Explore, Filter and Distill: Distilled Reinforcement Learning in Recommendation

Ruobing Xie*
WeChat, Tencent
Beijing, China
ruobingxie@tencent.com

Shaoliang Zhang*
WeChat, Tencent
Beijing, China
modriczhang@tencent.com

Rui Wang
WeChat, Tencent
Beijing, China
rysanwang@tencent.com

Feng Xia
WeChat, Tencent
Beijing, China
xiafengxia@tencent.com

Leyu Lin
WeChat, Tencent
Beijing, China
goshawklin@tencent.com

ABSTRACT

Reinforcement learning (RL) has been verified in real-world list-wise recommendation. However, RL-based recommendation suffers from huge memory and computation costs due to its large-scale models. Knowledge distillation (KD) is an effective approach for model compression widely used in practice. However, RL-based models strongly rely on sufficient explorations on the enormous user-item space due to the data sparsity issue, which multiplies the challenges of KD with RL models. What the teacher should teach and how much the student should learn from each lesson need to be carefully designed. In this work, we propose a novel Distilled reinforcement learning framework for recommendation (DRL-Rec), which aims to improve both effectiveness and efficiency in list-wise recommendation. Specifically, we propose an Exploring and filtering module before the distillation, which decides what lessons the teacher should teach from both teachers' and students' aspects. We also conduct a Confidence-guided distillation at both output and intermediate levels with a list-wise KL divergence loss and a Hint loss, which aims to understand how much the student should learn for each lesson. We achieve significant improvements on both offline and online evaluations in a well-known recommendation system. DRL-Rec has been deployed on WeChat Top Stories for more than six months, affecting millions of users. The source codes are released in https://github.com/modriczhang/DRL-Rec.

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

recommendation; reinforcement learning; knowledge distillation

*Both authors contributed equally to this research. Ruobing Xie is the corresponding author (ruobingxie@tencent.com).

CIKM '21, November 1–5, 2021, Virtual Event, QLD, Australia
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8446-9/21/11…$15.00
https://doi.org/10.1145/3459637.3481917

ACM Reference Format:
Ruobing Xie, Shaoliang Zhang, Rui Wang, Feng Xia, and Leyu Lin. 2021. Explore, Filter and Distill: Distilled Reinforcement Learning in Recommendation. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3459637.3481917

1 INTRODUCTION

Real-world recommendation systems should provide personalized and interesting items based on user diverse preferences. To learn users' personalized preferences, recommendation algorithms should capture informative (high-order) feature interactions between users, items and contexts, which derive from million or even billion level user behaviors [44]. Recent years have witnessed the great successes of neural networks in recommendations [4], while they inevitably suffer from the huge memory and computation costs due to their large-scale models. These memory and computation costs will even multiply when dealing with *list-wise recommendation* [38], which should generate a recommended list instead of a single item for each request. To capture the list-wise rewards, sophisticated reinforcement learning (RL) based models are often conducted in real-world systems [36, 38, 43]. Considering the billion-level behaviors, lots of model compression strategies are explored to balance effectiveness, efficiency, and expenses in practice.

Knowledge distillation (KD) [12], which eases the training of large-scale neural networks by following a teacher-student paradigm, is a classical and effective approach for knowledge transfer and model compression. It transfers the knowledge of cumbersome networks (i.e., *teachers*) into smaller networks (i.e., *students*), where the student is trained from both real labels and a soft version of the teacher's outputs. KD not only makes the student smaller and faster for deployment, but also improves the training by further considering negative logits via these softened outputs [12].

There are some works that have verified knowledge distillation in recommendation [19, 20]. However, there are very few works that conduct knowledge distillation between RL-based models in recommendation. The main challenges of conducting knowledge distillation on RL for (list-wise) recommendation locate in two aspects: (1) ***what lessons should the teacher pass on to the student?*** Conventional KD in classification tasks could learn additional knowledge from all negative logits of the teacher's softmax layer, which often has less than a few hundred categories. In contrast, it is not possible (and also unnecessary) to explore all state-action space (i.e., million-level users and items in various contexts) in RL-based recommendation models, especially in the list-wise recommendation that should further consider item positions. An effective KD in RL-based recommendation should *explore* items that are most likely to be clicked, *filter* other irrelevant items, and *distill* the refined informative knowledge to the student. (2) ***How much should the student learn from each lesson?*** The state-action space in list-wise RL-based recommendation is tremendous, which makes the interactions between users and item lists extremely sparse even compared to regular recommendations. Hence, the teacher cannot perfectly predict all possible state-action rewards, especially for unsupervised instances in offline RL exploration that do not have real user feedback. A confidence-aware distillation is required.

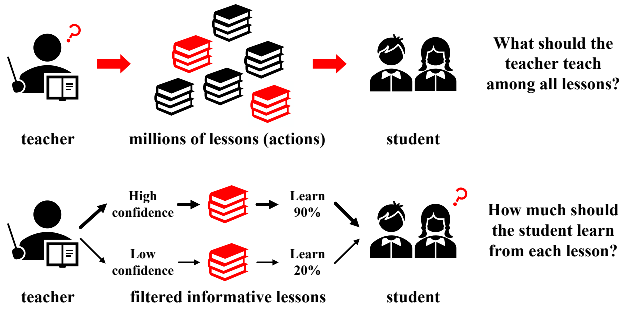
What should the teacher teach among all lessons?
teacher
millions of lessons (actions)
student
High confidence
Learn 90%
Low confidence
Learn 20%
How much should the student learn from each lesson?
teacher
filtered informative lessons
student

Figure 1: Challenges of KD in RL-based recommendation.

