Deep Energy Factorization Model for Demographic Prediction

Chih-Te Lai, Po-Kai Chang, Cheng-Te Li, Shou-De Lin

Abstract

Demographic information is important for various commercial and academic proposes, but, in reality, few of these data are accessible for analyses and researches. In order to solve this problem, several studies predict demographic attributes from users behavioral data. However, previous works suffer from different kinds of disadvantages. Specially, handling data sparseness and defining useful features for multiple demographics prediction remain challenge tasks. In this paper, we propose a novel Deep Energy Factorization Model (DEFM) to address these two drawbacks. The model is a designed encoder-decoder network which performs multi-label classification and feature representation in the same time. Experiments are conducted on four real-world recommendation datasets with four evaluation metrics. The empirical results show that DEFM significantly outperforms other state-of-the-art models most of the time.

1 Introduction

Demographic attributes are vital for business companies to plan marketing strategies, conduct market analysis and offer personalized recommendations. However, because of privacy concerns and other reasons, it's often difficult for companies to obtain users' demographic information such as age, gender, occupation, and educational background, etc. (Kobsa, Knijnenburg, and Livshits 2014) show that users are reluctant to disclose their demographic information to recommendation systems. (Dong et al. 2014) also point out that a large portion of mobile users don't supply their mobile service providers with any personal information.

This issue promotes studies of a popular research topic, demographic prediction. The primary objective is to infer users' demographic attributes by exploiting their behavioral data (e.g. purchase history, ratings on items, comments on news, etc.). Although it is difficult to collect demographic data, behavioral data, on the other hand, is relatively available for companies to acquire. In this paper, we focus on inferring multiple demographic attributes based on ratings in recommendation systems.

Several studies aim to predict demographic attributes by taking advantages of various behavioral data. (Weinsberg et al. 2012) use viewers' movie ratings to predict their genders. (Murray and Durrell 2000) infer users' profiles by their web browsing records. (Tang et al. 2008) extract researchers' profiles in an academic network. (Wang et al. 2016b) study demographic prediction in the retail scenario. These studies reveal the fact that demographic information is predictable and has strong connection with behavioral data.

Although demographic attributes can be inferred by behavioral data, the previous studies about demographic prediction imply limitations that do not meet real-world requests. First, some work (Culotta, Kumar, and Cutler 2015; Zhong et al. 2015) predict different attributes separately; however, because of the rarity of demographic attributes, it is difficult to train a model only in terms of one single attribute. Second, many studies (Dong et al. 2014; Hu et al. 2007; Schler et al. 2006; Culotta, Kumar, and Cutler 2015; Mislove et al. 2010) build their models relying on manually defined features. But defining features requires professional knowledge. Thus, it may not be feasible to generalize their models to other tasks. Finally, some work (Wang et al. 2016b; 2016a) utilize deep neural networks to deal with large-scale datasets that suffer from the data sparsity problem, but extracting useful features from the noisy data remains a difficult task for neural networks.

To alleviate these problems, we formalize demographic prediction as a multi-label classification task and propose *Deep Energy Factorization Model* (DEFM) to predict demographic attributes. DEFM is a special encoder-decoder neural network. The encoder part is a fully-connected neural network that aims at generating informative codes from behavioral data. The decoder part is a hybrid network with two branches. One branch is the counterpart of the encoder that reconstructs behavioral data, and the other is a variation of Restricted Boltzmann Machine (RBM), which is for multi-label classification. Both encoder and decoder of DEFM are updated jointly during the training process. There are three main advantages of DEFM: 1) DEFM can perform representation learning and support end-to-end training. 2) DEFM reconstructs data and handles the data sparsity problem in real-world datasets. 3) DEFM predicts multiple demographic attributes simultaneously. We verify our claims on real-world datasets with various evaluation metrics. The results show that DEFM significantly outperforms several state-of-the-art models. Our contributions are as follows:

- We present a novel learning framework, DEFM, for data

representation and multi-label classification. It boosts the performance of demographic prediction empirically.

- We propose a new hybrid model that can extract informative codes from behavioral data and infer demographic attributes jointly.
- We conduct experiments on real-world datasets, and our model outperforms state-of-the-art models in terms of several evaluation metrics.

The rest of this paper is organized as follows. Section 2 briefly summarizes related work. In Section 3, we introduce our problem statement and related approaches. Section 4 elaborates the details of our model. In Section 5, we provide the experimental results. Section 6 concludes this paper and provides prospect future work.

2 Related Work

In this section, we review related topics: demographic prediction and multi-label classification.

2.1 Demographic Prediction

Demographic prediction is a widely studied topic in academic areas. Several pioneers predict demographics on the basis of texts with clear semantic meaning. (Bhagat, Rozenbaum, and Cormode 2007) show that users' ages and genders information can be inferred from their blogs. (Schler et al. 2006) indicate that the strong differences in writing style and content can be utilized to infer genders and ages. (Otterbacher 2010) conduct users' reviews of items and use logistic regression to predict their genders.

Recently, due to the growth of online social networks, lots of user data offer opportunities for demographic prediction researches in social network analysis. (Mislove et al. 2010) reveal that users with common profiles are more likely to be friends. (Dong et al. 2014) propose a factor graph model to represent the interactions between users' profiles and social features on mobile communication networks. (Culotta, Kumar, and Cutler 2015) use logistic regression to predict demographic variables of Twitter users by whom they follow.

