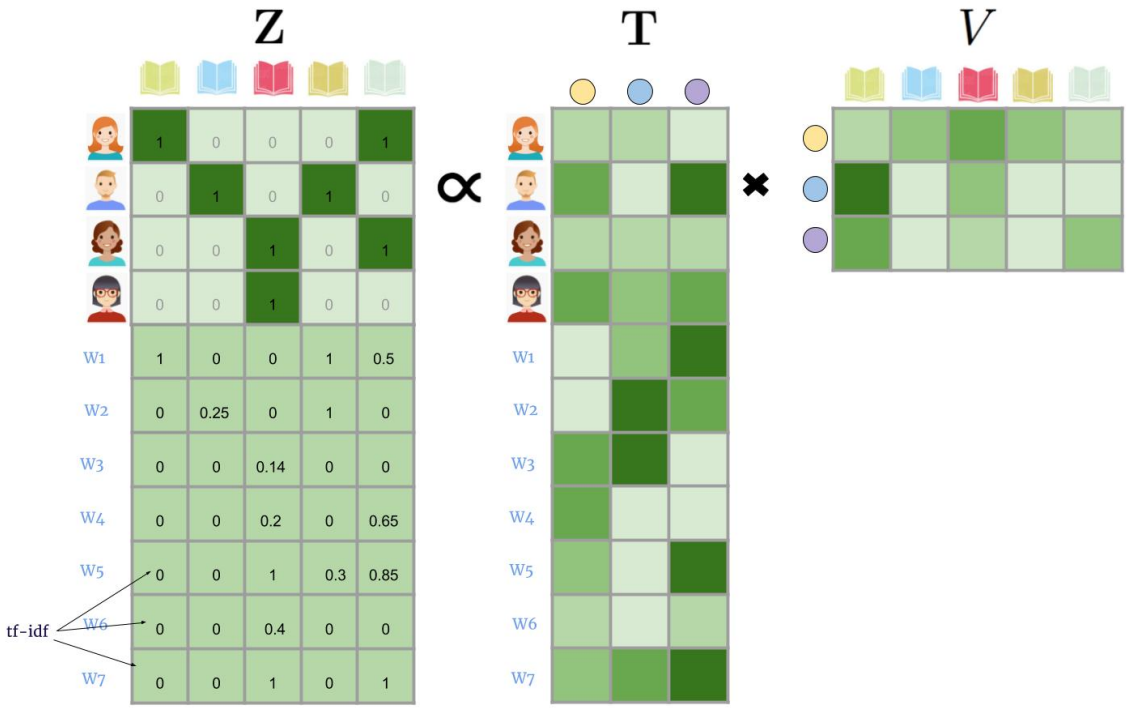

Figure 3.1. Schematic representation of coupled NMF. The colours on the matrix **Z** indicate weight values w_{pj}. Dark green is a, light green is b and mid green is c.

3.2. Collaborative Topic Regression (CTR)

Collaborative Topic Regression (CTR) uses probabilistic matrix factorization as latent factor model for rating analysis, and Latent Dirichlet Allocation (LDA) as a probabilistic topic model for content analysis [7]. The graphical model of CTR is given in Figure 3.2.

The generative model of LDA and PMF were given on Section: 2.4 and 2.3.3. The generative process of CTR is as follows:

(i) Draw a latent user vector for each user i, $\mathbf{u}_i \sim \mathcal{N}(0, \lambda_u^{-1}\mathbf{I}_K)$.

(ii) For each item j,

(a) Draw topic proportions $\boldsymbol{\theta}_i \sim Dirichlet(\alpha)$,

(b) Draw item latent offset $\boldsymbol{\epsilon}_j$ and get the item latent vector $\mathbf{v}_j$ as,

$\boldsymbol{\epsilon}_j \sim \mathcal{N}(0, \lambda_v^{-1}\mathbf{I}_K)$

$\mathbf{v}_j = \boldsymbol{\epsilon}_j + \boldsymbol{\theta}_j$

(c) For each word w_{js},

i. Draw topic assignment z_{js},

$z_{js} \sim Mult(\theta)$

ii. Draw word w_{js},

$w_{js} \sim Mult(\beta_{z_{js}})$

(iii) For each user-item pair (i, j), draw the rating,

$r_{ij} \sim \mathcal{N}(\mathbf{u}_i\mathbf{v}_j, c_{ij}^{-1})$

K is the number of topics and latent features for item feature vectors and user feature vectors. λ_u, λ_v and α are hyperparameters. The precision of the distribution for r_{ij}, c_{ij}, is confidence parameter. As c_{ij} grows, the surer we are of the value of r_{ij}. v_j is the combination of topic proportion and a latent offset. This allows adjusting the weight of the content in the recommendation. If there is no rating information on the item j, offset is 0. All we have is based on the content information. As the amount of rating for item j increases, that is more users are rated item j, we take rating information more into consideration than the content topic proportions [7].