To address these challenges, we propose a novel **Distilled reinforcement learning framework for recommendation (DRL-Rec)**, which wants to compress RL-based models for efficiency and even improve the performances in list-wise recommendation. DRL-Rec mainly contains three modules, namely the Teacher and student module, the Exploring and filtering module, and the Confidence-guided distillation module. (1) We first adopt two value-based RL models as the teacher and student models for list-wise recommendation. (2) In the Exploring and filtering module, we propose a novel dual-led teaching strategy to decide what lessons the teacher should teach the student. Specifically, we first conduct RL explorations to retrieve a group of item candidates, and then conduct both teacher-led and student-led lessons to select top-k teacher-concerned and student-concerned items for the following distillation. This module is regarded as the filtering process before distillation, which can eliminate most impurities from irrelevant items to improve the efficiency of knowledge distillation. (3) In the Confidence-guided distillation module, we conduct knowledge distillation at both output and intermediate levels. Precisely, we design a new list-wise KL divergence loss for the output distillation, and rely on the Hint loss [23] for the intermediate distillation. We further bring in the notion of confidence according to the teacher's accuracy to facilitate the KD. The advantages of DRL-Rec are mainly located in two aspects: (1) the Exploring and filtering module smartly selects informative items for teaching, which addresses the "what to teach" problem of the teacher. (2) The Confidence-guided distillation module enables more effective distillation between RL-based models, which solves the "how much to learn" issue of the student.

In experiments, we evaluate our DRL-Rec framework on a real-world recommendation system named WeChat Top Stories. DRL-Rec has significant improvements on both recommendation accuracy and model efficiency in memory and computation. To simulate the real-world scenarios, we further evaluate DRL-Rec with an online A/B test. Moreover, we also conduct several ablation tests to verify the effectiveness of different components in our model. The contributions of DRL-Rec are concluded as follows:

- We propose a novel DRL-Rec framework for RL-based knowledge distillation in list-wise recommendation. To the best of our knowledge, we are the first to conduct knowledge distillation between RL-based models in recommendation.
- We design an Exploring and filtering module with a dual-led teaching strategy to smartly select informative items for distillation. It solves what lessons the teacher should teach in KD with RL-based models.
- We also propose a Confidence-guided distillation with list-wise KL divergences and Hint losses, which enables more efficient knowledge transfer. It addresses how much the student should learn from each lesson in DRL-Rec.
- DRL-Rec achieves significant improvements on both offline and online evaluations and model efficiency. DRL-Rec has been deployed on a recommendation scenario in WeChat Top Stories, which affects millions of users.

2 RELATED WORKS

2.1 Recommendation Systems

Real-world recommendation systems should have the ability to capture informative feature interactions. Factorization machine (FM) [22] is a widely-used method to capture second-order feature interactions via trainable latent embeddings. Wide&Deep [3] builds a general framework that contains a Wide part and a Deep part. DeepFM [9] improves the Wide part of Wide&Deep with a bi-interaction layer to model second-order feature interactions as FM. NFM [11] and AFM [34] extend FM by adding DNN and attention layers after a neural FM model. Recently, AutoInt [25], BERT4Rec [26], and AFT [10] adopt self-attention to model feature fields' and user behaviors' interactions. Pre-training is also adopted in recommendation [33, 41]. DFN [35] jointly considers explicit/implicit and positive/negative feedbacks via self-attention. AFN [4] further introduces the logarithmic neural network to smartly and efficiently find informative high-order feature interactions. Practical large-scale recommendation systems often have more than dozens of feature fields. Hence, these introduced models are essential in recommendation, which could be used for directly recommending or as a component for feature interaction modeling.

Real-world recommendation systems, especially list-wise systems, usually need to consider multiple objectives and long-term rewards such as accuracy, diversity and user activeness [36, 43]. It is straightforward to introduce reinforcement learning (RL) to address these factors. Recently, RL methods have also been verified in recommendation with both value-based (Deep Q-network (DQN)) and policy-based (Policy gradient (PG)) strategies [16, 30]. DRN [44] and FeedRec [47] apply RL to model long-term rewards of user activeness and engagement. Wang et al. [31] adopts RL for explainable recommendation. As for the list-wise recommendation, Zhao et al. [43] conducts RL for E-commerce, and Zhao et al. [42] brings the multi-goals abstraction in the high-level agent of hierarchical RL. [38] also adopts a hierarchical RL that decouples the integrated recommendation into an item recommender and a channel selector. PAPERec [36] also builds a personalized approximate Pareto-efficient recommendation, using RL to find appropriate objective weights for different instances. These list-wise RL-based models regard the item recommendation at each position as an action, and thus can model the overall list-wise rewards conveniently. Inspired by their successes, we implement list-wise RL-based models as both teacher and student in DRL-Rec, and also compare with some competitive RL-based models as baselines.

2.2 Knowledge Distillation

Knowledge distillation aims to transfer the knowledge from a cumbersome model to a small model, which is instructed by a soften Softmax probability distribution under certain temperatures [12]. Romero et al. [23] further considers the intermediate representations from the teacher as hints to improve the training of students. Chen et al. [1] combines KD and hint loss with bounded regression loss in object detection. Fukuda et al. [8] conducts KD from a teacher ensemble, distilling different aspects of knowledge from multiple sources. Yim et al. [40] considers the inner product between features from two layers to transfer knowledge. Wang et al. [32] also combines KD with adversarial training for smarter distillation selection. Recently, with the thriving of pre-training models, lots of works such as Jiao et al. [13] and Sun et al. [27] focus on compressing pre-training models via certain combinations of MSE, Hint and KL divergence losses to distill BERT.