Commercial data (e.g. ratings on items, retail records) are also related to demographic prediction. (Narayanan and Shmatikov 2008) present a statistical attack for privacy breaches by calculating similarities between records. (Calandrino et al. 2011) use transaction history to infer other customers' transactions from their temporal changes in a recommendation system. (Weinsberg et al. 2012) utilize classifiers to infer their genders from movie ratings. (Bhagat et al. 2014) also suggest a possible attack based on matrix factorization in active learning setting. (Wang et al. 2016b; 2016a) propose Structured Neural Embedding (SNE) model and multi-task representation learning in retail scenario. Although these models address demographic prediction, extracting latent relations among commercial data and reducing the data sparsity are rarely conducted.

2.2 Multi-label Classification

Multi-label classification (MLC) is a classification task that predicts multiple labels at once for a given data point. That is, we assume there are a fixed number of categories for each predefined label. Then, given an arbitrary data point, algorithms in MLC aim to define the mapping from the data point to its labels.

The approaches can be either extending the algorithms from multi-class to multi-label or resolving the multi-label problems to multi-class problems. Adapt boosting (Schapire and Singer 2000), decision tree (Vickrey, Lin, and Koller 2010), k-nearest neighbors (Zhang and Zhou 2007), and neural networks (Zhang and Zhou 2006; Jing and Lin 2014) are the algorithms that utilize the first approach.

3 Preliminaries

In this section, we introduce notations and techniques in order to further explain our models. Briefly, we give the definition of demographic prediction problem and describe the neural conditional energy model (NCEM), which have close connections with our model. Then, we review the concept of denoising autoencoders (DAE).

3.1 Problem Definition

We aim to predict multiple demographic attributes based on users' ratings on items in recommendation systems. Particularly, a recommender system stores users' ratings and accesses to a dataset that contains demographic attributes shared by a part of users. Given users' ratings and available demographic attributes, the goal is to predict the other users' unknown attributes.

We formulate our problem mathematically. Consider a recommendation system comprises of ratings to M items given N users. We define r_{ij} as the rating of i-th user to j-th item, and $R \in \{1, 2, ..., N\} \times \{1, 2, ..., M\}$ denotes the set of ratings. Let $A = \{a_1, a_2, ..., a_K\}$ be a set of K demographic attributes. Each attribute $a_k \in A$ is associated with C_k possible values, where $C_k \geq 2, k = 1, 2, ..., K$. For each $i \in \{1, 2, ..., N\}$, we define $x^{(i)} = [r_{ij} | \forall j \in \{1, 2, ..., M\}, r_{ij} \in R]$ and $y^{(i)} = \left[y_1^{(i)}, y_2^{(i)}, ..., y_K^{(i)}\right]$, where $y_k^{(i)}$ is C_k-dimensional one-hot vector that indicates the specific value of i-th users' attribute a_k, for $k = 1, 2, ..., K$. Let $C = \sum_k C_k$. Then, $X = \left[x^{(i)}\right]$ denotes a $(N \times M)$-dimensional matrix of ratings, $Y = \left[y^{(i)}\right]$ denotes a $(N \times C)$-dimensional matrix of one-hot encodings of demographic attributes, where $i \in \{1, 2, ..., N\}$. Given above notations, the objective of multi-label demographic prediction is to learn a function $f : X \to Y$ such that $\hat{Y} = f(X)$, the demographic prediction, is as close to Y as possible.

We use a simple example to the problem notations in Figure 1. In this example, $K = 3$ and $A = \{a_1, a_2, a_3\}$, where a_1 denotes *age*, a_2 represents *gender*, and a_3 symbolizes *occupation*. For user $i = 3141$ and item $j = 81$, $r_{i,j} = 5$, $y^{(i)} = \left[y_1^{(i)}, y_2^{(i)}, y_3^{(i)}\right]$, where the exampled one-hot encodings are $y_1^{(i)} = [0, 0, 0, 1]$, $y_2^{(i)} = [1, 0]$, and $y_3^{(i)} = [0, 0, 0, 0, 1]$.

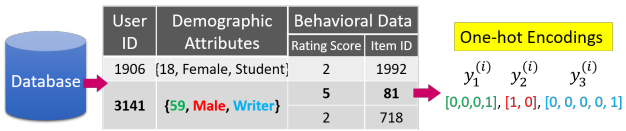


Figure 1: A toy example for our problem notations.

3.2 Neural Conditional Energy Model

Neural Conditional Energy Model (NCEM, shown in Figure 2(a)) (Jing and Lin 2014) is a hybrid model that combines the concepts of CRBM with deep neural network and is defined according to the following energy function,

$$E(v,h,y) = -f_{NN}(v;\lambda)^\top Qh - y^\top Uh - f_{NN}(v;\lambda)^\top Lt - g^\top h - s^\top y$$

where $f_{NN}(v;\lambda)$ denotes the output given by deterministic feed-forward neural network parameterized by λ. Thanks to back-propagation and other learning tricks, $f_{NN}(v;\lambda)$ can be trained efficiently. There is only one stochastic hidden layer, h; thus, the computation of inference is required at the last layer, which means the low-level representation can be learned efficiently using feed-forward passes.

