Neural Gaussian Mixture Model for Review-based Rating Prediction

Dong Deng
Beijing Key Lab of Traffic Data Analysis and Mining
Beijing, China
dengdong@bjtu.edu.cn

Liping Jing
Beijing Key Lab of Traffic Data Analysis and Mining
Beijing, China
lpjing@bjtu.edu.cn

Jian Yu
Beijing Key Lab of Traffic Data Analysis and Mining
Beijing, China
jianyu@bjtu.edu.cn

Shaolong Sun
Academy of Mathematics and Systems Science
Beijing, China
sunshl@amss.ac.cn

Haofei Zhou
Beijing Key Lab of Traffic Data Analysis and Mining
Beijing, China
17125275@bjtu.edu.cn

ABSTRACT

Review has been proven to be an important information in recommendation. Different from the overall user-item rating matrix, it can provide textual information that exhibits why a user likes an item or not. Recently, more and more researchers have paid attention on review-based rating prediction. There are two challenging issues: how to extract representative features to characterize users / items from reviews and how to leverage them for recommendation system. In this paper, we propose a **N**eural **G**aussian **M**ixture **M**odel (**NGMM**) for review-based rating prediction task. Among it, the review textual information is used to construct two parallel neural networks for users and items respectively, so that the users' preferences and items' properties can be sufficiently extracted and represented as two latent vectors. A shared layer is introduced on the top to couple these two networks together and model user-item rating based on the features learned from reviews. Specifically, each rating is modeled via a Gaussian mixture model, where each Gaussian component has zero variance, the mean described by the corresponding component in user's latent vector and the weight indicated by the corresponding component in item's latent vector. Extensive experiments are conducted on five real-world Amazon review datasets. The experimental results have demonstrated that our proposed **NGMM** model achieves the state-of-the-art performance in review-based rating prediction task.

CCS CONCEPTS

• **Computer systems organization → Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability;

KEYWORDS

Review-based Rating Prediction, Gaussian Mixture Model, Recommendation, Deep Learning

ACM Reference Format:
Dong Deng, Liping Jing, Jian Yu, Shaolong Sun, and Haofei Zhou. 2018. Neural Gaussian Mixture Model for Review-based Rating Prediction. In *Twelfth ACM Conference on Recommender Systems (RecSys '18), October 2–7, 2018, Vancouver, BC, Canada,* Jennifer B. Sartor, Theo D'Hondt, and Wolfgang De Meuter (Eds.). ACM, New York, NY, USA, Article 4, 9 pages. https://doi.org/10.1145/3240323.3240353

1 INTRODUCTION

With the rapid growth of e-commerce, rating prediction has become a very hot topic in recommendation. It aims to discover the preferences of users and properties of items based on historical rating information. Based on these preferences, product providers can help people figure out which products match their interests most.

However, most existing rating prediction methods encounter the problem of data sparsity. For instance, up to 99.99% of rating information is missing from Amazon review datasets. To solve this problem, many methods utilize other side information, such as reviews, social relations between users, items' content, etc. Among them, reviews can offer sufficient information for preferences discovery. Because, compared with numerical rating information, reviews often contain the details why users like an item or not. Thus, review-based rating prediction has attracted increasing interest.

Although reviews can provide useful information to help model user preferences and item properties, there are two challenging issues. Firstly, how to extract representative features from reviews to characterize users and items. Secondly, how to leverage them for rating prediction. Many existing methods utilize probabilistic topic models [2] to model reviews. In [12, 25], the authors introduce topic model to extract latent topics hidden in reviews. Then, these learnt features are incorporated into factorization-based methods for rating prediction. Although these methods can produce better results than approaches relying purely on rating information, they still encounter the problem of how to transfer the features in review textual space to rating space. Furthermore, topic model utilizes *bag-of-words* techniques to represent words and does not consider word sequences. Thus, these methods cannot efficiently capture the complex semantic and syntactic information contained in reviews. Recently, deep learning has been proven to be a powerful feature representation method for rating prediction. Such approaches [6, 27] usually construct a deep neural network to learn representations of users and items, and then incorporate these features into factorization-based methods for rating prediction, such as matrix factorization(MF) [7, 13, 16, 17] and factorization machine(FM) [15, 27]. Deep learning based methods usually utilize word embeddings to represent words which have been proven to be more efficiently than *bag-of-words*. However, these approaches ignore the intrinsic characteristics among the rating behaviours of users. In general, users consider preferences and weights over different *factors* of items when giving evaluations. Thus, unknown ratings are obtained by weighted summation over different factors.

To handle the above issues, in this paper, we propose a novel **N**eural **G**ussian **M**ixture **M**odel (**NGMM**). Firstly, we employ reviews to construct a parallel neural network. One of the network focuses on learning user preferences, and the other one learns item properties. Secondly, we utilize Gaussian mixture model to model rating information. In detail, our proposed model, NGMM, assumes that each rating is generated from a Gaussian mixture model which model user preferences over factors of items. In order to incorporate Gaussian mixture model into the neural network, a shared mixture of Gaussian layer is designed on top of the parallel neural network to simulate parameters of Gaussian mixture model, mean and mixture proportion. Specifically, each rating is modeled via a Gaussian mixture model, where each Gaussian component has zero variance, the mean described by the corresponding component in user's latent vector and the weight indicated by the corresponding component in item's latent vector.

