Front. Comput. Sci., 2020, 14(2): 430–450
https://doi.org/10.1007/s11704-018-8052-6

REVIEW ARTICLE

A survey of autoencoder-based recommender systems

Guijuan ZHANG, Yang LIU, Xiaoning JIN (✉)

Beijing Advanced Innovation Center for Future Internet Technology, Beijing University of Technology, Beijing 100124, China

Abstract In the past decade, recommender systems have been widely used to provide users with personalized products and services. However, most traditional recommender systems are still facing a challenge in dealing with the huge volume, complexity, and dynamics of information. To tackle this challenge, many studies have been conducted to improve recommender system by integrating deep learning techniques. As an unsupervised deep learning method, autoencoder has been widely used for its excellent performance in data dimensionality reduction, feature extraction, and data reconstruction. Meanwhile, recent researches have shown the high efficiency of autoencoder in information retrieval and recommendation tasks. Applying autoencoder on recommender systems would improve the quality of recommendations due to its better understanding of users' demands and characteristics of items. This paper reviews the recent researches on autoencoder-based recommender systems. The differences between autoencoder-based recommender systems and traditional recommender systems are presented in this paper. At last, some potential research directions of autoencoder-based recommender systems are discussed.

Keywords recommender system, autoencoder, deep learning, data mining

1 Introduction

The rapid development of Internet services and applications generates a large amount of information every day. Most users are struggling to find information relevant to their interests while using many applications with the rapid growth of information. Thus, recommender systems (RSs) become increasingly important to users. RSs provide appropriate items or services for users from a bunch of possible options [1]. Generally, the generation of recommendation lists requires user preferences, item features, user-item historical interactions and some additional sources of information about users or items (e.g., temporal and spatial data) [2]. Additional sources of information about users or items, also known as side information, which could be obtained from the users or items profiles.

Many recommendation models have been proposed during the last decade. They can be roughly classified into four categories: content-based models, collaborative filtering models, knowledge-based models and hybrid-based models. Nevertheless, these models do have their limitations in dealing with data sparsity and cold start problems. The recommendation performance drops significantly if the interactions between users and items are very sparse, which is known as data sparsity [3]. The cold start problems occur when RSs are unable to recommend for the system's new users and items [4–6]. To solve these problems, researchers have proposed some new recommendation models that exploit side information about users or items [7–12]. However, the improvement of recommended performance is not significant due to the limitations of these models in capturing users' preferences and features of items.

Autoencoder (AE) has become one of the most powerful approaches to capture main features of data. AE is a kind of neural network for unsupervised learning tasks, e.g., dimension reduction, efficient coding, and generative modeling

Received February 12, 2018; accepted November 9, 2018

E-mail: jinxn@bjut.edu.cn

[13]. AE has shown its superiority in learning latent feature representation in many application domains such as image recognition [14], computer vision [15] and speech recognition [16]. Recently, AE has reformed the recommendation architectures and brings more opportunities in reinventing user experiences to satisfy customers. In recent researches, merging other deep learning methods into the AE-based RSs has gained significant attention by overcoming the obstacles of traditional RSs and achieving high recommendation quality. In AE-based RSs, AE would help the system better understand users and items by learning the non-linear user-item relationship efficiently and encoding complex abstractions into data representations. Furthermore, AE can ease the impact of data sparsity by learning useful knowledge from abundant data sources such as contextual, textual and visual information.

The comparison between AE-based RSs and traditional RSs in four aspects is shown in Table 1. The first column of the table describes the differences between data sources. In general, traditional RSs deal with the single data source, such as rating or textual information. However, AE-based RSs also handle heterogeneous data sources including rating, audio, visual and video information [17, 18]. Compared with traditional RSs, AE-based RSs have better understanding of the users' demands and features of items, and AE-based RSs achieve higher recommendation accuracy than traditional RSs [19, 20], shown in the second column of Table 1. The comparison between AE-based RSs and traditional RSs about the adaptability in multimedia scenarios and noise handling is given in the third and fourth columns of the table respectively. AE-based RSs are more adaptable than the traditional RSs in multimedia scenarios [17, 21]. The capability of AE-based RSs to deal with noises is better than the traditional RSs [22, 23].

Table 1 Comparisons of traditional RSs and AE-based RSs

Models	Data sources	Accuracy	Adaptability in multimedia scenarios	Noise handling
Traditional RS	Single	Low	Bad	Bad
AE-based RS	Heterogeneous	Good	High	Good

This survey aims to help readers who are interested in AE-based RSs to quickly understand and step into the field. In this paper, we conduct a systematic review on AE-based RSs. Particularly, we propose a classification scheme to classify current related works and highlight the main prototypes of AE-based RSs as well as summarize the advantages and disadvantages of each. At last, we discuss future research directions in AE-based RSs. The remaining of this article is organized as follows. We introduce the preliminary knowledge and notations of this paper in Section 2. Section 3 presents our classification framework followed by the reviews of major AE-based RSs in each category. Furthermore, we conduct a qualitative analysis of published works under survey. Future research directions in AE-based RSs are discussed in Section 4. Section 5 draws a conclusion of this paper.

2 Preliminaries and terminologies

Before diving into the details of this survey, we first introduce the basic terminologies and concepts regarding RSs and AE. In this section, we first introduce RSs, then give a brief talk of AE and some of its important variants, as well as some common notations in this survey.

2.1 Recommender systems

RSs work as an information filter to solve the information overload problem. The goal of RSs is helping users find items or services that best match their personal tastes. RSs estimate users' preference for unseen items based on their past behaviors and preferences [24]. The tasks of RSs are often split into three main types: ranking prediction, rating prediction, and classification [2]. Ranking prediction generates a list of ordered items for users. Rating prediction is designed to fill the missing entries of the user-item rating matrix. Classification task aims to classify the candidate items into the correct categories for a recommendation. On the other hand, RSs can also be classified into collaborative filtering based, content-based, knowledge-based, and hybrid-based RSs [24].

1) Collaborative filtering is one of the most useful recommendation algorithms [25]. The main idea of collaborative filtering based RSs is assuming that similar users have similar interests [1, 26]. Collaborative filtering based RS recommends an item to a user by learning from user-item historical interactions, e.g., user's previous ratings and browsing history, without considering information about items [27].
2) Content-based RS is mainly based on side information of items and users' preferences, user-item historical interactions are not needed [28]. A content-based method would recommend items to a user by computing the items' similarities based on side information or similarities between users and items [29]. The side information, such as text, image, and video, all can be consid-

ered.

3) Knowledge-based RS generates recommendations based on user needs and preferences [30, 31]. Both collaborative filtering based and content-based methods assume that interests of users are stable and not changed over time. However, knowledge-based RS recommends items to a user based on the user's needs and more important it also takes probable changes of users' demands into account.

4) Hybrid-based RS combines two or more recommendation techniques to achieve a better performance while reducing defects of single RS. [32]. Hybrid-based RS is able to address common problems in RSs, such as data sparsity and cold start problems. There are three main ways to form a hybrid RS: combine several RSs, combine multiple recommendation algorithms, and combine features from different data sources as input [32].

2.2 Autoencoder and variants

In this subsection, we discuss the concepts of AE and its variants that are closely related to this survey. First, we introduce the concept, network structure and mechanism of basic AE.

AE is a kind of unsupervised neural network which appeared in the late 80's [33, 34]. AE has been considered as a powerful tool for automatically extracting nonlinear features [35]. As illustrated in Fig. 1, a basic AE consists of three layers: input layer, hidden layer and output layer. The number of neurons in each layer is n, m, n. The input layer and the hidden layer construct an encoder. The hidden layer and the output layer construct a decoder. The encoder encodes the high-dimensional input data $x = \{x_1, x_2, \ldots, x_n\}$ into a low-dimensional hidden representation $h = \{h_1, h_2, \ldots, h_m\}$ by a function f:

$$h = f(x) = s_f(Wx + b), \tag{1}$$

where s_f is an activation function. The encoder is parameterized by a $m \times n$ weight matrix W and a bias vector $b \in R^m$.

The decoder in Fig. 1 maps hidden representation h back to a reconstruction $x' = \{x'_1, x'_2, \ldots, x'_n\}$ by a function g:

$$x' = g(h) = s_g(W'h + b'), \tag{2}$$

where s_g represents the decoder's activation function. The decoder's parameters are comprised of a bias vector $b' \in R^n$ and a $n \times m$ weight matrix W'. The function s_f and s_g are usually non-linear activation function, e.g., the hyperbolic tangent function and the sigmoid function [36]. The nonlinear activation functions help AE learn more useful features than principal component analysis (PCA) do [37, 38].

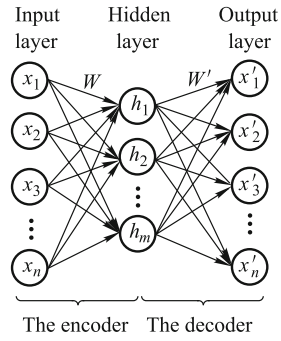

Fig. 1 The architecture of basic AE

AE is trained to minimize the reconstruction error between x and x'. There are two ways of formulating the reconstruction error: square error and cross-entropy. Their formulas are shown below:

- Square error:

$$E_{AE}(x, x') = \|x - x'\|^2. \tag{3}$$

- Cross-entropy:

$$E_{AE}(x, x') = -\sum_{i=1}^{n}(x_i \log x'_i + (1 - x_i)\log(1 - x'_i)). \tag{4}$$

A regularized term can be added to the calculation of reconstruction error to construct the lose function of AE:

$$L_{AE}(x, x') = (\sum_{x \in R^n} E_{AE}(x, x')) + \lambda \cdot \text{Regularization}. \tag{5}$$

The loss function can be optimized by stochastic gradient descent (SGD) [39] or alternative least squares (ALS) [40].

In recent years, various forms of AE have appeared in deep learning literature. Meanwhile, many variants of AE are used in RSs. Now, we briefly introduce four common variants of AE in RSs: denoising AE, stack denoising AE, marginalized denoising AE and Variational AE.

1) Denoising AE (DAE) [41]: DAE corrupts the inputs before mapping them into the hidden representation and then tries to reconstruct the original input x from its corrupted version $\tilde{x}$. The aim of DAE is to force the hidden layer to acquire more robust features and to prevent DAE from simply learning the identity function.

2) Stack denoising AE (SDAE) [42–44]: SDAE stacks several DAEs together to get higher level representations. The training is conducted greedily, e.g., layer by layer. Although SDAE has advanced performance, it still has some drawbacks. The main drawbacks of SDAE are the high computational cost of training and lack of scalability to high-dimensional features. These

drawbacks are caused by SDAE relying upon iterative and numerical optimization techniques to learn a large number of model parameters.

3) Marginalized denoising AE (MDAE) [35, 45]: In contrast to SDAE, MDAE avoids the high computational cost by marginalizing stochastic feature corruption. And, MDAE proposes a closed-form solution for learning model parameters. MDAE has a fast training speed, simple implementation and the ability to scale to large and high-dimensional data. In addition, MDAE can be stacked to form a deep architecture named marginalized stacked denoising AE (MSDAE).

4) Variational AE (VAE) [46]: VAE is an unsupervised latent variable model, which is used to learn a deep representation from high dimensional data. The basic idea of VAE is encoding the input x as a probability distribution z rather than a point encoding in conventional AE. VAE then uses a decoder network to reconstruct the original input by using samples from z. The encoder and decoder of VAE can be multi-player perceptron (MLP) [47], convolutional network (CNN) [13,48] or recurrent neural network (RNN) [48].

Assume there are M users and N items, and R represents the rating matrix. The vector $r^{(u)} = \{r^{u1}, r^{u2}, \ldots, r^{uN}\}$ denotes the observed ratings of user u, and $r^{(i)} = \{r^{1i}, r^{2i}, \ldots, r^{Mi}\}$ denotes the observed ratings of item i. M and N represent the number of users and items respectively. The rating of item i given by u is denoted by r_{ui}. $\hat{r}_{ui}$ represents the predicted rating of item i given by user u. U and V denote user latent factor and item latent factor, respectively. W and b represent the weight matrix and bias term in a neural network respectively. We summarize the aforementioned notations in Table 2.