For reinforcement learning, Teh et al. [29] distills multiple multi-task RL models into one student model. Czarnecki et al. [6] adopts KD with curriculum learning on policy gradient models. Rusu et al. [24] discusses the Q value distillation, which gives three methods including NLL, MSE and KL divergence based losses that enables KD on value-based RL. Czarnecki et al. [7] further explores the entire landscape of policy distillation via theoretical and empirical analyses. In DRL-Rec, we propose a novel list-wise KL divergence based loss with the Hint loss for distillation. We also bring in the notion of confidence in knowledge distillation.

2.3 Distillation in Recommendation

KD has also shown its power in recommendation. Tang and Wang [28] analyzes the differences and challenges in ranking distillation. To capture more information and reduce the sparsity in recommendation provides a solution with top-k items also regarded as positive instances. Zhou et al. [45] also adopts the knowledge distillation for training well-performing light networks. The teacher and student are jointly trained with a shared set of raw feature embeddings. Chen et al. [2] brings adversarial distillation in recommendation with external knowledge Pan et al. [21] designs an enhanced KD-based collaborative autoencoder for top-N recommendation. Recently, Ader [20] proposes the adaptive distilled exemplar replay that can help the continual learning in session-based recommendation. Kang et al. [14] proposes relaxed distillation for ranking. Liu et al. [19] attempts to distill multi-task teachers into a small DNN student to better model long-term rewards in recommendation. Besides model distillation, PFD [14] proposes the privileged feature distillation in E-commerce. It concentrates on making full use of informative privileged features that cannot be directly used in online via feature-based KD, such as the real-time duration in CVR prediction and the complicated time-consuming user-item feature interactions in the candidate generation module. Liu et al. [17] jointly conducts four KD tasks including label-based, feature-based, sample-based and model structure-based distillations for counterfactual recommendation. Differing from these models, we propose an Exploring and filtering module to effectively select informative instances for distillation between RL models. We also design the Confidence-guided list-wise KL divergence and Hint losses to guide student's training. To the best of our knowledge, we are the first to conduct KD between RL-based teacher and student in list-wise recommendation

3 METHODOLOGY

DRL-Rec aims to transfer knowledge from cumbersome RL models to small RL models in list-wise recommendation. We first introduce the key notions used in this work:

- **The teacher network.** We utilize *teacher* to represent the large RL-based recommendation model that will be distilled in knowledge distillation.
- **The student network.** We use *student* to indicate the small RL-based model in KD, which receives the distilled knowledge from the teacher. Without loss of generality, we implement the student with the same neural network as the teacher, which only differs in the dimensions of embeddings.
- **Filtered exploration candidate.** Knowledge distillation sometimes conducts an additional knowledge transfer on the *unlabelled data* guided by the teacher, which could enable a more sufficient training of students [18, 28]. In DRL-Rec, besides real training instances, the teacher network also instructs the student's training via informative unlabelled user-item pairs. Due to the enormous item space, these unlabelled pairs are pre-selected by an *Exploring and filtering module*. These filtered items are regarded as the *filtered exploration candidates*, which are used for the following knowledge distillation as supplements to real training instances.

Specifically, in Sec. 3.1, we first introduce the overall framework of our DRL-Rec. Second, we describe the details of our list-wise RL models and network structures in both student and teacher networks in Sec. 3.2. Third, we introduce the Exploring and filtering module in Sec. 3.3, which works as a pre-filter of the following knowledge distillation to smartly select informative unlabelled actions. Fourth, in Sec. 3.4, we discuss the Confidence-guided list-wise KL divergence and Hint losses for more effective KD in RL-based recommendation. Finally, we give the joint optimization objective in Sec. 3.5. We will show the details of the online system and deployment in Sec. 4.

3.1 Overall Framework

Fig. 2 displays the overall architecture of DRL-Rec, which mainly consists of three modules, including the Teacher and student module, the Exploring and filtering module, and the Confidence-guided

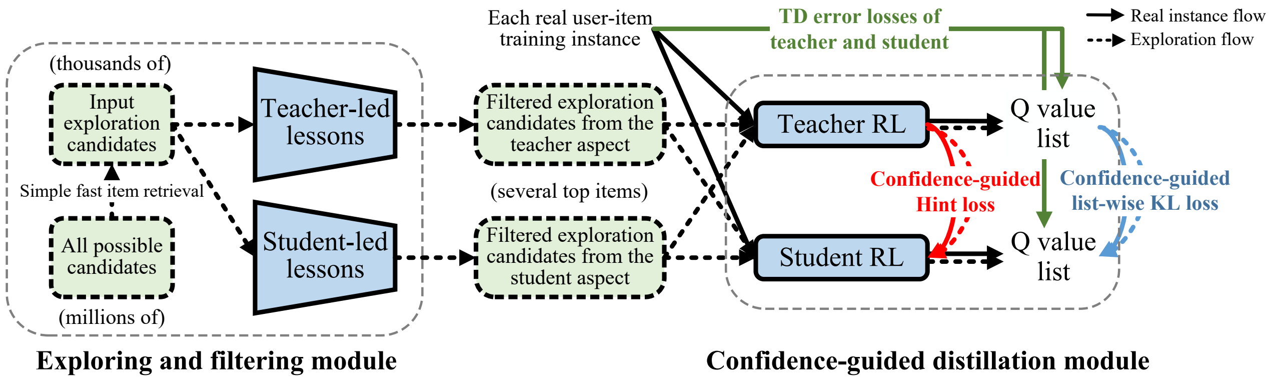
Each real user-item training instance
TD error losses of teacher and student
Real instance flow
Exploration flow
(thousands of)
Input exploration candidates
Simple fast item retrieval
All possible candidates
(millions of)
Teacher-led lessons
Student-led lessons
Filtered exploration candidates from the teacher aspect
(several top items)
Filtered exploration candidates from the student aspect
Teacher RL
Student RL
Q value list
Q value list
Confidence-guided Hint loss
Confidence-guided list-wise KL loss
Exploring and filtering module
Confidence-guided distillation module