Full posterior of the parameters $\mathbf{u}_i$, $\mathbf{v}_j$ and $\boldsymbol{\theta}_j$ is intractable. The authors developed an EM-style algorithm to learn MAP estimates. Maximizing the log likelihood of $\mathbf{U}$, $\mathbf{V}$, $\theta_{1:J}$ and $\mathbf{R}$ given λ_u, λ_v and β is equivalent the maximization of the posterior.

$$\begin{aligned}\mathcal{L} = &-\frac{\lambda_u}{2}\sum_i \mathbf{u}_i\mathbf{u}_i^T - \frac{\lambda_v}{2}\sum_j (\mathbf{v}_j - \boldsymbol{\theta}_j)^T(\mathbf{v}_j - \boldsymbol{\theta}_j) \\ &+ \sum_j \sum_s \log(\sum_k \theta_{jk}\beta_{kw_{js}}) - \sum_i \sum_j \frac{c_{ij}}{2}(r_{ij} - \mathbf{u}_i\mathbf{v}_j)^2\end{aligned}$$

U and V can be updated by fixing θ_j via gradient descent. Learning the topic proportions θ_j can be made by setting a lower bound on $\mathcal{L}(\theta_j)$ as described in [7], and using projection gradient. Prediction are made by point estimate of u_i, θ_j and ϵ_j to

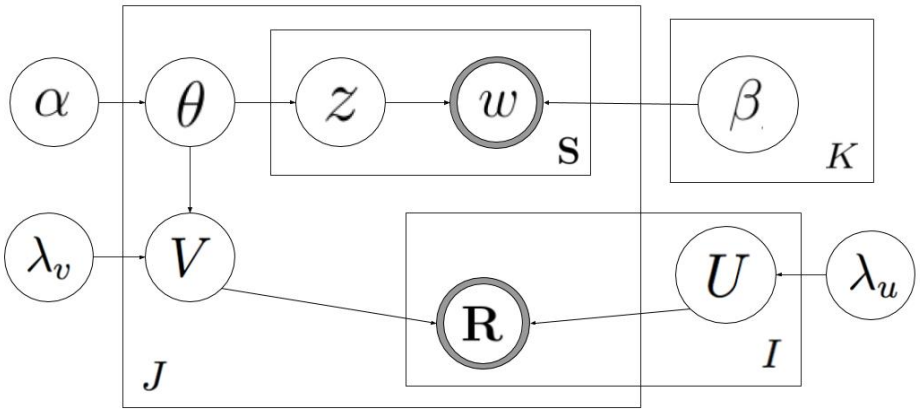

Figure 3.2. Graphical model of CTR [7]

approximate their expectations as stated in [7],

$$r_{ij}^* \approx \mathbf{u}_i^*(\boldsymbol{\theta}_j^* + \boldsymbol{\epsilon}_j^*) = \mathbf{u}_i^* \mathbf{v}_j^*$$

3.3. Collaborative Deep Learning (CDL)

Collaborative deep learning (CDL) is a hybrid model where the ratings are factorized by PMF and the content information is modeled via PSDAE [6]. The extracted features of the content data and the item vectors are combined by a bias vector as similar in the collaborative topic regression model as explained in Section 3.2. The graphical representation of the model is given in Figure 3.3. The generative process of the model is as follows:

(i) For each layer l of PSDAE, draw
   (a) $\mathbf{W}_{l,*n} \sim \mathcal{N}(\mathbf{0}, \lambda_w^{-1}\mathbf{I}_{K_l})$
   $\mathbf{b}_l \sim \mathcal{N}(\mathbf{0}, \lambda_w^{-1}\mathbf{I}_{K_l})$
   (b) $\mathbf{X}_{l,j*} \sim \mathcal{N}(\sigma(\mathbf{X}_{l-1,j*}\mathbf{W}_l + \mathbf{b}_l), \lambda_s^{-1}\mathbf{I}_{K_l})$

(ii) For each item j, draw,
   (a) $\mathbf{X}_{c,j*} \sim \mathcal{N}(\mathbf{X}_{L,j*}, \lambda_n^{-1}\mathbf{I}_S)$