Table 2 Notations and descriptions

Notation	Description
R	Rating matrix
$\hat{R}$	Predicted rating matrix
M	Number of users
N	Number of items
r_{ui}	Rating of item i given by user u
$\hat{r}_{ui}$	Predicted rating of item i given by user u
$r^{(i)}$	Partial observed vector for item i
$r^{(u)}$	Partial observed vector for user u
U	User latent factor
V	Item latent factor
W	Weight matrices for neural network
b	Bias terms for neural network

3 Autoencoder-based recommender systems

To give a better introduction and organization of this article, we propose a classification scheme to classify AE-based RSs. The classification scheme is shown in Fig. 2. We classify existing studies into two main categories: models rely solely on AE and integration models. Integration models can be further classified into two subcategories: integrated AE with traditional RSs and integrated AE with other deep learning techniques.

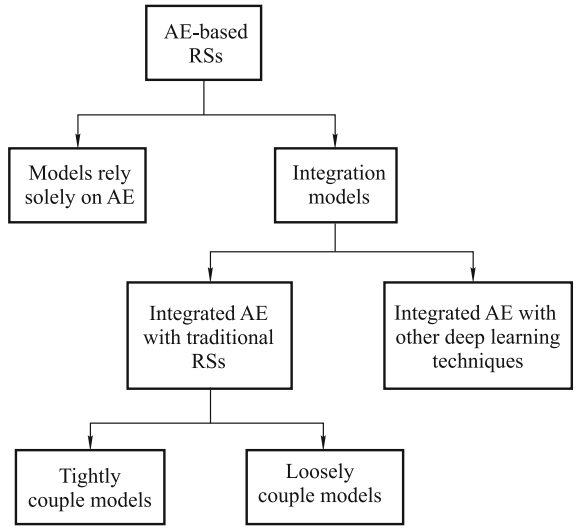

Fig. 2 The classification framework of AE-based RSs

Researches try to combine AE with traditional recommendation techniques, e.g., matrix factorization (MF) [49], probability matrix factorization (PMF) [50], factorization machine (FM) [51] and singular value decomposition (SVD) [52]. Based on how tightly the two methods are integrated, these models can be further classified into two types: loosely coupled models and tightly coupled models. In tightly coupled models, AE and traditional recommendation techniques mutually influence each other, and both are optimized simultaneously. In contrast, in loosely coupled models, AE and traditional recommendation techniques do not have interaction, and the two are optimized separately.

Some AE-based RSs try to combine AE with other deep learning techniques, e.g., CNN, RNN, deep semantic similarity model (DSSM) [53] and generative adversarial network (GAN) [54]. Combining AE with different deep learning techniques would complement each other and construct a more powerful hybrid model. For instance, integrating RNN with AE-based RSs help systems learn sequence features and improve recommendation accuracy.

Table 3 summarizes the shortlisted publications under this survey on the basis of the aforementioned classification scheme. The corresponding contents of each column in Table 3 are the articles that belong to this category in our survey. Then, we elaborate the important research prototypes in the proposed classification frameworks. We hope to identify the most significant advancements rather than provide an exhaustive list.

Table 3 Classification of shortlisted publications

Models rely solely on AE	Integration models		
	Integrate AE with traditional Rss		Integrate AE with other deep learning techniques
	Tightly coupled models	*Loosely coupled models*	
[55], [56], [57], [58], [59], [60], [61], [62], [63], [64], [65], [66], [67], [68]	[17], [21], [22], [23], [69], [70], [71], [72], [73], [74], [75], [76], [77]	[19], [20], [36], [78], [79], [80], [81], [82], [83], [84]	[18], [85], [86], [87], [88]

3.1 Models rely solely on AE

Autoencoder-based collaborative filtering (ACF) [55]. ACF is an user-specific AE-based RS. ACF does not directly take the original integer rating r_{ui} as input data. ACF first converts the r_{ui} into a vector only represented by 0 and 1 and then takes this vector as input data. For example, when $r^{(ui)}$ in the range of $[1, 5]$ and $r^{(u1)} = 1$, input of ACF needs to become $[1, 0, 0, 0, 0]$. Figure 3 presents an user-specific ACF, where integer rating range is $[1, 5]$. Each row in the left of the figure represents an item, and each square in the row corresponds to a rating. The black squares indicate that the user has rated the item as the corresponding rating. For instance, if the user gives 1 for the first item, the square corresponding to the 1 in the first row on the left of Fig. 3 is filled with black. Specific units denoted by a_j^k in the output layer represents the probability that item j will be rated value k. Therefore, the prediction rating for item j is computed by $r_q = \sum_{k=1} k \cdot a_j^k$. ACF uses a two-layer network, called Restricted Boltzmann Machine (RBM) [89,90], to pretrain model parameters to prevent local optimum. However, there are some problems of ACF:

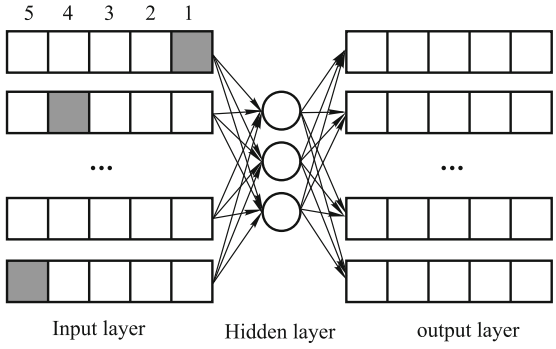

Fig. 3 An user-specific ACF [55]. The black squares indicate that the user has rated the item as the corresponding rating. e.g., the user gives 1 for item 1, 4 for item 2

(1) It fails to handle non-integer ratings.

(2) Stacking several AEs together slightly improves accuracy but increases computational overhead.

(3) The decomposition of partially observed vectors increases the sparseness of input data and reduces prediction accuracy.

AutoRec [56]. Unlike ACF, AutoRec directly takes user rating vectors $r^{(u)}$ or item rating vectors $r^{(i)}$ as input data and obtains the reconstructed rating at the output layer. There are two variants of AutoRec depending on two types of inputs: item-based AutoRec (I-AutoRec) and user-based AutoRec (U-AutoRec). Here, we only introduce I-AutoRec because I-AutoRec and U-AutoRec have the same structure. Figure 4 illustrates the structure of I-AutoRec. The input $r^{(i)} = (R_{1i}, R_{2i}, R_{3i}, \ldots, R_{mi})$ in the figure represents the ratings of item i given by users. W and V in the figure denote the weight matrix of the model. The bias of the model in the figure is 1. Only the basic AE structure is used in AutoRec. The objective function of the model is similar to the loss function of AE and can be optimized by resilient propagation (RProp) [91] or limited-memory Broyden Fletcher Goldfarb Shanno (L-BFGS) [92] algorithm. There are some important things about AutoRec that are worth mentioning:

(1) In AutoRec, RProp gives comparable performance to L-BFGS and much faster.

(2) I-AutoRec generally performs better than U-AutoRec. This is because the average number of ratings for each item is much more than those of each user.

(3) Different combinations of activation functions affect the performance of AutoRec.

(4) Increasing the number of hidden neurons or the number of layers would improve the result. It is because

that expanding the dimensionality of the hidden layer allows AutoRec to have more capacity to simulate the input features.

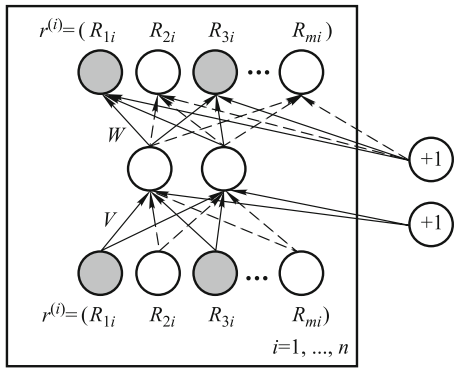

Fig. 4 I-AutoRec model [56]

Collaborative filtering neural network (CFN) [57, 58]. Unlike AutoRec, CFN is a collaborative filtering approach based on SDAE. CFN uses SDAE to make the model more robust [93]. Same as AutoRec, the input of CFN is partial observed ratings. Therefore, when CFN takes $r^{(i)}$ and $r^{(u)}$ as input respectively, there are two variants: I-CFN and U-CFN. Figure 5 presents the structure of I-CFN. To solve the cold start problem, as shown in Fig. 5, CFN integrates the side information, e.g., user profiles and item descriptions, in each layer. The input of CFN are corrupted ratings, e.g., $\tilde{r}^{(i)}$ in Fig. 5. Thus, the reconstruction of I-CFN becomes:

$$h(\{\tilde{r}^{(i)}, s_i\}) = f(W_2 \cdot \{g(W_1 \cdot \{\tilde{r}^{(i)}, s_i\} + \mu), s_i\} + b), \quad (6)$$

where s_i is side information, $\tilde{r}^{(i)}$ denotes the corrupted of original $r^{(i)}$, $\{\tilde{r}^{(i)}, s_i\}$ illustrates the concatenation of $\tilde{r}^{(i)}$ and s_i. W_1, μ and g are the weights matrix, bias vector and activation function of the encoder in I-CFN respectively. W_2, b and f are the weights matrix, bias vector and activation function of the decoder in I-CFN respectively. The objective function of I-CFN is defined as follows:

$$\begin{aligned} L = \alpha(& \sum_{i \in I(O) \cap I(\tilde{O})} [h(\{\tilde{r}^{(i)}, s_i\}) - r^{(i)}]^2) \\ & + \beta(\sum_{i \notin I(O) \cap I(\tilde{O})} [h(\{\tilde{r}^{(i)}, s_i\}) - r^{(i)}]^2) \\ & + \lambda \cdot \text{Regularizaion}. \end{aligned} \quad (7)$$

In Eq. (7), $I(O)$ and $I(\tilde{O})$ are the indices of observed and corrupted elements respectively. α and β are two hyperparameters which balance the influence of denosing the input and reconstruction the input. $h(\{\tilde{r}^{(i)}, s_i\})$ is calculated by the Eq. (6). CFN improves the prediction accuracy, accelerates the training process and enables the model to be more robust by combining side information and using SDAE.

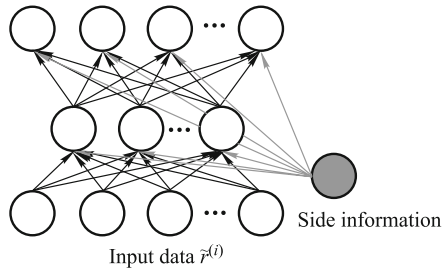

Fig. 5 The structure of I-CFN [58]

Collaborative denoising autoencoder (CDAE) [59]. CDAE is a neural network with one-hidden-layer. Comparing with previous models, CDAE has the following differences:

(1) The input of CDAE is not user-item ratings, but partial observed implicit feedback $r^{(u)}_{pref}$. If a user likes a movie, the corresponding entry value is 1, otherwise 0.

(2) Unlike earlier models that are mainly used for rating prediction, CDAE is principally used for ranking prediction.

Figure 6 shows a sample structure of CDAE. There are a total of $I + 1$ nodes in the input layer. The first I nodes represent user preferences, and each node of these I nodes corresponds to an item. The weight matrix of the first I is W_1. The last node is a user-specific node denoted by the black node in Fig. 6, which means different users have different node and associated weights. V_u in this figure represents the weight matrix of user-specific node. The bias vector of CDAE is added in the hidden layer of the model, as shown in Fig. 6. The weight matrix corresponding to the decoder is W_2. The corrupted input $\tilde{r}^u_{pref}$ of CDAE is drawn from a conditional Gaussian distribution $p(\tilde{r}^u_{pref}|r^u_{pref})$. The reconstruction of $\tilde{r}^{(u)}_{pref}$ is formulated as follows:

$$h(\tilde{r}^{(u)}_{pref}) = f(W_2 \cdot g(W_1 \cdot \tilde{r}^{(u)}_{pref} + V_u + b_1) + b_2), \quad (8)$$

where $V_u \in R^K$ is the weight matrix for the user node, and $b_1 \in R^K$ is the bias vector. b_2 is the bias vector. The parameters of CDAE are learned by minimizing the average reconstruction error, as follows:

$$\begin{aligned} \underset{W_1, W2, V, b_1, b_2}{\arg\min} \frac{1}{M} \sum_{u=1}^{M} & E_{p(\tilde{r}^{(u)}_{pref}|r^{(u)}_{pref})}[\ell(\tilde{r}^{(u)}_{pref}, h(\tilde{r}^{(u)}_{pref}))] \\ & + \lambda \cdot \text{Regularizaion}. \end{aligned} \quad (9)$$

The loss function $\ell(\cdot)$ in Eq. (9) can be square loss or logistic loss. CDAE uses the squared $\ell 2$ norm rather than Frobenius

norm to control the model complexity. SDAE applies SGD to learn model's parameters and adopts AdaGrad [94] to automatically adapt the training step size during the learning procedure. The authors of CDAE proposed a negative sampling technique to extract a small subset from items that user did not interact with for reducing the time complexity substantially without degrading the ranking quality.