To learn this model, a conditional log-likelihood is defined,

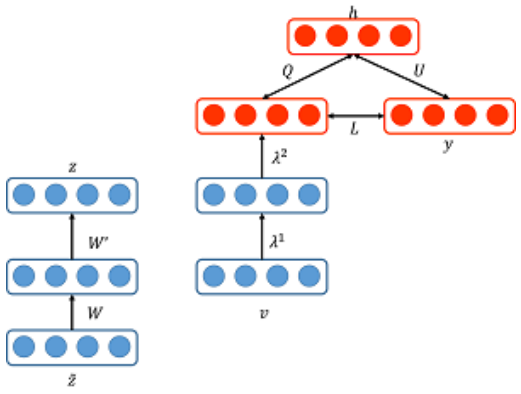
$$\frac{\partial \log p(y|v)}{\partial \lambda} = \sum_h p(h|v,y)\frac{\partial E(v,h,y)}{\partial \lambda} - \sum_{h,y^*} p(h,y^*|v)\frac{\partial - E(v,h,y^*)}{\partial \lambda}$$

Considering the energy function defined above, h denotes the stochastic hidden layer, and the deterministic hidden layers are already defined in the neural network with parameter λ. Given the training set $\{V, Y\}$, one can maximize the function and learn the parameters $\{Q, U, L, g, s, \lambda\}$ with mini-batch stochastic gradient ascent. Two expectation values are required to be computed. The first term can be calculated exactly, but the second term is intractable. In order to infer the expectation and train NCEM, (Jing and Lin 2014) propose a conditional stochastic back-propagation (CSBP) algorithm based on Contrastive Divergence (CD) method and back-propagation. Instead of starting the sampling from a random status, the CD method starts the sampling chain at a random training vector, which practically reduces plenty of time.

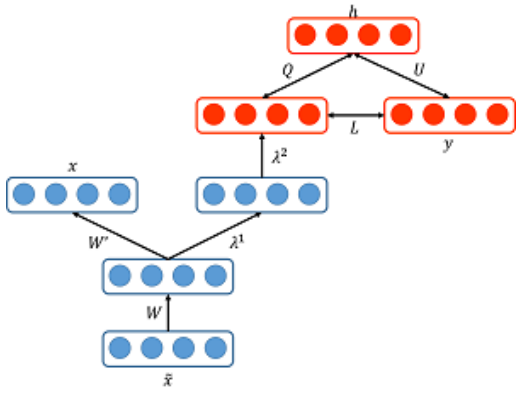
3.3 Denoising Autoencoder

The main goal behind denoising autoencoders (DAE, shown in Figure 2(a)) is to reconstruct data from inputs of corrupted data. Since learning an identity mapping like autoencoder may not come out useful features, giving the autoencoder the corrupted data is a way to avoid the issue. This approach forces the hidden layer to learn more robust features.

There are several methods for corrupting input data. The basic way is that randomly remove some parts of the data and lets the autoencoder predict these parts of data. Given



(a) DAE (left) and NCEM (right)



(b) DEFM

Figure 2: Architecture of models. (a): Denoising Autoencoder (DAE) and Neural Conditional Energy Model (NCEM). (b): Deep Energy Factorization Model (DEFM). Orange nodes stand for random variables; blue nodes stand for deterministic neurons. For simplicity, the biases between layers are not shown in figures.

input vector z, a corrupted vector z' is obtained by randomly assigning elements to zeros with some probability p, $0 < p < 1$.

Using the corrupted vector $\tilde{z}$, DAE computes a reconstruction vector $\hat{z}$,

$$z^c = f_{enc}(\tilde{z}) = \sigma(W\tilde{z} + b)$$
$$\hat{z} = f_{dec}(z^c) = \sigma(W'z^c + b')$$

DAE is trained by minimizing specific loss function l. Instead of $\tilde{z}$, the loss between z and $\hat{z}$, $l(\hat{z}, z)$, is calculated.

4 Methodology

In this section, we explain our model, DEFM, including the model architecture and the learning approach.

4.1 Deep Energy Factorization Model

DEFM (shown in Figure 2(b)) is a deep neural network combined with DAE and NCEM. We employ DAE to capture useful features and NCEM to perform multiple demographic

prediction. DEFM first encodes the input behavioral data to latent representations, and then the representations are used to reconstruct data and predict attributes. The whole model is trained and tested jointly. Specifically, given the notations in preliminaries, for each data pair $\{x, y\} \in \{X, Y\}$, the latent representation c is obtained by

$$c = f_{enc}(\tilde{x}) = W\tilde{x} + b$$

where $\tilde{x}$ is the corrupted version of x, and the energy function of DEFM is formulated by

$$\begin{aligned} E(c, h, y) = &- f_{NN}(c; \lambda)^\top Qh - y^\top Uh \\ &- f_{NN}(c; \lambda)^\top Ly - g^\top h - s^\top y \end{aligned}$$

Similar to NCEM, we use the following conditional probability distributions to infer y,

$$p(y|c) = \frac{\sum_h \exp(-E(c, h, y))}{\sum_{h,y'} \exp(-E(c, h, y'))}$$

$$p(h|c, y) = \prod_{j=1}^{|h|} \sigma\left(c^\top Q_{\cdot j} + y^\top U_{\cdot j} + g_j\right)$$

$$p(y|c, h) = \prod_{l=1}^{|y|} \sigma\left(c^\top L_{\cdot l} + U_{l\cdot} h + s_l\right)$$