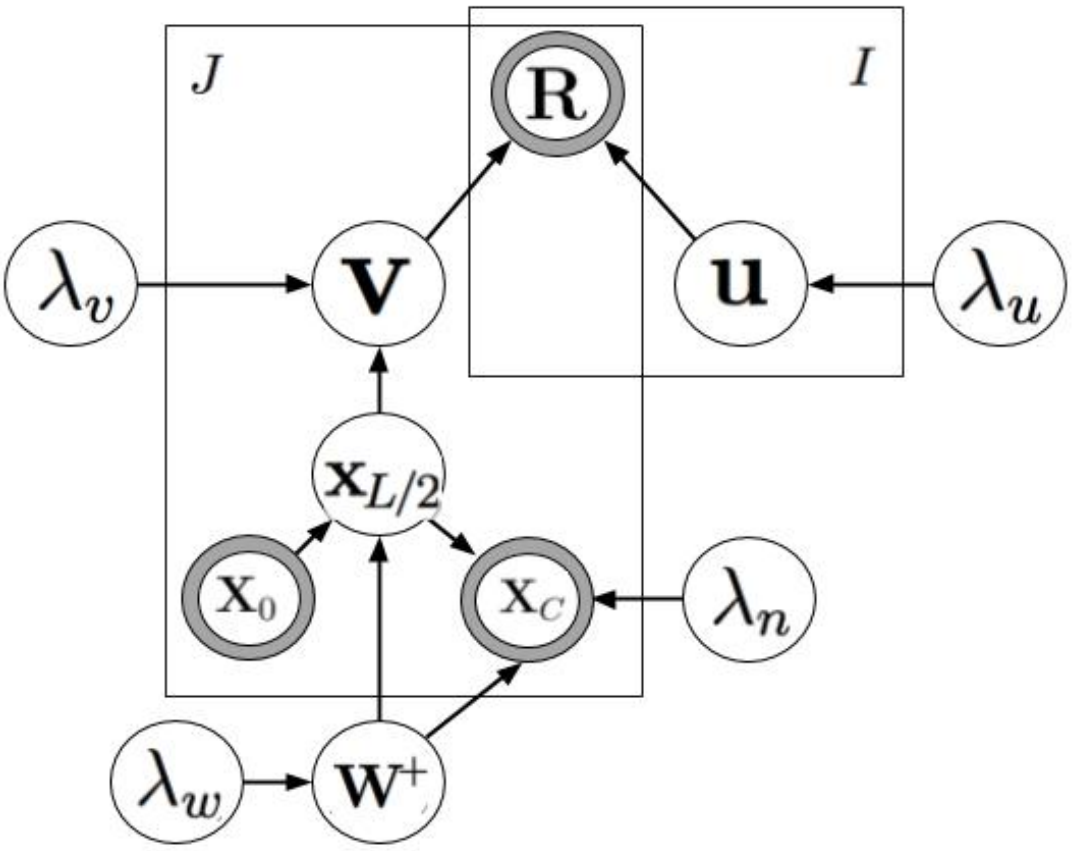

Figure 3.3. Graphical model of CDL [6]

(b) Draw a bias column vector, and set the latent item vector accordingly,

$\epsilon_j \sim \mathcal{N}(\mathbf{0}, \lambda_v^{-1}\mathbf{I}_K)$

$\mathbf{v}_j = \epsilon_j + \mathbf{X}^T_{\frac{L}{2},j*}$

(c) Draw a latent user row vector for each user i,

$\mathbf{u}_i \sim \mathcal{N}(\mathbf{0}, \lambda_u^{-1}\mathbf{I}_K)$

(d) For each user and item pair (i, j), draw a rating

$\mathbf{R}_{i,j},\ \mathbf{R}_{i,j} \sim \mathcal{N}(\mathbf{u}_i\mathbf{v}_j, \mathbf{C}_{ij}^{-1})$

λ_u, λ_v, λ_w, λ_s and λ_n are hyperparameters. The rating information of I users related to J items is represented with IxJ matrix $\mathbf{R}$. $\mathbf{X}_o$, the corrupted version of the input matrix $\mathbf{X}_c$, is the input of PSDAE. $\mathbf{R}_{ij}$ is a binary matrix, where $\mathbf{R}_{ij} = 1$ if the user i added to her or his library, 0 otherwise. $\mathbf{X}_l$ is the JxK_l output matrix of the l.layer of PSDAE. L is the number of all layers. Since the feature extraction is in the bottleneck layer, which is the $L/2$. layer, an L layered SDAE is considered as a $L/2$ level PSDAE. $\mathbf{W}_l$ and $\mathbf{b}_l$ are the weights and biases respectively. $\mathbf{W}^+$ represents the weights and biases of all layers. $\mathbf{C}_{ij}$ is the confidence parameter as stated in [7], if $\mathbf{R}_{ij} = 1$, $\mathbf{C}_{ij} = a$, $\mathbf{C}_{ij} = b$ otherwise. It indicates how much $\mathbf{R}_{ij}$ is reliable. λ_s hyperparameter is considered as infinite in order to simplify the calculations. That is, the Gaussian distribution converges to Dirac delta function. In this model, the collaborative filtering part utilizes the extracted information from the content data and the features of the content data are forced to enhance the performance of the PMF.

Training of the model is based on an EM based algorithm of the maximum a posteriori estimate. As stated in [7], maximization of the posterior distribution is to maximize the joint log-likelihood of the parameters $\mathbf{U}, \mathbf{V}$,$\{\mathbf{X}_l\}$, $\mathbf{X}_c$, $\{\mathbf{W}_l\}$,$\{\mathbf{b}_l\}$,$\mathbf{R}$, given the hyperparameters λ_u, λ_v, λ_w, λ_s and λ_n.

As λ_s goes to infinity the likelihood becomes as follows:

$$\begin{aligned}\mathcal{L} = &-\frac{\lambda_u}{2}\sum_i \|\mathbf{u}_i\|_2^2 - \frac{\lambda_w}{2}\sum_l (\|\mathbf{W}_l\|_F^2 + \|\mathbf{b}_l\|_2^2) \\ &- \frac{\lambda_v}{2}\sum_j \|\mathbf{v}_j - f_e(\mathbf{X}_{0,j*}, \mathbf{W}^+)^T\|_2^2 \\ &- \frac{\lambda_n}{2}\sum_j \|f_r(\mathbf{X}_{0,j*}, \mathbf{W}^+) - \mathbf{X}_{c,j*}\|_2^2 \\ &- \sum_{i,j} \frac{\mathbf{C}_{ij}}{2}(\mathbf{R}_{ij} - \mathbf{u}_i\mathbf{v}_j)^2\end{aligned}$$

The full expression is given in [6].