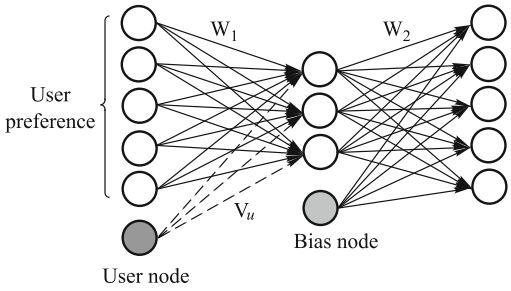

Fig. 6 The structure of CDAE [59]

Supervised neural recommendation (SNR) [60]. SNR employs stacked AE (SAE) [44] to extract the features of input and then reconstructs the input to make the recommendation. There are some important points about SNR that are worth noticing:

(1) Most of the AE-based recommendation models are unsupervised. However, SNR is supervised.

(2) SNR is not only used for rating prediction but also for classification prediction.

(3) In order to improve the recommendation performance, the side information of items or users is blended in the classification frame.

(4) To prevent the learned parameters from being oversmoothing and excessive dependent on data distribution, SNR adopts the Huber function [95] instead of the Frobenius norm to be the regularization term.

SNR has three procedures: features extraction, classification and reconstruction as shown in Fig. 7. Input is rating vector r of items. In Fig. 7, the black nodes denote the effective rating, the grey nodes in input layer represent the vacant rating, the grey nodes of the output layer during the reconstruction are predicted ratings, the grey nodes in the classification denote the results of classification. In SNR, SAE extracts the features of items, and then predicts the rating between users and items by reconstructing the features. The classification is used to extract the similarities of items to improve feature extraction. The SNR can be proposed as:

$$\min_{W} F_R(r, W_e, W_r, b_e, b_r) + \alpha \sum F_C(r, W_e, W_c, b_e, b_c) + \frac{\beta}{2}(H(W_e) + H(W_r) + H(W_c)), \tag{10}$$

where α and β are the regularization parameters to adjust the weight of each part in Eq. (10). w_e and b_e indicate the parameters in feature extraction process. w_r and b_r denote the parameters in reconstruction process. w_c and b_c are the parameters in classification process. $F_R(\cdot)$ is the cost function of the reconstruction process. $F_C(\cdot)$ is the cost function of the classification process. $H(\cdot)$ is the Huber function. SNR is optimized by gradient descent.

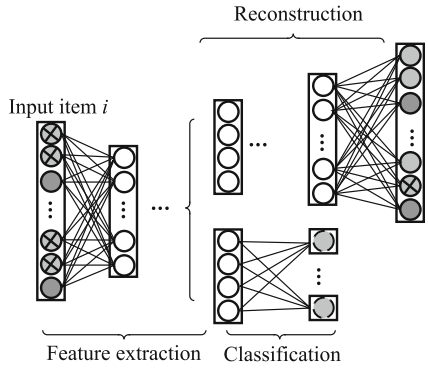

Fig. 7 The structure of SNR [60]

Trust-aware collaborative denoising autoencoder (TDAE) [61]. TDAE learns high-order correlations from rating and trust data through two DAEs for top-N recommendation. TDAE model is implemented by connecting the learned user preferences through two DAEs at a weighted layer. The weighted layer is used to balance the importance of rating and trust data. The graphical model of TDAE is illustrated in Fig. 8. The input of TDAE is corrupted by drop-out noise. As shown in Fig. 8, the input $\tilde{R}$ and $\tilde{T}$ of TDAE are corrupted version of rating and trust data. Z_u^R and Z_u^T in this figure represent the latent preferences of user u that learn from $\tilde{R}$ and $\tilde{T}$ respectively. However, the correlations between ratings and trust data are highly non-linear with different distribution [96]. It means Z_u^R and Z_u^T have higher variance. To merge Z_u^R and Z_u^T, the authors of TDAE developed a weighted hidden layer:

$$P_u = \alpha Z_u^R + (1 - \alpha) Z_u^T, \tag{11}$$

where P_u denotes the integrated user preference of user u. α is a hype-parameter to balance the effects of Z_u^R and Z_u^T. To make the model more robust, they proposed a robust correlative regularization to build the relationship between the rating and trust data in TDAE, which is given by:

$$L_C = \|Z_u^R - \theta_0 Z_u^T\|_F^2 + \|Z_u^T - \theta_1 Z_u^R\|_F^2. \tag{12}$$

In this equation, θ_0 and θ_1 denote the parameters to reconstruct data from its corresponding layer. TDAE uses SGD to train the model and sets $\ell 2$ norm as the regularization term to constrain the model complexity.

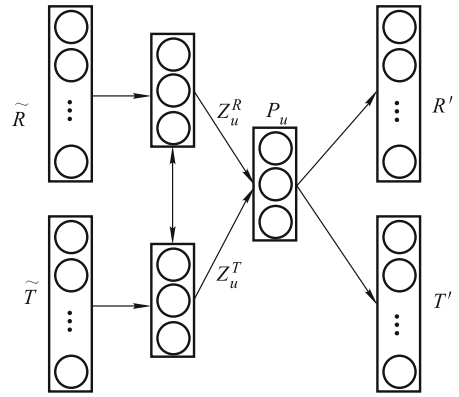

Fig. 8 Graphical model of TDAE [61]

Hybrid collaborative recommendation via semi-autoencoder (HCRSAE) [62]. In general, AE requires the dimension of input and output layer to be identical. However, Zhang et al. proposed semi-Autoencoder that the output layer shorter than the input layer, as shown in Fig. 9(a), breaking the limitations of output and input dimensionality [62]. Semi-autoencoder is applied to many areas such as extracting image features by adding captions or descriptions of images. It is convenient for semi-Autoencoder to combine side information in the input layer. HCRSAE is a recommendation model based on semi-Autoencoder. HCRSAE is usually used for ranking prediction and rating prediction depending on different types of input. Figure 9(b) shows the HCRSAE model for rating prediction. In Fig. 9(b), the black nodes in the input layer represent the rating vector, which has the same dimensions as the output layer. The grey nodes in the input layer denote the side information vector. To avoid over-fitting, HCRSAE takes $\ell 2$ norm as the regularization term and optimizes the model by SGD algorithm.

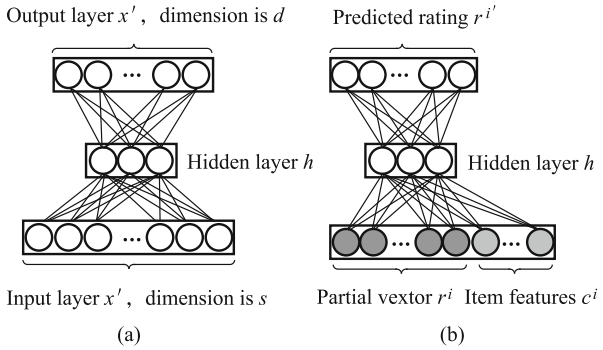

Fig. 9 Illustration of: (a) the structure of Semi-Autoencoder [62], where $h < d < s$; (b) the structure of HCRSAE for rating prediction [62]

Imputation-boosted denoising autoencoder (IDAE) [63]. IDAE model is designed for the top-N recommendation, consisting of two parts: imputing positive values and learning with imputed values. First, it infers positive user feedbacks from missing values using the basic AE to overcome the sparsity of positive data. Then, the correlation between items is learned by using the DAE from imputed values. While the existing DAE randomly corrupts the input, IDAE takes original user values as the input and the imputed values are regarded as the corrupted output. This denoising method would improve the accuracy of the top-N recommendation. Unlike previous models, IDAE takes cross-entropy to formulate the reconstruction loss. The authors adopted normalized discounted cumulative gain (NDCG) as the evaluation metric. In general, NDCG is more effective than other metrics for top-N recommendation.

Table 4 compares all aforementioned models that rely solely on AE in several aspects, e.g., recommendation tasks, input, corrupted input, side information, variants of AE, evaluation matric and data pretrain, to show the differences among them. As depicted in Table 4, ACF, AutoRec, CFN, SNR, and HCRSAE are rating predictions, CDAE, TDAE, HCRSAE, and IDAE are for ranking prediction, only SNR is for classification. TDAE adopts rating data and trust data as input, and other models take rating data as input. ACF, AutoRec, and SNR do not corrupt input, other models corrupt input data. ACF, AutoRec, and CDAE use side information in the model, and other models are not used. The most used AE variants in these models in Table 4 are DAE and SAE. The use of evaluation metrics in Table 4 are as follows: CDAE and TDAE adopt MAP, IDEA take NDCG as the evaluation metric, and RMSE for the remaining model. Only ACF and IDAE pretrain input data.

3.2 Integration models

Integration models can be classified into two subcategories: integrated AE with traditional RSs and integrated AE with other deep learning techniques. The motivation of the integration model is that different techniques complement each other and achieve a more powerful hybrid model. In this subsection, we introduce some important integration models.

3.2.1 Integrated AE with traditional RSs

Some researchers have tried to combine AE with traditional recommendation techniques to improve recommendation performance. Based on how tightly the two methods are integrated, these integration models formed by integrating

Table 4 Comparisons of recommendation models that rely solely on AE

Models	Recommendation task	Input	Corrupted input	Side information	Variants of AE	Evaluation metric	Pretrained
ACF [55]	Rating prediction	$r^{(i)}$ or $r^{(u)}$	No	No	SAE	RMSE	Yes
AutoRec [56]	Rating prediction	$r^{(i)}$ or $r^{(u)}$	No	No	AE	RMSE	No
CFN [57, 58]	Rating prediction	$r^{(i)}$ or $r^{(u)}$	Yes	Yes	SDAE	RMSE	No
CDAE [59]	Ranking prediction	$r^{(u)}$	Yes	No	DAE	MAP	No
SNR [60]	Rating prediction and Classification	$r^{(i)}$ or $r^{(u)}$	No	Yes	SAE	RMSE	No
TDAE [61]	Ranking prediction	$r^{(u)}$ and Trust data	Yes	Yes	DAE	MAP	No
HCRSAE [62]	Rating prediction or Ranking prediction	$r^{(i)}$ or $r^{(u)}$	Yes	Yes	Semi-Autoencoder	RMSE	No
IDAE [63]	Ranking prediction	$r^{(u)}$	Yes	Yes	AE and DAE	NDCG	Yes

AE with traditional recommendation techniques are further classified into loosely coupled models and tightly coupled models. MF is the most widely used traditional recommendation method in integration models. First, we give a brief introduction of MF. Then, we highlight several important research prototypes of integrating AE with traditional recommendation models within the proposed classification framework.

• Matrix factorization

MF is one of the most widely used methods in traditional RSs and has shown its advantages in the Netflix contest [97]. MF factorizes a user-item original rating matrix R into two low-rank matrices U and V. U and V consist of the user and item latent factor vectors respectively. And then MF utilizes the factorized matrices $U$and V to make further predictions, such that $R \approx UV$ [49, 98]. Non-negative matrix factorization (NMF) has previously shown to be a useful decomposition for multivariate data and is used in RSs to factorize the rating matrix into user and item profile [99, 100]. Paterek proposed an improved method of regularized singular value decomposition to predict users' preferences for items [101]. Another classical MF method is probabilistic matrix factorization (PMF) [50]. PMF adopts a probabilistic linear model with Gaussian observation noise to learn users and items latent feature representations from large and sparse rating data. Many other MF models have been proposed to enhance the performance of PMF by designing the Bayesian versions [102–105]. When side information is available, some MF models have been developed by incorporating side information. Various models show that side information, as a useful information prior, significantly improves recommendation performance [106–115].