4.2 Learning

In order to train DEFM, we maximize the objective function defined by

$$\log p(y|c) - \gamma l(\hat{x}, x)$$

where $\hat{x} = f_{dec}(c)$, and γ is a regularization scalar to balance the importance of the log-likelihood $\log p(y|c)$ and the loss function $l(\hat{x}, x)$. We aim to learn parameters $\{Q, U, L, W, W', g, s, b, b', \lambda\}$ from a given training dataset $\{X, Y\}$. For an instance $\{x, y\} \in \{X, Y\}$, the gradient of the objective function associated with parameter θ is formulated by

$$\begin{aligned} \frac{\partial \log p(\hat{x}, x)}{\partial \theta} - \gamma \frac{\partial l(\hat{x}, x)}{\partial \theta} \qquad & \theta \in \{W, b\} \\ - \gamma \frac{\partial l(\hat{x}, x)}{\partial \theta} \qquad & \theta \in \{W', b'\} \\ \frac{\partial \log p(y|c)}{\partial \theta} \qquad & \theta \in \{Q, U, L, g, s, \lambda\} \end{aligned}$$

The gradient $\frac{\partial l(\hat{x}, x)}{\partial \theta}$ can be calculated by using back-propagation. The gradient $\frac{\partial \log p(y|c)}{\partial \theta}$ can be written by

$$\begin{aligned} \frac{\partial \log p(y|c)}{\partial \theta} = &\sum_h p(h|c, y) \frac{\partial - E(c, h, y)}{\partial \theta} \\ &- \sum_{h, y^*} p(h, y^*|c) \frac{\partial - E(c, h, y^*)}{\partial \theta} \end{aligned}$$

To optimize the log-likelihood, we apply CSBP (Jing and Lin 2014) to estimate the second term, which is intractable. Briefly, CSBP is an optimization algorithm combining CD method and back-propagation. The gradient of log-likelihood is approximated with CD method on sample space $\{h, y\}$, and the parameters of NN in NCEM is updated by back-propagating the gradient. We make a few modification on CSBP to train DEFM. For further explanation of our learning algorithm, we rewrite the gradient $\frac{\partial \log p(y|c)}{\partial \theta}$ into free energy form

$$\frac{\partial \log p(y|v)}{\partial \lambda} = \frac{\partial - F(v, y)}{\partial \lambda} - \sum_{y^*} p(y^*|v) \frac{\partial - F(c, y^*)}{\partial \lambda}$$

where $F(c, y)$ is the free energy function that can be formulated by

$$\begin{aligned} F(c, y) &= -\log \sum_h \exp(-E(c, h, y)) \\ &= -\sum_j \log\left(1 + \exp\left(g_j + f_{NN}(c; \lambda)^\top Q_{\cdot j} + y^\top U_{\cdot j}\right)\right) \\ &\quad - f_{NN}(c; \lambda)^\top Ly - s^\top y \end{aligned}$$

We then can compute the derivatives of free energy with respect to parameters $\{Q, U, L, g, s\}$

$$\begin{aligned} \frac{\partial F(c, y)}{\partial Q_{ij}} &= -\frac{\exp\left(g_j + f_{NN}(c; \lambda)^\top Q_{\cdot j} + y^\top U_{\cdot j}\right)}{1 + \exp\left(g_j + f_{NN}(c; \lambda)^\top Q_{\cdot j} + y^\top U_{\cdot j}\right)} c_i \\ &= -\sigma\left(g_j + f_{NN}(c; \lambda)^\top Q_{\cdot j} + y^\top U_{\cdot j}\right) c_i \\ \frac{\partial F(c, t)}{\partial U_{lj}} &= -\frac{\exp\left(g_j + f_{NN}(c; \lambda)^\top Q_{\cdot j} + y^\top U_{\cdot j}\right)}{1 + \exp\left(g_j + f_{NN}(c; \lambda)^\top Q_{\cdot j} + y^\top U_{\cdot j}\right)} y_l \\ &= -\sigma\left(g_j + f_{NN}(c; \lambda)^\top Q_{\cdot j} + y^\top U_{\cdot j}\right) y_l \\ \frac{\partial F(c, y)}{\partial L_{il}} &= -y_l f_{NN}(c; \lambda)_i \\ \frac{\partial F(c, y)}{\partial g_j} &= -\frac{\exp\left(g_j + f_{NN}(c; \lambda)^\top Q_{\cdot j} + y^\top U_{\cdot j}\right)}{1 + \exp\left(g_j + f_{NN}(c; \lambda)^\top Q_{\cdot j} + y^\top U_{\cdot j}\right)} \\ &= -\sigma\left(g_j + f_{NN}(c; \lambda)^\top Q_{\cdot j} + y^\top U_{\cdot j}\right) \\ \frac{\partial F(c, y)}{\partial s_l} &= -y_l \end{aligned}$$

For the output of NN, we have

$$\begin{aligned} \frac{\partial F(c, y)}{\partial f_{NN}(c; \lambda)_i} = &-\sum_j \sigma\left(g_j + f_{NN}(c; \lambda)^\top Q_{\cdot j} + y^\top U_{\cdot j}\right) Q_{ij} \\ &- (Ly)_i \end{aligned}$$

We now summarize our learning approach. First, we feed our data in DAE and pass to DNN in NCEM directly. Second, we use Gibbs sampling to generate the hidden layer in NCEM. Last but not least, we back-propagate the error signals from both NECM and DAE and update all parameters of DEFM. Formally, for a training case (x, y), our learning algorithm can be represented as the followings:

1. Given an input vector x, compute encoded vector $c = f_{enc}(\tilde{x})$ and reconstructed vector $\hat{x} = f_{dec}(c)$.

