Deep Learning for Recommender Systems

Shuai Zhang, Yi Tay, Lina Yao, Aixin Sun, and Ce Zhang

1 Introduction

Since the reinvigoration of neural networks, deep learning has attracted significant attention. It provides a revolutionary step to actualize artificial intelligence and has fundamentally changed the landscape of a number of fields, including computer vision, natural language processing, robotics, and more. Applications of deep learning have become ubiquitous. For example, deep neural networks can be trained to recognize objects in images, translate text between languages, recognize speech, generate images/texts, just to mention a few. Very often, state-of-the-art performances are achieved with deep neural networks.

With this striking success, a variety of deep learning designs and methods have blossomed in the context of recommender systems. It has attracted a huge interest from academia and industry, evidenced by the exponentially increase of public research works and implementations. A growing number of companies are developing and deploying deep learning based algorithms to enhance their recommender systems, so as to attract more customers, improve user satisfaction and retention rate, and as a result boost their revenue. Clearly, deep learning

S. Zhang (✉) · C. Zhang
ETH Zurich, Zurich, Switzerland
e-mail: shuai.zhang@inf.ethz.ch; ce.zhang@inf.ethz.ch

Y. Tay
Google, Mountain View, CA, USA

L. Yao
UNSW Sydney, Sydney, NSW, Australia
e-mail: lina.yao@unsw.edu.au

A. Sun
NTU, Singapore, Singapore
e-mail: axsun@ntu.edu.sg

© Springer Science+Business Media, LLC, part of Springer Nature 2022

F. Ricci et al. (eds.), *Recommender Systems Handbook*,
https://doi.org/10.1007/978-1-0716-2197-4_5

based recommendation models are now in wide use across platforms and domains. Meanwhile, recommendation with deep learning techniques has been the major focus in the research community and the number of scientific publications on this topic grows exponentially.

There are numerous benefits of using deep learning techniques to build recommender systems. Firstly, deep learning requires less feature engineering effort as it can easily process unstructured data such as text, image, sound, video, etc. Secondly, deep learning opens up an opportunity for a range of challenging recommendation tasks. For example, we can solve the cold-start problem, improve the recommendation explainability/robustness, and handle temporal dynamics effortlessly with deep neural networks. Thirdly, deep neural networks have a high degree of flexibility in the sense that multiple neural building blocks can be composed end-to-end so that building powerful models (e.g., multitask models, etc.) becomes more convenient. Just as importantly, the increasing of computational resources as well as the easy access of deep learning computation libraries make the implementation, iteration, and deployment process more handily. All in all, these benefits have led to the rapid and revolutionary development of recommender systems.

In this chapter, we do not intend to give a thorough review of all related methods, but rather to overview various key approaches and highlight the impact of deep learning techniques on the recommender systems field. This chapter is divided into three parts. We begin with the basics of deep learning techniques that are widely adopted in the recommendation area, namely, multilayered perceptrons, convolutional neural networks, recurrent neural networks, graph neural networks, deep reinforcement learning, etc. In the second part, we review how deep learning methods are applied to specific recommendation problems including interaction modeling, user modeling, content representation learning, cold-start problem, and interpretability/robustness enhancement. Thereafter, we describe how recommendations in various application domains such as e-commerce, online entertainment, news, and point-of-interests, can benefit from deep learning techniques.

2 Deep Learning for Recommender Systems: Preliminary

We begin with the basics of recommender systems and widely adopted deep learning techniques.

2.1 Basics of Recommender Systems

In a typical recommender system setting, there are a bunch of users and a catalog of items. Items can refer to movies, books, jobs, jokes, news articles, music pieces, houses, products, and even services. There usually exist some historical interactions (e.g., purchases, ratings, likes, watches, etc.) between users

and items. If there is no interaction record for some users/items, we call them cold-start users/items. Oftentimes, additional information such as contexts like timestamp, item descriptions, and user profiles are available. With these available data, we can learn a recommendation model that anticipates user's preferences and performs personalised recommendations. A good recommender system should have the ability of accurately capturing users' preferences/intentions, modeling the characteristics of items, and fitting the context better. Popular methods such as matrix factorization [64] (Chapters "Advances in Collaborative Filtering" and "Item Recommendation from Implicit Feedback"), user-based/item-based collaborative filtering [92] (Chapter "Trust Your Neighbors: A Comprehensive Survey of Neighborhood-Based Methods for Recommender Systems"), content-based approaches (Chapter "Semantics and Content-based Recommendations"), context-aware recommender systems (Chapter "Context-Aware Recommender Systems: From Foundations to Recent Developments"), cross-domain recommender systems (Chapter "Design and Evaluation of Cross-Domain Recommender Systems"), and session-based recommender systems (Chapter "Session-Based Recommender Systems") have been extensively investigated in the past decades. We refer users to the corresponding chapters for more details.

2.2 Basics of Deep Learning Techniques

Deep learning utilizes a hierarchical level of neural networks to carry out the process of machine learning. Neural network is made up of neurons which are inspired by biological neurons. A typical deep neural network consists of multiple layers and each layer has multiple neurons. Neurons between two successive layers are connected. It accumulates signals from a previous layer and the information transmission between layers is controlled with activation functions [36]. Unlike traditional linear methods, activation functions such as *ReLU*, *Sigmoid*, and *Tahn* in neural networks enable machines to process data in a nonlinear way.

At a high level, deep learning is a representation learning approach. It offers the potential to identify complex patterns and relationships hidden in the data. As such, deep learning has been widely used to learn representations of text, image, audio, video, etc. The representation learning process can operate either in a supervised manner or in an unsupervised manner. Based on the way how neurons are organized in the network, there are various neural network architectures. Major (but not exhaustive) neural networks include: multi-layer perceptrons (MLPs), convolutional neural networks (CNNs), recurrent neural networks (RNNs), autoencoders, generative adversarial networks, and graph neural networks [36, 63].

To learn parameters of deep learning models, an algorithm known as back-propagation is widely adopted [36]. Back-propagation computes the derivatives of the loss function like mean squared error loss, cross-entropy loss, or triplet loss, with respect to the weights and biases in a neural network. The computed gradients are then used by optimizers such as gradient descent, stochastic gradient descent, and

Adam to update the corresponding weights and biases. Same as traditional machine learning techniques, deep learning also suffers from the overfitting problem. To overcome this problem, regularization methods such as dropout, batch normalization, and layer normalization are usually employed. We will briefly introduce some widely used neural network architectures in the following sections.

2.2.1 Multi-Layer Perceptrons

A multilayer perceptron is composed of multiple perceptrons. It has an input layer to receive the signal, an output layer to make predictions, and an arbitrary number of hidden layers in between. Theoretically, an MLP is able to learn any mapping functions and has been proven to be a universal approximation algorithm. The predictive capability of MLPs comes from the hierarchical and multi-layered structure. It is an ideal representation learning approach as it can represent features at different scales, orders, and resolutions.

MLPs are usually used to model the correlations or dependencies between inputs and outputs. An example of MLP is shown in Fig. 1. The input layer takes the dataset as input and passes it to the next layer. The layer after the input layer is a hidden layer. It is possible to stack many hidden layers to form a very deep architecture. Deep learning refers to learning with networks that have many hidden layers. The last layer is the output layer which outputs a scalar or a vector depending on the task. For example, a regression problem has a single output neuron and there are more than one neuron in binary or multi-class classification tasks. Generally, the network outputs are converted to probabilities with *sigmoid* or *softmax* function.

2.2.2 Convolutional Neural Networks

Convolutional neural networks (CNNs) are suitable for grid topology data. For grid-like data such as images, MLPs might not scale well. For example, a small image of size 32 (width) $\times$ 32 (height) $\times$ 3 (RGB channels) has 3072 dimensions. The full connectivity will bring in a huge number of parameters and is more likely to cause overfitting. CNNs are precisely designed to circumvent this limitation. CNN can capture the locality and spatial-invariance (e.g., position of an object in an image

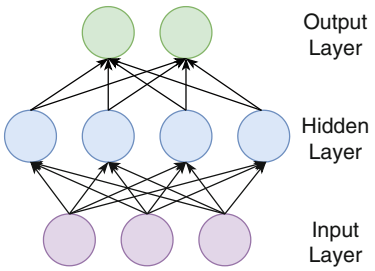

Fig. 1 Multi-layer perceptrons

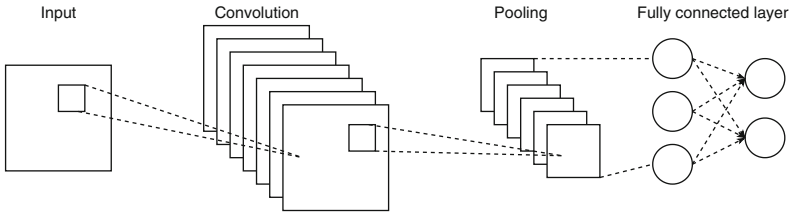

Fig. 2 Illustration of a convolutional neural network which has a convolutional layer, a pooling layer, and a fully-connected layer

does not change the class of the object) in features with merely a small number of free-parameters. CNNs can also process other types of data structure such as sequences [2].

A typical convolutional neural network consists of an input layer, several convolutional layers and pooling layers, and fully-connected layers (same as MLPs). An example is shown in Fig. 2. Convolutional layers are a major building block of CNNs. A convolution layer defines a window (or filter, kernel), by which the model can examine a subset of input (e.g., a region of an image), and subsequently scan the entire input by looking through this window. The window is small and parameterized so that it needs fewer parameters than fully connected layers. In addition, this window mechanism provides a way to look at the local regions of the input and enables a more efficient learning of local patterns. Oftentimes, we can use multiple windows to get multiple feature mappings. Pooling layer is also an important component in CNNs. Pooling operations are used to compress spatial information rather than extract certain features. It is useful when we care more about whether some features are present or not instead of where they are. Pooling is also operated with windows and various pooling operations including max pooling, average pooling, and minimum pooling are usable. Among them, max pooling is the most commonly used operation as it can return the most notable features.

Recent years have witnessed the emergence of numerous CNN architectures such as LeNet, AlexNet, VGG, Inception, ResNet, ResNeXt, and many more [36]. These networks have got very deep (up to 100 layers) and achieved huge performance boost. We omit the details for brevity.

2.2.3 Recurrent Neural Networks

Many practical problems such as time series prediction, sentence classification, machine translation, text generation, and speech recognition, require the ability of modeling sequential and temporal patterns. In these tasks, current outputs depend on previous inputs and outputs. Recurrent neural network is such a model that can deal with sequential dynamics in the data and are becoming increasingly popular.

RNNs allow a model to operate over sequences in the input, output, or both of them [36]. The most important feature of RNNs is the hidden state which remembers

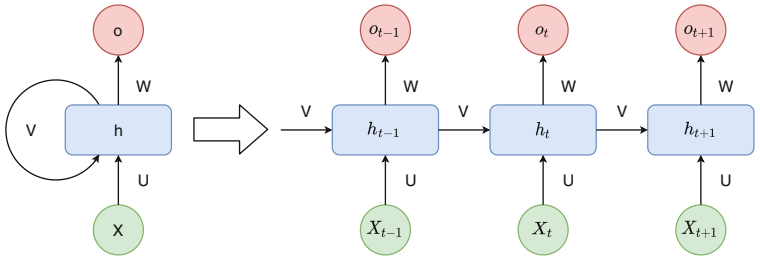

Fig. 3 Illustration of recurrent neural networks in folded (left) and unfolded (right) forms

what information has been calculated about a sequence. Figure 3 shows the structure of RNNs in both folded and unfolded forms. It has several distinct components: (1) X is the sequential input; (2) o is the output; (3) h is the hidden states; (4) V, W, U are trainable model parameters. The hidden state h is the memory of the network and is calculated based on the previous hidden state and the current input. A simple calculation is:

$$h_t = f(UX_t + Vh_{t-1}), \tag{1}$$

where f is a nonlinear activation function.

Vanilla RNNs may suffer from problems like gradient vanishing and exploding problems. They are less effective in processing very long sequences. Therefore, there are many variants of RNNs and Long short-term memory networks (LSTMs) [52] and gated recurrent units (GRUs) [22] are the most popular ones. LSTMs are proposed to address the problem of keeping or resetting context. LSTM unit is composed of a cell, an input gate, an output gate, and a forget gate. It utilizes the cell and gate mechanism to memorize or filter long-term dependencies. The sibling architecture GRUs operate in a similar fashion and can achieve the same goal. In some tasks like speech recognition and handwriting recognition, it is important to look beyond the current state, to fix the past. Knowing what is coming next can help understand the context and alleviate the ambiguity faced in the past. This gives rise to bi-directional RNNs [36].