In sum, our main contributions are outlined as follows:

- The proposed neural Gaussian mixture model (**NGMM**) jointly model users' preferences and items' properties from review textual information. Then, a mixture of Gaussian layer on top of the parallel neural network can capture the interaction between users and items, and learn the ratings and weights over different factors. It imitates the rating behaviour of users to items.
- NGMM utilizes deep neural network to model reviews and uses word embeddings to represent words. Compared with existing topic model based methods which use the traditional *bag-of-words* techniques, NGMM achieves a significant improvement in rating prediction task.
- Experiments over five real-world Amazon review datasets demonstrate that NGMM achieves state-of-the-art performance in review-based rating prediction task.

The rest of the paper is organized as follows. Section2 gives a short review of related works. Problem statement and some preliminaries are introduced in Section3. In Secion4, we present the architecture and details of proposed model. A series of experiments on five real-world Amazon review datasets are conducted in Section5. In Section6, a brief conclusion and future work are given.

2 RELATED WORK

There are two categories of studies which are related to our work: approaches which model users and items by exploiting both ratings and reviews, and deep learning techniques employed for review-based rating prediction task. In this paper, we will give a short review of these researches and distinguish our proposed model from them.

The first category of studies incorporates reviews into historical user-item rating matrix for unknown rating prediction. One of the pioneer works from [4] which found that reviews usually contain some opinion *targets* or *aspects*, e.g., price, quality, time, that can be useful for rating prediction. In [8], authors utilize reviewer, product and text features to form a three-dimension tensors and use tensor factorization techniques to predict unknown ratings. Furthermore, in [12, 21, 25], topic model techniques are used to model reviews and combine with matrix factorization based methods. A similar work is proposed in [11] with the main difference that it models rating information using Gaussian mixture model instead of popular matrix factorization. Although These methods can exploit reviews to solve or alleviate the problem of data sparsity, all of them just consider the statistics information of words among each review. One limitation of the above studies is that they model reviews with *bag-of-words* technique which can't capture the complex semantic information and doesn't consider their sequences.

Recently, with the rapid growth of deep learning, its' ability for representation learning has attracted many researchers to use it in rating prediction task. The authors in [3, 10, 18, 24] utilize popular neural network architecture like denoising autoencoder , restricted boltzmann machines (RBM) or feedforward neural network to model users and items from historical rating information. However, these methods don't exploit other side information. Several works in [20, 23] incorporate content of items into rating information and use CNN or deep belief network (DBN) to learn latent representations of users and items. On the other hand, in [1, 6, 22], authors use reviews to form content of items and construct many neural network architectures, like CNN, auto-encoder or RNN to capture efficient representation of items. Although items' content can provide valuable information for items, they can't provide useful information for users. Reviews, as a user textual information, may describe the details why users like or hate products. Thus, reviews can provide useful information for exploring both users' preference and items' properties. To utilize such information in reviews. recent works in [9, 27] use CNN architecture to jointly model representation of users and items from reviews. Then, a shared layer on top of the neural network capture the interaction of users and items and predict the unknown ratings. Although, above neural network based methods can efficiently take advantage of review textual information. They don't consider the rating behaviour of users.

In this paper, we construct a neural Gaussian mixture model (NGMM) to solve the above problems. Firstly, we employ review textual information to construct a parallel neural network which can fully capture the complex semantic information among reviews. Secondly, NGMM can capture the ratings of each user to different factors of an item. To incorporate Gaussian mixture model(GMM) into the neural network, we design a mixture of Gaussian layer on

Table 1: Notations

Symbols	Descriptions
$\lvert\mathcal{U}\rvert$	number of users in corpus.
$\lvert\mathcal{V}\rvert$	number of items in corpus.
K	number of components in each mixture of Gaussian model.
μ_u	mean of Gaussian mixture model corresponding to user u.
θ_v	mixture proportion of Gaussian mixture model corresponding to item v.
$f_{u,v}$	components assigned by mixture of Gaussian model for pair of user u and item v.
$r_{u,v}$	observed ratings of user u to item v.
μ_0	mean of Gaussian distribution for μ.
σ_0^2	variance of Gaussian distribution for μ.
σ^2	variance of Gaussian mixture distribution for $r_{u,v}$
$d_{u,v}$	a single review written by u-th user to v-th item
d^u	concatenating all reviews written by user u
d^v	concatenating all reviews of item v rated by any user

top of the parallel neural network and simulate the parameters of GMM, mean and mixture proportion. Finally, an unknown rating is predicted by weighted summation over ratings generated from each component in GMM.

3 PRELIMINARY

3.1 Problem Formulation

Let Q be a collection of training data consisting of N samples. Each sample is a 4-tuple $(u, v, r_{u,v}, d_{u,v})$, where $u \in \{1, 2, \cdots, |\mathcal{U}|\}$ is the user index, $v \in \{1, 2, \cdots, |\mathcal{V}|\}$ is the item index. $r_{u,v}$ indicates the numerical ratings of user u to item v. It is scaled from 1 to 5. $d_{u,v}$ is the review written by user u to item v. It is a piece of free format natural language which describes the detail information why user u like item v or not. Furthermore, for simplicity, we use d^u and d^v to indicate the concatenation of all reviews written by user u and all reviews related to item v. we use them to represent user reviews and item reviews. The primary goal in this paper is to predict the unknown ratings of items that the users have not rated yet. The mathematical notations used in this paper are summarized in Table 1.