Figure 2: Overall architecture of DRL-Rec. The solid lines represent the real instances' forward and backward flows, while the dotted lines indicate the explored unlabelled items' flows. Both real user-item instances and filtered exploration candidates are used for the following Confidence-guided distillation.

distillation module. (1) The Teacher and student module contains two value-based RL models, which attempts to predict the scores of recommending certain items in the current states. Without knowledge distillation, the teacher and student are trained separately. (2) The Exploring and filtering module focuses on exploring countless user-item combinations and filtering informative user-item pairs for the teacher to teach the student in the following KD. We design various criteria to decide which unlabelled user-item explorations should be taught from both teacher's and student's aspects. These filtered exploration candidates are utilized as additional lessons besides real training instances for the following knowledge distillation. (3) After filtering, the Confidence-guided distillation module conducts both intermediate and output distillations with the teacher's confidences being considered. We design a list-wise KL divergence based manner to distill the Q values in a list, and conduct a classical Hint loss for intermediate distillation. Besides the original RL-based ranking loss, the student can learn from both real training instances and filtered exploration candidates via KDs. In online, we only deploy the student network for recommendation to fit the time and memory requirements.

3.2 Teacher and Student Module

In this subsection, we introduce the details of our RL-based teacher and student. We follow most conventional list-wise RL-based models [36, 38, 43], which regard recommending a whole item list as a round, and generating each item as the action. Specifically, we adopt a value-based RL method Double DQN [30], and define the key RL notions used in list-wise recommendation as:

- **State** s: the state s contains information of user profiles, user historical behaviors, recommendation contexts, and previous recommended item in the list.
- **Action** a: the action a is generating an item in the list.
- **Reward** r: the reward r is measured by the click times of the current item in the recommended list.
- **Discount factor** γ: $\gamma \in [0, 1]$ measures the importance of future rewards to the current state.

The RL model sequentially generates items to form an item list for each request. For faster and better model convergence, we adopt the same neural networks for the teacher and student, which only differ in model size. It is also not difficult to conduct different RL models for the student.

3.2.1 *Network Structures of Teacher and Student.*

In list-wise recommendation, the RL states should describe not only user preferences and current item features, but also previously recommended items in the list. Therefore, differing from conventional ranking models that use click behaviors to learn user interests, DRL-Rec uses the *impression* behaviors (i.e., items being exposed to users) in the current session to model the list-level state. Precisely, inspired by [38], when recommending the t-th item in the list, we use the previous m impressed items before t to form the input feature sequence seq_t, noted as $seq_t = \{f_1, \cdots, f_m\}$.

The i-th feature $\boldsymbol{f}_i$ mainly contains four feature groups: (a) user basic information (e.g., age, gender, long-term interests), (b) recommendation contexts (e.g., timestamp, city), (c) current item features (e.g., ID, tag, category embeddings), and (d) cumulative features of the current session (e.g., cumulative item-specific information of the former $i-1$ items in the input sequence, reflecting the list-level information). Inspired by the Transformer-based feature interactions used in Song et al. [25], we divide these feature groups into n feature fields, and conduct a multi-head self-attention to capture useful feature interactions. For the i-th feature matrix $\boldsymbol{F}'_i \in \mathbb{R}^{n \times d_r}$ that consists of four feature groups, we calculate its flatten feature embedding $\boldsymbol{f}_i$ after a field-level self-attention as follows:

$$\boldsymbol{f}_i = \mathrm{Flatten}(\mathrm{Transformer}(\boldsymbol{F}'_i)), \quad \boldsymbol{f}_i \in \mathbb{R}^{nd_r}. \tag{1}$$

d_r is the dimension of raw feature fields. The Flatten$(\cdot)$ operation concatenates all field embeddings after Transformer to form a 1D embedding $\boldsymbol{f}_i$. Next, we conduct an RNN with GRU to represent the state embedding $\boldsymbol{s}_t$ as the last hidden state in GRU. We have:

$$\boldsymbol{s}_t = \mathrm{GRU}(seq_t) = \mathrm{GRU}(\{\boldsymbol{f}_1, \cdots, \boldsymbol{f}_m\}). \tag{2}$$

With the learned state embedding $\boldsymbol{s}_t$ and the target item embedding $\boldsymbol{a}_t$ (a trainable embedding), we conduct an MLP to generate the final predicted Q value q_t as follows:

$$q_t = \mathrm{MLP}(\mathrm{Concat}(\boldsymbol{s}_t, \boldsymbol{a}_t)). \tag{3}$$

In DRL-Rec, the Q value is defined as the expected return of the current and rest items in the list. We adopt the Temporal difference (TD) learning [30], which is formalized as:

$$Q(s_t, a_t) = \mathbb{E}_{s_{t+1}, r_t \sim E}[r_t + \gamma Q(s_{t+1}, a_{t+1})], \tag{4}$$

where a_t indicates the t-th recommended item, and r_t is calculated as the clicked number of the t-th item. The discount factor γ considers the position bias factors in list-wise recommendation. Items with higher ranks should be more concerned (for their higher impression rates and user experience), and thus the future rewards of low-rank items in the list are discounted via γ. Both teacher and student models share this structure with different parameters. Fig. 3 shows the detailed structure of the student network.

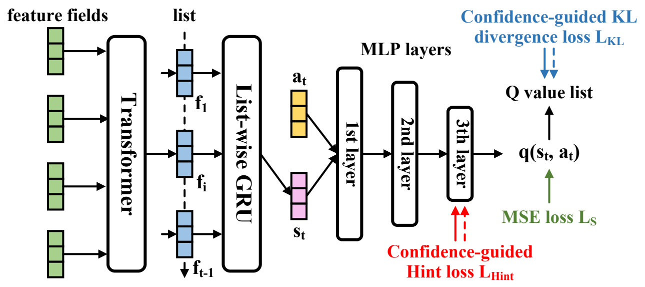

Figure 3: The network structures of the student.