2.2.4 Encoder and Decoder Architectures

The encoder-decoder architecture is quite common in the deep learning field. We first introduce auto-encoder (commonly written as autoencoder).

Autoencoder [111] is an unsupervised learning framework which consists of an encoder, a decoder, and a bottleneck layer (see Fig. 4). The output can be viewed as a copy of the input. It aims to reconstruct the input in the output layer by minimizing the reconstruction error measured by the differences between the input and the reconstruction. It is a widely used representation learning approach as it can compress the high-dimensional input into a low-dimensional hidden representation

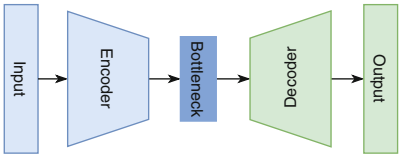

Fig. 4 Architecture of autoencoder

at the bottleneck layer, while maintaining the important characteristics. A variant of autoencoder called variational autoencoder (VAE) can generate new data. VAE encodes the input as a distribution over a latent space and this distribution is regularised to ensure continuity and completeness.

Conceptually, in an typical encoder-decoder architecture, an encoder is used to map the input to a compact hidden representation and a decoder is used to generate a high-dimensional output by up-sampling the compressed representation. It is not exclusive to autoencoder. Many sequence-to-sequence models such as transformer [109] are also structured in this way.

2.2.5 Graph Neural Networks

Graph data (e.g., social networks, chemical molecules, knowledge graph, transaction graph, etc.) is commonplace in real world applications. A graph consists of a set of nodes and a set of edges between nodes. Usually, nodes and edges are associated with some feature information for descriptive purposes.

The network architectures discussed earlier (e.g., CNNs or RNNs) cannot handle graph data effectively. As such, graph neural networks (GNNs) [63, 110] come into play. GNNs learn node/graph representations from node (edge) features and graph structures via message propagation and aggregation. The most critical process, message passing, pushes messages from surrounding nodes around a given reference node through its edges. At a single time step, each node is updated by its current presentation and the aggregation of its neighbors' representations. This step is repeated multiple times in deep GNNs, allowing information from multi-hops away to be propagated. A general update rule of GNNs is as follows:

$$h_i^{(\ell+1)} = \sigma\left(h_i^{(\ell)} W_0^{(\ell)} + \sum_{j \in \mathcal{N}_i} \frac{1}{c_{ij}} h_j^{(\ell)} W_1^{(\ell)}\right), \tag{2}$$

where $h_i^{(\ell+1)}$ is the representation of node i at layer $\ell + 1$; $\mathcal{N}_i$ is a set of neighbors of node i; c_{ij} is a fixed/trainable norm; W_*^* is a weight matrix.

2.2.6 Deep Reinforcement Learning

Deep reinforcement learning [32] is the combination of deep learning and reinforcement learning. Reinforcement learning provides a formalism for behavior

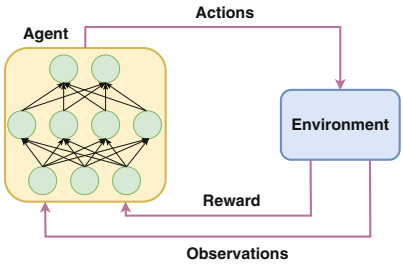

Fig. 5 The architecture of deep reinforcement learning

and allows us to solve many types of decision-making problems. Deep learning is powerful in handling unstructured data such as raw sensor or image signals. By integrating deep learning techniques, it allows reinforcement learning to solve more complex problems end-to-end. Deep neural networks enable the agent to get knowledge from raw data and derive efficient representations without handcrafted features and domain heuristics. Thus, deep reinforcement learning opens up the possibility for applications in domains such as robotics, games, healthcare, and many more (Fig. 5).

Reinforcement learning is learning through interaction with an *environment*, The *agent* learns to achieve a goal in a complex *environment*. It employs a trial and error strategy to learn from the consequence of its *actions* and selects its actions on the basis of past experiences (exploitation) and new choices (exploration). The agent receives rewards or penalties for the actions it performs and seeks to learn to select actions that maximize the accumulated reward over time.

2.2.7 Adversarial Neural Networks

We are concerned with two techniques: adversarial training and generative adversarial neural networks (GANs) [24].

An adversarial example is an instance with small, intentional feature perturbations that cause a machine learning model to make a mistake. Adversarial examples will cause model performance degradation and, even worse, be dangerous in applications like autonomous driving. Adversarial training is a straightforward way to combat with adversarial examples. It generates adversarial examples during the training process and the model is learned to combat these generated examples such that it cannot be fooled easily after training.

Generative adversarial neural networks (GANs), inspired by the minimax game in game theory, have two neural networks which compete with each other. The two networks in GANs are: generator and discriminator. The generator creates new data instances that resemble the training data, and the discriminator is trained to distinguish the generated fake data from real data. The generator and the discriminator are simultaneously trained with back-propagation. In the end, the generator can generate images that are not present in the training data.

2.2.8 Restricted Boltzmann Machines

A restricted Boltzmann machine (RBM) [31] can lean a probability distribution over its input. RBMs have found applications in dimensionality reduction, classification, topic modeling, recommendation, etc. A classical RBM can be considered as a shallow two-layer neural network which in general consists of a visible layer (also known as an input layer) and a hidden layer. The restriction indicates that there is no intra-layer communication. Same as MLPs, activation functions and bias terms can also be used. To optimize the parameters of RBMs, the algorithm contrastive divergence is usually adopted. RBMs are the building blocks of deep belief networks (DBNs) [51] where multiple hidden layers are needed.

3 Deep Learning for Recommender Systems: Algorithms

This section focuses on how various challenges (e.g., interaction/user modeling, cold-start problems, robustness, explainability, etc.) in recommender systems can be tackled with deep learning techniques.

3.1 Deep Learning for Interaction Modeling

Interaction modeling lives at the heart of recommender systems. Past interactions recorded between users and items such as ratings, purchases, likes, and views are the major source for performing collaborative filtering. As such, a good recommendation model should be able to capture the interactions/crossings between objects/features. Modeling features in their raw form will rarely provide optimal results and interaction modeling becomes necessary.

3.1.1 User-Item Interaction Modeling

Interactions between users and items form a user-item interaction matrix. This interaction matrix can be extremely sparse as many interactions are missing. Modeling user item interaction from this partially observed matrix forms the bedrock of collaborative filtering (CF) algorithms. A standard CF approach is to model user item interactions with dot product (e.g., matrix factorization). As an alternative, we can also use deep neural networks to fulfill this goal.

Formally, we adopt the following formula to define the interaction between users and items.

$$s = f(h(g_1(u), g_2(i))), \tag{3}$$

Table 1 Comparison of various user item interaction modelling approaches

Models	h	f	g_*
NeuMF [47]	Concatenation: $[U_u; V_i]$ Hadamard product: $U_u \odot V_i$	MLPs	Embedding look-up
He et al. [45]	Outer product: $U_u \otimes V_i$	CNNs	Embedding look-up
AutoRec [93], CDAE [127]	Encoder	Decoder	MLPs
DFM [132]	Dot product: $U_u \cdot V_i$	Identity mapping	MLPs
CML [53], LRML [53]	Euclidean distance: $\parallel U_u - V_i \parallel_2^2$	Identity mapping	Embedding look-up
HyperML [112]	Hyperbolic distance: $\cosh^{-1}(1 + 2\frac{\|U_u - V_i\|^2}{(1-\|U_u\|^2)(1-\|V_i\|^2)})$	Identity mapping	Embedding look-up
Yao et al. [137]	Searched	MLPs	Embedding look-up

where g_* is used to get the representation of user/item. For example, if g_* is a sparse embedding look-up function, it will output the corresponding user embedding U_u and item embedding V_i; s is the recommendation score; f can be neural networks, identity mapping, etc., and h is the interaction operation such as concatenation, Hadamard product, etc. Table 1 summarizes popular user-item interaction modeling approaches.

Using neural networks to model the interactions between users and items has aroused extensive interest. Recently, there is an increasing popularity to replace dot product with MLPs. For example, NeuMF [47] models the interaction with multilayer perceptron. It takes the concatenation of U_u and V_i or the element-wise multiplication of U_u and V_i as the input of MLPs to capture the interactions. A follow-up work [45] replaces dot product with outer product where the output is treated as an interaction map and CNNs are applied to learn high-order correlations among embedding dimensions. Despite its popularity, a recent study [88] suggests that it is non-trivial to learn the dot product with an MLP, and it might be too costly to be used in production environments.

The autoencoder architecture can also be applied to interaction modeling [93, 127]. These methods take as inputs the columns or rows of the partially observed interaction matrix, then recover the columns/rows in the output. The gradients of unobserved entries are masked out in the training phase. Nonlinear activation functions, dropout, and denoising strategies are usually used to enhance the model expressiveness and robustness. These methods can be regarded as generalizations of latent factor models. Different from conventional autoencoders, the goal of these models is to complete the interaction matrix in a column-wise (row-wise) manner instead of learning compact representations of inputs. The generated scores in

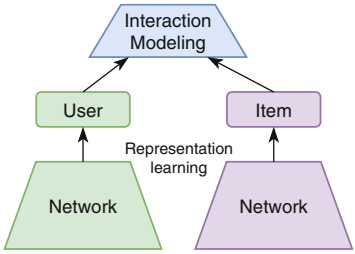

Fig. 6 Deep learning based interaction modeling for recommendation

the outputs for missing entries are viewed as the recommendation scores for the corresponding user-item pair.

As shown in Fig. 6, learning user or item representations is also an indispensable step in interaction modeling. Instead of using a simple one-hot identifier to learn user/item embeddings, an alternative way is to enrich the representation information by incorporating all historical interactions of each user/item [132, 148]. For example, we can use the rows (all items that the user rated) or columns (all users who liked the item) as the input and obtain the corresponding user/item representations; the pioneering neural network based recommendation model, RMB-CF [35, 91], aims to learn a user-specific feature vector from the observed interaction matrix (e.g., rating matrix, implicit feedback, etc.) via RBMs. The learned hidden units, in return, can help approximate the distribution of the whole user-item interaction input, thus, making predictions on unobserved data possible.

The idea of word2vec [82] can also be utilized for item representation learning. Representative models are item2vec [4], prod2vec [37], and [38]. In essence, they treat items as words and the sets of items generated from purchase logs/click sequences as sentences and a skip-gram algorithm is utilized to learn the item embeddings. For example, prod2vec learns the product embeddings by minimizing the following objective function:

$$\mathcal{L} = \sum_{s \in \mathcal{S}} \sum_{p_i \in s} \sum_{-c \leq j \leq c, j \neq 0} \log P(p_{i+j} | p_i), \tag{4}$$

where $\mathcal{S}$ is the entire "sentence" set; c defines the window size; p_{i+j} is the neighboring product of current product p_i. The probability $P(p_{i+j}|p_i)$ is obtained with a *softmax* function. Works [4, 38] take a similar form so that the details are omitted for brevity.

Another line of research is modeling interactions between users and items via distance metric learning where a shorter distance between a user and an item indicates stronger fondness. These methods aim to learn the distance $\| U_u - V_i \|_2^2$ between users and items [53]. To enhance the geometrical flexibility, Yi et al. [104] propose adding a trainable relation vector r and the distance function becomes $\| U_u + r - V_i \|_2^2$. Extending this metric learning approach to non-Euclidean space for recommender systems also shows promising performance. An ideal option is the hyperbolic space where hierarchical structures and exponentially expansion proper-

ties can be easily captured. Representative methods such as HyperML [112, 145] have shown promising results. We review these methods because most of these methods are constructed and optimized within standard deep learning operators and they can also be integrated into other deep learning architectures.

However, due to the complex nature of interactions in real-world applications, it is difficult for a specific interaction function to have consistently good performance across different application scenarios. As such, Yao et al. [137] propose using automated machine learning (AutoML) to search for the neural interaction functions for different scenarios. The search space includes inner product, Euclidean distance, outer product, concatenation, maximum, minimum, etc. The proposed method can train the search architecture and the recommendation model simultaneously in an end-to-end manner.

3.1.2 Feature Interaction Modeling

Apart from the user-item interaction matrix, there are abundant side information/feature[1] available that can be predictive for making recommendations. Nevertheless, using features in their raw form, in general, will not lead to optimal solutions. Two features combined via some operations could be more predictive than the same two features used independently. However, exhaustive manual search for feature interaction is infeasible in real world applications. The capability of learning feature interactions automatically is of great importance to recommendation models.