3.2 Gaussian Mixture Model for Rating Prediction

Inspiration from recently work in [11], Gaussian mixture model for rating prediction assumes that each rating rated by a user to an item is generated under a mixture of Gaussian distribution. The generative process can be shown as follows:

(1) For each user $u \in \mathcal{U}$:
  (a) For each component $k \in [1, K]$ in mixture of Gaussian:
    i. Draw $\mu_{u,k} \sim \text{Gaussian}(\mu_0,\sigma_0^2)$
(2) For each item $v \in \mathcal{V}$:
  (a) Draw mixture proportion $\theta_v \sim \text{Dirichlet}(\beta)$

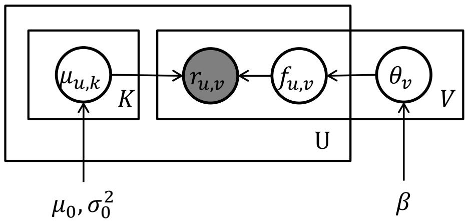


Figure 1: the graphical model of mixture of Gaussian model.

(3) For each observed rating assigned by u to v:
  (a) Draw distribution assignment $f \sim \text{Multinomial}(\theta_v)$
  (b) Draw rating $r_{u,v} \sim \text{Gaussian}(\mu_{u,f}, \sigma^2)$

From the above generative process, for a specific rating $r_{u,v}$, we have the following probabilistic distribution.

$$p(r_{u,v}) = \sum_{k=1}^{K} \theta_{v,k} \cdot P_k(r_{u,v}|\mu_{u,k}, \sigma^2) \tag{1}$$

where $\theta_{v,k}$ is the mixture proportion indicating the weight of k-th latent factor it possess. $P_k(x)$ corresponds to a Gaussian function which is the k-th component of mixture of Gaussian. $\mu_{u,k}$ is its' mean and σ^2 is variance. Then, for all items which are rated by user u, The likelihood of these ratings can be calculated.

$$\mathcal{L} = \prod_{v=1}^{V} \sum_{k=1}^{K} \theta_{v,k} \cdot P_k(r_{u,v}|\mu_{u,k}, \sigma^2) \tag{2}$$

Meanwhile, θ is defined as a prior dirichlet distribution with a prior parameter β. Thus, given the overall generative process, the distribution of all observed ratings can be formulated as:

$$P(r|\alpha, \beta, \mu_0, \sigma_0, \sigma^2; \theta, \mu) = \prod_{v=1}^{|\mathcal{V}|} P(\theta_v|\alpha)$$
$$\prod_{u=1}^{|\mathcal{V}|}(\sum_{f=1}^{K} P(f|\theta_v)P(r_{u,v}|\mu_{u,k}, \mu_{u,f}, \sigma^2)) \tag{3}$$
$$(\prod_{u=1}^{|\mathcal{U}|}\prod_{k=1}^{K} P(\mu_{u,k}|\mu_0, \sigma_0^2))$$

Gaussian mixture model for rating prediction can be optimized through collapsed Gibbs sampling method. After estimating the model parameters $\Theta = (\theta, \mu)$, unknown rating of user u to item v can be predicted by inner product of $\mu_u * \theta_v$.

3.3 Neural Network View of Gaussian Mixture Model for Rating Information

According to the description in the above subsection, In Gaussian mixture model, each user and item are represented as a mixture of latent variable μ and f respectively which indicate the expected value of a user's preference and f-th component of an item it belongs. To formulate, suppose $r_{u,v}$ is a rating rated by user u and item v. Gaussian mixture model computes the conditional

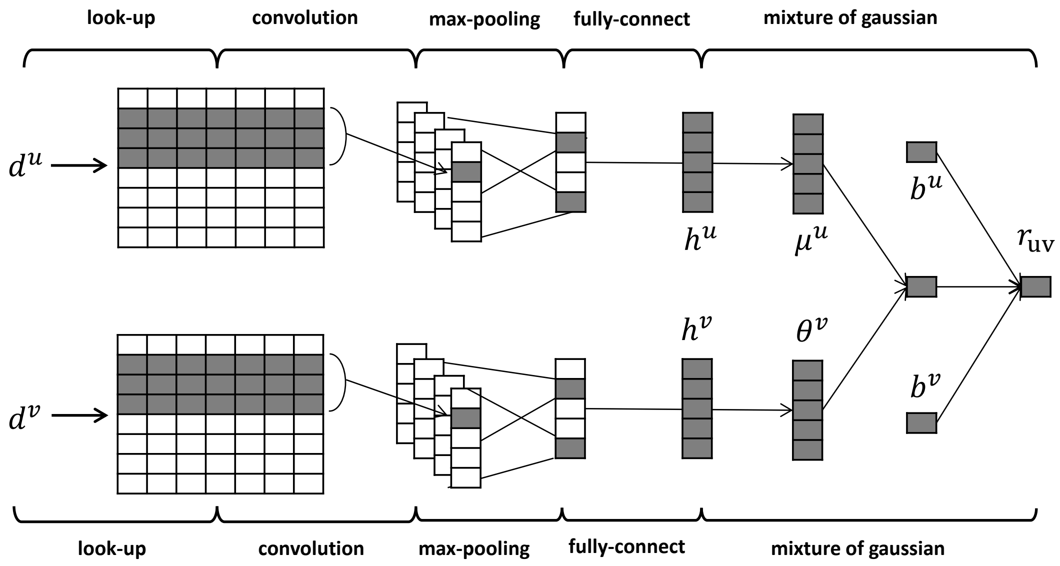


Figure 2: the architecture of NGMM model.