3.2.2 Training Objectives of Teacher and Student. For teacher's Double DQN model training, we use the classical mean squared loss (MSE) [30] for value-based RL training as:

$$\begin{aligned} L_T &= \mathbb{E}_{s_t, r_t \sim E}[(y_t - Q(s_t, a_t|\theta_T))^2], \\ y_t &= r_t + \gamma Q(s_{t+1}, \text{argmax}_a Q(s_{t+1}, a|\theta_T)|\theta'_T). \end{aligned} \tag{5}$$

y_t indicates the t-th target Q value. θ_T is the online parameter to be trained, and θ'_T is the parameters of the target network fixed during training. Similarly, the student's MSE-based loss L_S is defined the same as that of the teacher. We have:

$$\begin{aligned} L_S &= \mathbb{E}_{s_t, r_t \sim E}[(y_t - Q(s_t, a_t|\theta_S))^2], \\ y_t &= r_t + \gamma Q(s_{t+1}, \text{argmax}_a Q(s_{t+1}, a|\theta_S)|\theta'_S). \end{aligned} \tag{6}$$

θ_S and θ'_S are the parameters of the trained and target networks. In DRL-Rec, we train the teacher and student simultaneously with L_T, L_S and two distillation losses in Sec. 3.4.

3.3 Exploring and Filtering Module

Conventional knowledge distillation models could benefit from the softmax distributions and negative logits of the teacher as supplements to the hard labels [12]. However, the action in recommendation (i.e., recommending an item) may have million-level candidates. The action combinations even multiply in list-wise scenarios for each user request. It is extremely time-consuming (even impossible) and not necessary to explore and learn from *all "negative" instances*. In real-world physical scenarios, scientists usually conduct the filtration process before distillation, which filters all irrelevant impurities that may reduce the distillation efficiency. Inspired by this, we assume that an effective KD should also filter irrelevant user-item candidates before the following KD, focusing on the high-quality informative user-item pairs in teaching. Hence, we propose the Exploring and filtering module to find what lessons (i.e., state-action pairs) the teacher should teach the student.

Specifically, we first conduct an offline exploration to generate informative user-item pairs as additional unlabelled instances beyond real instances in KD. A simple fast item retrieval model (e.g., two-tower matching model [5] or random selection) is first conducted to generate thousand-level item candidates (noted as the input exploration candidates) for each state, working as a coarse-grained filtration considering the efficiency. Next, we propose a combination of teacher-led and student-led lessons to smartly collect appropriate user-item instances from both aspects for the following distillation.

Teacher-led Lessons. The teacher-led lessons collect user-item instances that the teacher thinks are worth teaching. It selects *items with top-k Q values generated by the teacher* in Eq. (3) for the following exploration-based distillation. The lessons with the teacher's top-k Q value items make full use of the generalization ability of the teacher, which bring additional informative knowledge in KD. Note that differing from Tang and Wang [28], our teacher-led lessons distill the Q values of top-k items given by the teacher, not directly viewing them as true labels.

Student-led Lessons. The student-led lessons concentrate on instances the student thinks are worth learning. It focuses on the *items with top-k Q values predicted by the student.* In such lessons, instead of passively learning from the teacher, the student actively predicts what it thinks is appropriate and then asks the teacher for advice. In teacher-led lessons, the teacher teaches based on what it thinks will be clicked, while in student-led lessons, the teacher corrects the student's current perceptions on what the student thinks will be clicked, which also matters in online recommendation.

By combining the teacher-led and student-led lessons, the Exploring and filtering module can generate possible clicked items from both teacher's and student's opinions for a more effective distillation. It is also convenient to conduct other high-quality lessons to select items for KD, such as the teacher-student deviation-led lessons. This filtering process is essential, since most item candidates are irrelevant and inappropriate to be recommended, which may even disturb the distillation. The filtered exploration candidates provided by the Exploring and filtering module are useful supplements to real training instances, which are used in the following Confidence-guided knowledge distillation.

3.4 Confidence-guided Distillation Module

The Confidence-guided distillation module aims to distill useful knowledge from the teacher to the student, where the distilled instances derive from both real training instances and filtered exploration instances. We conduct KDs on both intermediate embeddings and output Q values in our list-wise RL scenario. We also bring in the notion of confidence according to the teacher's accuracy to further enhance the effectiveness of distillation.

3.4.1 Confidence-guided List-wise Output Distillation. Differing from Hinton et al. [12] that focuses on a classifier, the teacher is a value-based RL model predicting Q value lists, which is a difficult regression task. Inspired by Rusu et al. [24], we propose a confidence-guided list-wise output distillation. Specifically, we use the Kullback-Leibler divergence (KL) between the teacher's and student's Q value lists as our distillation objective, and then conduct distillation with temperature τ to distill the Q value lists as follows:

$$L_{KL} = \sum_{q_i \in Q} \mathrm{softmax}(\frac{\boldsymbol{q}_i^T}{\tau}) \ln \frac{\mathrm{softmax}(\frac{\boldsymbol{q}_i^T}{\tau})}{\mathrm{softmax}(\boldsymbol{q}_i^S)}. \quad (7)$$

Q indicates the overall Q value list set. $\boldsymbol{q}_i^T$ and $\boldsymbol{q}_i^S$ are the Q value embeddings of the i-th Q value list given by the teacher and student respectively, where the j-th element in $\boldsymbol{q}_i^T$ indicates the Q value of the j-th item in the i-th list. Comparing with simply using point-wise MSE loss between teacher's and student's Q values, our list-wise KL-based distillation further considers the overall distribution in the list, which helps RL modeling in list-wise scenarios. τ is the distillation temperature, which is used to adjust the sharpness of the Q value distributions in the lists.