A popular feature interaction modeling approach is factorization machines which capture the pairwise interactions in linear time complexity. However, modeling higher-order feature interaction with factorization machines can be costly. Recently, there is a growing interest in applying deep neural networks to model feature interaction. For instance, Cheng et al. [19] propose a wide and deep framework to combine MLPs with a linear model (shown in Fig. 7a). In this framework, the deep part captures feature combinations and has good generalization for unseen feature combinations, and the wide part keeps good memorization for feature co-occurrence or correlation. The recommendation score is defined as:

$$s = \sigma(f_{LM}(x_{wide}) + f_{MLP}(x_{deep})) \tag{5}$$

where f_{LM} and f_{MLP} represent a linear model and multilayered perceptrons respectively. The input feature x is manually separated into x_{wide} and x_{deep}.

Obviously, the wide and deep network requires expert knowledge in splitting features into two parts. As such, Guo et al. [40] propose an improved deep and wide architecture, DeepFM (shown in Fig. 7b). In this model, the deep part and the wide have shared input features. The deep part is an MLP utilized to capture the

[1] In this section, we mainly refer to categorical features.

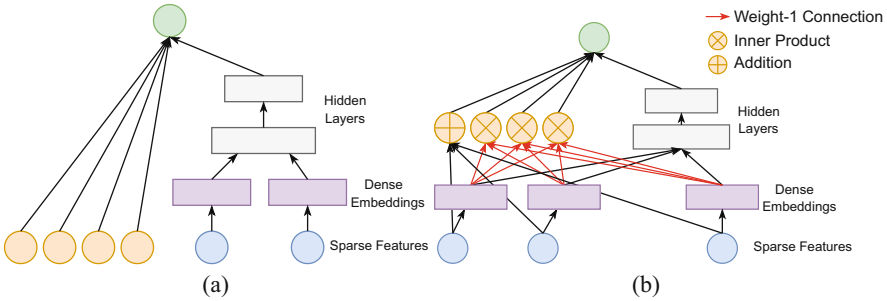

Fig. 7 Architectures for feature interaction modeling: (**a**) Wide and Deep Model [19]; (**b**) DeepFM [40]

high-level and nonlinear feature interactions and the wide part is an FM. DeepFM can capture both high-level and lower-order feature interactions. A concurrent work NFM [44] also has a similar design as DeepFM. The scoring function of DeepFM is defined as:

$$s = \sigma(f_{FM}(x) + f_{MLP}(x)) \tag{6}$$

where f_{FM} represents factorization machines. Clearly, the two components share the same input.

It is noted that MLPs can only model interactions implicitly which is not necessarily effective for all types of cross features. To circumvent this limitation, Wang et al. [118] propose a deep and cross network (CrossNet) to explicitly apply feature crossings across layers. This model requires no manual feature engineering and will not incur much additional computational cost. The downside of CrossNet is that the interactions come in a bit-wise fashion. As such, Lian et al. [74] design a generic version of CrossNet to enable vector-wise feature interaction. The proposed model, xdeepfm [74], is a combination of a generalized CrossNet and MLPs.

Evidently, not all feature interactions are created equally. Inappropriate feature interactions might be useless and even harmful. To better distinguish the importance of different feature combinations, Xiao et al. [129] propose an attention based deep factorization recommendation algorithm (AFM). This model utilizes a parameterized attention network to get an attention score for each pair of interactions in FM. The output is an attentive summation of all possible interactions. A similar attention based FM model, A3NCF [20], is proposed for the rating prediction task. Essentially, FM model adopts inner product to model the pairwise interaction. Same as the methods used to improve user item interactions, we can replace the interaction function with a few alternatives. For example, HFM [107] uses circular convolution operations or circular correlation operations to replace the inner product in FM. LorentzFM [130] substitutes the inner product with Lorentzian distance and generalizes FM to non-Euclidean space. It is worth noting that AFM, A3NCF, HFM, and LorentzFM lose the efficiency advantage of FM and have quadratic training/inference complexity.

3.2 Deep Learning for User Modeling

To provide personalised recommendations to users, a key step is to accurately infer users' demands, intentions, and interests from their profiles and past behaviors. However, the highly dynamic interaction data and extremely entangled/complex user interests hinder the use of traditional machine learning methods. We scratched the surface of user representation learning in the previous section. Here, we will dive deeper into more specific challenges on this topic.

3.2.1 Temporal Dynamics Modeling

Many traditional recommendation algorithms ignore the chronological order of user historical interactions. Nonetheless, the user item interactions are essentially sequential. User's short-term interests have huge impact on her decisions. Time context (e.g., holiday, black Friday, etc.) also affects user behaviors. Moreover, items' popularity are dynamic rather than static over time [59]. The temporal dynamics call for sequence-aware recommender systems. Learning preference representation from the sequence of actions becomes the fundamental task in sequence-aware recommendation. The preference learning process is shown in Fig. 8.

To tackle sequential patterns, popular sequence modeling approaches such as RNN come into play. Wu et al. [124] propose recurrent recommender networks (RRN) which uses LSTMs to capture the temporal dependencies for both users and items. In RRN, the state evolution for a user depends on which items she liked previously. An item's state is dependent on the users who liked it in the past. At last, short-term and long-term impacts (e.g., long-term preference of a user, fixed properties of an item) are integrated for final prediction. Using standard LSTMs or GRUs for temporal dynamics modeling in recommender systems might not be the optimal solution since users and items are usually modeled separately. As such, Donkers et al. [27] devise a novel gated GRU cell for personalized next item recommendations. The proposed cell integrates user characteristics into the recommendation model, which is beneficial to the network's predictive power. Thenceforth, customizing the gated cell in recurrent neural networks has gain growing interest. For instance, Bharadhwaj et al. [6] propose a customized gated recurrent unit to capture latent features of users and items. Guo et al. [41] propose

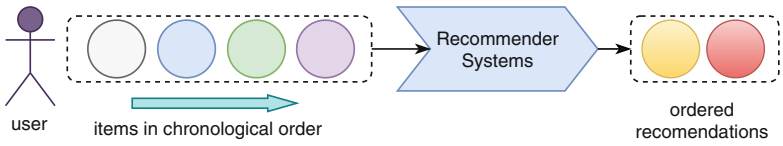

Fig. 8 Sequence aware recommender systems for temporal dynamics modeling

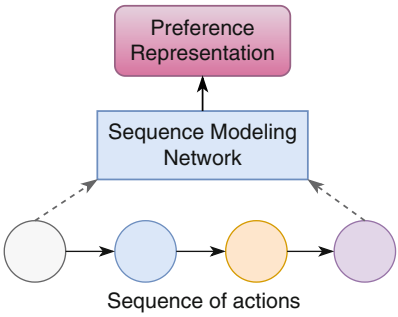

Fig. 9 Sequence modeling for preference learning

PcGRU, an improved GRU, to model the preference drifts and dynamics of user preferences (Fig. 9).

Convolutional neural networks have demonstrated superior performance in a diverse set of sequence modeling tasks such as polyphonic music modeling, character-level language modeling, and word-level language modeling, over canonical recurrent neural networks [2]. CNNs based recommendation models also show promising results in user representation learning from historical sequences. Tang et al. [103] present a convolutional sequence embedding learning approach (Caser) for sequential recommendation. Caser consists of a vertical convolutional network and a horizontal convolutional network to capture both union-level patterns and point-level sequential patterns. You et al. [139] propose a hierarchical temporal convolutional networks (HierTCN) for sequential aware recommendation. HierTCN is composed of a GRU to model the long-term interests across sessions, and a temporal convolutional network [2] to learn dynamic user embeddings. Yan et al. [133] propose a 2-D convolutional network (CosRec) to capture complex item correlations in user's historical actions. Very often, CNNs based sequential learning architectures are more efficient than RNNs based models.

Self-attention makes huge performance boost in the natural language processing field and is gradually becoming a key component in sequence models [109]. This mechanism can also be applied to address the task of sequential recommendation. Atrank [156] takes the lead in using self-attention network to capture temporal dynamics in user behaviors and achieves encouraging results. Kang et al. [61] use multi-head self-attention networks to model user's dynamic interests. The adopted framework is the same as the transformer model [109]. Zhang et al. [146] propose integrating self-attention module into metric learning for sequence dependencies modeling. Huang et al. [58] adopt a self-attention network to model the heterogeneous user behaviors, including diversity of actions and multi-modal property of content. Zhang et al. [150] combine vanilla attention with self-attention to capture feature importance and item transition patterns. The original self-attention model does not consider the time span even with positional encoding. Nonetheless, the absolute time span also matters. Intuitively, if a user has no activities for a relative long time, her last action might have less impact on her current decision. Thus, Xu et al. [131] propose a time kernel to learn functional time representations in a self-

attention model. The improved model obtains promising results on sequence-aware recommendation tasks. Another limitation of self-attention is that it can only model left-to-right unidirectional patterns, which might lead to sub-optimal performance. Inspired by the recent success of BERT [26], Sun et al. [99] adopt a BERT-like model which uses a deep bidirectional self-attention module to fuse both left-to-right and right-to-left dependencies in behavior sequences.

There are many other methods for temporal dynamics modeling. For example, Chen et al. [79] propose a hierarchical gating networks to discriminate item importance based on users' preferences via gating mechanism. Chen et al. [16] design a memory-augmented network based sequential recommendation algorithm. It utilizes an external memory matrix to store, access, and manipulate users' historical records in a more explicit fashion. Tang et al. [101] propose a mixture model which combines MLPs, RNNs, CNNs, and self-attention to capture tiny, small, and long range sequential dependencies.

In the sequence-aware recommendation task, users' identifier may be missing in some scenarios. For example, some websites/applications do not require user registration or login. In this case, the system can only record the activities in the current active session. Making recommendations in this scenario is known as session-based recommendations. GRU4Rec [49] is the pioneering work that solves this task with neural networks. Specifically, it adopts GRUs to model the sequential patterns in the sessions and predict the next event. In their extension work, the authors design a new ranking loss that is tailored to RNNs in session-based recommendation settings [48], and propose a parallel RNNs architecture to model both the click sessions and the features of the clicked items [50]. Li et al. [71] present an attention based model to capture users' intentions in the current session. Tuan et al. [108] propose a content-based session recommendation approach with CNNs and a three-dimensional CNN is used to learn representations from item textual descriptions and categorical features.

3.2.2 Diverse Interest Modeling

Users might have a diverse set of preferences and be interested in certain aspects of items. For example, a user might like history documentaries and romantic comedies for different reasons. Yet, common modeling approaches usually force all these interests to be encoded into one latent factor. It is critical to enable models to be aware of the diversity in user interests and to have the capability of distinguishing them. To reach the goal, Li et al. [70] present a multi-interest learning approach with dynamic routing. Specifically, to achieve richer user representations, the dynamic routing method disentangles user's interests from her historical interactions. The dynamic routing component acts as a soft clustering approach which groups user's historical behaviors into several clusters, with each cluster corresponding to a particular interest. Similar idea is presented in [8]. The difference is that this work adopts a dynamic routing mechanism from Capsnet [90]. Multiple interests is also investigated in [149] where quaternions are used to represent users. Each

component of the quaternion (three imagery components and one real component) represents one aspect of user's interests. In their follow-up work [144], multiple hypercuboids (i.e., concentric hypercuboids, multiple independent hypercuboids) are utilized to represent user interests. The benefits of using hypercuboids is that they can naturally embed the range of preference such as the preferred price range, enabling greater expressiveness. To disentangle the complex user interests, Ma et al. [80] use disentangled variational autoencoder to infer both macro and micro disentanglements for user interests. Macro disentanglement refers to high-level intentions (e.g., buy a book or smartphone) while micro disentanglement reflects low-level factors (e.g., color, size, etc.). Learning disentanglement representations makes the recommendation lists controllable and enriches the interpretability of the learned representations.

3.3 Deep Learning for Content Representation Learning

Features (e.g., text, image, sound, video, etc.) associated with items or users can be predictive for recommender systems. Unsurprisingly, these features play a pivotal role in conventional content-based recommender systems. Nonetheless, processing these content and mapping them into latent factors are non-trivial with traditional feature extraction methods. The representation learning capability of deep learning makes it easy to integrate these raw features into modern recommendation models.

3.3.1 Textual Feature Extraction

Text data is a rich source of information. Text can be collected from different places such as user reviews, news content, social media, and many more. In recommender systems, text data can be leveraged to better understand items and users, to alleviate the sparsity and even to address the cold-start problem. Processing text data and extracting useful representations can be challenging due to its unstructured nature. Recently, deep learning is becoming the mainstream option for various text processing tasks. Deep learning cannot only process unstructured data easily, but also has the ability to uncover hidden patterns in text data.