probability $P(r_{u,v}|u,v)$ as the combination of component-item and rating-gaussian distribution:

$$p(r_{u,v}|u,v) = \sum_{k=1}^{K} P(f_k|v)P(r_{u,v}|f_k) \tag{4}$$

where K is the pre-defined component number. As describe in Fig. 1, $\theta = [P(f_1), P(f_2), \cdots, P(f_K)]$. $P(r_{u,v}|f_k)$ is the k-th Gaussian distribution in mixture model. Thus, in order to maximize the probability of given observed rating $r_{u,v}$, expected value of k-th component Gaussian distribution is assigned to $r_{u,v}$. Thus, it's equal to fit the following equation for each rating $r_{u,v}$:

$$r_{u,v} = \sum_{k=1}^{K} \mu_{u,f_k}\theta_{v,k} \tag{5}$$

4 NEURAL GAUSSIAN MIXTURE MODEL

Different from factorization-based methods, such as matrix factorization, Gaussian mixture model can efficiently imitate the rating behaviour of users to items. However, in order to alleviate the problem of data sparsity by incorporating review textual information, [11] introduces topic model to model the review texts and incorporate learned features into Gaussian mixture model. However, as described above, it uses *bag-of-words* model to represent each word and does not consider the word sequential information among reviews. Thus, this model can not capture the complex semantic structure information. In this paper, we propose a neural Gaussian mixture model to solve this problem. Full description of this model is shown in the following subsection.

4.1 Architecture

The architecture of our proposed model, NGMM, is shown in Fig. 2. We take inspiration from [27], and then construct a parallel neural network to extract features from user and item reviews respectively. One network for items Net_{item} and the other one for users Net_{user}. The first layer, denoted as look-up layer, map the user and item reviews to a uniform matrices of word embeddings. Next are some common layers used in CNN based models to discover multiple levels of features including convolution, pooling and fully connected layer. Finally, a mixture of Gaussian layer is added on the top of the parallel neural network to simulate parameters of mixture of Gaussian model, mean μ and mixture proportions θ. In order for simplicity, we use $\Theta = (\theta, \mu)$ to represent these two parameters in this paper. In the following subsections, Net_{user} and Net_{item} only differ in their inputs and activation function within some layers, we will focus on illustrating the detail process for Net_{item}. The same process is similar applied for Net_{user}.

4.1.1 Look-Up Layer. Word embedding is a popular representation technique which maps a word to a dense low dimensional real-value vector. Compared with *bag-of-words*, word embedding can capture more semantic and syntactic information among words. Let $x_i^v \in \mathbb{R}^C$ be a C-dimensional word embedding corresponding to the i-th word in the v-th item reviews. Each review of length n (padded where necessary) can be represented as

$$d_{1:n}^v = \Phi(d_1^v) \oplus \Phi(d_2^v) \oplus \cdots \oplus \Phi(d_n^v) \tag{6}$$

where $\oplus$ is the concatenation operator. $d_{i:i+j}^v$ refer to the concatenation of words $d_i^u, d_{i+1}^u, \cdots, d_{i+j}^u$. It's worth mention that we can utilize pre-trained word embedding to initialize these parameters which can avoid bad random initialization.

4.1.2 CNN Layer. CNN layer contains a set of commonly operations in CNN architecture based models. Firstly, a convolution layer applies a *filter* $\mathbf{w} \in \mathbb{R}^{c \times h}$ in a window of h words to produce a new feature. For example, a feature z_i is generated from a window of

Table 2: Data summary on Dataset I (random splitting dataset into training, validation and test)

DataSets	#items	#users	#reviews	#words	#words/review	#reviews/item	#reviews/user	density
Musical Instruments	83046	339231	500176	35.2M	70.39	6.02	1.47	0.0018%
Amazon Instant Video	23965	426922	583933	24.8M	58.09	24.37	1.37	0.0057%
Baby	64426	531890	915446	60.0M	65.54	14.21	1.72	0.0027%
Grocery and Gourment Food	166049	768438	1297156	63.0M	48.61	7.81	1.69	0.0010%
Movies and TV	200941	2088620	4607047	399.0M	86.78	22.93	2.21	0.0011%
Average on all datasets	107685	831020	1580752	116.5M	62.76	15.07	1.69	0.0025%

Table 3: Data summary on Dataset II (preprocessing dataset following [6])

DataSets	#items	#users	#reviews	#words	#words/review	#reviews/item	#reviews/user	density
Musical Instruments	48792	29040	120566	14.0M	115.71	2.47	4.15	0.0085%
Amazon Instant Video	15149	29756	108148	7.7M	70.90	7.14	3.63	0.024%
Baby	42523	71826	304035	27.5M	90.50	7.15	4.23	0.010%
Grocery and Gourment Food	108467	86400	407035	29.1M	71.56	3.75	4.71	0.0043%
Movies and TV	169065	319406	2026664	281.8M	139.07	11.99	6.35	0.0038%
Average on all datasets	76799	107285	593289	72.0M	97.55	6.50	4.62	0.0101%

words $d^v_{i:i+h-1}$ by

$$z_i^v = f^v(\mathbf{w}^v * d^v_{i:i+h-1} + b) \tag{7}$$

Here $b \in \mathbb{R}$ is a bias term and f^v is an activation function. In Net_{item} architecture, we use *relu* as our activation function in this layer. However for Net_{user} architecture, we use tanh activation function. Symbol $*$ is convolution operator. Then, this filter is applied to each possible window of words in a document $\{d^v_{1:h}, d^v_{2:h+1}, \cdot, d^v_{n-h+1:n}\}$ to produce a *feature map*:

$$\mathbf{z}^v = [z_1^v, z_2^v, \cdot, z^v_{n-h+1}] \tag{8}$$

where $\mathbf{z}^v \in \mathbb{R}^{n-h+1}$. Secondly, we apply a max pooling operation [5] over the feature map to select the maximum value as our features.