• Tightly coupled models

Deep collaborative filtering (DCF) [22]. DCF is a general deep architecture for collaborative filtering by integrating PMF with MSDAE. DCF is a hybrid model that makes use of both rating matrix and side information and bridges MF and feature learning together. Figure 10 illustrates the DCF framework. The inputs of DCF are user-item rating R, user feature set X and item feature set Y, as shown in Fig. 10. DCF has two key components:

(1) Factorizing the rating matrix R into user latent features matrices U and item latent features V through matrix factorization.

(2) Using MSDAE extracts user contextual features and item contextual features from X and Y, and then connects the extracted user contextual features and items contextual features with U and V

DCF decomposes R and learns latent factors from side information through the following formulation:

$$\begin{aligned}\arg\min_{U,V} l(R,U,V) + \beta(\|U\|_F^2 + \|V\|_F^2) \\ +\gamma L(X,U) + \delta L(Y,V),\end{aligned} \tag{13}$$

where β, γ and δ are the trade-off parameters. $l(\cdot)$ is the loss of collaborative filtering model. $L(X, U)$ and $L(Y, V)$ represent the loss of connecting user and item contextual features with latent features respectively.

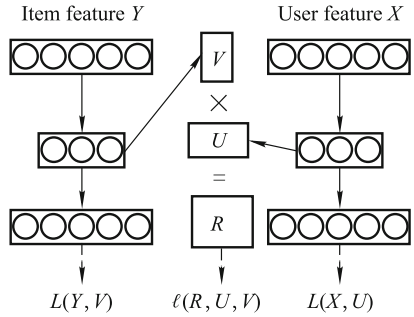

Fig. 10 Illustration of DCF framework [22]

Recommendation with social relationships via deep learning (RSRDL) [23]. Like DCF, RSRDL is a deep learning architecture based on MSDAE and MF. However, there are two key differences from DCF:

(1) RSRDL takes the user-item matrix X as input. X_{ij} denotes the number of times that user i has interacted with item j via implicit feedback, e.g., number of views, clicks, or explicit feedback, e.g., the rating that user i has given to item j.

(2) RSRDL believes that social relationships have an impact on recommendations. Therefore, a social relationships matrix A is added to the RSRDL. $A_{ij} = 1$ denotes user i and j are friends and 0 otherwise. RSRDL uses MSDAE to learn the latent representation of social relationships. Incorporating social relationships into RSRDL would reduce the impact of data sparseness and improve the accuracy of recommendations.

RSRDL is implemented by developing a joint objective function that enforces the latent representation of social relationships to be as close as possible to user latent factor U factorized from the user-item matrix X. The objective function is defined as follows:

$$\min_{U,V,W} L = \|X - UV^T\|_F^2 + \|\bar{A} - W\tilde{A}\|_F^2 + \|A^T UU^T A - WA\|_F^2 + \lambda(\|U\|_F^2 + \|V\|_F^2), \quad (14)$$

where W is the weight matrix in the hidden layer of MSDAE, $\tilde{A}$ represents the corrupted version of A. $\bar{A}$ is the c-times repeated version of A. The objective function is optimized by an alternating optimization algorithm or gradient descent.

Relational stacked denoising autoencoder (RSDAE) [69]. RSDAE adopts SDAE and PMF to improve the tag recommendation performance significantly. The authors of RSDAE developed a probabilistic SDAE to satisfy the requirement for relational deep learning. RSDAE can simultaneously learn the feature representation from the content information and the relation between items. The relational latent matrix is drawn from the items' relation data using a matrix variate normal distribution [116]. In the middle layer of SDAE, RSDAE draws the representation vector of an item from the product of two Gaussians (PoG) [117]. Meanwhile, RSDAE can be naturally extended to handle multi-relation data because of its probabilistic nature. And RSDAE is sufficient to adapt other deep learning models like RNN as well.

Collaborative deep learning (CDL) [70]. Like RSDAE, CDL also uses SDAE and PMF to build a hierarchical model. To seamlessly integrate deep learning and recommendation models, a general Bayesian deep learning framework was proposed in [118], consisting of two tightly components: a perception component (deep neural network) and a task-specific component. In CDL, the Bayesian SDAE is the perception component and PMF acts as the task-specific component. The tight combination enables CDL to seamlessly integrate deep representation learning for content information and collaborative filtering for the rating matrix. CDL handles both the sparse rating matrix and sparse text information, and learn a more effective latent representation for each item and each user. Figure 11 shows the graphical model of CDL, $X_{L/2}$ in this figure represents the middle layer of SDAE. Using the Bayesian SDAE as a component, the generative process of CDL is as follows:

1) For each layer l of the SDAE, as depicted in the dashed rectangle on the left side of Fig. 11:
   a) For each column n of the weight matrix W_l, draw $W_{l,*n} \sim N(0, \lambda_W^{-1} I_{Dl})$.
   b) Draw the bias vector $b_l \sim N(0, \lambda_W^{-1} I_{Dl})$.
   c) For each row i of X_l , draw $X_{l,i*} \sim N(\sigma(X_{l-1,i*} W_l + b_l), \lambda_s^{-1} I_{Dl})$.
2) For each item i, as depicted in the rectangle labeled J in Fig. 11:
   a) Draw a clean input $X_{c,i*} \sim N(X_{l,i*}, \lambda_n^{-1} I_{I_l})$.
   b) Draw a latent item offset vector $\varepsilon_i \sim N(0, \lambda_v^{-1} I_D)$ and then set the latent item vector to be: $V_i = \varepsilon_i + X_{\frac{L}{2},i*}^T$.
3) Draw a latent user vector for each user u, $U_u \sim N(0, \lambda_u^{-1} I_D)$, as depicted in the rectangle labeled I in Fig. 11.
4) Draw a rating for each user-item pair (u,i), $r_{ui} \sim N(U_u^T V_i, C_{ui}^{-1})$.

Here, W_l and b_l are the weight matrix and biases vector for layer l. X_l represents the l layer of the neural network. $\lambda_w, \lambda_s, \lambda_n, \lambda_v, \lambda_u$ are hyper-parameters, and C_{ui} is a confidence parameter used to measure the observations [119]. The authors exploited an EM-style algorithm to learn the parameters and developed a sampling-based algorithm to avoid the local optimum.

Collaborative variational autoencoder (CVAE) [17]. CVAE is a Bayesian generative model that considering both rating and content information for a recommendation. Unlike the previous models, CVAE adopts VAE as the perception component and learns deep latent representations and implicit relationships between items and users. CVAE can be easily ex-

tended to other multimedia modalities, e.g., images, video, and not just text. CVAE model does not corrupt the input but seeks the probabilistic latent variable model of the content. CVAE model has an inference network and a generation network, as shown in Fig. 12. In the inference network, CVAE learns a latent distribution z for content in latent space instead of observation space. In the generation network, CVAE constructs the input through latent item variables. CVAE is also applicable to other deep learning models such as CNN, GAN, and RNN depending on the input data type. RVAE [21] is an extension of CVAE, but RVAE considers both links and content information for link prediction with multimedia data.

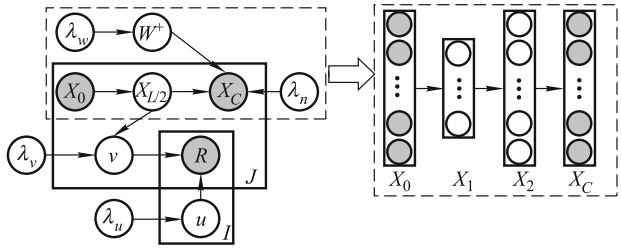

Fig. 11 On the left is the graphical model of CDL [70]. The part inside the dashed rectangle represents a 2-layer SDAE

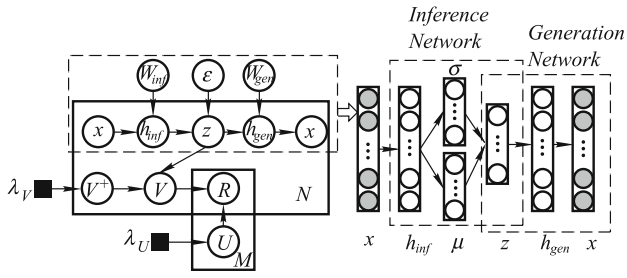

Fig. 12 On the left is the graphical model of CVAE [17]. On the right is the zoom-in of the inference network and generation network in CVAE

Collaborative deep ranking (CDR) [72]. CDR uses SDAE to extract deep feature representations from side information and then integrates them into a pair-wise ranking model to reduce the negative effects of data sparsity on top-N recommendation tasks. This class of collaborative filtering which only considers positive samples in a sample set is known as one-class collaborative filtering (OCCF) [120]. There are two kinds of existing methods for solving OCCF: point-wise and pair-wise [121]. The pair-wise methods usually achieve better performance for ranking recommendation tasks in empirical studies [59, 72, 122]. Figure 13 shows the graphics model of CDR. From the figure, we know that CDR and CDL have the same first two steps. The remaining steps of CDR are as follows:

3) For each user u, as depicted in the rectangle labeled n in Fig. 13:

   a) Draw a user factor vector $U_u \sim N(0, \lambda_u^{-1} I_D)$.

   b) For each pair-wise preference $(j, k) \in P_i$, where $P_i = \{(j, k) : r_{ij} - r_{ik} > 0\}$, drew the estimator, $\delta_{ijk} \sim N(u_i^T v_j - u_i^T v_k, c_{ijk}^{-1})$.

Here, δ_{ijk} represents the paired relationship between item i and item j on the user's preference. c_{ijk}^{-1} is a confidence parameter which denotes how much user u prefers item i than item k. CDR has a same optimization process with CDL. As a general framework, CDR can incorporate with other deep learning methods, such as CNN and RNN.

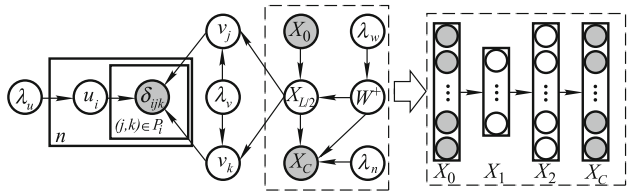

Fig. 13 The graphic model of CDR [72]

Recommendation via dual-autoencoder (ReDa) [73]. ReDa uses two AEs to learn hidden latent representations for users and items simultaneously, and minimizes the deviations of training data by the learned latent representations. The structure of ReDa is shown in Fig. 14. Two AEs simultaneously learn user's latent feature vector ξ_1 and item's latent feature vector ξ_2 from the rating matrix R_1 and R_2 respectively. Here, $R_1 = R$ and $R_2 = R^T$. ReDa only considers explicit feedback information between users and items, such as rating matrix and check-in matrix, ignoring the side information about users or items. When ξ_1 and ξ_2 are obtained, the rating matrix R' is calculated. ReDa hopes to minimize the deviations between original data R and R', the objective function of ReDa is defined as follows:

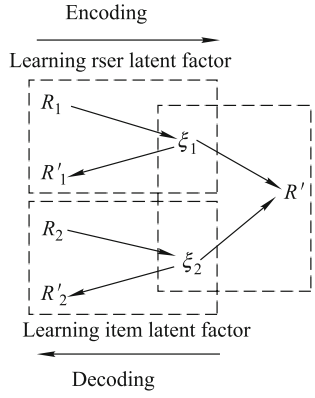

Fig. 14 The graphic model of ReDa [73] (R is a rating matrix, $R_1 = R, R_2 = R^T$)

$$L = L_c + \alpha \cdot L_b + \beta \cdot L_a + \gamma \cdot \text{Regularization}, \quad (15)$$

where L_c represents the deviations of training data, L_a and L_b are the loss function for learning users latent factors and items latent factors respectively. α, β and γ are trade-off parameters. α and β balance the influences of learning latent factors using AEs. $\ell 2$ norm is used as the regularization term. The objective function Eq. (15) is optimized by SGD. Noticing that the rating matrix should be normalized as $R = \frac{R}{r_{\max}}$ in ReDa, where $r_{\max}$ is the maximum in R.