$f_e(.)$ is the encoder function, which takes the corrupted version of the content data as input and provides the encoding of the data, which is the $L/2$. layer of the PSDAE. $f_r(.)$ is the reconstruction function, which takes the corrupted input, encodes it and then reconstructs the uncorrupted content data, that is basically the PSDAE.

For given $\mathbf{W}^+$, $\mathbf{u}_i$ and $\mathbf{v}_j$ can be updated by coordinate ascent as in [7] and [32] as follows:

$$\begin{aligned}\mathbf{u}_i^T &\leftarrow (\mathbf{V}\mathbf{C}_i\mathbf{V}^T + \lambda_u\mathbf{I}_K)^{-1}\mathbf{V}\mathbf{C}_i\mathbf{R}_i \\ \mathbf{v}_j &\leftarrow (\mathbf{U}\mathbf{C}_j\mathbf{U}^T + \lambda_v\mathbf{I}_K)^{-1}(\mathbf{U}\mathbf{C}_j\mathbf{R}_j + \lambda_v f_e(\mathbf{X}_{0,j*}, \mathbf{W}^+)^T)\end{aligned}$$

$\mathbf{U}$ and $\mathbf{V}$ represent the feature matrices of all users and items respectively. $\mathbf{C}_i = diag(\mathbf{C}_{i1}, ..., \mathbf{C}_{iJ})$ is the diagonal matrix of the confidence values. $\mathbf{R}_i$ is the ith user's vector representing the all items' ratings for that user and $\mathbf{R}_j$ is the jth item's vector representing all users' ratings for that item.

There are two extreme sides of the hyperparameter settings. One of them occurs when λ_n/λ_v ratio reaches to positive infinity. In this case, PMF and PSDAE parts of the model disconnect from each other and the features learned by PSDAE are used directly. In the other case, when λ_n/λ_v ratio approaches to zero, PSDAE's decoder

vanishes. It is stated in [6] that in either case, the performance drops significantly.

$\mathbf{W}_l$ and $\mathbf{b}_l$ can be learned by back-propagation given $\mathbf{U}$ and $\mathbf{V}$. The gradients of the likelihood with respect to $\mathbf{W}_l$ and $\mathbf{b}_l$ are as follows:

$$\begin{aligned}\nabla_{\mathbf{W}_l}\mathcal{L} &= -\lambda_w \mathbf{W}_l \\ &- \lambda_v \sum_j \nabla_{\mathbf{W}_l} f_e(\mathbf{X}_{0,j*}, \mathbf{W}^+)^T (f_e(\mathbf{X}_{0,j*}, \mathbf{W}^+)^T - \mathbf{v}_j) \\ &- \lambda_n \sum_j \nabla_{\mathbf{W}_l} f_r(\mathbf{X}_{0,j*}, \mathbf{W}^+)(f_r(\mathbf{X}_{0,j*}, \mathbf{W}^+) - \mathbf{X}_{c,j*})\end{aligned}$$

$$\begin{aligned}\nabla_{\mathbf{b}_l}\mathcal{L} &= -\lambda_w \mathbf{b}_l \\ &- \lambda_v \sum_j \nabla_{\mathbf{b}_l} f_e(\mathbf{X}_{0,j*}, \mathbf{W}^+)^T (f_e(\mathbf{X}_{0,j*}, \mathbf{W}^+)^T - \mathbf{v}_j) \\ &- \lambda_n \sum_j \nabla_{\mathbf{b}_l} f_r(\mathbf{X}_{0,j*}, \mathbf{W}^+)(f_r(\mathbf{X}_{0,j*}, \mathbf{W}^+) - \mathbf{X}_{c,j*})\end{aligned}$$

By updating $\mathbf{U}, \mathbf{V}, \mathbf{W}_l$ and $\mathbf{b}_l$ alternatingly, the local optimum of the likelihood can be reached.

The rating of an index is to be predicted similarly as stated in [7] according to the observed test data, D.

$$E[\mathbf{R}_{ij}|D] \approx E[\mathbf{u}_i|D](E[(f_e(\mathbf{X}_{0,j*}, \mathbf{W}^+)^T|D] + E[\epsilon_j|D])$$

That is, the predicted rating can be calculated as follows:

$$\mathbf{R}^*_{ij} \approx (\mathbf{u}^*_j)(f_e(\mathbf{X}_{0,j*}, \mathbf{W}^{+*})^T + \epsilon^*_j) = \mathbf{u}^*_i \mathbf{v}^*_j$$

In this chapter, we explained two hybrid models, coupled Bayesian nonnegative matrix factorization (CBNMF) and collaborative deep learning (CDL), that we compared in our experiments and another hybrid model,collaborative topic regression (CTR), that collaborative deep learning is mainly based on.

In the following chapter, we give the details of our experiments and their results.

4. EXPERIMENTS AND RESULTS

CDL is a state-of-art hybrid model, which uses a deep structure for content feature extraction. CBNMF aims to use the advantage of BNMF in order to combine the rating data and content information. We compared these two methods in our work. We reproduced CDL and applied CBNMF to the same dataset.