$$o = \max\{\mathbf{z}^v\} \tag{9}$$

It's worth noting that max pooling scheme can naturally deal with any varied length text. Then, we use multiple filters to capture various features and get the output vector of the max pooling layer

$$O^v = \{o_1^v, o_2^v, \cdot, o_s^v\} \tag{10}$$

where s denotes the kernel size in the convolutional layer. Finally, a fully connected layer with weight $\mathbf{W}$ and activation function g^v is applied and get the final output

$$h^v = g^v(\mathbf{W}^v \times O^v + b^v) \tag{11}$$

Similarly, we use *relu* and *tanh* activation function for Net_{item} and Net_{user} architecture respectively. Furthermore, we can get the output of fully-connected layer for Net_{user} neural network as follows.

$$h^u = g^u(\mathbf{W}^u \times O^u + b^u) \tag{12}$$

h^v and h^u can be respectively viewed as item and user embeddings.

4.1.3 Mixture of Gaussian Layer. Outputs of the fully-connected layer can be viewed as a latent representation of users and items. However, in order to capture the interaction between users and items, some techniques have been proposed like Factorization Machine or Matrix Factorization. In this paper, in order to consider the importance of each factor to overall ratings, we design a mixture of Gaussian layer on the top of the fully-connected layer. Its' goal is to simulate the parameters of mixture of Gaussian model, mean μ and mixture proportions θ. In detail, we firstly utilize a fully-connected layer to simulate $\Theta = (\theta, \mu)$ based on user and item embeddings.

$$\theta^v = l^v(\mathbf{W}' \times h^v + b') \tag{13}$$

Similarly, we can get μ^u as follows.

$$\mu^u = l^u(\mathbf{W}' \times h^u + b') \tag{14}$$

In addition, we consider the bias of each user and item to correct biased ratings as shown in Fig.2. Finally, the objective function of NGMM is given by

$$\mathbb{L} = \sum_{u=1}^{|\mathcal{U}|}\sum_{v=1}^{|\mathcal{V}|} \mathbb{I}_{u,v}\Big(\sum_{k=1}^{K} \mu_k^u * \theta_k^v + p^u + q^v - r_{u,v}\Big)^2 \tag{15}$$

where $\mathbb{I}_{u,v}$ is indicator function which is 1 when $r_{u,v}$ is observed in training data and 0 vice versa. p^u and q^v are respectively the bias of user and item.

4.2 Training Algorithm

Our network is trained by minimizing Eq. 15. The derivatives of parameters including $\Phi = (\theta, \mu, b^u, b^v)$ and other parameters in different layers of CNN can be calculated by applying differentiation chain rule.

Given a training set Q consisting of N samples, we optimize the model through *Adam* over mini-batches. In addition, in order to prevent over-fitting, the dropout[19] strategy is applied on the input layer of the parallel network.

Table 4: MSE Comparison with baselines on Dataset I. Best results are indicated in bold.

DataSets	PMF	RMR	ConvMF	DeepCoNN	NGMM
Musical Instruments	1.454 ± 0.000	1.425 ± 0.000	3.505 ± 0.014	1.445 ± 0.011	**1.306 ± 0.001**
Amazon Instant Video	1.364 ± 0.000	1.319 ± 0.000	3.674 ± 0.003	1.229 ± 0.020	**1.089 ± 0.006**
Baby	1.666 ± 0.000	1.634 ± 0.000	3.110 ± 0.006	1.583 ± 0.012	**1.433 ± 0.001**
Grocery and Gourment Food	1.577 ± 0.000	1.532 ± 0.000	3.302 ± 0.006	1.533 ± 0.009	**1.417 ± 0.000**
Movies and TV	1.383 ± 0.000	1.247 ± 0.002	2.836 ± 0.007	1.303 ± 0.008	**1.120 ± 0.000**
Average on all datasets	1.515 ± 0.000	1.478 ± 0.000	3.39775 ± 0.007	1.448 ± 0.013	**1.311 ± 0.002**

Table 5: MSE Comparison with baselines on on Dataset II. Best results are indicated in bold.

DataSets	PMF	RMR	ConvMF	DeepCoNN	NGMM
Musical Instruments	1.039 ± 0.000	0.981 ± 0.001	1.154 ± 0.022	1.012 ± 0.005	**0.936 ± 0.000**
Amazon Instant Video	1.223 ± 0.000	1.192 ± 0.002	1.148 ± 0.015	1.057 ± 0.007	**0.991 ± 0.000**
Baby	1.390 ± 0.000	1.331 ± 0.000	1.267 ± 0.018	1.279 ± 0.009	**1.202 ± 0.000**
Grocery and Gourment Food	1.257 ± 0.000	1.173 ± 0.000	1.207 ± 0.016	1.178 ± 0.008	**1.110 ± 0.000**
Movies and TV	1.183 ± 0.000	1.074 ± 0.000	1.165 ± 0.011	1.073 ± 0.002	**1.041 ± 0.000**
Average on all datasets	1.227 ± 0.000	1.169 ± 0.001	1.194 ± 0.018	1.132 ± 0.007	**1.060 ± 0.000**

5 EXPERIMENT

In this section, we evaluate the empirical performance of NGMM on five real-world Amazon review datasets. Our experiments demonstrate that: 1) NGMM outperformed other baselines in the review-based rating prediction task when the datasets are very sparse. 2) existing word embeddings im most cases can help us improve the prediction accuracy, but it sometimes may introduce *noise* which decreases the performance. 3) we evaluate the effect of number of components in GMM for our proposed NGMM.