Table 5 shows the differences among the aforementioned tightly coupled models in several aspects, e.g., recommendation task, input, corrupted input, etc. The R in the table can be user-item rating matrix, check-in matrix, and tag-item matrix. In Table 5, only DCF and ReDa make the rating prediction as the recommendation task, while other models make ranking prediction. These models in the table, except that ReDa take R as input, and other models use R and side information as input. The most used side information in the table is content information. The models in Table 5, DCF, RSRDL, RSDAE and CDL corrupt the input data. RSRDL is used in a social recommendation, RSDAE is used in tag recommendation, and CDR is used in link recommendation. As shown in Table 5, we see that in the tightly coupled models, the most used AE variants and traditional recommendation method are SDAE and MF, respectively. The commonly used evaluation metric of the tightly coupled models in the table is Recall. In this part, we find that in tightly coupled models, most AE variants are always used to learn effective latent factor representation for feeding into traditional recommendation method, and the parameters of AE and traditional recommendation method are optimized simultaneously. Traditional recommendation methods and AE mutually influence each other.

Table 5 Comparisons of tightly coupled models

Models	Recommendation task	Input	Corrupted input	Used in specific fields	Variants of AE	Traditional recommendation model	Evaluation metric
DCF [22]	Rating prediction	R, user and item feature set	Yes	No	MDAE,MSDAE	PMF	RMSE
RSRDL [23]	Ranking prediction	R, user preferences data, social relationship data	Yes	Social recommendation	MSDAE	MF	NDCG
RSDAE [69]	Ranking prediction	R, content information, relation data	Yes	Tag recommendation	SDAE	PMF	Recall
CDL [70]	Ranking prediction	R, content information	Yes	No	SDAE	PMF	Recall
CVAE [17]	Ranking prediction	R, content information	No	No	VAE	MF	Recall
RVAE [21]	Ranking prediction	R, content information, relation data	No	No	VAE	MF	Link rank and AUC
CDR [72]	Ranking prediction	R, content information and links data	No	Link recommendation	SDAE	MF	Recall
ReDa [73]	Rating prediction	R	No	No	AE	MF	RMSE

• Loosely coupled models

Tag-aware recommender systems based on deep neural networks (TARSBDNN) [78]. TARSBDNN adopts sparse AE [38] to process tag information for tag recommendation [123]. A tag recommendation allows users to label items freely using arbitrary words, namely user-defined tags [124]. User-defined tags reflect both users' preferences and estimates on items. Tag recommender system includes three parts: users, items and tags, which are usually represented as a tuple $F = (U, I, T, Y)$. U, I, T are finite sets and denote users set, items set and tags set respectively. Y represents the relationships among users, items and tags. If user u has assigned tag t to item i, $Y = y_{u,i,t} = 1$, otherwise $Y = y_{u,i,t} = 0$. Y can be decomposed into two 2-dimensional matrices: user-item matrix and user-tag matrix. To learn more interpretable features, TARSBDNN adopts sparse coding [125]. To integrate tags and items information, the authors of TARSBDNN adopted the method proposed in [126]. TARSBDNN predicts the rating of target user u to item i by:

$$S_{u,i} = \sum_{v \in N_u} sim_{u,v} \cdot (\Pi_{UI} Y)_{u,i}. \quad (16)$$

Here, N_u is a neighborhood of the target user u. $sim_{u,v}$ which indicates the similarity of user u and v under the same tag. Π_{UI} represents the user-item matrix.

Integrated recommendation models with collaborative filtering and deep learning (IRCD) [79, 80]. IRCD is a hybrid collaborative model based on SADE and timeSVD++ [127]. IRCD uses SDAE to learn item features from online items to address the cold items problem. IRCD is a time-aware

model due to timeSVD++ is able to track time-changing behaviors in the data and consider the temporal dynamics. The cold start problems can be classified into two classes: complete cold start (CCS) and incomplete cold start (ICS). Usually, the ICS problem indicates that the sparsity of item rating is higher than 85% and less than 100%, whereas the sparsity of rating for CCS items is 100% [128]. For solving the CCS problem and ICS problem, the authors of IRCD proposed two integrated recommendation models called IRCD-CCS and IRCD-ICS respectively. To predict the ratings for items, the author of IRCD proposed two methods:

- Top-of-All (ToA): in this way, the rating of item i is predicted by the top M most similar non-cold start (non-CS) items which selected from the entire non-CS item set.
- Top-of-User (ToU): in this way, the rating of item i given by user u is predicted by the top M most similar non-CS items which selected from the set of non-CS items rated by user u. ToU approach is better than ToA based on experimental results.

ToA and ToU only consider the similar non-CS, ignoring the other information in the rating matrix. The authors of IRCD put the ToU method together with timeSVD++ to process the other information for higher accuracy, as shown in Fig. 15. In Fig. 15, the left black dash rectangle is the graphical model of IRCD-CCS, the right black dash rectangle is the graphical model of IRCD-ICS, the middle part is a traditional MF model. IRCD uses SDAE to learn the item content feature θ_i from the raw item content information C. In IRCD-CCS, the ToU approach is used to obtain a predicted rating r_{ui} based on the similarity measure of item content feature. For the IRCD-ICS model the item content feature θ_i is used to learn item factor q_i.

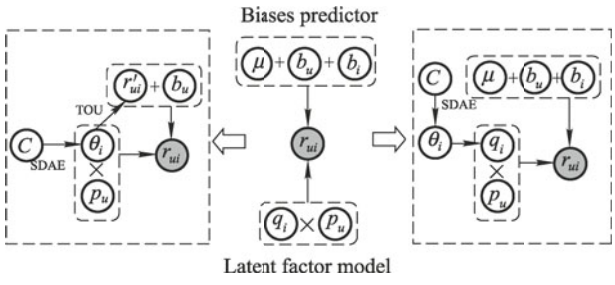

Fig. 15 The left black dash rectangle is the graphical model of IRCD-CCS; The right black dash rectangle is the graphical model of IRCD-ICS [79]

AutoSVD++ [36]. AutoSVD++ is a hybrid model by generalizing contractive AE [129] into a matrix factorization framework, allowing the model to have good scalability and computational efficiency. The learned feature representations from contractive AE are robust towards small disturbances around the training points. AutoSVD++ also models content information to achieve efficient and compact representations, and uses implicit user feedback to provide an accurate recommendation. The model parameters are learned by SGD. For reducing the training time, the authors of AutoSVD++ proposed an efficient training algorithm.

In this part, according to the aforementioned several recommendation models, we find that in the loosely couple recommendation models, AE is usually used to learn more useful feature representations. In the entire model, AE is an independent part and has no interaction with the traditional recommendation methods.

3.2.2 Integrated AE with other deep learning techniques

Collaborative knowledge based embedding (CKE) [18]. CKE integrates AE with CNN. CKE uses heterogeneous network embedding and deep learning embedding methods to extract semantic representations from structural information, textual information and visual information in the knowledge to improve the quality of RSs. The structural information contains the properties of items and the relationships among items and users. CKE applies a heterogeneous embedding method TransR [130] to find the latent representation from the structural information. For textual information, CKE adopts Bayesian SDAE to obtain latent representations. Similarly, CKE employs Bayesian stacked convolutional AE (Bayesian SCAE) to extract item entities' semantic representations from the visual knowledge, e.g., images, video. Bayesian SCAE uses convolution by replacing the fully-connected layers of SDAE with convolutional layers, making CKE more robust and efficient.

Collaborative recurrent autoencoder (CRAE) [85]. CRAE is a hierarchical Bayesian recommendation model which integrates RNN with DAE. Most of the aforementioned models lack robustness and incapable of modeling the sequences of text information. CRAE has a good ability to deal with this limitation. CRAE replaces feedforward neural layers with RNN, which enables CRAE to capture the sequential information of item content information [131, 132]. The authors of CRAE first proposed a robust recurrent network (RRN) as a type noisy gated RNN. In RRN, the gates and other latent variables are designed to incorporate noise in order to make the model more robust. The authors of CRAE designed a wildcard denoising method to avoid overfitting, and also

proposed a novel beta-pooling approach for pooling variable-length sequences into fixed-length vectors. With its Bayesian nature, CRAE can seamlessly incorporate side information to improve the accuracy of recommendations. In addition, multiple Bayesian recurrent layers can be stacked together to strengthen representation.

Variational autoencoders for collaborative filtering (VAE-CFs). Lee et al. [86] proposed a set of model-based collaborative filtering with VAE, which called a set of VAE-CFs that are varied through the dependency structures in modeling side information and implicit user feedback. To model different types of data, the authors of VAE-CFs proposed two types of CF models based on VAE: CVAE-CF and JVAE-CF. VAE-CFs have five different model structures. These five models reflect the different perspectives of modeling hypotheses listed in below.

- VAE-CF: only the distribution of users' responses is modeled.
- CVAE-CF: a conditional distribution of users' responses given side information is modeled.
- JVAE-CF: a joint distribution of users' responses and side information is modeled.
- VAE-AR: two independent distributions of users' responses and side information are modeled, and then utilize GAN [54] to merge the models.
- CLVAE: a conditional distribution of users' responses given side information is modeled by utilizing the ladder VAE [133].

The authors of VAE-CFs used negative example sampling for dealing with implicit feedback in the VAE framework. The auto-encoding of previous VAE models is combined into a single latent variable whereas the VAE-CFs have two latent variables, which lead to a richer representation.

Semantics-aware autoencoders (SEM-AUTO) [87]. SEM-AUTO uses the semantic information encoded in a knowledge-based graph (KG) to build connections between neurons in an AE. The authors used the categorical information related to items rated by users to map the AE network topology. The mapping with KG makes the number of neurons in the hidden layer of SEM-AUTO to be of variable length, and the number depends on how much categorical information is available for items rated by a specific user. Noticing that the neural network of SEM-AUTO is not fully connected and does not need bias nodes. The vectors of weights learned in SEM-AUTO are used to estimate the utility associated with items that unrated by the user, thus calculating a top-N recommendation list.

Fashion coordinates model (FCM) [88]. FCM is a hybrid fashion coordinates recommendation model that considers both user behaviors and visual fashion styles, such as the pictures of products. FCM utilizes extended latent factor model (E-LFM) [88] to deal with user behavioral features, and uses CNN-based DAE to process visual features. The final recommendation list is obtained by the combination of the recommendation of both E-LFM and CNN-based DAE. Experimental results show that FCM is more accurate than traditional methods on multi-items recommendations and is not affected by the cold start.

Table 6 shows the comparison of five recommendation models that integrate AE with other deep learning techniques. Note that, integrating AE with other deep learning for recommendation is still a relatively new direction. From the five recommendation models in Table 6, we find that integrating AE with other deep learning methods can handle different types of input, making the model more robust, and improving the quality of recommendations.

Table 6 Comparisons of five recommendation models that integrate AE with other deep learning

Models	Variants of AE	Combined techniques	Input
CKE [18]	SDAE	CNN	R, structural information, textual information, visual information
CRAE [85]	DAE	RNN	R, content information
VAE-CFs [86]	VAE	GAN	R, social relation data
SAE-AUTO [87]	AE	KG	R, categorical information
FCM [88]	DAE	CNN, E-LFM	R, visual information

3.3 Qualitative analysis

In this subsection, we conduct basic statistical analysis on experimental datasets, evaluation metrics and the use of AE and its variants in AE-based RSs in this survey. Figure 16 shows the datasets used in the autoencoder-based RSs. As shown in the figure, MovieLens (see GroupLens), CiteUlike and Netflix (see Netflix Prize) are the three most used datasets. Other datases such as Douban (collected from Douban), Yelp and Epinions are also frequently adopted.

Figure 17 presents the evaluation metrics used in the review works. Root mean square error (RMSE) and recall are the most-used evaluation metrics. As for evaluation metrics, RMSE and mean average error (MAE) are usually used for rating prediction evaluation, while recall, mean average precision (MAP) and NDCG give greater credit to correctly rec-

ommended items in top ranks. Precision and recall are widely used for classification result evaluation.