2. Compute the loss of reconstruction $l(\hat{x}, x)$.
3. Feed c through DNN and obtain an output $f_{NN}(c; \lambda)$.
4. Given the output $f_{NN}(c; \lambda)$ and a label vector y, run Gibbs sampling for k steps by alternately updating h and y.
5. Collect the sample as y^k.
6. For $\theta \in \{Q, U, L, W, g, s, b, \lambda\}$, update it using the following rule,

$$\theta \leftarrow \theta + \alpha \left(-\frac{\partial F(x, y)}{\partial \theta} + \frac{\partial F\left(x, y^k\right)}{\partial \theta} \right)$$

7. For $\theta \in \{W, W', b, b'\}$, update it using the following rule,

$$\theta \leftarrow \theta - \alpha\gamma \frac{\partial l(\hat{x}, x)}{\partial \theta}$$

where α is the learning rate, γ is the regularization weight.

4.3 Prediction

When predicting demographic attributes on a testing dataset $\{X, Y\}$, maximum a posterior inference (MAP) is required to search for the demographic predictions $\bar{Y}$ such that the posterior probability $p\left(\bar{Y}|X\right)$ is maximized. Because MAP is intractable in DEFM, Gibbs sampling is performed to approximate the posterior probability. The prediction algorithm is similar to the one we mention above. Specifically, for one testing case (x, y), the prediction algorithm is described as the followings:

1. Compute the encoded vector $c = f_{enc}(x)$.
2. Feed c through the deterministic DNN and obtain $f_{NN}(c; \lambda)$.
3. Randomly generate the hidden vector h and the label vector $\hat{y}$.
4. Given the output $f_{NN}(c; \lambda)$ and a label vector $\hat{y}$, run Gibbs sampling for k steps by alternately updating h and $\hat{y}$.
5. Use the label vector $\hat{y}$ to generate the final prediction $\bar{y}$

Notice that we do not corrupt the input and use the reconstruction of input during prediction phase.

5 Experiments

We conduct experiments to illustrate the ability of demographic attributes prediction of NEFM by comparing it with baseline models on several datasets. The experimental settings are introduced first. Then for each dataset, we list several tables of demographic attributes prediction evaluation results to compare DEFM with other baseline models.

5.1 Dataset

Our experiments are run on several real-world datasets including *MovieLens 100k Dataset*, *MovieLens 1M Dataset*, *Facebook Social Network Dataset*, and *Youtube Communities Dataset*. Both *MovieLens 1M Dataset* and *MovieLens 100k Dataset* comprise exactly three demographic attributes including age, gender, and occupation. *Facebook Social Network Dataset* and *Youtube Communities Dataset* contain users' information of social circles and communities respectively. For research propose, we preprocess the data from each dataset in our experiments as follows:

- *MovieLens 100K Dataset*:
  MovieLens 100K Dataset contains 1000 ratings from 1000 users on 1700 movies. We split ages to groups as 0-17, 18-35, 36-65, and 66-100. Each group is a label. We treat each occupation as a label including "retired", "none", and "other".
- *MovieLens 1M Dataset*:
  MovieLens 1M Dataset contains 1 million from 6000 users on 4000 movies. We follow the setting of MovieLens 1M to categorize the age groups as age 0-17, age 18-24, age 25-34, age 35-44, age 45-49, age 56-100. The occupation's labeling method is the same as the one in *MovieLens 100K Dataset*.
- *Facebook Social Network Dataset*:
  Facebook Social Network Dataset (Leskovec and Krevl 2014) consists of 10 networks of social circles (lists of friends) from Facebook. In each network, every person has at least one friend link to another, which is recorded in its corresponding .edges file. For each person, whether he belongs to is viewed as an attribute. Each trained model aims to predict the existence in circles through the edges information for each person.
- *Youtube Communities Dataset*:
  Youtube Communities Dataset (Leskovec and Krevl 2014) consists one huge social network with 5000 communities. We select the largest 10 communities and the corresponding links to serve our experiment. Given a person's links, each model aims to predict his existence in the 10 communities.

5.2 Model Settings

We compare DEFM with several state-of-the-art methods on demographic attributes prediction. It should be mentioned that all models in experiments have predefined number of latent features. We perform the experiments with 100, 200, 300, and 400 latent features (code sizes) respectively.

- **SNE.** The negative sampling numbers is set to 1. In addition, for *MovieLens 100K Dataset* and *MovieLens 1M Dataset*, only records of ratings greater than or equivalent to 3 are used. We implemented SNE in Python at this link [1].
- **BPMF with NCEM (BFEM).** We employ Bayesian probabilistic matrix factorization (Salakhutdinov and Mnih 2008) to factorize the ratings and use the latent features as the input of NCEM. The hidden sizes of NCEM are set to 800, 600 for a deterministic and a stochastic layer respectively. The negative sampling numbers, learning rate, and the iterations of sampling in training, testing

[1] https://github.com/LplusKira/SNE_lab

phase are set to 5, 0.005, 2000, and 100. We refer the implementation of BPMF at the link[2] and the implementation of NCEM at the link[3].