5.1 Datasets

To demonstrate the effectiveness of our proposed NGMM model, we evaluate it on five real-world Amazon datasets [12]. Firstly, we randomly split the whole data into a training dataset (80%), a validation dataset (10%) and a test dataset (10%), just like [11, 26, 27]. It's called **Dataset** I. Secondly, as for ConvMF in [6], the authors require a different pre-process for dataset. Except for randomly splitting each dataset into a training set (80%), a validation set (10%) and a test set (10%). It also requires that the training set contains at least a rating on every user and item. At the same time, it removes users that have less than 3 ratings. Thus, following the strategy in [6], we construct a second dataset, **Dataset** II. The final statistics of **Dataset** I and **Dataset** II are shown in table 2 and 3. From table 2, we can see that Amazon review dataset is very sparse (up to 99.99% of ratings or reviews information are missing). Even though we remove some users which have less than 3 ratings in **Dataset** II. Almost 99.9% of the ratings information are missing.

5.2 Baselines

We compare the proposed NGMM with four baselines.

- PMF [17]: Probability Matrix Factorization (PMF) is a popular used rating prediction model which only utilizes user-item ratings matrix.
- RMR [11]: 'Ratings Meet Reviews' (RMR) is a combined approach of content-based filtering and collaborative filtering. It utilizes the mixture of Gaussian to model rating information and incorporates the item description documents into the mixture proportion.
- ConvMF [6]: Convolutional Matrix Factorization (ConvMF) integrates CNN into PMF method so as to capture contextual information in item description documents.
- DeepCoNN [27]: Deep Cooperative neural Networks (DeepCoNN) uses a parallel deep neural network to directly learn the representation of items and users from their corresponding description documents. Then it utilizes a shared layer to map the features from users and items to a same feature space and predict unknown ratings.

5.3 Evaluation Metrics

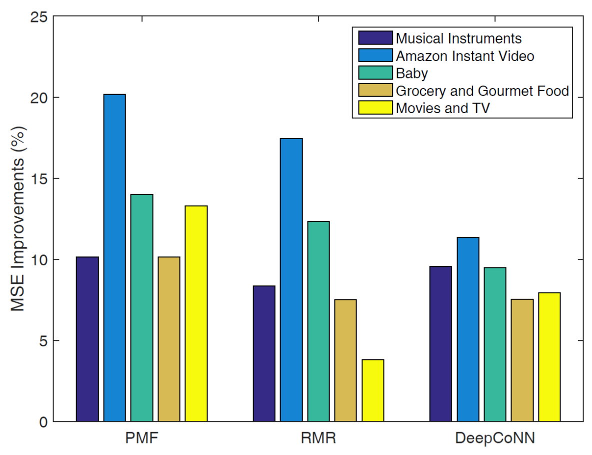
In experiments, we use training dataset to train each model and use validation dataset to tune the hyper-parameters. Finally, we report test error of each model which gives the lowest validation errors. Furthermore, we use Mean Squared Error (*MSE*) as our evaluation measures, which are defined as follows:

$$MSE = \sqrt{\frac{\sum_{u=1}^{U}\sum_{v=1}^{V}\mathbb{I}_{u,v}(r_{u,v} - \hat{r}_{u,v})^2}{\# of\ ratings}}$$

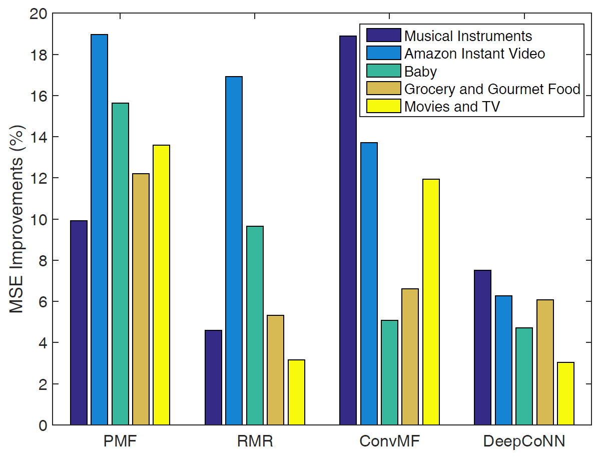
where $r_{u,v}$ is the observed ratings rated by u-th user to i-th item. $\hat{r}_{u,v}$ is the corresponding predicted ratings. $\mathbb{I}_{u,v}$ is the indicator function which is 1 when user u have rated item v and 0 vice verse. U and V are the total number of users and items.

5.4 Experimental Settings

We implemented NGMM with the following settings: 1) we set the maximum length of each user or item description document to cover the 85% percent of the whole documents. Then we cut or pad each document to the same length. 2) we initialize the word



(a) **Dataset I**



(b) **Dataset II**

Figure 3: MSE Improvements of DMM over other baselines on (a) Dataset I (b) on Dataset II.

embedding randomly with dimensional size of 100. The impact of dimension size will also be considered in the experiments. 3) we utilize the dropout layer and set dropout rate to 0.2 so as to prevent model from over-fitting. 4) we compute the empirical mean g from the training set. Then we feed the $r'_{u,v} = r_{u,v} - g$ to NGMM model, when NGMM makes predictions, we add g back. 5) in Net_u, we use *relu* activation function in multiple convolutional layers and utilize tanh in the fully-connected layer. In Net_v, we use tanh activation function both in multiple convolutional layers and fully-connected layer. As for other methods, the optimal experimental settings are determined either by experiments or suggested by the authors.