Textual description of items (e.g., abstract of an academic paper, plot summary of a movie, news content, tweets) is one of the most used text data in recommender systems.[2] To make use of the textual descriptions, Wang et al. [114] propose a framework, CDL (short for collaborative deep learning), to fuse stacked autoencoder with Bayesian probabilistic matrix factorization (BPMF). An EM-style optimization algorithm is devised to alternatively update the parameters of autoencoder and

[2] Since there are a large body of related work on news recommendation, we refer readers to Sect. 4 for more details.

BPMF. Nonetheless, CDL uses bag-of-words representation as the input and ignores the contextual information such as surrounding words and word orders. To overcome the limitations, Kim et al. [62] replaces the autoencoder of CDL with convolutional neural networks. Pre-trained word embeddings (e.g., Glove embeddings) are used for better semantics and expressiveness. Bansal et al. [3] present an end-to-end collaborative filtering model which leverages GRUs to encode the text associated with items into latent vectors for both warm-start and cold-start recommendations. Tan et al. [100] present a quote recommendation method that uses LSTMs to learn the distributed representations for quotes. Lee et al. [67] design a quotes recommender system which combines CNNs and RNNs for quotes process.

Reviews-based recommendation is another typical text-intensive recommendation scenario. On many e-commerce platforms, writing reviews is a strongly encouraged act. The rich semantic information in text reviews cannot be conveyed via the implicit interaction data. For example, users explain the reasons behind their ratings and their additional opinions in reviews. Reviews also provide a wealth of knowledge for prospective customers. In recent years, there has been a widespread interest in using deep learning to exploit reviews for better recommendations. Zheng et al. [155] present a deep cooperative neural network for jointly modeling of users and items reviews. Each user is represented by all reviews she has written and an item is represented by all reviews it received. Specifically, reviews for a single user (item) are concatenated and processed by convolutional networks to get the user(item) representation. To further improve model interpretability, Seo et al. [94] propose adding an attention layer before the convolutional layer. The attention layer contains a local attention window to select informative keywords and a global attention window to filter out irrelevant and noisy words. Yet, the former paradigm of simple concatenation of reviews is unnatural. For example, when deciding if a book should be recommended or not, the user's historical reviews on other books are highly relevant, while her reviews on clothes can be ignored. To this end, Tay et al. [105] present a multi-pointer co-attention network for review based recommendation where a pointer-based learning scheme is employed to extract important and predictive reviews from user and item reviews for subsequent user-item interaction. The pointer-based method is implemented with a gumbel-softmax based pointer mechanism that incorporates discrete vectors within differentiable neural architectures. When reviewing an item, different users may have opinions on different aspects of the item. For example, in restaurant recommendation, some users might care more about tastes while others might pay more attention to locations or ambiences. As such, Guan et al. [39] design an aspect-aware method for review-based recommendation. It extracts the aspects of reviews and uses an attention network to dynamically learn the importance of each aspect.

3.3.2 Image Feature Extraction

Images are commonplace on many e-commerce and social platforms. On platforms like Amazon, product images are what users typically scan through most intently.

Images can attract user's attention easily, hence incorporating visual features into recommender systems catches potential preferences of users.

Benefiting from the computer vision research, extracting and processing feature representations from images become easier than ever before. CNNs are the most popular tool for image processing. A number of works attempt to incorporate image features into recommender systems. He et al. [43] incorporate visual features into the conventional Bayesian personalised ranking (BPR) framework and propose VBPR for top-n recommendations. VBPR uses a pretrained-CNN architecture for image pre-processing. Niu et al. [84] propose a neural personalised ranking model for image recommendation on Flickrnote[3] where a hybrid-CNN model is used for visual feature extraction. Lei et al. [69] present a comparative deep learning method for Flickr image recommendation. In this dual network, images are processed with an Alexnet-like CNN architecture. Geng et al. [34] use CNNs to learn visual feature representations for image-based recommendation in social networks on Pinterest. Chu et al. [21] present a visual-aware restaurant recommender system which adopts CNNs to extract visual features from restaurant environment and food images. Additionally, in fashion recommendation domain, incorporation of visual content is also a matter of prime importance. Likewise, CNNs are the top option for feature extraction. For example, McAuley et al. [81], He et al. [42], Yu et al. [140] and Liu et al. [76] all utilize CNNs to infer user's preference/aesthetic on visual styles of fashion products (e.g., clothes. accessories, shoes, etc.). Kang et al. [60] present a system that recommends the best fit products based on 'scene' images (e.g., recommending hats based on a given selfie), where ResNet is employed for image representation learning.

3.3.3 Video and Audio Feature Extraction

Deep learning has achieved tremendous success in audio/video analysis such as speech recognition and video surveillance analytics. Compared with the aforementioned features, learning from audio and video features is relatively less common in the context of recommender systems.

Music recommendation is a representative audio-based recommendation scenario. In general, audio information such as rhythm, melody, and timbre is of critical importance to listeners. In the music recommender model designed by Van et al. [85], CNNs are used to extract music features from audio clips. CNNs are also adopted by Huang et al. [56] for music representation learning from acoustic inputs to address the task of music co-listen prediction. CNNs are suited for audio representation learning as they can operate on multiple timescales. Concurrently, Wang et al. [121] use deep belief networks (DBNs) for automatic music feature extraction for music recommendation and they claim that DBNs do well in modeling rhythms and melodies.

[3] https://www.flickr.com/.

Video features can also be leveraged to improve recommendations. Usually, videos are converted into a sequence of frames and audio waves. As such, CNNs based models become a desirable option for video analysis. For instance, Xu et al. [17] propose a key frame recommender system to select the key frames from a video for each user. A CNN with five convolutional layers and three fully-connected layers is used for frame representation learning. In the video recommender system proposed by Lee et al. [68], both video frames and audio features are included. The model employs an Inception-v3 network to extract frame features, then it aggregates the frame-level features into video-level ones with average pooling. The audio features are extracted with a modified version of ResNet-50.

In summary, for each type of data, various deep learning techniques can be applied. For text data, sequence modeling approaches are more suitable. While for image, video, and audio data, CNN based architectures are preferable.

3.4 Deep Learning for Graph-Structured Data in Recommendation

The interaction data generated from recommender systems can be formulated as a bipartite graph (shown in Fig. 10a). Users and items are two disjoint and independent sets. The interactions between users and items compose the edges of the graph. The user item interaction matrix is the biadjacency matrix of this bipartite graph. Furthermore, connections between users can also formulate a relationship/social network.

The idea of incorporating social networks into recommendation has been circulating for quite a while. Only recently has it been connected to deep learning approaches. In recent years, there is a surge of interest in graph neural networks which has achieved state-of-the-art performances on a number of graph related tasks. Given the interaction logs, it is natural to employ GNNs to learn from the bipartite graph so as to better understand the characteristics of nodes and their relationships. Using GNNs for recommender systems is beneficial for two

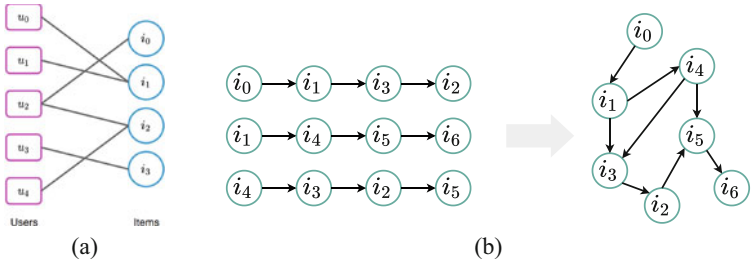

Fig. 10 Example graphs in recommender systems. (**a**) Interaction graph; (**b**) item session graph

reasons: (1) GNN can better model graph-structural relationships; (2) GNN can help incorporate other graph related data (e.g., social graph, knowledge graph).

Recommendation on the user-item bipartite graph is performed from the link prediction view, where ratings/purchase are represented as links. In doing so, the graph-structural information can be naturally captured. Berg et al. [5] propose a graph convolutional matrix completion model which produces latent factors of user and item nodes via message passing on the bipartite interaction graph. A bilinear decoder with the learnt latent factors is used to reconstruct the links. The same neighbourhood message passage is adopted in Wang et al. [120]. In addition, multi-hop propagation is also taken into consideration in this model. To avoid computation on the entire graph, Ying et al. [138] present a sampling based graph convolution approach for pins recommendation at Pinterest. The proposed model performs a localized convolution by sampling the neighborhood around a node, and dynamically constructing aggregations from the sampled neighborhood. Monti et al. [83] present a multi-graph convolutional neural networks to extract local features from the interaction matrix with row and column similarities encoded by user and item graphs.

GNNs can also be used to capture inter-item relatedness by mining from the items graph. For instance, in session-based recommendation task, the click sequences and transition patterns from sessions can be represented with a session graph (shown in Fig. 10b). As such, it is natural to use graph neural networks to learn click representations [87, 126] in session-aware recommender systems.

Social graph is another type of graph that is useful for recommender systems. When making decisions, users can be easily influenced by their friends. GNNs can be utilized for influence diffusion on social graphs in social recommendations. Fan et al. [28] propose learning user representations by simultaneously aggregating information from her neighbourhood items in the interaction graph and her neighbourhood users in the social network and an attention mechanism is adopted for information aggregation. Influences maybe context-dependent. For example, users might trust different groups of friends for different types of products. To model such dynamic effects, Song et al. [97] propose a dynamic graph attention network to attend the influence of friends with user's short-term actions.

Knowledge graph is also a widely available information source in recommender systems. Knowledge graph is a directed graph of linked data where nodes correspond to entities (e.g., person, movie, etc.) and edges correspond to relations. For example, a triple in a knowledge graph ⟨James Cameron, direct, Titanic⟩ has two entities (James Cameron and Titanic) and one relation (direct). Knowledge graphs are usually associated with items, e.g., movies in this example. As such, this supplementary information can be used to infer connections between items. For example, the main idea behind the model by Wang et al. [117] is to consider items as entities in the knowledge graph, and then the model aggregates neighborhood information for learning entity representations. In their later study [116], they add flexibility to the model by enabling user-specific relation learning. The authors also introduce a label-smoothing regularization to overcome the outfitting issue.

Graph neural networks have exhibited great potential in learning from graph-structured data in recommender systems. However, there are challenges that remain to be solved. A worrisome aspect of these models is the low computational efficiency. Real-world recommendation tasks on the other hand, often come with large scale data, so more efficient GNNs based recommender systems are expected.

3.5 Deep Learning for Cold-Start Recommendation

Cold-start problems happen when new users or new items arrive in the system. Because most recommendation models are built on historical interactions, the lack of interaction records for these new items/users makes the cold-start problem challenging. Leveraging the associated side information accounts for the major paradigm in cold-start scenarios.

Bansal et al. [3] present deep recurrent neural networks for cold-start article recommendation. It adopts GRUs to encode text into latent factors, and a multi-task learning framework is proposed to enable both recommendations and item metadata predictions. Instead of incorporating additional content-based objective terms, Volkovs et al. [113] focus on the optimization process. They demonstrate that neural network models can be explicitly trained for cold-start recommendation via dropout. A key observation is that cold-start problem is equivalent to the problem of missing data. As such, they make DNNs generalize to cold-start settings by selecting an appropriate amount of dropout. This approach shows superior performance on both warm start and cold start scenarios. Pan et al. [86] propose a meta-learning approach to address the cold-start problem which trains an embedding generator for new items through gradient-based meta learning. This model learns embeddings by taking as input item content and attributes. Zhang et al. [142] present Star-GCN, a graph neural networks based recommendation model. Star-GCN predicts the embeddings of unseen nodes by the means of masking a part of (or the whole) user/item embeddings and then reconstructing the masked embeddings. Zhang et al. [143] propose a graph based matrix completion algorithm under the inductive setting. It trains GNNs based on one-hop subgraphs, and can generalize to unseen users/items. This approach does not use any side information. However, it cannot address extreme cold-start scenarios because it needs the cold user/item to have a few interactions with neighbors.

3.6 Deep Learning for Recommendation: Beyond Accuracy

Beyond accuracy, some other characteristics of recommender systems are also crucial. Here, we review methods that enhance the explainability and robustness of recommender engines using deep learning techniques.

3.6.1 Explainable Recommendations

Explainable recommender systems not only generate personalised recommendations but also produce intuitive explanations to the recommendations. In doing so, model transparency is enhanced and the persuasiveness and trustworthiness of recommendations are improved. Moreover, the explanations also provide a way for system designers to diagnose and refine the recommendation algorithms.