In addition, we also introduce the notion of confidence in distillation. For each real user-item instance, the confidence c_t of the t-th item is defined via the gap between the teacher's predicted Q value q_t^T and the real Q value y_t, noted as:

$$c_t = 1 - \frac{||q_t^T - y_t||_2^2}{\eta}, \quad \eta = \max ||q_t^T - y_t||_2^2 + \epsilon. \quad (8)$$

y_t is the Q value in Eq. (5). This confidence measures the reliability of teacher's predictions, which is added to the learning rates of corresponding items as discount weights when updating the student. For filtered exploration instances, we simply set the average confidence equally for all instances, since we do not have real user feedbacks on these unlabelled instances. In distillation, to avoid the disturbances from the student, we also conduct a gradient block for the teacher as [45]. The confidence-guided KL distillation enables a more effective learning for the student via emphasized lessons.

3.4.2 Confidence-guided Intermediate Distillation. We also conduct a confidence-guided intermediate distillation following the Hint loss [23]. We conduct a point-wise distillation on the last MLP hidden embeddings in Eq. (3), which is formalized as follows:

$$L_{Hint} = \sum_{i=1}^{|T|} c_i ||\boldsymbol{h}_i^T - \boldsymbol{h}_i^S||_2^2. \quad (9)$$

T indicates the overall instances (including real and filtered exploration instances). $\boldsymbol{h}_i^T$ and $\boldsymbol{h}_i^S$ are the last hidden states of the teacher and student. c_i is the i-th confidence of current instance introduced in Eq. (8). The Hint loss with confidence also improves the distillation from the point-wise intermediate level, distilling useful information from another aspects.

3.5 Optimization Objective

In training, we combine the teacher/student RL losses L_T and L_S in Eq. (5) and Eq. (6) with the two confidence-guided distillation losses L_{KL} and L_{Hint} in Eq. (7) and Eq. (9) as our final training objective:

$$L = \lambda_1 L_T + \lambda_2 L_S + \lambda_3 L_{KL} + \lambda_4 L_{Hint}. \quad (10)$$

We empirically set $\lambda_1 : \lambda_2 : \lambda_3 : \lambda_4 = 2 : 2 : 1 : 1$ according to both offline and online performances. The teacher network is trained only via L_T, while the student network is directly influenced by L_S, L_{KL} and L_{Hint}.

4 ONLINE DEPLOYMENT

We have deployed our DRL-Rec framework on a well-known recommendation system named WeChat Top Stories for more than six months. In this section, we will introduce the online system, serving, and efficiency of our real-world list-wise system.

4.1 Online System and Serving

WeChat Top Stories is a popular recommendation system in WeChat. It has nearly million-level users and items, generating tens of millions of daily user behaviors. To model future rewards and recommendation diversity, we adopt a list-wise recommendation system. Specifically, we directly deploy the student RL model of DRL-Rec on the ranking module of a recommendation scenario, with other modules unchanged for control experiments. Inspired by [36, 38], we also adopt the online RL exploration on small online traffic to collect real user feedbacks on diversified state-action instances. The online exploration randomly recommends one of the top-k best items pre-selected by the matching module (rather than the item with the highest score) as a recommendation trial. It improves the diversity of RL explorations and thus helps RL models to discover users' potential interests. Note that the offline exploration in the Exploring and filtering module aims to select informative items as unsupervised information for KD, while the online exploration focuses on collecting diverse real user feedbacks as supervised information. Specifically, we conduct on TensorFlow PS-Worker architecture. We have 50 ps machines and 150 worker machines for training (4 cores and 6G memory). We spend an average of 2 hours completing the training. We have also tested various parameter combinations to confirm the model convergence.

4.2 Online Efficiency

DRL-Rec attempts to conduct knowledge distillation on RL-based recommendations, where the computation and memory costs are essential. In both offline training and online serving, the computation costs mainly locate in the state encoder. In training, the Exploring and filtering module should first evaluate all candidates via the teacher and the student, which is extremely time-consuming. Fortunately, the Transformer and GRU calculations at each position in Eq. (1) to Eq. (3) can be reused for all candidates, which greatly reduces the amount of computation. Moreover, a fast retrieval model (e.g., matching models or simple random selection) is adopted as a coarse filter. In online, we only need to conduct the forward Q value calculation for all item candidates (i.e., top 200 items retrieved by the previous matching module), whose time cost is related to the student network. The online time complexity is $O(kN)$, where k is the number of input item candidates of DRL-Rec, and N is the time cost of the student. The online memory and computation costs of the deployed DRL-Rec are only 49.7% and 76.7% of the teacher.

5 EXPERIMENTS

We propose DRL-Rec for distilling value-based RL models in list-wise recommendation. In this section, we conduct extensive offline and online experiments, aiming to answer the following three research questions: **RQ1**: How does DRL-Rec perform against competitive baselines in offline (see Sec. 5.4)? **RQ2**: How does DRL-Rec perform in online recommendation systems with various real-world metrics (see Sec. 5.5)? **RQ3**: What are the influence of different components in DRL-Rec (see Sec. 5.6)? **RQ4**: What are the influence of different parameters on the performance (see Sec. 5.7)?

5.1 Datasets

Since there is no large-scale dataset for list-wise recommendation and the online exploration requires real user feedbacks for RL training, we build a new dataset LRec-1B from a widely-used list-wise system WeChat Top Stories. This system generates 10 items for each user request. Precisely, we randomly select nearly 36 million users, and utilize their 1.1 billion item impression instances in 151 million lists. We split these impressed instances into a train set and a test set following the chronological order. The instance numbers of the train set and the test set are approximately set as 7 : 3 in LRec-1B. All data are preprocessed via data masking to protect user privacy. Table 1 shows the detailed statistics of LRec-1B.