5.5 Performance Evaluation on Amazon Datasets

The performance of NGMM with four baselines are reported in table 4 and 5 which corresponds to **Dataset** I and **Dataset** II respectively. Each result is repeated 3 times. The average and variance are presented with the best performance in bold. Meanwhile, the percentage of improvements gained by NGMM over other baselines are shown in Fig. 3.

From table 4, we can observe that all methods perform better than PMF except for ConvMF. It's because that ConvMF can't deal with the situation where some users or items don't occur in training data. Thus, following the pre-process in ConvMF [6], we construct another dataset, **Dataset** II, which guarantee that each user and item are in training data and users are removed when its' rating numbers are less than 3. The performance in table 5 shows that almost all approaches achieve better result than PMF except for the ConvMF in *Musical Instruments* dataset. It's because that this dataset is the most sparse with least reviews 2.47 for each item. As is describe above, ConvMF utilizes CNN architecture to extract features from the reviews of item and incorporate them into the user-item rating matrix. When the reviews of each item increase in other four datasets, it's not surprising to observe that ConvMF performs better than pMF. In a word, reviews can actually improve the performance of rating prediction task. It's worth noting that we didn't show the improvements of NGMM over ConvMF in Fig. 3, because ConvMF performs worse than PMF.

From table 4 and 5, NGMM performs better than RMR on all five datasets. The improvements of MSE ranges from almost 4% to 17% on **Dataset I** and 3% to 17% on **Dataset II**. The main reason is that RMR utilizes the traditional topic model to extract features from reviews. It can't capture the complex semantic relations between words and ignore the words' sequential information. Meanwhile, compared with RMR, NGMM performs the most significant improvements on *Amazon Instant Video*. It's because that *Amazon Instant Video* is the most dense dataset among the five Amazon review datasets. It also demonstrates that our proposed model, NGMM, can utilize reviews more efficiently than RMR.

Although ConvMF and DeepCoNN also use deep learning technique to model reviews, it can be observed that, from table 4 and 5, NGMM performs better than both ConvMF and DeepCoNN. Compared with ConvMF, the improvements of MSE ranges from almost 6% to 19% on **Dataset II**. Furthermore, NGMM gets an improvement of MSE over DeepCoNN from almost 7% to 11% and 3% to 7% on **Dataset I** and **Dataset II** respectively. It demonstrates the effectiveness of Gaussian mixture model for rating information which can efficiently imitate the rating behaviour of users to items and improve the prediction accuracy.

5.6 Effect of Number of Latent Components

In this subsection, we evaluate the effects of number of latent components of Gaussian mixture model in NGMM on *Amazon Instant Video*. It's worth noting that we almost have the observation on other datasets. In Fig.4, we show the performance of NGMM on *Amazon Instant Video* with varying K from 2 to 10. As it can be seen, the best performance is located at $k = 8$. When the values of K is bigger than 8 or smaller than 3, the performance of NGMM decreases.

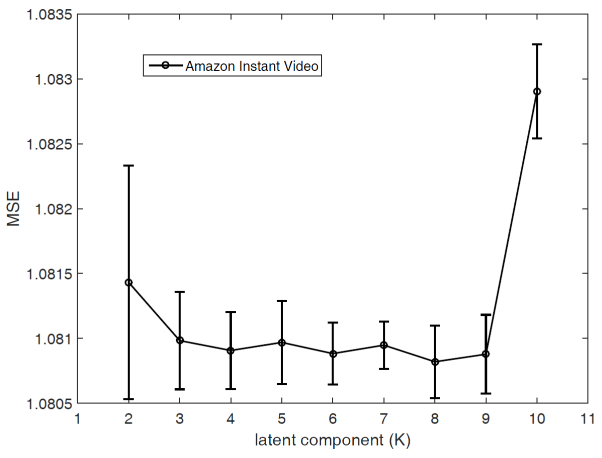


Figure 4: MSE (%) of NGMM over different number of latent components on Amazon Instant Video dataset.

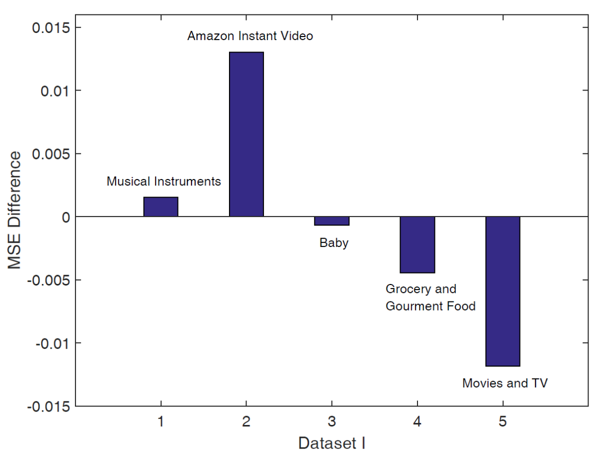


Figure 5: MSE difference of NGMM with or without existing word embeddings. value of y-axis is computed by $MSE_{not\ use\ word\ embedding}$ - $MSE_{use\ existing\ word\ embedding}$

5.7 Effect of pre-trained word embedding

We empirically evaluate the impacts of pre-trained word embeddings in NGMM. In Fig.5, we show the performance of NGMM on five Amazon review datasets with or without existing word embeddings. The MSE difference is computed by $MSE_{not\ use\ word\ embedding}$ - $MSE_{use\ existing\ word\ embedding}$. In this experiment, we use the pre-trained embeddings from glove[1] [14]. As it can be seen, for *Musical Instruments* and *Amazon Instant Video* which is two smallest dataset among five datasets, using pre-trained word embedding can improve the accuracy. However, when our dataset gets bigger for *Baby*, *Grocery and Gourmet Food* and *Movies and TV*, using pre-trained word embeddings on the contrary decreases the performance. It's because that existing embeddings from other domains may also introduce much *noise*. These *noise* corresponds to the knowledge which is not proper for the current task.