In our experiments, we chose CiteUlike-a dataset [33]. CiteULike is a platform for researchers to search and bookmark the articles they are interested in. The details of the dataset are given in Section 4.1. The dataset was obtained and used in [7] at the first time. After work can be found in [6, 34, 35]. The advantage of this dataset is that it is available for researchers. There is another dataset called CiteULike-t in which there are more users and items and it is more sparse. However, we used the CiteUlike-a dataset in the scope of our work.

Another commonly used dataset for recommendation systems is Netflix dataset. Although the rating dataset is available, it is not directly applicable to the models that we are working on. It does not contain plot summary or synopsis data, which should be collected separately. This process is out of our work's scope.

4.1. CiteULike-a Dataset

As mentioned before, we used CiteUlike-a dataset. It contains 5551 users, 16980 items, and 204987 user-item pairs. The density of the data is 0.22 % [34]. User-item pair indicates that the user added that item to her/his library. Figure 4.1 shows the density of the selected, preferred articles per user. The amount of each user's preferred articles among the dataset is not balanced. 81% of the users bookmarked less than 50 articles among 16980 articles and there are very few users, just 15 users, who bookmarked more than 300 articles. The content is represented by the bag-of-words representation of the titles and abstracts of the articles. It is the bag-of-words matrix with normalized term frequency-inverse document frequency (TF-IDF) values of the top 8000 discriminating

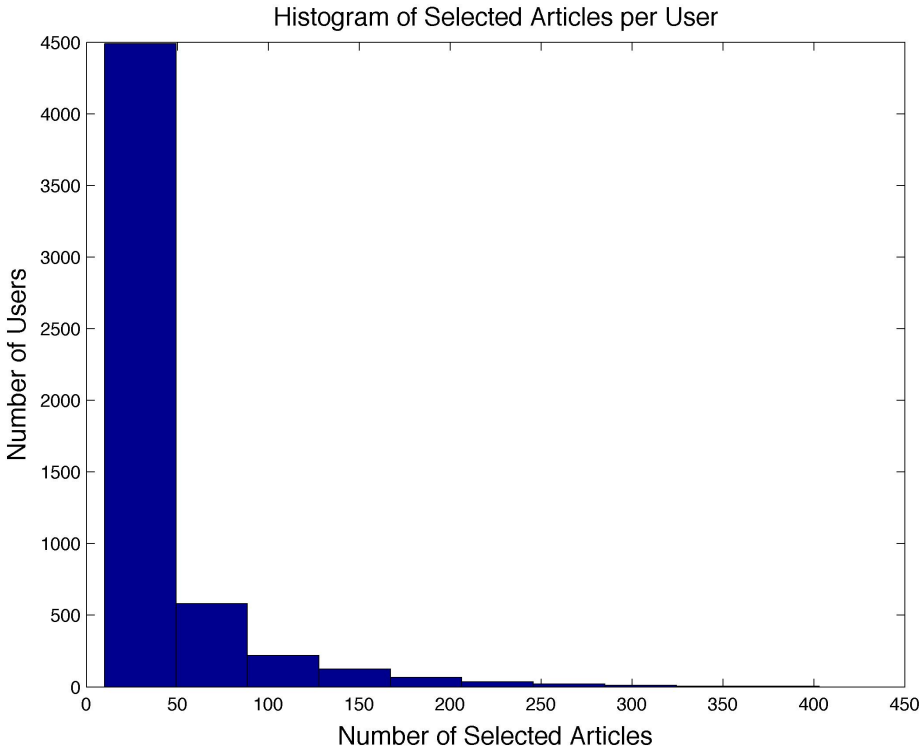

Figure 4.1. The density of the bookmarked articles per user

words according to their TF-IDF values. TF-IDF value of word i, in document j is calculated as follows:

$$tf\_idf_{ij} = tf_{ij} \log(\frac{N}{df_i})$$

$$tf_{ij} = \frac{\text{\# of word } i \text{ in document } j}{\text{\# of words in document } j}$$
$$df_i = \frac{\text{\# of documents word } i \text{ appears}}{\text{\# of all documents in corpus}}$$

tf_{if} is the frequency of word i in document j, df_i is the number of documents which contains word i in the corpus, and N is the total number of documents in the corpus. The higher the value of TF-IDF indicates high relevancy and high discriminating value. Example of content representation is shown in Figure 3.1 as the lower part of matrix

Z.

Two different settings are used in the experiments as in [6].

(i) Sparse Setting: In training set, 1 item is randomly selected for each user. The remaining pairs are reserved for the test. The sparsity of the sparse training set is 0.0059%.

(ii) Dense Setting: In the training set, 10 items are randomly selected for each user. The remaining pairs are reserved for the test. That is, the sparsity of the sparse training set is 0.0589%.

4.2. Experiments

We have four different setups for experiments. Two methodologies for two different settings. We made grid-search for optimum hyperparameters in CDL, and for the best starting point for hyperparameter selection in CBNMF. The final results are the average values of five experiments for the best hyperparameter choices.

4.2.1. Collaborative Deep Learning

Before training the whole system, the content fed to SDAE for pretraining of the content. After pretraining, the whole system has been trained. Hyperparameter search for $\lambda_u, \lambda_v, \lambda_n$, is done by grid search. Best results for sparse setting and dense setting are obtained with $\lambda_u = 10, \lambda_v = 1000, \lambda_n = 100$, and $\lambda_u = 10, \lambda_v = 1, \lambda_n = 10^4$, respectively.