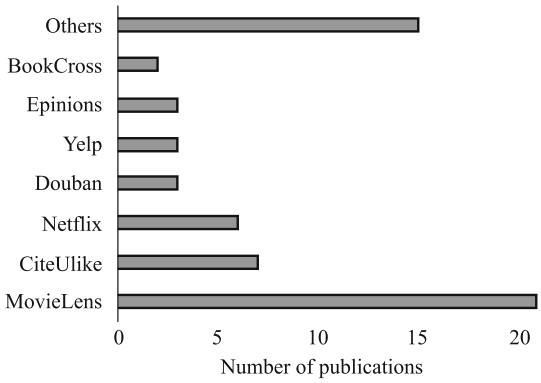

Fig. 16 Datasets in use

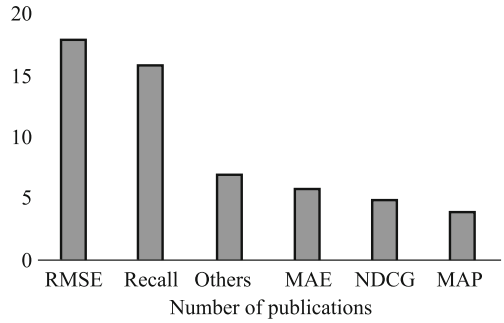

Fig. 17 Evaluation metrics in use

AE and its variants used in AE-based recommendation models are shown in Fig. 18. Basic AE, SDAE, and DAE are the most-used in the reviewed works. Another thing we would like to mention is that in order to improve and validate the performance of the model, most models often adopt a variety of datasets, evaluation metrics, and AE variants. As for recommendation tasks, rating prediction has gained the most popularity, followed by the ranking prediction, while few works take classification problems as the recommendation tasks.

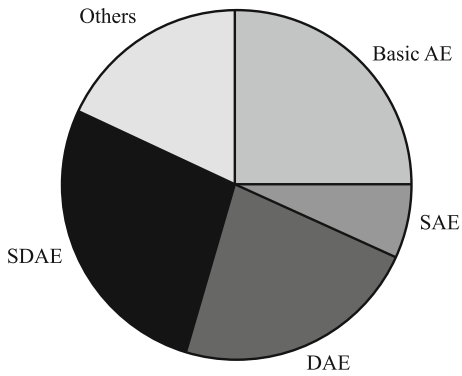

Fig. 18 Variants of AE in use

4 Future research directions

Through the above reviews on existing AE-based RSs, we find that the application of AE in RSs is still in a preliminary stage. In this section, we identify several emerging research directions in this area.

4.1 Auxiliary information

Nowadays, a critical problem in RSs is data sparsity, which means the user-item choice matrix is parsimoniously filled. This problem can be solved by exploiting abundant auxiliary information for regularization. Although existing works have investigated the efficiency of side information in a recommendation, neither they make full use of these various types of side information, nor take full advantages of the available data. Therefore, when facing different recommendation requirements, it is necessary to choose appropriate auxiliary information to help understand users and items to further improve the accuracy of recommendation. Moreover, there are few works to investigate the changes in users' interests or intentions. The changes in users' interests can be inferred from the users' footprint from social media, e.g., Facebook or WeChat posts [134] and physical world, e.g., Internet of things [135]. The capability of AE in processing heterogeneous data sources also brings more opportunities in recommending diverse items with unstructured data such as textual, visual, audio and video features.

4.2 Development of AE

Recently, many effective unsupervised learning techniques based on AE have emerged. Especially, autoencoder variants include importance weighted autoencoders [136], ladder variational autoencoders [133] and discrete variational autoencoders [137]. These variants allow AE to have stronger learning ability and better scalability. Therefore, applying these newly emerged AE variants to RSs will help improve the recommended performance.

4.3 Integration model

As we demonstrated in Subsection 3.2, integration models can model the heterogeneous features of determining factors, e.g., user, item, and context, in RSs. There are also many studies that integrate AE with traditional recommendation methods. However, only a few studies have been integrated AE with other deep learning methods. For example, AE could be combined with deep semantic similarity mod-

els to learn semantic representations of items in a common continuous semantic space and measure the semantic similarities of items. Most models that integrate AE with traditional recommendation algorithms adopt collaborative filtering methods. There are few models adopt content-based or knowledge-based methods. Therefore, the integration model is a promising but largely under-explored area where more studies are expected.

4.4 Model performance

The performance of a model is an important indicator to judge the merits of this model. Most models can improve performance by adopting some methods during their implementations. The following are some aspects help improve the performance of AE-based recommendation models.

- Multi-task learning multi-task learning is successful in many fields, e.g., computer version and natural language processing [138,139]. Among the reviewed studies, the work in [60] applied multi-task learning to an RS and achieved some improvements over single task learning. Applying multi-task learning to RSs has the following advantages: (1) learning several tasks at once can prevent overfitting by sharing the hidden representations; (2) multi-task provides an implicit data enhancement to address the sparsity problem; (3) multi-task learning can be easily deployed for cross-domain recommendations, each of which generates recommendations for the corresponding specific domain.
- Temporal dynamics user demands are not static and will change over time. Session-based RS is designed to capture the dynamic and temporal user demands. Moreover, user and item features can evolve independently or co-evolve dependently over time [140, 141]. Therefore, the evolution and co-evolution of items and users are also important aspects of temporal influence. More intensive studies on temporal dynamics for AE-based RS will be a promising research direction.
- Interpretability most RSs directly give recommendations, without giving the reasons or process for the recommendations. Suitable explanations of recommendation results can help users accept the recommendation results [142–144], as well as improve the user experiences in system transparency, credibility, effectiveness and satisfaction [145, 146]. There already have been a few studies that adopt topic models [147] or item features [148] in traditional RSs to explain recommendation. However, there is no study on explanations in AE-based RSs. Therefore, it is a direction for future research work about AE-based RSs.
- Attention mechanism attention mechanism is an intuitive but effective technique, which can be applied to deep neural networks. Attention mechanism provides a good solution for dealing with long-range dependencies and helps the network to better memorize inputs. By applying attention mechanism, AE-based RSs can filter out non-meaningful content and select the most representative items while providing good interpretability [149].
- Quick solution of the recommendation model quick solution of recommendation models is a hot topic in research. The ever-increasing volume of data in the big data era is a challenge to the quick solution of the recommendation model. Taking multimedia data sources as input or merging side information to models, while providing strong data support for the RS, also exacerbates the complexity of recommendation model. SGD is widely used in RS due to its low computational complexity and good parallelism. There are a few studies on parallelized SGD [150, 151]. Parallelized SGD can speed up the solution of recommendation model.
- Scalability scalability is critical to the actual use of recommendation models. MF is the most used method in AE-based RS because of its high scalability. However, how to develop an AE-based recommendation model with high scalability and effectively combine more auxiliary information will be one of the focuses of future research works.

4.5 Evaluation metrics

As we indicated in Subsection 3.3, most research works adopt RESM, Recall or NDCG as evaluation metrics to test the accuracy of a recommendation model. However, being accurate is far from enough for a high-quality RS in practical, and can even lead to over-specialization. Apart from accuracy, other evaluation metrics, such as diversity [66, 74], novelty, serendipity, coverage, trustworthiness, privacy, interpretability, should also be considered in RSs [152–155]. These evaluation metrics enable RSs to capture the users' unclear interests and be friendlier to users. Therefore, RSs not only perform accurate modeling but also offer a comprehensive experience to users.

5 Conclusion

In this paper, we provide a systematic survey of the most novel works to date on AE-based RSs. We propose a classification scheme for organizing and clustering existing related works. We elaborate some important research prototypes of AE-based RS and summarize their advantages and disadvantages. We also conduct a brief statistical analysis of these publications to identify the contributions and characteristics of these studies. In addition, we discuss new trends and future directions in this research field to share the prospects and expand the horizons of AE-based RSs. We hope this survey will be helpful to researchers and educators who are interested in this area.

Acknowledgements This work was supported by Beijing Advanced Innovation Center for Future Internet Technology (110000546617001).

References

1. Soghra L, Ebrahimpour-komleh H. Improving collaborative recommender systems via emotional features. In: Proceedings of the 10th IEEE International Conference on Application of Information and Communication Technologies (AICT). 2016, 1–5
2. Zhang S, Yao L, Sun A, Tay Y. Deep learning based recommender system: a survey and new perspectives. 2017, arXiv preprint arXiv:1707.07435
3. Cacheda F, Carneiro V, Fernández D, Formoso V. Comparison of collaborative filtering algorithms: limitations of current techniques and proposals for scalable, high-performance recommender systems. ACM Transactions on the Web (TWEB), 2011, 5(1): 1–33
4. Nguyen A T, Denos N, Berrut C. Improving new user recommendations with rule-based induction on cold user data. In: Proceedings of the 2007 ACM Conference on Recommender Systems. 2007, 121–128
5. Rashid A M, Albert I, Cosley D, Lam S K, McNee S W, Konstan J A, Riedl J. Getting to know you: learning new user preferences in recommender systems. In: Proceedings of the 7th ACM International Conference on Intelligent User Interfaces. 2002, 127–134
6. Ebesu T, Fang Y. Neural semantic personalized ranking for item cold-start recommendation. Information Retrieval Journal, 2017, 20(2): 109–131
7. Chow R, Jin H, Knijnenburg B, Saldamli G. Differential data analysis for recommender systems. In: Proceedings of the 7th ACM Conference on Recommender Systems. 2013, 323–326
8. Gomez-Uribe C A, Hunt N. The netflix recommender system: algorithms, business value, and innovation. ACM Transactions on Management Information Systems (TMIS), 2015, 6(4): 1–19
9. Sottocornola G, Stella F, Zanker M, Canonaco F. Towards a deep learning model for hybrid recommendation. In: Proceedings of the International Conference on Web Intelligence. 2017, 1260–1264
10. Yan S, Lin K J, Zheng X, Zhang W, Feng X. An approach for building efficient and accurate social recommender systems using individual relationship networks. IEEE Transactions on Knowledge and Data Engineering, 2017, 29(10): 2086–2099
11. Wu H, Zhang Z, Yue K, Zhang B, Zhu R. Content embedding regularized matrix factorization for recommender systems. In: Proceedings of the 2017 IEEE International Congress on Big Data (BigData Congress). 2017, 209–215
12. McAuley J, Leskovec J. Hidden factors and hidden topics: understanding rating dimensions with review text. In: Proceedings of the 7th ACM Conference on Recommender Systems. 2013, 165–172
13. Goodfellow I, Bengio Y, Courville A. Deep Learning. MA: MIT Press, 2016
14. Peng X, Li Y, Wei X, Luo J, Marphey Y L. Traffic sign recognition with transfer learning. In: Proceedings of the 2017 IEEE Symposium Series on Computational Intelligence (SSCI). 2017, 1–7
15. Dehghan A, Ortiz E G, Villegas R, Shah M. Who do I look like? Determining parent-offspring resemblance via gated autoencoders. In: Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition. 2014, 1757–1764
16. Lu X, Yu T, Matsuda S, Hori C. Speech enhancement based on deep denoising autoencoder. Interspeech, 2013, 436–440
17. Li X, She J. Collaborative variational autoencoder for recommender systems. In: Proceedings of the 23rd ACM International Conference on Knowledge Discovery and Data Mining. 2017, 305–314
18. Zhang F, Yuan N J, Lian D, Xie X, Ma W Y. Collaborative knowledge base embedding for recommender systems. In: Proceedings of the 22nd ACM International Conference on Knowledge Discovery and Data Mining. 2016, 353–362
19. Unger M. Latent context-aware recommender systems. In: Proceedings of the 9th ACM Conference on Recommender Systems. 2015, 383–386
20. Unger M, Bar A, Shapira B, Rokach L. Towards latent context-aware recommendation systems. Knowledge-Based Systems, 2016, 104: 165–178
21. Li X, She J. Relational variational autoencoder for link prediction with multimedia data. In: Proceedings of the Thematic Workshops of ACM Multimedia. 2017, 93–100
22. Li S, Kawale J, Fu Y. Deep collaborative filtering via marginalized denoising auto-encoder. In: Proceedings of the 24th ACM International Conference on Information and Knowledge Management. 2015, 811–820
23. Rafailidis D, Crestani F. Recommendation with social relationships via deep learning. In: Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval. 2017, 151–158
24. Adomavicius G, Tuzhilin A. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering, 2005, 17(6): 734–749
25. Lu J, Guo Y, Mi Z, Yang Y. Trust-enhanced matrix factorization using pagerank for recommender system. In: Proceedings of the International Conference on Computer, Information and Telecommunication Systems (CITS). 2017, 123–127
26. Linden G, Smith B, York J. Amazon. com recommendations: item-to-item collaborative filtering. IEEE Internet Computing, 2003, 7(1): 76–80
27. Resnick P, Varian H R. Recommender systems. Communications of the ACM, 1997, 40(3): 56–58
28. Mooney R J, Roy L. Content-based book recommending using learn-

ing for text categorization. In: Proceedings of the 5th ACM Conference on Digital Libraries. 2000, 195–204