It is well known that the internal decision process of deep learning models is hard to control and explain. By circumventing direct explanation on the internal architectures, we can use deep learning methods to generate explicit explanations for the recommendations. For example, we can generate a user-specific sentence for each item in a recommendation list. For example, the model by Li et al. [73] can simultaneously make recommendations and generate readable tips that describe possible reviews a user might give to the recommended items. The model combines a recommendation module, a neural rating regression module, and a text generation module, RNNs. The text generation module is similar to the decoder of sequence-to-sequence models for machine translation, where user reviews are used as the supervision signals of the tips generation module. To generate controllable textual explanations, Li et al. [72] adopt a cue word based GRU [136] to control the generated sentence with a given cue word. For example, for hotel recommendations, given a cue word "room", the generated tips would be "the room is spacious and comfortable". It offers the option to users to select aspects that they care about.

Attention mechanism is also a desirable tool to enhance explanation. It can be employed to highlight critical information of the recommended items. For example, attention mechanism can be used to highlight the prime information in the reviews for each recommendation. Chen et al. [10] propose an attention based approach to pick up valuable and useful reviews from all reviews of a target item. They further improve the accuracy of explanations by considering the dynamics of user preferences. In specific, the model attentively learns the importance of review information according to the user's current state and preference [18]. Attention networks are also used to highlight key phrases/words in the reviews in Seo et al. [94]. Key regions in images can also be highlighted to provide visual explanations. In the area of fashion recommendation, Chen et al. [15] adopt CNNs to extract region representations of each image and use an attention mechanism to select the most impactful regions to the prediction. This model therefore can tell users, for recommended items, which parts that they are likely to be interested in.

The rich linked data in knowledge graphs has also been leveraged to provide tailored explanations for recommendations. Huang et al. [55] present a knowledge graph based approach to enhance explanations in sequential recommendation settings. A memory network is used to capture attribute-level preferences by leveraging external knowledge graphs. This approach provides explanations on which attributes (e.g., genre, album, singer, actor, etc.) are taking effects when making recommendations. Knowledge graph can also used to identify the paths

that lead to the recommendations. Huang et al. [57] adopt a self-attention network to explicitly model the knowledge graph aware paths between user-item pairs for path-wise explanations. The attention scores help identify the most influential path for each recommendation. Xian et al. [128] propose a deep reinforcement learning approach for pathfinding in knowledge graphs as the interpretable evidence for recommendations. For example, item b is recommended to user u because of the path: {user $u \rightarrow$ item $a \rightarrow$ brand $e \rightarrow$ item b}. This reasoning path gives explicit explanations for why a certain item is recommended.

3.6.2 Robust Recommender Systems

Recent work shows that recommendation models are not robust and are vulnerable to adversarial attacks. Defending against those adversaries lives at the core pertaining to the robustness. This task can be modeled as a minimax game similar to that in GANs where the adversary crafts adversarial examples to degrade recommendation performances while the recommender engine is trained to become robust to such noises.

He et al. [46] propose an adversarial personalized ranking approach to improve Bayesian personalised ranking (BPR) method by performing adversarial training. The adversarial perturbations are added on the embedding vectors of users and items, and are trained to maximize the BPR objective. The recommendation model is trained to minimize the BPR loss plus an additional loss with adversary. This adversarial training method can also be combined with autoencoder [141], which could further improve both the robustness and performance. Adversarial permutations can also be added to the content of users/items. For instance, Tang et al. [102] apply this adversarial training approach to multimedia recommender systems where the adversary adds perturbations on the multimedia content of items to maximize VBPR (visual BPR) loss. The recommendation model is learned by minimizing the VPBR and adversary's loss.

3.7 Deep Reinforcement Learning for Recommendation

Recommender systems are usually solved in a supervised learning paradigm. There are two major limitations of this line of work: (1) recommendation is an interactive process; supervised learning will bring the so-called system bias that only observed feedback from the current system is considered. (2) it tends to give myopic recommendations by only recommending catchy items to get immediate response instead of long term user utility. In recent years, DRL has begun to attract attention in the recommender systems community. Using DRL, the recommendation process can be formulated as a dynamic decision-making problem. DRL aims to maximize the overall accumulated reward and considers the long term reward of the recommendations. In order to build a recommender system based on reinforcement

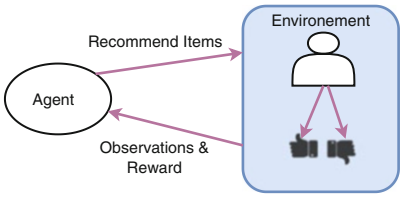

Fig. 11 Reinforcement learning based recommender systems

learning, the key elements are defined as: *Agent* is a candidate generator; *State* corresponds to user interests and contexts; *Reward* is defined as user satisfaction; and *action* represents the nomination of items for recommendation. The definition might be different in some cases, we refer readers to corresponding papers (Fig. 11).

To overcome the aforementioned limitations of supervised learning, Chen et al. [12] propose REINFORCE, for recommendations at YouTube. REINFORCE is a policy-gradient algorithm which can scale to production environment with an action spaces up to millions. Chen et al. [11] present a tree-structured policy gradient approach to handle the problem of large discrete action space in recommender systems. In specific, to reduce time complexity, a balanced hierarchical clustering tree is built over items and selecting an item is formulated as finding a path from the root in this tree. Zhao et al. [153] apply DRL to capture distinct contributions of both positive and negative feedback in a sequential interaction setting. During scanning of the recommendation list, users may skip some recommended items. Incorporating such negative feedback helps the system gain better understandings about user's preference. Zhao et al. [152] explore the page-wise recommendation scenario with DRL. The proposed framework DeepPage is able to adaptively optimize a page of items based on user's real-time actions. Zheng et al. [154] present DRN, a reinforcement learning based news recommender method which considers: (1) the dynamic changes of news content and user preferences, (2) incorporating return patterns (to the service) of users, and (3) the increase diversity of recommendations. Xian et al. [128] propose a reinforcement learning approach to find the paths in knowledge graph to enhance the explainability of recommendation. Zou et al. [159] propose PDQ where real customers are replaced with a customer simulator. The simulator is used to simulate the environment and is optimized according to the current recommendation policy. Using a customer simulator reduces the instability of convergence and provides unlimited interactions without involving real users. In many works like [11, 152, 153], the embeddings of users and items are usually pretrained and kept fixed during the training of the reinforcement learning algorithms. However, the pretrained and fixed embeddings cannot reflect the dynamics in user preferences and item characteristics, and might lead to sub-optimal solutions. To solve this problem, Liu et al. [75] propose an end-to-end framework to overcome the limitations. As the embedding component cannot be stably trained with reinforcement learning algorithms, they introduce an additional supervised learning classifier to stabilize the embedding learning process by predicting whether a user offers positive feedback to the recommended item or not.

4 Deep Learning for Recommender Systems: Applications

In what follows, we introduce how deep learning techniques are used to build a variety of recommendation applications. We categorize existing publications according to their target domains which are prevalent in today's life. We contextualize closely related domains and review how they address specific recommendation problems in these domains. Table 2 summarizes a representative publication. Note that this is not an exhaustive enumeration.

4.1 Deep Learning for E-commerce Recommendation

Recommender systems have become serious business tools in many e-commerce sites such as Amazon,[4] eBay,[5] and Alibaba,[6] and are re-shaping the world of e-commerce. Most large commerce web sites are using recommender systems to help their customers to identify the products they are interested in. Nowadays, deep learning has been playing a primary role in the recommendation process on e-commerce platforms. Here, we give some examples to illustrate how deep learning is used in some e-commerce platforms.[7]

Lake et al. [66] present a deep learning based product embedding approach for product recommendations on Amazon. The model adopts an attention network to select relevant pieces of information from a user's historical interactions. It then learns a joint representation from a specific user-item pair, instead of representing user and items in a common latent space. Both online and offline

Table 2 Deep learning based recommendation applications and corresponding publications

Applications	Publications
E-commerce	Lake et al. [66], Galron et al. [33], Zhou et al. [156, 157], Wu et al. [125], Li et al. [70], Feng et al. [29, 30], Cen et al. [8], Chen et al. [13, 14]
Entertainment	Covington et al. [23], Chen et al. [12], Van et al. [85], Huang et al. [56], Wang et al. [121], Cheng et al. [19], Ying et al. [138], Yang et al. [135]
News	Wu et al. [122, 123], An et al. [1], De et al. [98], Wang et al. [115] Hu et al. [54]
Point-of-Interests	Yang et al. [134], Wang et al. [119], Chang et al. [9], Liu et al. [77] Zhou et al. [158]
Other domains	Shang et al. [95, 96], Biswal et al. [7], Tay et al. [106]

[4] https://www.amazon.com/.

[5] https://www.ebay.com/.

[6] https://www.alibaba.com/.

[7] Note that there is no guarantee that the listed methods are currently in use.

experiments demonstrate promising performance of this method for personalized product recommendation. Galron et al. [33] propose an item embedding network for recommendation on eBay. It has an item embedding network to learn continuous item representations from its features (e.g., attributes, categories, and title tokens), and a prediction network to compute the similarity between seed items (e.g. a recently purchased item) and recommendation candidates. Deep learning based recommender algorithms are also widely employed in Alibaba [8, 13, 14, 29, 30, 70, 156, 157]. Zhou et al. [157] use deep learning to learn user representations from user profiles and her behaviors. A local activation unit is used to attend the related user interests by soft-searching for relevant parts from historical behaviors. In the follow-up work [13, 30, 156], the sequential order of user behaviors is also considered. In specific, sequence modeling approaches RNNs, self-attention mechanism, and transformer are adopted to capture the sequential signals underlying users' behavior data. To enhance the capability of modeling the diversity of user interests, [70] and [8] adopt dynamic routing approaches to learn each user multiple interest representations. Knowledge graph is also leveraged to improve the recommendation performance [29]. Xu et al. [14] explore the use of deep learning in fashion outfit recommendations on Alibaba. In this model, an encoder-decoder model is designed to generate personalized fashion outfits. Wu et al. [125] deploy a deep recurrent neural network based method for recommendation at NetEase (Kaola platform[8]). In this model, deep RNNs are used to model the sequential behaviors of users.

4.2 Deep Learning for Online Entertainment Recommendation

Online entertainment platforms are taking over theatrical and home entertainment business. Online entertainment covers a range of domains such as music, video, book, etc. As such, personalized entertainment recommendation is critical to help people narrow the universe of potential items to fit their unique tastes. In recent years, deep learning also plays an important part in online entertainment recommendations.

Covington et al. [23] propose a deep neural network based recommendation method for recommendation on YouTube. It consists of a deep candidate generation module and a separate deep ranking module. Both modules are fully connected deep nonlinear neural networks. The deep candidate generation module is used for candidate selection from the large pool of videos and the ranking module ranks candidates. To improve the long-term user utility and tackle system biases, Chen et al. [12] further propose a deep reinforcement learning based recommendation algorithm which is currently in use on YouTube. Deep learning techniques can also be applied for music recommendations [56, 85, 121]. In specific, these models use CNNs to extract music representations from audio signals. Cheng et al. [19]

[8] https://www.kaola.com/.

present a wide and deep learning based model to make app recommendations in the Google play application store. A two-tower neural network with mixed negative sampling is also employed in Google play [135]. Ying et al. [19] design a graph convolutional neural network based recommender that is deployed to perform Pins recommendations at Pinterest.[9]

4.3 Deep Learning Based News Recommendation

Reading news online has become a routine in many people's lives. To avoid overwhelming readers, personalization of the recommended articles is important for online news services. Generally, news is text extensive and sometimes also contains images, audio, and video pieces. An encoder that gets the representation for each piece of news is indispensable, and deep learning techniques are suitable tools for news resources.

Wu et al. [122] propose a news recommender model which adopts CNNs to learn news representations from titles. The model applies attention networks to select important words. User representation is learned via an attentive multi-view learning framework from user's search queries, clicked news, and browsed webpages. They further improve the news recommender by learning both news and users representations with multi-head self-attention [123]. An et al. [1] use an attention-based CNNs method for news representation learning, and incorporate sequential behaviors and short-term preferences of users with GRUs. User's long- and short-term representations are integrated by concatenation, or by using the long-term representation as the initialization of the hidden state of the GRUs. De et al. [98] propose a deep learning architecture for session-based recommendation. This architecture has two modules: a news article representation module and a session-based recommendation module with RNNs. News readers usually have diverse interests. Their interests in different topics or events can be revealed from their historical browsed news. To learn the fine-grained interests, Wang et al. [115] construct representations for each news from multiple semantic views with stacked dilated convolutional encoder, and perform fine-grained matching between candidate news and user's browsed news at different semantic levels. Hu et al. [54] model the diverse interests by disentangling a user's latent preference factors. In specific, they construct a bipartite graph from the user-news interactions and apply graph neural networks to relationships encoding via information propagation. The learned representations are disentangled by a neighborhood routing algorithm to enhance the expressiveness and interpretability.