- **NMF with NCEM (NFEM).** We employ nonnegative matrix factorization (NMF) to factorize the ratings and input use the latent features as the input of NCEM. The setting of NCEM is the same as BFEM.
- **AE with NCEM (AEEM).** We employ denoising autoencoder (DAE) to learn the latent representation and use the representation as the input of NCEM. The setting of NCEM is the same as BFEM.
- **DEFM.** The hidden sizes of NCEM are set to 800, 600 for a deterministic and a stochastic layer respectively. The regularization scalar γ between reconstruction loss and log-likelihood is 1. We use single layer neural network as f_{enc}. We also try f_{enc} with two layers and it generate the same results roughly. Our source code is available at this link[4].

5.3 Experiment Settings

For each train data and each of the four latent features sizes, all models generate the corresponding average and standard deviation of metrics through 10-fold cross-validation.

5.4 Evaluation Metrics

We consider 4 metrics in our experiments: Micro-F1, Ranking Loss, Average Precision, and Hamming Loss. Details can be referred in (Zhang and Zhou 2014).

5.5 Performance on Demographic Prediction

The performance of DEFM can be examined in Table 1 and Table 2, which deliver two findings. The performance of DEFM can be examined in Table 1 and Table 2, which delivers two findings.

1. DEFM has better results on *MovieLens 100k Dataset*, *MovieLens 1M Dataset*, and *Youtube Communities Dataset* in almost all metrics and code sizes. It may be due to users' attributes in training users' latent features that makes DEFM generally wins AEEM over MLC considering the four metrics. In the training phase of AEEM, since the training of DAE and NCEM are performed separately, the correlation of raw behavioral data and users' attributes are not taken into account in training DAE. It turns out that, by comparing DEFM with AEEM, the MLC task done by NCEM part can be improved if users' demographic attributes are considered in training DAE. Thus, the backpropagation from NCEM part helps generate better users' latent features. In addition, although SNE considers the correlation between users' latent features and demographic attributes in training, the generated features may not be representative enough. In our settings, SNE employs average pooling method (Wang et al. 2016b) to transform items features into each user's features. Therefore, SNE might not perform as well as our model.
2. All models follow consistent trends on metrics for almost all datasets. For positive metrics, exactly Micro-F1 Score and Average Precision, the values on any model increases as the code size grows up. For negative metrics, the error values decrease consistently along with the increment of code size. The main reason could be that the complexity between attributes and behavioral data can't be depicted well without more latent features.

6 Conclusion

This paper addresses the problem of multiple demographic prediction on recommendation systems. We propose the novel DEFM model, which predicts multi-label attributes and extract useful representations in a jointly learning framework. Experiments on real-world datasets demonstrate that DEFM outperforms other state-of-the-art models on each dataset under five evaluation metrics.

Although DEFM achieves better results in demographic prediction, it has limitations. First, the usage of memory for DEFM is proportional to the number of users, which may be unacceptable when dealing with large-scale datasets. Second, a better mechanism to automatically determine the parameters in DEFM is not available. Finally, it is obscure how to select proper behavioral data and demographic attributes. We plan to tackle these issues in our extensive work.

We also perform some potential attempts. First, we aim to jointly do the recommendation and predict demographic attribute based on DEFM. A mutually-reinforced mechanism may boost the performance of both tasks. Second, we plan to extend our learning algorithm to optimize a square error function and a log-likelihood function simultaneously to allow solving the tasks about multi-task learning. Finally, we aim at integrating DEFM with DAE and RBM, which may help to create new kinds of generative models.

References

Bhagat, S.; Weinsberg, U.; Ioannidis, S.; and Taft, N. 2014. Recommending with an agenda: Active learning of private attributes using matrix factorization. In *Proceedings of the 8th ACM conference on recommender systems*, 65–72. ACM.

Bhagat, S.; Rozenbaum, I.; and Cormode, G. 2007. Applying link-based classification to label blogs. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, 92–101. ACM.

Calandrino, J. A.; Kilzer, A.; Narayanan, A.; Felten, E. W.; and Shmatikov, V. 2011. " you might also like:" privacy risks of collaborative filtering. In *Security and Privacy (SP), 2011 IEEE Symposium on*, 231–246. IEEE.

Culotta, A.; Kumar, N. R.; and Cutler, J. 2015. Predicting the demographics of twitter users from website traffic data. In *AAAI*, 72–78.

Dong, Y.; Yang, Y.; Tang, J.; Yang, Y.; and Chawla, N. V. 2014. Inferring user demographics and social strategies in mobile social networks. In *Proceedings of the 20th ACM*

[2] https://www.cs.toronto.edu/~rsalakhu/BPMF.html

[3] https://github.com/Kublai-Jing/NCEM

[4] https://github.com/LplusKira/DEFM

Table 1: Ranking Loss and Hamming Loss (mean±std) of each model in datasets (**bold** indicates the best).