29. Bhumichitr K, Channarukul S, Saejiem N, Jiamthapthaksin R, Nongpong K. Recommender systems for university elective course recommendation. In: Proceedings of the 14th International Joint Conference on Computer Science and Software Engineering (JCSSE). 2017, 1–5
30. Carrer-Neto W, Hernández-Alcaraz M L, Valencia-García R, Garcìa-Sánchez F. Social knowledge-based recommender system application to the movies domain. Expert Systems with Applications, 2012, 39(12): 10990–11000
31. Tarus J K, Niu Z, Yousif A. A hybrid knowledge-based recommender system for e-learning based on ontology and sequential pattern mining. Future Generation Computer Systems, 2017, 72: 37–48
32. Burke R. Hybrid recommender systems: survey and experiments. User Modeling and User-Adapted Interaction, 2002, 12(4): 331–370
33. Rumelhart D E, Hinton G E, Williams R J. Learning representations by back-propagating errors. Nature, 1986, 323(6088): 533
34. Baldi P, Hornik K. Neural networks and principal component analysis: learning from examples without local minima. Neural Networks, 1989, 2(1): 53–58
35. Chen M, Xu Z, Weinberger K, Sha F. Marginalized denoising autoencoders for domain adaptation. In: Proceedings of the 29th International Conference on Machine Learning. 2012, 1627–1634
36. Zhang S, Yao L, Xu X. Autosvd++: an efficient hybrid collaborative filtering model via contractive auto-encoders. 2017, arXiv preprint arXiv:1704.00551
37. Japkowicz N, Hanson S J, Gluck M A. Nonlinear autoassociation is not equivalent to PCA. Neural Computation, 2000, 12(3): 531–545
38. Hinton G E, Salakhutdinov R R. Reducing the dimensionality of data with neural networks. Science, 2006, 313(5786): 504–507
39. Bertsekas D P, Tsitsiklis J N. Gradient convergence in gradient methods with errors. Society for Industrial and Applied Mathematics Journal on Optimization, 1999, 10(3): 627–642
40. Takane Y, Young F W, Leeuw J D. Nonmetric individual differences multidimensional scaling: an alternating least squares method with optimal scaling features. Psychometrika, 1977, 42(1): 7–67
41. Vincent P, Larochelle H, Bengio Y, Manzagol P A. Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th International Conference on Machine Learning. 2008, 1096–1103
42. Hinton G E, Osindero S, Teh Y M. A fast learning algorithm for deep belief nets. Neural Computation, 2014, 18(7): 1527–1554
43. Bengio Y, Lamblin P, Popovici D, Larochelle H. Greedy layer-wise training of deep networks. In: Proceedings of the International Conference on Neural Information Processing Systems. 2007, 153–160
44. Vincent P, Larochelle H, Lajoie I, Bengio Y. Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. Journal of Machine Learning Research, 2010, 11(12): 3371–3408
45. Chen M, Weinberger K, Sha F, Bengio Y. Marginalized denoising auto-encoders for nonlinear representations. In: Proceedings of the International Conference on Machine Learning. 2014, 1476–1484
46. Kingma D P, Welling M. Auto-encoding variational bayes. In: Proceedings of the 2nd International Conference on Learning Representations (ICLR). 2013
47. Gardner M W, Dorling S R. Artificial neural networks (the multilayer perceptron)–a review of applications in the atmospheric sciences. Atmospheric Environment, 1998, 32(14-15): 2627–2636
48. LeCun Y, Bengio Y, Hinton G. Deep learning. Nature, 2015, 521(7553): 436
49. Yehuda K, Robert B, Chris V. Matrix factorization techniques for recommender systems. Computer, 2009, 42(8): 30–37
50. Mnih A, Salakhutdinov R R. Probabilistic matrix factorization. In: Proceedings of the 20th International Conference on Neural Information Processing Systems. 2007, 1257–1264
51. Rendle S. Factorization machines. In: Proceedings of the 10th International Conference on Data Mining (ICDM). 2010, 995–1000
52. Koren Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2008, 426–434
53. Huang P S, He X, Gao J, Deng L, Acero A. Learning deep structured semantic models for Web search using clickthrough data. In: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management. 2013, 2333–2338
54. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y. Generative adversarial nets. In: Proceedings of the International Conference on Neural Information Processing Systems. 2014, 2672–2680
55. Ouyang Y, Liu W, Rong W, Xiong Z. Autoencoder-based collaborative filtering. In: Processings of the International Conference on Neural Information. 2014, 284–291
56. Sedhain S, Menon A K, Sanner S, Xie L. Autorec: autoencoders meet collaborative filtering. In: Proceedings of the 24th International Conference on World Wide Web. 2015, 111–112
57. Strub F, Mary J. Collaborative filtering with stacked denoising autoencoders and sparse inputs. In: Proceedings of the NIPS Workshop on Machine Learning for eCommerce. 2015
58. Strub F, Mary J, Gaudel R. Hybrid collaborative filtering with autoencoders. 2016, arXiv preprint arXiv:1603.00806
59. Wu Y, DuBois C, Zheng A X, Ester M. Collaborative denoising autoencoders for top-n recommender systems. In: Proceedings of the 9th ACM International Conference on Web Search and Data Mining. 2016, 153–162
60. Yi B, Shen X, Zhang Z, Shu J, Liu H. Expanded autoencoder recommendation framework and its application in movie recommendation. In: Proceedings of the 10th International Conference on Software, Knowledge, Information Management & Applications (SKIMA). 2016, 298–303
61. Pan Y, He F, Yu H. Trust-aware collaborative denoising auto-encoder for top-n recommendation. 2017, arXiv preprint arXiv:1703.01760
62. Zhang S, Yao L, Xu X, Wang S, Zhu L. Hybrid collaborative recommendation via semi-autoencoder. In: Proceedings of the International Conference on Neural Information. 2017, 185–193
63. Lee J W, Lee J. IDAE: imputation-boosted denoising autoencoder for collaborative filtering. In: Proceedings of the 2017 ACM Conference on Information and Knowledge Management (CIKM). 2017, 2143–2146
64. Zhuang F, Luo D, Yuan N J. Representation learning with pair-wise constraints for collaborative ranking. In: Proceedings of the 10th ACM International Conference on Web Search and Data Mining. 2017, 567–575
65. Liang H, Baldwin T. A probabilistic rating auto-encoder for personalized recommender systems. In: Proceedings of the 24th ACM In-

ternational Conference on Information and Knowledge Management. 2015, 1863–1866

66. Suzuki Y, Ozaki T. Stacked denoising autoencoder-based deep collaborative filtering using the change of similarity. In: Proceedings of the 31st International Conference on Information Networking and Applications Workshops (WAINA). 2017, 498–502
67. Majumdar A, Jain A. Cold-start, warm-start and everything in between: an autoencoder based approach to recommendation. In: Proceedings of International Joint Conference on Neural Networks. 2017, 3656–3663
68. Krstic M, Bjelica M. Personalized program guide based on one-class classifier. IEEE Transactions on Consumer Electronics. 2016, 62(2): 175–181
69. Wang H, Shi X, Yeung D Y. Relational stacked denoising autoencoder for tag recommendation. In: Proceedings of AAAI Conference on Artificial Intelligence. 2015, 3052–3058
70. Wang H, Wang N, Yeung D Y. Collaborative deep learning for recommender systems. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2015, 1235–1244
71. Liang D, Krishnan R G, Hoffman M D, Jebara T. Variational autoencoders for collaborative filtering. In: Proceedings of the 2018 Conference on World Wide Web. 2018, 689–698
72. Ying H, Chen L, Xiong Y, Wu J. Collaborative deep ranking: a hybrid pair-wise recommendation algorithm with implicit feedback. In: Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining. 2016, 555–567
73. Zhuang F, Zhang Z, Qian M, Shi C, Xie X, He Q. Representation learning via dual-autoencoder for recommendation. Neural Networks, 2017, 90: 83–89
74. Bai B, Fan Y, Tan W, Zhang J. DLTSR: a deep learning framework for recommendation of long-tail Web services. IEEE Transactions on Services Computing, 2017, 99: 1
75. Dong X, Yu L, Wu Z, Sun Y, Yuan L, Zhang F. A hybrid collaborative filtering model with deep structure for recommender systems. In: Proceedings of the 31st AAAI Conference on Artificial Intelligence. 2017, 1309–1315
76. Nguyen T T, Lauw H W. Collaborative topic regression with denoising autoencoder for content and community co-representation. In: Proceedings of the ACM Conference on Information and Knowledge Management. 2017, 2231–2234
77. Mori K, Ito S, Harada T, Thawonmas R, Kim K. Feature extraction of gameplays for similarity calculation in gameplay recommendation. In: Proceedings of the 10th IEEE International Workshop on Computational Intelligence and Applications. 2017, 171–176
78. Zuo Y, Zeng J, Gong M, Jiao L. Tag-aware recommender systems based on deep neural networks. Neurocomputing, 2016, 204: 51–60
79. Wei J, He J, Chen K, Zhou Y, Tang Z. Collaborative filtering and deep learning based hybrid recommendation for cold start problem. In: Proceedings of the 14th IEEE International Conference on Dependable, Autonomic and Secure Computing. 2016, 874–877
80. Wei J, He J, Chen K, Zhou Y, Tang Z. Collaborative filtering and deep learning based recommendation system for cold start items. Expert Systems with Applications, 2017, 69: 29–39
81. Cao S, Yang N, Liu Z. Online news recommender based on stacked auto-encoder. In: Proceedings of the 16th IEEE/ACIS International Conference on Computer and Information Science (ICIS). 2017, 721–726
82. Niu B, Zou D, Niu Y. A stacked denoising autoencoders based collaborative approach for recommender system. In: Proceedings of the International Symposium on Parallel Architecture, Algorithm and Programming. 2017, 172–181
83. Deng S, Huang L, Xu G, Wu X, Wu Z. On deep learning for trust-aware recommendations in social networks. IEEE Transactions on Neural Networks and Learning Systems, 2017, 28(5): 1164–1177
84. Qian Y, Wai L. Review-aware answer prediction for product-related questions incorporating aspects. In: Proceedings of the 11th ACM International Conference on Web Search and Data Mining. 2018, 691–699
85. Wang H, Shi X, Yeung D Y. Collaborative recurrent autoencoder: recommend while learning to fill in the blanks. In: Proceedings of the 30th International Conference on Neural Information Processing Systems. 2016, 415–423
86. Lee W, Song K, Moon I C. Augmented variational autoencoders for collaborative filtering with auxiliary information. In: Proceedings of the ACM Conference on Information and Knowledge Management. 2017, 1139–1148
87. Bellini V, Anelli V W, Noia T D, Sciascio E D. Auto-encoding user ratings via knowledge graphs in recommendation scenarios. In: Proceedings of the 2nd Workshop on Deep Learning for Recommender Systems. 2017, 60–66
88. Gu S, Liu X, Cai L, Shen J. Fashion coordinates recommendation based on user behavior and visual clothing style. In: Proceedings of the 3rd International Conference on Communication and Information Processing. 2017, 185–189
89. Salakhutdinov R, Mnih A, Hinton G. Restricted boltzmann machines for collaborative filtering. In: Proceedings of the 24th International Conference on Machine Learning. 2007, 791–798
90. Nair V, Hinton G E. Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10). 2010, 807–814
91. Riedmiller M, Braun H. A direct adaptive method for faster backpropagation learning: the rprop algorithm. In: Proceedings of the IEEE International Conference on Neural Networks. 1993, 586–591
92. Zhang A, Wei E, Parker B B. Optimal estimation of tidal open boundary conditions using predicted tides and adjoint data assimilation technique. Continental Shelf Research, 2003, 23(11–13): 1055–1070
93. Kim M, Smaragdis P. Adaptive denoising autoencoders: a fine-tuning scheme to learn from test mixtures. In: Proceedings of the International Conference on Latent Variable Analysis and Signal Separation. 2015, 100–107
94. Duchi J, Hazan E, Singer Y. Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research, 2011, 12(7): 2121–2159
95. Huber P J. Robust estimation of a location parameter. The Annals of Mathematical Statistics, 1964, 35(1): 73–101
96. Ngiam J, Khosla A, Kim M, Nam J, Lee H, Ng A Y. Multimodal deep learning. In: Proceedings of the International Conference on Machine Learning (ICML). 2011, 689–696
97. Bennett J, Lanning S. The netflix prize. In: Proceedings of KDD Cup and Workshop. 2007, 35
98. Nathan S, Tommi J. Weighted low-rank approximations. In: Proceedings of the International Conference on Machine Learning. 2003, 720–727