We used two leveled SDAE with [8000-200-50] architecture as in [6]. Masking noise with a fraction of 0.3 is used for input corruption, that is randomly chosen elements of $\mathbf{x}$ set to 0 with the fraction of 0.3. Dropout rate of 0.1 is used as in [6].

4.2.2. Coupled Bayesian Nonnegative Matrix Factorization

The content was pretrained before training the coupled matrix. Hyperparameters are searched via grid search. The confidence parameters a and b are fixed for 1 and 0.01, respectively. c for the content is searched by grid search and set to 0.4. Even tough, we updated hyperparameters during training, starting point for shape and scale parameters are sensitive to starting point. Hence, we also applied grid search for initialization of shape parameters, $a^t_{p,k}, a^v_{k,j}$. Mean values $b^t_{p,k}$ and $b^v_{k,j}$ are set to 1. In the final setup of the experiment the initial values of hyperparameter were $a^t_{p,k} = 100$, $b^t_{p,k} = 1$, $a^v_{k,j} = 1$, $b^v_{k,j} = 1$ for both sparse and dense settings.

4.3. Evaluation

In this thesis, we evaluated the results according to two metrics, recall and mean average precision, as in [6, 7, 34]. Recall measure is more suitable than precision for implicit feedback [14, 32].

The recall@M is defined as follows as explained in Section 2.6:

$$\text{recall@M} = \frac{\text{number of items user preferes among the recommended top M items}}{\text{total number of items that user preferes}}$$

The second metric, mean average precision (mAP), is the mean of average precisions [36].

$$\text{AP@M} = \frac{1}{|R|} \sum_{i=1}^{M} \left(\frac{r_i}{i} \sum_{j=1}^{i} r_j \right)$$

$$\text{mAP@M} = \frac{\sum_n \text{AP@M}}{n}$$

$|R|$ is the number of relevant documents at top M recommendations. r_i is 1 if ith recommendation is relevant, 0 otherwise. n is the number of users whom recommendations are made in our case. In the average precision calculation, each recommendation's precision is considered from the beginning. Only the relevant recommendation's preci-

Table 4.1. Mean Average Precision (mAP@500) results of CDL and CBNMF

mAP@500	Sparse Setting	Dense Setting
CDL	0.0457	0.0922
CBNMF	0.0133	0.0324

sion is taken into account [36]. The order of the recommendations matters in average precision metric, where the order is not taken into consideration in recall metric.

4.3.1. Results of Our Experiments

Recall for top 300 recommendation results are given in Figure- 4.2. mAP results for top 500 recommendation are given in Table-4.1. The results are the mean of five different training sets for best-known hyperparameters.

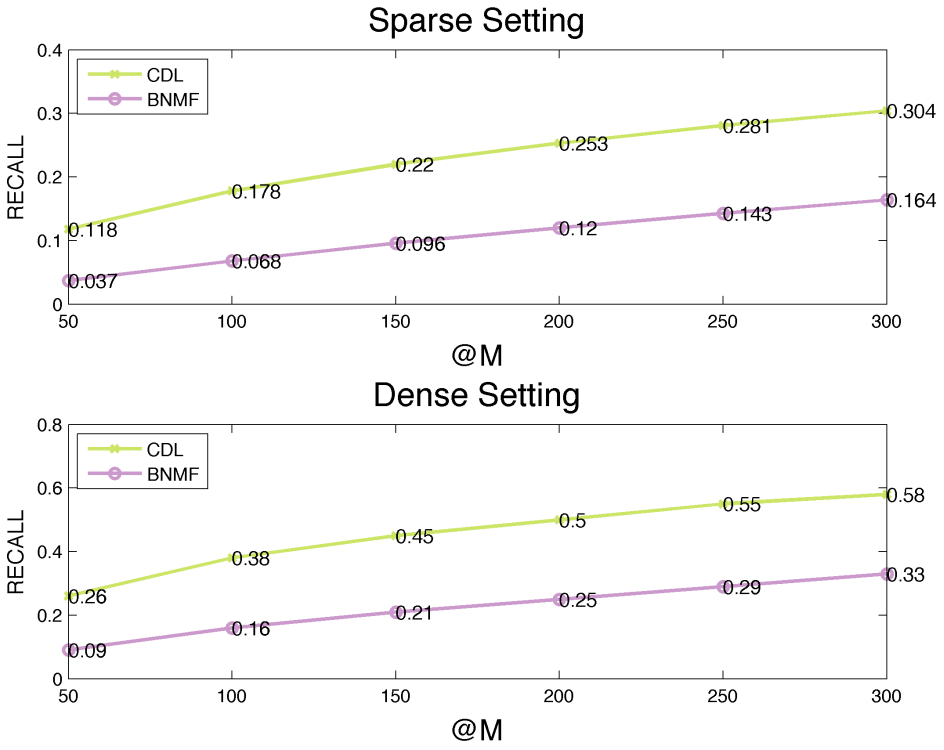

Figure 4.2. Recall@M results of our experiments

5. CONCLUSIONS AND FUTURE WORK

In this study, we compared two hybrid models, coupled Bayesian nonnegative matrix factorization (CBNMF) and collaborative deep learning (CDL), as well as reviewing some models that are related to these hybrid models, namely, matrix factorization, probabilistic matrix factorization, Bayesian nonnegative matrix factorization (BNMF), restricted Boltzmann machines, latent Dirichlet allocation, autoencoders, denoising autoencoders (DAE), stacked denoising autoencoders (SDAE) and probabilistic SDAE.