Metrics	Dataset	Code	SNE	NFEM	BFEM	AEEM	DEFM
Ranking Loss	*MovieLens 100K*	100	0.192±0.002	0.169±0.012	0.180±0.019	0.152±0.007	**0.137±0.019**
		200	0.187±0.003	0.185±0.012	0.201±0.025	0.145±0.014	**0.139±0.015**
		300	0.189±0.003	0.192±0.011	0.213±0.044	0.165±0.015	**0.145±0.013**
		400	0.188±0.002	0.193±0.004	0.218±0.031	0.161±0.022	**0.144±0.006**
	MovieLens 1M	100	0.263±0.006	**0.168±0.005**	0.211±0.022	0.187±0.008	0.177±0.003
		200	0.260±0.005	0.169±0.005	0.199±0.006	0.166±0.005	**0.160±0.010**
		300	0.260±0.005	0.170±0.005	0.206±0.018	**0.155±0.008**	**0.155±0.007**
		400	0.258±0.008	0.169±0.006	0.208±0.014	0.161±0.006	**0.156±0.007**
	Facebook Social Network	100	0.382±0.034	0.023±0.008	0.081±0.022	0.026±0.010	**0.021±0.007**
		200	0.349±0.025	0.015±0.004	0.048±0.014	**0.012±0.004**	0.014±0.005
		300	0.329±0.026	**0.009±0.002**	0.030±0.009	**0.009±0.004**	0.013±0.004
		400	0.295±0.023	**0.007±0.002**	0.029±0.007	0.008±0.002	0.008±0.004
	Youtube Communities	100	0.477±0.005	0.057±0.002	0.092±0.004	0.097±0.004	**0.048±0.003**
		200	0.473±0.005	0.056±0.003	0.091±0.004	0.096±0.009	**0.046±0.002**
		300	0.469±0.007	0.058±0.002	0.096±0.009	0.088±0.009	**0.047±0.002**
		400	0.462±0.005	0.057±0.004	0.096±0.007	0.085±0.005	**0.048±0.002**
Hamming Loss	*MovieLens 100K*	100	0.481±0.005	0.092±0.012	0.100±0.010	0.091±0.010	**0.089±0.008**
		200	0.462±0.010	0.093±0.002	0.111±0.006	**0.087±0.002**	0.091±0.004
		300	0.467±0.006	0.092±0.001	0.099±0.008	**0.090±0.007**	0.092±0.006
		400	0.467±0.008	0.094±0.005	0.110±0.008	0.091±0.014	**0.088±0.003**
	MovieLens 1M	100	0.573±0.017	**0.084±0.001**	0.091±0.008	0.088±0.006	0.087±0.002
		200	0.562±0.012	**0.085±0.002**	0.093±0.007	**0.085±0.001**	**0.085±0.003**
		300	0.562±0.015	0.085±0.002	0.095±0.014	0.083±0.002	**0.082±0.002**
		400	0.557±0.019	0.086±0.001	0.092±0.007	0.083±0.002	**0.081±0.002**
	Facebook Social Network	100	0.382±0.034	**0.049±0.012**	0.118±0.020	0.056±0.015	0.051±0.011
		200	0.349±0.025	0.037±0.006	0.077±0.015	0.036±0.009	**0.035±0.007**
		300	0.329±0.026	**0.026±0.004**	0.057±0.011	0.027±0.006	0.030±0.008
		400	0.295±0.023	**0.023±0.004**	0.053±0.009	0.025±0.005	0.026±0.007
	Youtube Communities	100	0.477±0.005	0.119±0.002	0.137±0.012	0.128±0.006	**0.090±0.006**
		200	0.473±0.005	0.120±0.002	0.135±0.014	0.143±0.015	**0.083±0.002**
		300	0.469±0.007	0.120±0.001	0.135±0.013	0.144±0.016	**0.084±0.003**
		400	0.462±0.005	0.120±0.001	0.146±0.015	0.142±0.011	**0.084±0.004**

Table 2: Micro-F1 and Average Precision (mean±std) of each model in datasets (**bold** indicates the best).

Metrics	Dataset	Code	SNE	NFEM	BFEM	AEEM	DEFM
Micro-f1 Score	*MovieLens 100K*	100	0.519±0.005	0.544±0.050	0.519±0.013	0.554±0.037	**0.555±0.006**
		200	0.538±0.010	0.540±0.010	0.469±0.031	0.558±0.024	**0.566±0.014**
		300	0.533±0.006	0.543±0.017	0.492±0.036	0.541±0.026	**0.547±0.028**
		400	0.533±0.008	0.535±0.013	0.456±0.030	0.550±0.055	**0.556±0.012**
	MovieLens 1M	100	0.427±0.017	**0.476±0.014**	0.426±0.017	0.439±0.020	0.462±0.015
		200	0.438±0.012	0.482±0.015	0.427±0.016	0.473±0.012	**0.484±0.012**
		300	0.438±0.015	0.478±0.017	0.435±0.026	0.495±0.012	**0.502±0.006**
		400	0.443±0.019	0.489±0.005	0.429±0.015	0.490±0.004	**0.508±0.017**
	Facebook Social Network	100	0.618±0.034	**0.951±0.011**	0.885±0.018	0.944±0.013	0.950±0.010
		200	0.651±0.025	0.962±0.005	0.924±0.015	**0.964±0.009**	**0.964±0.007**
		300	0.671±0.026	**0.974±0.004**	0.944±0.010	0.973±0.006	0.969±0.007
		400	0.705±0.023	**0.977±0.004**	0.947±0.009	0.975±0.005	0.974±0.006
	Youtube Communities	100	0.523±0.005	0.881±0.002	0.864±0.011	0.873±0.006	**0.911±0.005**
		200	0.527±0.005	0.880±0.003	0.865±0.014	0.860±0.011	**0.917±0.003**
		300	0.531±0.007	0.880±0.002	0.866±0.011	0.858±0.014	**0.916±0.003**
		400	0.602±0.004	0.880±0.001	0.855±0.014	0.860±0.010	**0.916±0.003**
Average Precision	*MovieLens 100K*	100	0.600±0.005	0.630±0.035	0.580±0.020	0.639±0.039	**0.646±0.024**
		200	0.614±0.009	0.630±0.007	0.547±0.036	**0.651±0.021**	0.643±0.020
		300	0.611±0.004	0.619±0.008	0.559±0.048	0.630±0.027	**0.637±0.028**
		400	0.610±0.005	0.609±0.020	0.534±0.031	0.630±0.052	**0.639±0.011**
	MovieLens 1M	100	0.470±0.017	**0.581±0.006**	0.515±0.033	0.542±0.018	0.557±0.009
		200	0.480±0.014	**0.585±0.013**	0.522±0.024	0.577±0.005	0.582±0.018
		300	0.482±0.015	0.577±0.019	0.513±0.046	0.595±0.012	**0.600±0.008**
		400	0.486±0.018	0.589±0.010	0.522±0.022	0.591±0.009	**0.606±0.015**
	Facebook Social Network	100	0.648±0.029	0.978±0.008	0.921±0.021	0.975±0.010	**0.980±0.007**
		200	0.670±0.023	0.984±0.006	0.948±0.016	**0.987±0.004**	0.985±0.006
		300	0.690±0.023	**0.990±0.003**	0.966±0.011	**0.990±0.003**	0.986±0.004
		400	0.721±0.021	**0.993±0.003**	0.968±0.009	0.992±0.002	0.991±0.004
	Youtube Communities	100	0.590±0.004	0.948±0.001	0.910±0.004	0.904±0.005	**0.957±0.003**
		200	0.593±0.004	0.949±0.003	0.910±0.004	0.904±0.010	**0.957±0.003**
		300	0.597±0.005	0.946±0.003	0.904±0.010	0.915±0.009	**0.956±0.002**
		400	0.538±0.005	0.948±0.004	0.905±0.007	0.918±0.006	**0.955±0.002**