99. Lee D D, Seung H S. Learning the parts of objects by non-negative matrix factorization. Nature, 1999, 401(6755): 788–791
100. Lee D D, Seung H S. Algorithms for non-negative matrix factorization. In: Proceedings of the International Conference on Neural Information Processing Systems. 2001, 556–562
101. Arkadiusz P. Improving regularized singular value decomposition for collaborative filtering. In: Proceedings of KDD Cup and Workshop. 2007, 5–8
102. Salakhutdinov R, Mnih A. Bayesian probabilistic matrix factorization using markov chain monte carlo. In: Proceedings of the 25th International Conference on Machine Learning. 2008, 880–887
103. Srebro N, Rennie J, Jaakkola T S. Maximum-margin matrix factorization. In: Proceedings of the International Conference on Neural Information Processing Systems. 2005, 37(2): 1329–1336
104. Xu M, Zhu J, Zhang B. Fast max-margin matrix factorization with data augmentation. In: Proceedings of the International Conference on Machine Learning. 2013, 978–986
105. Shi J, Wang N, Xia Y, Yeung D Y, King I, Jia J. SCMF: sparse covariance matrix factorization for collaborative filtering. In: Proceedings of the International Conference on Artificial Intelligence. 2013, 2705–2711
106. Ma H, Zhou D, Liu C, Lyu M R, King I. Recommender systems with social regularization. In: Proceedings of the 4th ACM International Conference on Web Search and Data Mining. 2011, 287–296
107. Adams R P, Dahl G E, Murray I. Incorporating side information in probabilistic matrix factorization with gaussian processes. In: Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence. 2010, 1–9
108. Zhao T, McAuley J, King I. Leveraging social connections to improve personalized ranking for collaborative filtering. In: Proceedings of the 23rd ACM International Conference on Information and Knowledge Management. 2014, 261–270
109. Porteous I, Asuncion A U, Welling M. Bayesian matrix factorization with side information and dirichlet process mixtures. In: Proceedings of the 24th AAAI Conference on Artificial Intelligence. 2010, 563–568
110. Kim Y D, Choi S. Scalable variational Bayesian matrix factorization with side information. In: Proceedings of the 17th International Conference on Artificial Intelligence and Statistics. 2014, 493–502
111. Singh A P, Gordon G J. Relational learning via collective matrix factorization. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2008, 650–658
112. Park S, Kim Y D, Choi S. Hierarchical Bayesian matrix factorization with side information. In: Proceedings of the International Joint Conference on Artifical Intelligence. 2013, 1593–1599
113. Hu L, Cao J, Xu G, Cao L, Gu Z, Zhu C. Personalized recommendation via cross-domain triadic factorization. In: Proceedings of the 22nd International Conference on World Wide Web. 2013, 595–606
114. Menon A K, Chitrapura K P, Garg S, Agarwal D, Kota N. Response prediction using collaborative filtering with hierarchies and side-information. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2011, 141–149
115. Li S, Kawale J, Fu Y. Predicting user behavior in display advertising via dynamic collective matrix factorization. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2015, 875–878
116. Gupta A K, Nagar D K. Matrix Variate Distributions. Boca Raton: CRC Press. 1999
117. Gales M J F, Airey S S. Product of gaussians for speech recognition. Computer Speech & Language, 2006, 20(1): 22–40
118. Wang H, Yeung D Y. Towards bayesian deep learning: a framework and some existing methods. IEEE Transactions on Knowledge and Data Engineering, 2016, 28(12): 3395–3408
119. Hu Y, Koren Y, Volinsky C. Collaborative filtering for implicit feedback datasets. In: Proceedings of the 8th IEEE International Conference on Data Mining. 2008, 263–272
120. Pan R, Zhou Y, Cao B, Liu N N, Lukose R, Scholz M, Yang Q. One-class collaborative filtering. In: Proceedings of the 8th IEEE International Conference on Data Mining. 2008, 502–511
121. Yao W, He J, Wang H, Zhang Y, Cao J. Collaborative topic ranking: leveraging item meta-data for sparsityreduction. In: Proceedings of the AAAI Conference on Artificial Intelligence. 2015, 374–380
122. Rendle S, Freudenthaler C, Gantner Z, Zhang Y, Cao J. BPR: Bayesian personalized ranking from implicit feedback. In: Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence. 2009, 452–461
123. Chi E H, Mytkowicz T. Understanding navigability of social tagging systems. In: Proceedings of ACM CHI Conference. 2007
124. Hotho A, Jäschke R, Schmitz C, Stumme G. Information retrieval in folksonomies: search and ranking. In: Proceedings of the European Conference on the Semantic Web: Research and Applications. 2006, 411–426
125. Lee H, Battle A, Raina R, Ng A Y. Efficient sparse coding algorithms. In: Proceedings of the International Conference on Neural Information Processing Systems. 2007, 801–808
126. Ricci F, Rokach L, Shapira B. Introduction to Recommender Systems Handbook. Springer, Boston, MA, 2011, 1–35
127. Koren Y. Collaborative filtering with temporal dynamics. Communications of the ACM, 2010, 53(4): 89–97
128. Zhang D, Hsu C H, Chen M, Chen Q, Xiong H, Uoret J. Cold-start recommendation using bi-clustering and fusion for large-scale social recommender systems. IEEE Transactions on Emerging Topics in Computing, 2014, 2(2): 239–250
129. Rifai S, Vincent P, Muller X, Glorot X, Bengio Y. Contractive auto-encoders: explicit invariance during featureextraction. In: Proceedings of the 28th International Conference on Machine Learning. 2011, 833–840
130. Lin Y, Liu Z, Sun M, Liu Y, Zhu X. Learning entity and relation embeddings for knowledge graph completion. In: Proceedings of the AAAI Conference on Artificial Intelligence. 2015, 2181–2187
131. Cho K, Van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. 2014, arXiv preprint arXiv:1406.1078
132. Hochreiter S, Jürgen S. Long short-term memory. Neural Computation, 1997, 9(8): 1735–1780
133. Sønderby C K, Raiko T, Maaløe L, Sønderby S K, Winter O. Ladder variational autoencoders. In: Proceedings of the 29th Annual Conference on Neural Information Processing Systems. 2016, 3738–3746
134. Hsieh C K, Yang L, Wei H, Naaman M, Estrin D. Immersive recommendation: news and event recommendations using personal digital traces. In: Proceedings of International Conference on World Wide

Web. 2016, 51–62

135. Yao L, Sheng Q Z, Ngu A H H, Li X. Things of interest recommendation by leveraging heterogeneous relations in the internet of things. Acm Transactions on Internet Technology, 2016, 16(2): 9
136. Burda Y, Grosse R, Salakhutdinov R. Importance weighted autoencoders. Computer Science, 2015
137. Rolfe J T. Discrete variational autoencoders. In: Proceedings of International Conference on Learning Representations. 2017
138. Collobert R, Weston J. A unified architecture for natural language processing: deep neural networks with multitask learning. In: Proceedings of International Conference on Machine Learning. 2008, 160–167
139. Deng L, Yu D. Deep learning: methods and applications. Foundations & Trends in Signal Processing, 2014, 7(3): 197–387
140. Dai H, Wang Y, Trivedi R, Song L. Recurrent coevolutionary latent feature processes for continuous-time recommendation. In: Proceedings of the Workshop on Deep Learning for Recommender Systems. 2016, 29–34
141. Wang Y, Nan D, Trivedi R, Song L. Coevolutionary latent feature processes for continuous-time user-item interactions. In: Proceedings of the 30th International Conference on Neural Information Processing Systems. 2016, 4554–4562
142. Herlocker J L, Konstan J A, Riedl J. Explaining collaborative filtering recommendations. In: Proceedings of ACM Conference on Computer Supported Cooperative Work. 2000, 241–250
143. Gedikli F, Jannach D, Ge M. How should I explain? a comparison of different explanation types for recommender systems. International Journal of Human - Computer Studies, 2014, 72(4): 367–382
144. Cramer H, Evers V, Ramlal S, Someren M V, Rutledge L, Stash N, Aroyo L. The effects of transparency on trust in and acceptance of a content-based art recommender. User Modeling and User-Adapted Interaction, 2008, 18(5): 455–496
145. Friedrich G, Zanker M. A taxonomy for generating explanations in recommender systems. AI Magazine, 2011, 32(3): 90–98
146. Sharma R, Ray S. Explanations in recommender systems: an overview. International Journal of Business Information Systems, 2016, 23(2): 248
147. Wang C, Blei D M. Collaborative topic modeling for recommending scientific articles. In: Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2011, 448–456
148. Zhao W X, Wang J, He Y, Wen J R, Chang E Y, Li X. Mining product adopter information from online reviews for improving product recommendation. ACM Transactions on Knowledge Discovery from Data, 2016, 10(3): 1–23
149. Chen J, Zhang H, He X, Nie L, Liu W, Chua T. Attentive collaborative filtering: multimedia recommendation with item- and component-level attention. In: Proceedings of International ACM SIGIR Conference on Research and Development in Information Retrieval. 2017, 335–344
150. Gemulla R, Nijkamp E, Haas P J, Sismanis Y. Large-scale matrix factorization with distributed stochastic gradient descent. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2011, 69–77
151. Zhao S Y, Li W J. Fast asynchronous parallel stochastic gradient descent: a lock-free approach with convergence guarantee. In: Proceedings of the 30th AAAI Conference on Artificial Intelligence. 2016, 2379–2385
152. Ge M, Delgado-Battenfeld C, Jannach D. Beyond accuracy: evaluating recommender systems by coverage and serendipity. In: Proceedings of ACM Conference on Recommender Systems. 2010, 257–260
153. Khan M M, Ibrahim R, Ghani I. Cross domain recommender systems: a systematic literature review. ACM Computing Surveys, 2017, 50(3): 1–34
154. Mobasher B, Burke R, Bhaumik R, Williams C. Toward trustworthy recommender systems: an analysis of attack models and algorithm robustness. ACM Transactions on Internet Technology, 2007, 7(4): 23
155. Varges S, Castells P. Rank and relevance in novelty and diversity metrics for recommender systems. In: Proceedings of ACM Conference on Recommender Systems. 2011, 109–116

Guijuan Zhang received the BS degree in computer science from Zhengzhou University, China in 2014. She is currently working toward the MS degree in Beijing Advanced Innovation Center for Future Internet Technology, Beijing University of Technology, China. Her research interests include recommender system and deep learning.

Yang Liu received the BS degree in network engineering from Tianjin University of Finance and Economics, China in 2016. He is currently working toward the MS degree in the Beijing Advanced Innovation Center for Future Internet Technology, Beijing University of Technology, China. His research interests include intelligent recommedation, data mining, and machine learning.

Xiaoning Jin received his BS degree and the PhD degree in information and signal processing with the signal detecting and processing laboratory in the Institute of Acoustics of the Chinese Academy of Sciences, China in 2011. Now he is a lecturer of Beijing University of Technology, China. His current research interests include networking technology, data science, and artificial intelligence.