Table 1: Statistics of the LRec-1B dataset.

#user	#list	#click	#impression
36,984,728	151,530,216	92,802,521	1,092,114,138

5.2 Competitors

We implement several competitive methods as our baselines, which are roughly grouped into two categories, including the classical point-wise ranking models and the list-wise RL-based models.

Point-wise Ranking Models. We implement seven representative point-wise ranking models, including FM [22], Wide&Deep [3], NFM [11], AFM [34], DeepFM [9], AutoInt [25], and AFN [4] for baselines, which are widely verified in industry. Compared to DRL-Rec, these models do not optimize the list-wise rewards in their training objectives. Precisely, we have:

- **FM [Rendle 2010].** Factorization machine (FM) is a classical model that captures the second-order feature interactions via latent vectors. It is considered as the base model in our offline evaluation.
- **Wide&Deep [Cheng et al. 2016].** Wide&Deep jointly models raw features via a LR-based Wide part, and conducts a DNN-based Deep part to learn high-order feature interactions. It is widely used in industry.
- **NFM [He and Chua 2017].** NFM uses a neural FM module followed by a DNN module to model feature interactions.
- **AFM [Xiao et al. 2017].** AFM brings in an attention layer after a neural FM layer to weight different second-order feature combinations.
- **DeepFM [Guo et al. 2017].** DeepFM adopts the Wide&Deep framework, using a neural FM layer as the Wide part.
- **AutoInt [Song et al. 2019].** AutoInt introduces the popular self-attention across feature fields, which is widely used as a fundamental module in recommendation. DRL-Rec also adopts a self-attention layer to model feature interactions.
- **AFN [Cheng et al. 2020].** AFN is a recent SOTA ranking model, which adopts the logarithmic neural network to automatically select effective high-order feature interactions.

Note that all raw features of users, items and contexts and all user historical behaviors used in DRL-Rec are also considered in these baselines for fair comparisons.

List-wise RL-based Models. We further conduct comparisons with two competitive list-wise RL-based recommendation models.

- **Teacher network**. In knowledge distillation, the teacher network is always a strong and essential competitor. Precisely, we directly adopt the teacher network introduced in Sec. 3.2, which is trained via Eq. (5) with Double DQN [30].
- **HRL-Rec [Xie et al. 2021b].** HRL-Rec is the state-of-the-art list-wise RL-based recommendation framework. It is specially designed for the integrated recommendation with heterogeneous items, which adopts a hierarchical RL model for channel selection and item recommendation. For this task, we directly use the item recommender of HRL-Rec trained with DDPG as a challenging baseline.

For fair comparisons, all RL baselines share the same input features and list-wise rewards as DRL-Rec. Moreover, the dimensions of feature embeddings in all baselines are set to the same, while the DRL-Rec model is compressed (i.e., 50%) via KD. Sec. 5.3 and Sec. 5.7 give detailed analyses on the model compression rates.

5.3 Experimental Settings

In DRL-Rec, the discount factor γ, the numbers of input exploration candidates and filtered exploration candidates in the Exploring and filtering module, and the temperature τ are the most essential parameters, which have great impacts on the final performances. We conduct a grid search and test the discount factor γ among $\{0.0, 0.2, 0.4, 0.6, 0.8\}$, the number of input candidates among $\{100, 500, 1000\}$, the number of filtered exploration candidates for KD among $\{1, 3, 5, 10\}$, and the temperature among $\{1, 2, 3, 5, 10, 20\}$. In conclusion, a larger discount factor γ indicates that DRL-Rec values more about future rewards. The discount factor γ is set to be 0.4 considering the position bias. Larger numbers of input and filtered output candidates in the Exploring and filtering module imply that more additional lessons are explored and taught. Too few candidates may be insufficient in finding informative KD lessons via explorations. However, too many lessons may also bring in tremendous computation costs and noises. Hence, we select $1,000$ candidates as inputs and top 3 filtered items as outputs for both teacher- and student- led lessons to balance effectiveness and efficiency. These top 3 filtered items are combined with the original train set for KD. In distillation, the temperature τ is set as 10 according to the offline and online performances.

For the Teacher and student module, the teacher and student share the same neural network structure and feature fields. The compression rate (i.e., student model size / teacher model size) is the key parameter in KD. We have tested various compression rates (see Sec. 5.7), and set it as 50% balancing both effectiveness and efficiency. Precisely, we have more than 15 feature fields in online. Each feature field's dimensions d_r of teacher and student are 16 and 8, and other hidden embeddings also follow the 50% compression rate. The maximum length m of the impression sequence is 10. We conduct a 3-layer MLP to get q_t for both the teacher and student.

In optimization, we jointly train both teacher and student simultaneously, and train DRL-Rec once over all training instances considering the efficiency. We deploy DRL-Rec on a value-based DQN model, since it is more convenient and stable for DQN to converge in knowledge distillation. We use Adam [15] for optimization with the batch size set as 256. We conduct a grid search for parameter selection. All models share the same features (if beneficial) and experimental settings. More parameter analyses are in Sec. 5.7.

5.4 Offline Evaluation (RQ1)

We first evaluate DRL-Rec with other baselines on LRec-1B to show its effectiveness on recommendation accuracy.

5.4.1 Evaluation Protocols. In offline evaluation, we first score all instances with all models, and then use the classical metrics Area Under Curve (AUC) and RelaImpr widely-utilized in real-world (list-wise) recommendation [36, 38, 46] to evaluate models. AUC measures the probability of clicked instances having higher ranks than unclicked instances, while RelaImpr [39] reflects the relative improvements of current models over the base model (i.e., FM in offline evaluations). In offline evaluation, we conduct 3 runs and report the average results for all models.