6 CONCLUSION AND FUTURE WORK

In this paper, we propose a neural Gaussian mixture model (NGMM) to exploit review textual information for rating prediction. It uses a parallel neural network to extract features from user and item reviews and design a mixture of Gaussian layer on top to capture the interaction between users and items. Furthermore, we compare NGMM with four baselines on five real world Amazon review datasets. In experiments, our proposed model, NGMM, achieves 3% to 20% improvements. The results show that except for historical ratings, review textual information is also an important information to improve the rating prediction. At the same time, it also demonstrates the effectiveness of neural Gaussian mixture model to model rating information.

This work focuses on predicting unknown ratings of users to items. In the future work, we will try to extend this model to not only deal with the prediction task, but also introduce the explanation mechanism. In addition, we will evaluate our proposed model, NGMM, on more datasets, such as the whole Amazon review datasets and yelp.

[1]https://nlp.stanford.edu/projects/glove/

REFERENCES

[1] Amjad Almahairi, Kyle Kastner, Kyunghyun Cho, and Aaron Courville. 2015. Learning distributed representations from reviews for collaborative filtering. In *Proceedings of the 9th ACM Conference on Recommender Systems*. ACM, 147–154.

[2] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research* 3 (2003), 993–1022.

[3] Jianbing Zhang Shujian Huang Jiajun Chen Hongjian Xue, Xinyu Dai. 2017. Deep matrix factorization models for recommender systems. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*. 3203–3209.

[4] Niklas Jakob, Stefan Hagen Weber, Mark Christoph Müller, and Iryna Gurevych. 2009. Beyond the stars: exploiting free-text user reviews to improve the accuracy of movie recommendations. In *Proceedings of the 1st International CIKM Workshop on Topic-sentiment Analysis for Mass Opinion (TSA '09)*. ACM, 57–64.

[5] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Baltimore, Maryland, 655–665.

[6] Donghyun Kim, Chanyoung Park, Jinoh Oh, Sungyoung Lee, and Hwanjo Yu. 2016. Convolutional matrix factorization for document context-aware recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16)*. ACM, 233–240.

[7] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009).

[8] Fangtao Li, Nathan Liu, Hongwei Jin, Kai Zhao, Qiang Yang, and Xiaoyan Zhu. 2011. Incorporating reviewer and product information for review rating prediction. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Three (IJCAI'11)*. AAAI Press, 1820–1825.

[9] Piji Li, Zihao Wang, Zhaochun Ren, Lidong Bing, and Wai Lam. 2017. Neural rating regression with abstractive tips generation for recommendation. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '17)*. ACM, New York, NY, USA, 345–354.

[10] Sheng Li, Jaya Kawale, and Yun Fu. 2015. Deep collaborative filtering via marginalized denoising auto-encoder. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (CIKM '15)*. ACM, New York, NY, USA, 811–820.

[11] Guang Ling, Michael R. Lyu, and Irwin King. 2014. Ratings meet reviews, a combined approach to recommend. In *Proceedings of the 8th ACM Conference on Recommender Systems (RecSys '14)*. ACM, 105–112.

[12] Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: Understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems (RecSys '13)*. ACM, 165–172.

[13] Arkadiusz Paterek. 2007. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD Cup and Workshop*. 39–42.

[14] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1532–1543.

[15] Steffen Rendle and Lars Schmidt-Thieme. 2010. Pairwise interaction tensor factorization for personalized tag recommendation. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining (WSDM '10)*. ACM, New York, NY, USA, 81–90.

[16] Ruslan Salakhutdinov and Andriy Mnih. 2008. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proceedings of the 25th International Conference on Machine Learning (ICML '08)*. ACM, New York, NY, USA, 880–887.

[17] Ruslan Salakhutdinov and Andriy Mnih. 2008. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, Vol. 20.

[18] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. 2007. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th International Conference on Machine Learning (ICML '07)*. ACM, New York, NY, USA, 791–798.

[19] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15 (2014), 1929–1958.

[20] Aaron van den Oord, Sander Dieleman, and Benjamin Schrauwen. 2013. Deep content-based music recommendation. In *Advances in Neural Information Processing Systems*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 2643–2651.

[21] Chong Wang and David M. Blei. 2011. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '11)*. ACM, 448–456.

[22] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '15)*. ACM, New York, NY, USA, 1235–1244.

[23] Xinxi Wang and Ye Wang. 2014. Improving content-based and hybrid music recommendation using deep learning. In *Proceedings of the 22Nd ACM International Conference on Multimedia (MM '14)*. ACM, New York, NY, USA, 627–636.

[24] Yao Wu, Christopher DuBois, Alice X. Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-N recommender systems. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining (WSDM '16)*. ACM, New York, NY, USA, 153–162.

[25] H. Fang Y. Bao and J. Zhang. 2014. TopicMF: Simultaneously exploiting ratings and reviews for recommendation. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*. 2–8.

[26] Dacheng Tao Yali Du, Chang Xu. 2017. Privileged matrix factorization for collaborative filtering. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*. 1610–1616.

[27] Lei Zheng, Vahid Noroozi, and Philip S. Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining (WSDM '17)*. ACM, 425–434.