Our main contribution to this field is to apply Bayesian nonnegative matrix factorization method [2] as a hybrid recommendation system to propose preferable academic articles to users. Collaborative filtering and content based approach are applied as coupling the rating matrix and the content matrix. We compared our proposed method with a state-of-art hybrid method, collaborative deep learning [6].

We used CiteULike-a dataset, where the structure of the data consists of two parts. One of them is the main rating information which contains sparse and missing implicit binary values. The other part is the titles and abstracts of the articles.

We reproduced the experimental results of CDL [6], and adapted BNMF [2] as a hybrid model. We made grid search to find the optimal best hyperparameters on both of the models. Top 300 recommendations of the models are inferred. Recall and mean average precision metrics are calculated for the evaluation, since these metrics are more appropriate for both implicit feedback and characteristics of the information in this study.

Experimental results showed that CDL outperforms CBNMF, although both are using content data beside the rating information. This is primarily caused by Bayesian stacked DAE in CDL. It seems to extract more effective features from the content data. CDL benefits from robustness because of the Bayesian nature and also benefits from

extracting more useful features because of its deep structure by means of nonlinearity on the hidden layers.

To be more precise, Bayesian NMF allows robust and effective modelling in the cases where provided information obeys nonnegativity constraint. It is a hierarchical latent factor analysis which facilitates Bayesian model selection. On the other hand, Principal component analysis (PCA) is a statistical method to span the dataset into pricipal components, directions of the highest variance. NMF is an alternative to PCA, and it can also handle missing information on the contrary to PCA. In nonnegative matrix factorization, template matrix can be considered as basis matrix and excitation matrix is the weights of the bases for the corresponding entries. Each entry on the input matrix is represented as a linear combination of the bases [2]. Autoencoders with linear transformation and mean squared error objective and the same number of hidden neurons as with the PCA's principal component, learn the same span as PCA. Also, linear autoencoder is also can be considered as a matrix factorization method. If the hidden-layer is non-linear, then autoencoder behaves differently [37].

An autoencoder learns a representation of the inputs. It is deterministic, whereas RBMs are statistical and learn probability distributions. On the other hand, denoising autoencoders are probabilistic and Bayesian treatment of DAE is generative. Nonlinearity in DAE and RBMs, allows to build deep structures, namely stacked DAE and deep belief nets respectively [5, 6]. So, nonlinearity and deep structure of Bayesian SDAE seem the causes of out-performance of CDL.

As future work, our study can be extended in the following directions:

(i) For coupled BNMF, other observation models, such as truncated Gaussian or Gamma, can be used instead of Poisson distribution because of the binary rating information, and other cost functions like Euclidian or Itakura-Saito divergences can be used, other than KL divergence [38].
(ii) Different representation of the content data can be explored. Bag-of-word representation does not consider order of the words. Regarding the order of the

words in represenation may allow extract more effective features. For example, in [39] order of the words was taken into consideration in the skip-gram model as a distributed representation, or different word embedding methods can be used as in [40].

(iii) Another direction would be to use different deep structures for feature extraction, such as recurrent neural networks as in [35] or convolutional neural networks. So that, the order of the words would be taken into consideration and stronger representations can be used as in (ii).

REFERENCES

1. Salakhutdinov, R. and A. Mnih, "Probabilistic Matrix Factorization", *Nips*, Vol. 1, pp. 1257–1264, 2007.

2. Cemgil, A. T., "Bayesian inference for nonnegative matrix factorisation models", *Computational Intelligence and Neuroscience*, Vol. 2009, 2009.

3. Salakhutdinov, R., A. Mnih and G. Hinton, "Restricted Boltzmann machines for collaborative filtering", *Proceedings of the 24th international conference on Machine learning*, pp. 791–798, ACM, 2007.

4. Blei, D. M., A. Y. Ng and M. I. Jordan, "Latent dirichlet allocation", *Journal of machine Learning research*, Vol. 3, No. Jan, pp. 993–1022, 2003.

5. Vincent, P., H. Larochelle, I. Lajoie, Y. Bengio and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion", *Journal of Machine Learning Research*, Vol. 11, No. Dec, pp. 3371–3408, 2010.

6. Wang, H., N. Wang and D.-Y. Yeung, "Collaborative deep learning for recommender systems", *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1235–1244, ACM, 2015.

7. Wang, C. and D. M. Blei, "Collaborative topic modeling for recommending scientific articles", *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 448–456, ACM, 2011.

8. Spotify, *Spotify Companyinfo*, 2018, `https://newsroom.spotify.com/companyinfo/`, accessed at May 2018.

9. DMR, *Spotify Statistics and Facts*, 2018, `https://expandedramblings.com/index.php/spotify-statistics/`, accessed at May 2018.

10. uNoGS, *Netflix Global Search on uNoGS*, 2018, `https://unogs.com/`, accessed at May 2018.

11. Koren, Y., R. Bell and C. Volinsky, "Matrix factorization techniques for recommender systems", *Computer*, Vol. 42, No. 8, 2009.