[9] https://www.pinterest.com/.

4.4 Deep Learning for Point-of-Interest Recommendation

Point-of-Interest recommendation is useful and acts as a link between internet users and real-world places such as stores, cinemas, restaurants, and tourist attractions. Location-aware apps like Yelp,[10] Foursquare,[11] Google map,[12] and Meituan[13] all provide location based services that are extensively used by millions of users.

Yang et al. [134] adopt fully connected neural networks to model the interaction between users and POIs, and regularize users' and POIs' latent factors with an additional context prediction task. Wang et al. [119] propose to enrich the POIs recommendation by incorporating the visual contents of users' posts (e.g., photos of landmarks and food). In their model, CNNs is used for visual feature extraction which is utilized for user intention identification. Textual information such as reviews/comments on locations can also be utilized. Chang et al. [9] use both multi-head attention mechanism and character-level CNNs to encode user's generated textual content into content embeddings. Those content embeddings are integrated with LSTMs to capture user's overall interests. To model the sequential behaviors, Liu et al. [77] propose a spatial temporal recurrent neural networks to model both time-specific transition and distance specific transition in POIs. Zhou et al. [158] integrate a temporal latent Dirichlet allocation topic model and memory network for personalized POI recommendation. This model integrates the POI-specific geographical influence to enhance recommendations. Zhao et al. [151] present a spatio-temporal gated network (STGN) for POI recommendation by enhancing the long-short term memory network with gating mechanisms. Specifically, a time gate and a distance gate are proposed to control the updates of short-term and long-term preference representations.

4.5 Deep Learning Based Recommendation on Other Domains

Besides the domains described above, there exist substantial studies on many other domains. We use healthcare as an example domain here. An important healthcare related application is medication recommendations. Shang et al. [96] present a graph augmented memory networks for medication combination recommendation. It integrates both drug-to-drug interaction knowledge and patient health records to provide safe and personalized recommendation. The problem in medication recommendation is that the records of patients with only single visit (visit hospital only once) are usually discarded. To leverage those data, Shang et al. [95] propose

[10] https://www.yelp.com/.

[11] https://foursquare.com/.

[12] https://www.google.com/maps.

[13] http://www.meituan.com/.

a pre-training approach to leverage the single visit health records and fine-tune it for downstream predictive tasks on longitudinal electronic health records from patients with multiple visits. Biswal et al. [7] present doctor2vec to help identify the appropriate doctors for clinical trial based on trial description and patient EHR data. Friends recommendation is also popular in applications such as Twitter and Facebook. For instance, Tay et al. [106] present an attention based GRU model for friends recommendation on Twitter.

5 Discussion and Conclusion

Each task in recommender systems has its own challenges and specific factors to consider. This chapter provides a categorised overview and perspective on the published academic literature on deep learning based recommender systems. For example, deep learning is advantageous in interaction modeling by introducing nonlinearity and high-order interactions; user modeling by capturing the temporal dynamics via sequential models and by disentangling user complex interests; representation learning from unstructured content information; interpretability and robustness. We hope that this chapter can shed some light on those who are confronting these tough problems.

Nonetheless, some attractive properties come with the cost of time complexity. Not all models reviewed in the chapter are computationally efficient, and some models have high running time even with the help of GPU acceleration. In addition, deep learning models are black boxes and neurons interact in complex ways to produce the results. It is difficult to interpret the learned model and understand how the outputs are arrived at. Luckily, we can partially bypass this problem by producing human understandable explanations for recommendations. Deep learning based models usually have many hyperparameters to tune. For example, the number of layers, the number of neurons, dropout rate, how to initialize parameters, just to name a few. There are no principles to guide the hyperparameter selection process. Moreover, it usually requires a large amount of data to learn an effective deep learning model. It might lead to sub-optimal solutions if the system does not have enough data available. Expectantly, these limitations can be reduced with the development of deep learning techniques.

Moreover, criticism on the field cannot be overlooked [25, 65, 89, 147]. One major concern is that the evaluation of those newly proposed recommendation models are not rigorous, which is manifested in several aspects. Firstly, the selection of baselines and datasets in most papers are seemingly arbitrary. Authors have free rein over the choices of datasets(e.g., random splits of train/val/test)/baselines. This is understandable to a certain extent when the number of baselines/datasets increases too quickly. Nonetheless, without a fair and comprehensive comparison, the reported improvements do not add up and the advancements do not seem convincing. Secondly, the inappropriate use of evaluation methodology exaggerates the problem. Recent study shows that a popular evaluation methodology based on

sampling that are widely adopted to measure the effectiveness of recommendation models does not reflect the true model effectiveness [65, 89]. Having a full grasp of the chosen evaluation methodology is critical in preventing inconsistent results. Thirdly, many traditional methods are not fully tuned for comparison. It is reported that some simple traditional methods can easily outperform neural networks on some tasks with sufficient tuning [78]. Although these issues are not only tied to this field, it is a good practice to bear these pitfalls in mind when evaluating your models. We also encourage that standard benchmarks with consistent evaluation metrics and data splits should be constructed in order to better judge incoming recommendation algorithms. Also, codes and datasets shall be released to enhance reproducibility when appropriate.

This chapter introduces the main deep learning techniques that can be applied in the design of recommender systems. We have reviewed their use in the literature and characterized them from different perspectives. The aim is to give researchers and practitioners an understanding and a thorough summary of the field's breadth and depth. There are broad types of recommender systems; some of them are rarely touched in the moment and some aspects might require more modelling efforts. We hope that deep learning could lead to the advancement of next-generation recommender systems that possess better intelligence and offer better customer experience.

References

1. M. An, F. Wu, C. Wu, K. Zhang, Z. Liu, X. Xie, Neural news recommendation with long- and short-term user representations, in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence* (2019), pp. 336–345
2. S. Bai, J.Z. Kolter, V. Koltun, An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. Preprint, arXiv:1803.01271 (2018)
3. T. Bansal, D. Belanger, A. McCallum, Ask the GRU: multi-task learning for deep text recommendations, in *Recsys, New York, NY* (2016), pp. 107–114
4. O. Barkan, N. Koenigstein, Item2vec: neural item embedding for collaborative filtering, in *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)* (IEEE, Piscataway, 2016), pp. 1–6
5. R. van den Berg, T.N. Kipf, M. Welling, Graph convolutional matrix completion. CoRR abs/1706.02263 (2017). http://arxiv.org/abs/1706.02263
6. H. Bharadhwaj, H. Park, B.Y. Lim, RecGAN: recurrent generative adversarial networks for recommendation systems, in *Recsys* (2018), pp. 372–376
7. S. Biswal, C. Xiao, L.M. Glass, E. Milkovits, J. Sun, Doctor2vec: dynamic doctor representation learning for clinical trial recruitment, in *Proceedings of AAAI* (2020), pp. 557–564
8. Y. Cen, J. Zhang, X. Zou, C. Zhou, H. Yang, J. Tang, Controllable multi-interest framework for recommendation, in *Proceedings of SIGKDD* (2020)
9. B. Chang, Y. Koh, D. Park, J. Kang, Content-aware successive point-of-interest recommendation, in *Proceedings of the 2020 SIAM International Conference on Data Mining* (SIAM, Philadelphia, 2020), pp. 100–108
10. C. Chen, M. Zhang, Y. Liu, S. Ma, Neural attentional rating regression with review-level explanations, in *Proceedings of WWW, WWW '18, Republic and Canton of Geneva, CHE* (2018), pp. 1583–1592

11. H. Chen, X. Dai, H. Cai, W. Zhang, X. Wang, R. Tang, Y. Zhang, Y. Yu, Large-scale interactive recommendation with tree-structured policy gradient, in *Proceedings of AAAI*, vol. 33 (2019), pp. 3312–3320
12. M. Chen, A. Beutel, P. Covington, S. Jain, F. Belletti, E.H. Chi, Top-k off-policy correction for a reinforce recommender system, in *Proceedings of WSDM, WSDM '19* (Association for Computing Machinery, New York, 2019), p. 456–464
13. Q. Chen, H. Zhao, W. Li, P. Huang, W. Ou, Behavior sequence transformer for e-commerce recommendation in Alibaba, in *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data, DLP-KDD '19, New York* (2019)
14. W. Chen, P. Huang, J. Xu, X. Guo, C. Guo, F. Sun, C. Li, A. Pfadler, H. Zhao, B. Zhao, POG: personalized outfit generation for fashion recommendation at Alibaba iFashion, in *Proceedings of SIGKDD, KDD '19, New York* (2019), pp. 2662–2670
15. X. Chen, H. Chen, H. Xu, Y. Zhang, Y. Cao, Z. Qin, H. Zha, Personalized fashion recommendation with visual explanations based on multimodal attention network: towards visually explainable recommendation, in *Proceedings of SIGIR, SIGIR'19, New York* (2019), pp. 765–774
16. X. Chen, H. Xu, Y. Zhang, J. Tang, Y. Cao, Z. Qin, H. Zha, Sequential recommendation with user memory networks, in *Proceedings of WSDM* (2018), pp. 108–116
17. X. Chen, Y. Zhang, Q. Ai, H. Xu, J. Yan, Z. Qin, Personalized key frame recommendation, in *Proceedings of SIGIR, New York* (2017), pp. 315–324
18. X. Chen, Y. Zhang, Z. Qin, Dynamic explainable recommendation based on neural attentive models, in *Proceedings of AAAI*, vol. 33 (2019), pp. 53–60
19. H.T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir et al., Wide & deep learning for recommender systems, in *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems* (2016), pp. 7–10
20. Z. Cheng, Y. Ding, X. He, L. Zhu, X. Song, M. Kankanhalli, A3NCF: an adaptive aspect attention model for rating prediction, in *Proceedings of IJCAI* (2018), pp. 3748–3754
21. W.T. Chu, Y.L. Tsai, A hybrid recommendation system considering visual information for predicting favorite restaurants. WWW **20**(6), 1313–1331 (2017)
22. J. Chung, C. Gulcehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling. Preprint, arXiv:1412.3555 (2014)
23. P. Covington, J. Adams, E. Sargin, Deep neural networks for youtube recommendations, in *Proceedings of Recsys* (2016), pp. 191–198
24. A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, A.A. Bharath, Generative adversarial networks: an overview. IEEE Signal Process. Mag. **35**(1), 53–65 (2018)
25. M.F. Dacrema, P. Cremonesi, D. Jannach, Are we really making much progress? A worrying analysis of recent neural recommendation approaches, in *Proceedings of the 13th ACM Conference on Recommender Systems* (2019), pp. 101–109
26. J. Devlin, M.W. Chang, K. Lee, K. Toutanova, Bert: pre-training of deep bidirectional transformers for language understanding, in *Proceedings of NAACL-HLT* (2019)
27. T. Donkers, B. Loepp, J. Ziegler, Sequential user-based recurrent neural network recommendations, in *Proceedings of Recsys* (2017), pp. 152–160
28. W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, D. Yin, Graph neural networks for social recommendation, in *Proceedings of WWW, New York* (2019), pp. 417–426
29. Y. Feng, B. Hu, F. Lv, Q. Liu, Z. Zhang, W. Ou, ATBRG: adaptive target-behavior relational graph network for effective recommendation, in *Proceedings of SIGIR* (2020)
30. Y. Feng, F. Lv, W. Shen, M. Wang, F. Sun, Y. Zhu, K. Yang, Deep session interest network for click-through rate prediction, in *International Joint Conferences on Artificial Intelligence Organization, IJCAI* (2019), pp. 2301–2307
31. A. Fischer, C. Igel, An introduction to restricted Boltzmann machines, in *Iberoamerican Congress on Pattern Recognition* (Springer, New York, 2012), pp. 14–36
32. V. François-Lavet, P. Henderson, R. Islam, M.G. Bellemare, J. Pineau et al., An introduction to deep reinforcement learning. Found. Trends Mach. Learn. **11**(3–4), 219–354 (2018)