SIGKDD international conference on Knowledge discovery and data mining, 15–24. ACM.

Hu, J.; Zeng, H.-J.; Li, H.; Niu, C.; and Chen, Z. 2007. Demographic prediction based on user's browsing behavior. In *Proceedings of the 16th international conference on World Wide Web*, 151–160. ACM.

Jing, H., and Lin, S.-D. 2014. Neural conditional energy models for multi-label classification. In *Data Mining (ICDM), 2014 IEEE International Conference on*, 240–249. IEEE.

Kobsa, A.; Knijnenburg, B. P.; and Livshits, B. 2014. Let's do it at my place instead? attitudinal and behavioral study of privacy in client-side personalization. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 81–90. ACM.

Leskovec, J., and Krevl, A. 2014. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`.

Mislove, A.; Viswanath, B.; Gummadi, K. P.; and Druschel, P. 2010. You are who you know: inferring user profiles in online social networks. In *Proceedings of the third ACM international conference on Web search and data mining*, 251–260. ACM.

Murray, D., and Durrell, K. 2000. Inferring demographic attributes of anonymous internet users. *Web Usage Analysis and User Profiling* 7–20.

Narayanan, A., and Shmatikov, V. 2008. Robust deanonymization of large sparse datasets. In *Security and Privacy, 2008. SP 2008. IEEE Symposium on*, 111–125. IEEE.

Otterbacher, J. 2010. Inferring gender of movie reviewers: exploiting writing style, content and metadata. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, 369–378. ACM.

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

Schapire, R. E., and Singer, Y. 2000. Boostexter: A boosting-based system for text categorization. *Machine learning* 39(2-3):135–168.

Schler, J.; Koppel, M.; Argamon, S.; and Pennebaker, J. W. 2006. Effects of age and gender on blogging. In *AAAI spring symposium: Computational approaches to analyzing weblogs*, volume 6, 199–205.

Tang, J.; Zhang, J.; Yao, L.; Li, J.; Zhang, L.; and Su, Z. 2008. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 990–998. ACM.

Vickrey, D.; Lin, C. C.; and Koller, D. 2010. Non-local contrastive objectives. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 1103–1110.

Wang, P.; Guo, J.; Lan, Y.; Xu, J.; and Cheng, X. 2016a. Multi-task representation learning for demographic prediction. In *European Conference on Information Retrieval*, 88–99. Springer.

Wang, P.; Guo, J.; Lan, Y.; Xu, J.; and Cheng, X. 2016b. Your cart tells you: Inferring demographic attributes from purchase data. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, 173–182. ACM.

Weinsberg, U.; Bhagat, S.; Ioannidis, S.; and Taft, N. 2012. Blurme: Inferring and obfuscating user gender based on ratings. In *Proceedings of the sixth ACM conference on Recommender systems*, 195–202. ACM.

Zhang, M.-L., and Zhou, Z.-H. 2006. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE transactions on Knowledge and Data Engineering* 18(10):1338–1351.

Zhang, M.-L., and Zhou, Z.-H. 2007. Ml-knn: A lazy learning approach to multi-label learning. *Pattern recognition* 40(7):2038–2048.

Zhang, M.-L., and Zhou, Z.-H. 2014. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering* 26(8):1819–1837.

Zhong, Y.; Yuan, N. J.; Zhong, W.; Zhang, F.; and Xie, X. 2015. You are where you go: Inferring demographic attributes from location check-ins. In *Proceedings of the eighth ACM international conference on web search and data mining*, 295–304. ACM.