12. Ricci, F., L. Rokach, B. Shapira and P. B. Kantor, *Recommender systems handbook*, Springer, 2015.

13. Burke, R., "Hybrid web recommender systems", *The adaptive web*, pp. 377–408, Springer, 2007.

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

15. Lang, K., "Newsweeder: Learning to filter netnews", *Proceedings of the 12th international conference on machine learning*, pp. 331–339, 1995.

16. Sevil, S. G., O. Kucuktunc, P. Duygulu and F. Can, "Automatic tag expansion using visual similarity for photo sharing websites", *Multimedia Tools and Applications*, Vol. 49, No. 1, pp. 81–99, 2010.

17. Georgiev, K. and P. Nakov, "A non-IID Framework for Collaborative Filtering with Restricted Boltzmann Machines.", *ICML (3)*, pp. 1148–1156, 2013.

18. Hidasi, B., A. Karatzoglou, L. Baltrunas and D. Tikk, "Session-based recommendations with recurrent neural networks", *arXiv preprint arXiv:1511.06939*, 2015.

19. Van den Oord, A., S. Dieleman and B. Schrauwen, "Deep content-based music recommendation", *Advances in neural information processing systems*, pp. 2643–

2651, 2013.

20. Wang, X. and Y. Wang, "Improving content-based and hybrid music recommendation using deep learning", *Proceedings of the 22nd ACM international conference on Multimedia*, pp. 627–636, ACM, 2014.

21. Lü, L., M. Medo, C. H. Yeung, Y.-C. Zhang, Z.-K. Zhang and T. Zhou, "Recommender systems", *Physics Reports*, Vol. 519, No. 1, pp. 1–49, 2012.

22. Bobadilla, J., F. Ortega, A. Hernando and A. Gutiérrez, "Recommender systems survey", *Knowledge-based systems*, Vol. 46, pp. 109–132, 2013.

23. Su, X. and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques", *Advances in artificial intelligence*, Vol. 2009, p. 4, 2009.

24. Billsus, D. and M. J. Pazzani, "A hybrid user model for news story classification", *UM99 User Modeling*, pp. 99–108, Springer, 1999.

25. Salakhutdinov, R. and A. Mnih, "Bayesian probabilistic matrix factorization using Markov chain Monte Carlo", *Proceedings of the 25th international conference on Machine learning*, pp. 880–887, ACM, 2008.

26. Berry, M. W., M. Browne, A. N. Langville, V. P. Pauca and R. J. Plemmons, "Algorithms and applications for approximate nonnegative matrix factorization", *Computational statistics & data analysis*, Vol. 52, No. 1, pp. 155–173, 2007.

27. Hinton, G. E., "Training products of experts by minimizing contrastive divergence", *Training*, Vol. 14, No. 8, 2006.

28. Hinton, G. E., "A practical guide to training restricted Boltzmann machines", *Neural networks: Tricks of the trade*, pp. 599–619, Springer, 2012.

29. Bengio, Y., I. J. Goodfellow and A. Courville, "Deep learning", *Nature*, Vol. 521, pp. 436–444, 2015.

30. Hinton, G. E. and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks", *science*, Vol. 313, No. 5786, pp. 504–507, 2006.

31. Herlocker, J. L., J. A. Konstan, L. G. Terveen and J. T. Riedl, "Evaluating collaborative filtering recommender systems", *ACM Transactions on Information Systems (TOIS)*, Vol. 22, No. 1, pp. 5–53, 2004.

32. Hu, Y., Y. Koren and C. Volinsky, "Collaborative filtering for implicit feedback datasets", *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pp. 263–272, Ieee, 2008.

33. Wang, H., *Collaborative Deep Learning for Recommender Systems*, 2015, `http://www.wanghao.in/CDL.htm`, accessed at May 2018.

34. Wang, H., B. Chen and W.-J. Li, "Collaborative Topic Regression with Social Regularization for Tag Recommendation.", *IJCAI*, pp. 2719–2725, 2013.

35. Wang, H., S. Xingjian and D.-Y. Yeung, "Collaborative recurrent autoencoder: Recommend while learning to fill in the blanks", *Advances in Neural Information Processing Systems*, pp. 415–423, 2016.

36. Yilmaz, E. and J. A. Aslam, "Estimating average precision with incomplete and imperfect judgments", *Proceedings of the 15th ACM international conference on Information and knowledge management*, pp. 102–111, ACM, 2006.

37. Bengio, Y. *et al.*, "Learning deep architectures for AI", *Foundations and trends® in Machine Learning*, Vol. 2, No. 1, pp. 1–127, 2009.

38. Févotte, C. and A. T. Cemgil, "Nonnegative matrix factorizations as probabilistic inference in composite models", *Signal Processing Conference, 2009 17th European*, pp. 1913–1917, IEEE, 2009.

39. Mikolov, T., I. Sutskever, K. Chen, G. S. Corrado and J. Dean, "Distributed representations of words and phrases and their compositionality", *Advances in neural information processing systems*, pp. 3111–3119, 2013.

40. Musto, C., G. Semeraro, M. De Gemmis and P. Lops, "Word Embedding Techniques for Content-based Recommender Systems: An Empirical Evaluation.", *RecSys Posters*, 2015.