33. D.A. Galron, Y.M. Brovman, J. Chung, M. Wieja, P. Wang, Deep item-based collaborative filtering for sparse implicit feedback (2018)
34. X. Geng, H. Zhang, J. Bian, T.S. Chua, Learning image and user features for recommendation in social networks, in *Proceedings of ICCV* (2015), pp. 4274–4282
35. K. Georgiev, P. Nakov, A non-IID framework for collaborative filtering with restricted Boltzmann machines, in *Proceedings of ICML* (2013), pp. 1148–1156
36. I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning* (MIT Press, 2016). https://www.deeplearningbook.org
37. M. Grbovic, V. Radosavljevic, N. Djuric, N. Bhamidipati, J. Savla, V. Bhagwan, D. Sharp, E-commerce in your inbox: product recommendations at scale, in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2015), pp. 1809–1818
38. A. Greenstein-Messica, L. Rokach, M. Friedman, Session-based recommendations using item embedding, in *Proceedings of the 22nd International Conference on Intelligent User Interfaces* (2017), pp. 629–633
39. X. Guan, Z. Cheng, X. He, Y. Zhang, Z. Zhu, Q. Peng, T.S. Chua, Attentive aspect modeling for review-aware recommendation. ACM Trans. Inf. Syst. **37**(3) (2019)
40. H. Guo, R. Tang, Y. Ye, Z. Li, X. He, DeepFM: a factorization-machine based neural network for CTR prediction, in *Proceedings of IJCAI* (2017), pp. 1725–1731
41. X. Guo, C. Shi, C. Liu, Intention modeling from ordered and unordered facets for sequential recommendation, in *Proceedings of WWW* (2020), pp. 1127–1137
42. R. He, J. McAuley, Ups and downs: modeling the visual evolution of fashion trends with one-class collaborative filtering, in *Proceedings of WWW, Republic and Canton of Geneva, CHE* (2016), pp. 507–517
43. R. He, J. McAuley, VBPR: visual Bayesian personalized ranking from implicit Feedback, in *Proceedings of AAAI* (2016), pp. 144–150
44. X. He, T.S. Chua, Neural factorization machines for sparse predictive Analytics, in *Proceedings of SIGIR* (2017), pp. 355–364
45. X. He, X. Du, X. Wang, F. Tian, J. Tang, T.S. Chua, Outer product-based neural collaborative filtering, in *Proceedings of IJCAI* (2018), pp. 2227–2233
46. X. He, Z. He, X. Du, T.S. Chua, Adversarial personalized ranking for recommendation, in *Proceedings of SIGIR, SIGIR '18, New York* (2018), pp. 355–364
47. X. He, L. Liao, H. Zhang, L. Nie, X. Hu, T.S. Chua, Neural collaborative filtering, in *Proceedings of WWW* (2017), pp. 173–182
48. B. Hidasi, A. Karatzoglou, Recurrent neural networks with top-k gains for session-based recommendations, in *Proceedings of CIKM* (2018), pp. 843–852
49. B. Hidasi, A. Karatzoglou, L. Baltrunas, D. Tikk, Session-based recommendations with recurrent neural networks. Preprint, arXiv:1511.06939 (2015)
50. B. Hidasi, M. Quadrana, A. Karatzoglou, D. Tikk, Parallel recurrent neural network architectures for feature-rich session-based recommendations, in *Proceedings of Recsys* (2016), pp. 241–248
51. G.E. Hinton, Deep belief networks. Scholarpedia **4**(5), 5947 (2009)
52. S. Hochreiter, J. Schmidhuber, Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
53. C.K. Hsieh, L. Yang, Y. Cui, T.Y. Lin, S. Belongie, D. Estrin, Collaborative metric learning, in *Proceedings of WWW* (2017), pp. 193–201
54. L. Hu, S. Xu, C. Li, C. Yang, C. Shi, N. Duan, X. Xie, M. Zhou, Graph neural news recommendation with unsupervised preference disentanglement, in *Proceedings of Association for Computational Linguistics* (2020)
55. J. Huang, W.X. Zhao, H. Dou, J.R. Wen, E.Y. Chang, Improving sequential recommendation with knowledge-enhanced memory networks, in *Proceedings of SIGIR, SIGIR '18, New York* (2018), pp. 505–514
56. Q. Huang, A. Jansen, L. Zhang, D.P.W. Ellis, R.A. Saurous, J. Anderson, Large-scale weakly-supervised content embeddings for music recommendation and tagging, in *ICASSP 2020–*

2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2020), pp. 8364–8368
57. X. Huang, Q. Fang, S. Qian, J. Sang, Y. Li, C. Xu, Explainable interaction-driven user modeling over knowledge graph for sequential recommendation, in *Proceedings of the 27th ACM International Conference on Multimedia, MM '19, New York* (2019), pp. 548–556
58. X. Huang, S. Qian, Q. Fang, J. Sang, C. Xu, CSAN: contextual self-attention network for user sequential recommendation, in *Proceedings of the 26th ACM International Conference on Multimedia* (2018), pp. 447–455
59. Y. Ji, A. Sun, J. Zhang, C. Li, A re-visit of the popularity baseline in recommender systems, in *Proceedings of SIGIR* (2020), pp. 1749–1752
60. W.C. Kang, E. Kim, J. Leskovec, C. Rosenberg, J. McAuley, Complete the look: scene-based complementary product recommendation, in *Proceedings of CVPR* (2019)
61. W.C. Kang, J. McAuley, Self-attentive sequential recommendation. in *Proceedings of ICDM* (IEEE, Piscataway, 2018), pp. 197–206
62. D. Kim, C. Park, J. Oh, S. Lee, H. Yu, Convolutional matrix factorization for document context-aware recommendation, in *Proceedings of Recsys, New York* (2016), pp. 233–240
63. T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks. Preprint, arXiv:1609.02907 (2016)
64. Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)
65. W. Krichene, S. Rendle, On sampled metrics for item recommendation, in *Proceedings of SIGKDD, KDD '20, New York* (2020), pp. 1748–1757
66. T. Lake, S.A. Williamson, A.T. Hawk, C.C. Johnson, B.P. Wing, Large-scale collaborative filtering with product embeddings. Preprint, arXiv:1901.04321 (2019)
67. H. Lee, Y. Ahn, H. Lee, S. Ha, S.g. Lee, Quote recommendation in dialogue using deep neural network, in *Proceedings of SIGIR, New York* (2016), pp. 957–960
68. J. Lee, S. Abu-El-Haija, B. Varadarajan, A.P. Natsev, Collaborative deep metric learning for video understanding, in *Proceedings of SIGKDD, New York* (2018), pp. 481–490
69. C. Lei, D. Liu, W. Li, Z.J. Zha, H. Li, Comparative deep learning of hybrid representations for image recommendations, in *Proceedings of CVPR* (2016)
70. C. Li, Z. Liu, M. Wu, Y. Xu, H. Zhao, P. Huang, G. Kang, Q. Chen, W. Li, D.L. Lee, Multi-interest network with dynamic routing for recommendation at Tmall, in *Proceedings of CIKM, CIKM '19, New York* (2019), pp. 2615–2623
71. J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, J. Ma, Neural attentive session-based recommendation, in *Proceedings of CIKM* (2017), pp. 1419–1428
72. L. Li, L. Chen, Y. Zhang, Towards controllable explanation generation for recommender systems via neural template, in *Proceedings of WWW, WWW '20, New York* (2020), pp. 198–202
73. P. Li, Z. Wang, Z. Ren, L. Bing, W. Lam, Neural rating regression with abstractive tips generation for recommendation, in *Proceedings of SIGIR, SIGIR '17, New York* (2017), pp. 345–354
74. J. Lian, X. Zhou, F. Zhang, Z. Chen, X. Xie, G. Sun, xDeepFM: combining explicit and implicit feature interactions for recommender systems, in *Proceedings of SIGKDD* (2018), pp. 1754–1763
75. F. Liu, H. Guo, X. Li, R. Tang, Y. Ye, X. He, End-to-end deep reinforcement learning based recommendation with supervised embedding, in *Proceedings of WSDM, WSDM '20, New York* (2020), pp. 384–392
76. Q. Liu, S. Wu, L. Wang, Deepstyle: learning user preferences for visual recommendation, in *Proceedings of SIGIR, New York* (2017), pp. 841–844
77. Q. Liu, S. Wu, L. Wang, T. Tan, Predicting the next location: a recurrent model with spatial and temporal contexts, in *Thirtieth AAAI Conference on Artificial Intelligence* (2016)
78. M. Ludewig, D. Jannach, Evaluation of session-based recommendation algorithms. User Model. User-Adapted Interact. **28**(4–5), 331–390 (2018)

79. C. Ma, P. Kang, X. Liu, Hierarchical gating networks for sequential recommendation, in *Proceedings of SIGKDD* (2019), pp. 825–833
80. J. Ma, C. Zhou, P. Cui, H. Yang, W. Zhu, Learning disentangled representations for recommendation, in *Proceeding of NeurIPS*, ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett (Curran Associates, Red Hook, 2019), pp. 5711–5722
81. J. McAuley, C. Targett, Q. Shi, A. van den Hengel, Image-based recommendations on styles and substitutes, in *Proceeding of SIGIR, New York* (2015), pp. 43–52
82. T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space. Preprint, arXiv:1301.3781 (2013)
83. F. Monti, M. Bronstein, X. Bresson, Geometric matrix completion with recurrent multi-graph neural networks, in *Advances in Neural Information Processing Systems*, ed. by I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Curran Associates, Red Hook, 2017), pp. 3697–3707
84. W. Niu, J. Caverlee, H. Lu, Neural personalized ranking for image recommendation, in *Proceedings of WSDM, New York* (2018), pp. 423–431
85. A. van den Oord, S. Dieleman, B. Schrauwen, Deep content-based music recommendation, in *Advances in Neural Information Processing Systems*, ed. by C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, K.Q. Weinberger (2013), pp. 2643–2651
86. F. Pan, S. Li, X. Ao, P. Tang, Q. He, Warm up cold-start advertisements: improving CTR predictions via learning to learn ID embeddings, in *Proceedings of SIGIR, SIGIR'19, New York* (2019), pp. 695–704
87. R. Qiu, Z. Huang, J. Li, H. Yin, Exploiting cross-session information for session-based recommendation with graph neural networks. ACM Trans. Inf. Syst. **38**(3), 1–23 (2020)
88. S. Rendle, W. Krichene, L. Zhang, J. Anderson, Neural collaborative filtering vs. matrix factorization revisited. Preprint, arXiv:2005.09683 (2020)
89. S. Rendle, W. Krichene, L. Zhang, J. Anderson, Neural collaborative filtering vs. matrix factorization revisited, in *Fourteenth ACM Conference on Recommender Systems, RecSys '20, New York* (2020), pp. 240–248
90. S. Sabour, N. Frosst, G.E. Hinton, Dynamic routing between capsules, in *Proceedings of NeurIPS* (2017), pp. 3856–3866
91. R. Salakhutdinov, A. Mnih, G. Hinton, Restricted Boltzmann machines for collaborative filtering, in *Proceedings of ICML* (2007), pp. 791–798
92. B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in *Proceedings of WWW* (2001), pp. 285–295
93. S. Sedhain, A.K. Menon, S. Sanner, L. Xie, Autorec: autoencoders meet collaborative filtering, in *Proceedings of WWW, WWW '15 Companion, New York* (2015), pp. 111–112
94. S. Seo, J. Huang, H. Yang, Y. Liu, Interpretable convolutional neural networks with dual local and global attention for review rating prediction, in *Proceedings of Recsys, New York* (2017), pp. 297–305
95. J. Shang, T. Ma, C. Xiao, J. Sun, Pre-training of graph augmented transformers for medication recommendation, in *Proceedings of IJCAI* (2019), pp. 5953–5959
96. J. Shang, C. Xiao, T. Ma, H. Li, J. Sun, Gamenet: graph augmented memory networks for recommending medication combination, in *Proceedings of AAAI*, vol. 33 (2019), pp. 1126–1133
97. W. Song, Z. Xiao, Y. Wang, L. Charlin, M. Zhang, J. Tang, Session-based social recommendation via dynamic graph attention networks, in *Proceedings of WSDM, New York* (2019), pp. 555–563
98. G. de Souza Pereira Moreira, F. Ferreira, A.M. da Cunha, News session-based recommendations using deep neural networks, in *Proceedings of the 3rd Workshop on Deep Learning for Recommender Systems* (2018), pp. 15–23
99. F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, P. Jiang, Bert4rec: sequential recommendation with bidirectional encoder representations from transformer, in *Proceedings of CIKM* (2019), pp. 1441–1450

100. J. Tan, X. Wan, J. Xiao, A neural network approach to quote recommendation in writings, in *Proceedings of CIKM, New York* (2016), pp. 65–74
101. J. Tang, F. Belletti, S. Jain, M. Chen, A. Beutel, C. Xu, H. Chi, E.: Towards neural mixture recommender for long range dependent user sequences, in *Proceedings of WWW* (2019), pp. 1782–1793
102. J. Tang, X. Du, X. He, F. Yuan, Q. Tian, T. Chua, Adversarial training towards robust multimedia recommender system. IEEE Trans. Knowl. Data Eng. **32**(5), 855–867 (2020)
103. J. Tang, K. Wang, Personalized top-n sequential recommendation via convolutional sequence embedding, in *Proceedings of WSDM* (2018), pp. 565–573
104. Y. Tay, L. Anh Tuan, S.C. Hui, Latent relational metric learning via memory-based attention for collaborative ranking, in *Proceedings of WWW* (2018), pp. 729–739
105. Y. Tay, A.T. Luu, S.C. Hui, Multi-pointer co-attention networks for recommendation, in *Proceedings of SIGKDD, New York* (2018), pp. 2309–2318
106. Y. Tay, L.A. Tuan, S.C. Hui, Couplenet: paying attention to couples with coupled attention for relationship recommendation, in *Twelfth International AAAI Conference on Web and Social Media* (2018)
107. Y. Tay, S. Zhang, A.T. Luu, S.C. Hui, L. Yao, T.D.Q. Vinh, Holographic factorization machines for recommendation, in *Proceedings of AAAI*, vol. 33 (2019), pp. 5143–5150
108. T.X. Tuan, T.M. Phuong, 3D convolutional networks for session-based recommendation with content features, in *Proceedings of RecSys, New York* (2017), pp. 138–146
109. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in *Proceedings of NeurIPS* (2017), pp. 5998–6008
110. P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks. Preprint, arXiv:1710.10903 (2017)
111. P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.A. Manzagol, L. Bottou, Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. J. Mach. Learn. Res. **11**(12), 3371–3408 (2010)
112. L. Vinh Tran, Y. Tay, S. Zhang, G. Cong, X. Li, Hyperml: a boosting metric learning approach in hyperbolic space for recommender systems, in *Proceedings of WSDM* (2020), pp. 609–617
113. M. Volkovs, G. Yu, T. Poutanen, Dropoutnet: addressing cold start in recommender systems, in *Proceedings of NIPS*, ed. by I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Curran Associates, Red Hook, 2017), pp. 4957–4966
114. H. Wang, N. Wang, D.Y. Yeung, Collaborative deep learning for recommender systems, in *Proceedings of SIGKDD, New York* (2015), p. 1235–1244
115. H. Wang, F. Wu, Z. Liu, X. Xie, Fine-grained interest matching for neural news recommendation, in *Proceedings of ACL* (2020), pp. 836–845
116. H. Wang, F. Zhang, M. Zhang, J. Leskovec, M. Zhao, W. Li, Z. Wang, Knowledge-aware graph neural networks with label smoothness regularization for recommender systems, in *Proceedings of SIGKDD, New York* (2019), pp. 968–977
117. H. Wang, M. Zhao, X. Xie, W. Li, M. Guo, Knowledge graph convolutional networks for recommender systems, in *Proceedings of WWW, New York* (2019), pp. 3307–3313
118. R. Wang, B. Fu, G. Fu, M. Wang, Deep & cross network for ad click predictions, in *Proceedings of the ADKDD'17, New York* (2017)
119. S. Wang, Y. Wang, J. Tang, K. Shu, S. Ranganath, H. Liu, What your images reveal: exploiting visual contents for point-of-interest recommendation, in *Proceedings of WWW* (2017), pp. 391–400
120. X. Wang, X. He, M. Wang, F. Feng, T.S. Chua, Neural graph collaborative filtering, in *Proceedings of SIGIR, New York* (2019), pp. 165–174.
121. X. Wang, Y. Wang, Improving content-based and hybrid music recommendation using deep learning, in *Proceedings of the 22nd ACM International Conference on Multimedia, New York* (2014), pp. 627–636
122. C. Wu, F. Wu, M. An, T. Qi, J. Huang, Y. Huang, X. Xie, Neural news recommendation with heterogeneous user behavior, in *Proceedings of EMNLP-IJCNLP* (Association for Computational Linguistics, Hong Kong, 2019), pp. 4874–4883

123. C. Wu, F. Wu, S. Ge, T. Qi, Y. Huang, X. Xie, Neural news recommendation with multi-head self-attention, in *Proceedings of EMNLP-IJCNLP* (2019), pp. 6390–6395
124. C.Y. Wu, A. Ahmed, A. Beutel, A.J. Smola, H. Jing, Recurrent recommender networks, in *Proceedings of WSDM* (2017), pp. 495–503
125. S. Wu, W. Ren, C. Yu, G. Chen, D. Zhang, J. Zhu, Personal recommendation using deep recurrent neural networks in NetEase, in *Proceedings of ICDE* (IEEE, Piscataway, 2016), pp. 1218–1229
126. S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, T. Tan, Session-based recommendation with graph neural networks, in *Proceedings of AAAI*, vol. 33 (2019), pp. 346–353
127. Y. Wu, C. DuBois, A.X. Zheng, M. Ester, Collaborative denoising auto-encoders for top-n recommender systems, in *Proceedings of WSDM, New York* (2016), pp. 153–162
128. Y. Xian, Z. Fu, S. Muthukrishnan, G. de Melo, Y. Zhang, Reinforcement knowledge graph reasoning for explainable recommendation, in *Proceedings of SIGIR, SIGIR'19, New York* (2019), pp. 285–294
129. J. Xiao, H. Ye, X. He, H. Zhang, F. Wu, T.S. Chua, Attentional factorization machines: learning the weight of feature interactions via attention networks, in *Proceedings of IJCAI* (2017), pp. 3119–3125
130. C. Xu, M. Wu, Learning feature interactions with Lorentzian factorization machine, in *Proceedings of AAAI* (2019)
131. D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, K. Achan, Self-attention with functional time representation learning, in *Proceedings of NeurIPS* (2019), pp. 15915–15925
132. H.J. Xue, X. Dai, J. Zhang, S. Huang, J. Chen, Deep matrix factorization models for recommender systems, in *Proceedings of IJCAI* (2017), pp. 3203–3209
133. A. Yan, S. Cheng, W.C. Kang, M. Wan, J. McAuley, Cosrec: 2d convolutional neural networks for sequential recommendation, in *Proceedings of CIKM* (2019), pp. 2173–2176
134. C. Yang, L. Bai, C. Zhang, Q. Yuan, J. Han, Bridging collaborative filtering and semi-supervised learning: a neural approach for poi recommendation. in *Proceedings of SIGKDD, KDD '17, New York* (2017), pp. 1245–1254
135. J. Yang, X. Yi, D. Zhiyuan Cheng, L. Hong, Y. Li, S. Xiaoming Wang, T. Xu, E.H. Chi, Mixed negative sampling for learning two-tower neural networks in recommendations, in *Companion Proceedings of the Web Conference 2020* (2020), pp. 441–447
136. L. Yao, Y. Zhang, Y. Feng, D. Zhao, R. Yan, Towards implicit content-introducing for generative short-text conversation systems. in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing* (Association for Computational Linguistics, Copenhagen, 2017), pp. 2190–2199
137. Q. Yao, X. Chen, J.T. Kwok, Y. Li, C.J. Hsieh, Efficient neural interaction function search for collaborative filtering. in *Proceedings of WWW* (2020), pp. 1660–1670
138. R. Ying, R. He, K. Chen, P. Eksombatchai, W.L. Hamilton, J. Leskovec, Graph convolutional neural networks for web-scale recommender systems, in *Proceedings of SIGKDD, New York* (2018), pp. 974–983
139. J. You, Y. Wang, A. Pal, P. Eksombatchai, C. Rosenburg, J. Leskovec, Hierarchical temporal convolutional networks for dynamic recommender systems, in *Proceedings of WWW* (2019), pp. 2236–2246
140. W. Yu, H. Zhang, X. He, X. Chen, L. Xiong, Z. Qin, Aesthetic-based clothing recommendation, in *Proceedings of WWW, Republic and Canton of Geneva, CHE* (2018), pp. 649–658
141. F. Yuan, L. Yao, B. Benatallah, Adversarial collaborative neural network for robust recommendation, in *Proceedings of SIGIR, SIGIR'19, New York* (2019), pp. 1065–1068
142. J. Zhang, X. Shi, S. Zhao, I. King, Star-GCN: stacked and reconstructed graph convolutional networks for recommender systems, in *Proceedings of IJCAI* (AAAI Press, Palo Alto, 2019), pp. 4264–4270
143. M. Zhang, Y. Chen, Inductive matrix completion based on graph neural networks. Preprint, arXiv:1904.12058 (2019)

144. S. Zhang, H. Liu, A. Zhang, Y. Hu, C. Zhang, Y. Li, T. Zhu, S. He, W. Ou, Learning user representations with hypercuboids for recommender systems, in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining* (2020)
145. S. Zhang, Y. Tay, W. Jiang, D.c. Juan, C. Zhang, Switch spaces: learning product spaces with sparse gating. Preprint, arXiv:2102.08688 (2021)
146. S. Zhang, Y. Tay, L. Yao, A. Sun, J. An, Next item recommendation with self-attentive metric learning. in *AAAI Workshop*, vol. 9 (2019)
147. S. Zhang, L. Yao, A. Sun, Y. Tay, Deep learning based recommender system: a survey and new perspectives. ACM Comput. Surv. **52**(1), 1–38 (2019)
148. S. Zhang, L. Yao, A. Sun, S. Wang, G. Long, M. Dong, Neurec: on nonlinear transformation for personalized ranking, in *Proceedings of IJCAI* (2018), pp. 3669–3675
149. S. Zhang, L. Yao, L. Vinh Tran, A. Zhang, Y. Tay, Quaternion collaborative filtering for recommendation, in *Proceedings of IJCAI* (2019), pp. 4313–4319
150. T. Zhang, P. Zhao, Y. Liu, V.S. Sheng, J. Xu, D. Wang, G. Liu, X. Zhou, Feature-level deeper self-attention network for sequential recommendation, in *Proceedings of IJCAI* (2019), pp. 4320–4326
151. P. Zhao, H. Zhu, Y. Liu, J. Xu, Z. Li, F. Zhuang, V.S. Sheng, X. Zhou, Where to go next: a spatio-temporal gated network for next poi recommendation. in *Proceedings of AAAI*, vol. 33 (2019), pp. 5877–5884
152. X. Zhao, L. Xia, L. Zhang, Z. Ding, D. Yin, J. Tang, Deep reinforcement learning for page-wise recommendations, in *Proceedings of Recsys, RecSys '18* (Association for Computing Machinery, New York, 2018), pp. 95–103
153. X. Zhao, L. Zhang, Z. Ding, L. Xia, J. Tang, D. Yin, Recommendations with negative feedback via pairwise deep reinforcement learning, in *Proceedings of SIGKDD, KDD '18, New York* (2018), pp. 1040–1048
154. G. Zheng, F. Zhang, Z. Zheng, Y. Xiang, N.J. Yuan, X. Xie, Z. Li, DRN: a deep reinforcement learning framework for news recommendation. in *Proceedings of WWW* (2018), pp. 167–176
155. L. Zheng, V. Noroozi, P.S. Yu, Joint deep modeling of users and items using reviews for recommendation, in *Proceedings of WSDM, New York* (2017), pp. 425–434
156. C. Zhou, J. Bai, J. Song, X. Liu, Z. Zhao, X. Chen, J. Gao, Atrank: an attention-based user behavior modeling framework for recommendation, in *Proceedings of AAAI* (2018)
157. G. Zhou, X. Zhu, C. Song, Y. Fan, H. Zhu, X. Ma, Y. Yan, J. Jin, H. Li, K. Gai, Deep interest network for click-through rate prediction, in *Proceedings of SIGKDD* (2018), pp. 1059–1068
158. X. Zhou, C. Mascolo, Z. Zhao, Topic-enhanced memory networks for personalised point-of-interest recommendation, in *Proceedings of SIGKDD* (2019), pp. 3018–3028
159. L. Zou, L. Xia, P. Du, Z. Zhang, T. Bai, W. Liu, J.Y. Nie, D. Yin, Pseudo Dyna-Q: a reinforcement learning framework for interactive recommendation, in *Proceedings of WSDM, WSDM '20, New York* (2020), pp. 816–824