Table 2: Results of offline evaluation on LRec-1B. Note that the online memory cost and computation cost of DRL-Rec are only 49.7% **and** 76.7% **of the teacher network.**

model	AUC	RelaImpr
FM (Rendle 2010)	0.7321	0.00%
Wide&Deep (Cheng et al. 2016)	0.7413	3.96%
NFM (He and Chua 2017)	0.7384	2.71%
AFM (Xiao et al. 2017)	0.7382	2.63%
DeepFM (Guo et al. 2017)	0.7426	4.52%
AutoInt (Song et al. 2019)	0.7463	6.12%
AFN (Cheng et al. 2020)	0.7459	5.95%
The teacher network	0.7619	12.84%
HRL-Rec (Xie et al. 2021b)	0.7625	13.10%
DRL-Rec (ours)	**0.7653**	**14.30%**

5.4.2 Experimental Results. Table 2 shows the results of all models, from which we can observe that:

(1) DRL-Rec achieves significant improvements on AUC over all representative point-wise and list-wise RL-based baselines. We should highlight that the improvements of DRL-Rec are significant (significance level $\alpha = 0.01$) with our billion-level dataset. Based on the 50% compression rate, the online memory cost and computation cost of DRL-Rec are only 49.7% and 76.7% of the teacher network. It indicates that DRL-Rec has successfully learned informative knowledge from the large-scale teacher with a smaller size, which can jointly improve both effectiveness and efficiency.

(2) The advantages of DRL-Rec over simple knowledge distillation mainly locate in two aspects: (a) the Exploring and filtering module can smartly select informative lessons from tremendous action space, which solves the “what the teacher should teach” issue respecting the wishes of both teachers and students. These high-quality explorations bring in additional training opportunities for the student. (b) The Confidence-guided list-wise KL loss and Hint loss enable a more effective knowledge distillation via high-quality instances according to the teacher’s accuracy, which addresses the “how much the student should learn” for each KD lesson. In Sec. 5.6 and Sec. 5.7, we will give detailed analyses on the effectiveness of different components and parameters.

(3) It is astonishing that DRL-Rec (i.e., the distilled student network) even outperforms the teacher network which has the same neural network, features, and RL method with a twice model size. It is because that: (a) DRL-Rec jointly uses both real user-item instances and filtered exploration candidates in knowledge distillation. These filtered exploration candidates could be viewed as additional unlabelled training lessons for the student. The student is learned from (i) direct training with real instances, (ii) KL/Hint losses with real instances, and (iii) KL/Hint losses with top-3 filtered exploration candidates, while the teacher only learns from real instances. (b) The Confidence-guided KD further improves the knowledge transfer, where the student can concentrate more on the correct experiences of the teacher. (c) The list-wise KL divergence loss in KD further considers the overall reward distributions in the recommended list. We also show the results of students with different compression rates in Sec. 5.7.

5.5 Online A/B Tests (RQ2)

To evaluate DRL-Rec on online scenarios, we further conduct an online A/B test on a real-world list-wise recommendation scenario in WeChat Top Stories.

5.5.1 Evaluation Protocols. Following Sec. 4, we deploy DRL-Rec and the online baseline (i.e., the teacher network) in the ranking module with other modules unchanged. We use the average click number per capita (ACN) as the main metric. ACN is a practical and widely-used online metric, which could jointly measure both recommendation accuracy and user activeness [37]. Besides, we also report the online serving computation cost per request and the online memory cost. We conduct the online A/B tests for 5 days, affecting nearly 2 million users.

Table 3: Online A/B test on WeChat Top Stories.

metrics	ACN	computation cost	memory cost
DRL-Rec	+1.38%	-23.3%	-50.3%

5.5.2 Experimental Results. Table 3 shows the improvement percentages of DRL-Rec, from which we can find that:

(1) DRL-Rec significantly outperforms the online baseline on ACN with the significance level $\alpha = 0.05$, which indicates that DRL-Rec is capable of improving both accuracy and efficiency in online scenarios. It reconfirms that DRL-Rec could capture informative knowledge from the teacher and learn to be better via the exploring, filtering, and distillation training framework.

(2) Real-world recommendation systems should balance effectiveness, efficiency, and time/memory expenses in online deployment. Comparing with the original teacher network, DRL-Rec achieves 1.38% ACN improvement with only 50% memory cost and 77% computation cost. The time and memory surplus could be used for more complex algorithms, high-order features, or multiple objectives to improve recommendation accuracy and user experience.

5.6 Ablation Tests (RQ3)

To verify that different components are essential in DRL-Rec, we implement seven ablation versions with different model settings.

5.6.1 Evaluation Protocols. Precisely, we categorize these ablation versions into three groups. The first group evaluates the two essential contributions of DRL-Rec, namely the Exploring and filtering module and the Confidence-guided KD. The second group tests the effectiveness of two KD losses, namely the list-wise KL divergence loss in Eq. (7) and the Hint loss in Eq. (9). The third group attempts to demonstrate the necessity of three technics in the neural network, including the sequence modeling, the self-attention based feature interaction modeling, and the consideration of future rewards.

5.6.2 Experimental Results. From Table 4 we can find that:

(1) The first ablation group indicates that both the Exploring and filtering module and the Confidence-guided manner are beneficial in the knowledge distillation of RL models. The improvements of these modules are significant. Besides the performance improvements, the Exploring and filtering module filters most irrelevant exploration candidates, which greatly reduces the computation cost. Moreover, the Confidence-guided manner also plays an important role in ensuring a stable RL training, which is essential in daily model updates of online systems.

(2) The second ablation group represents that both the intermediate Hint loss and the output list-wise KL divergence loss are essential in KD. The Hint loss focuses on the point-wise knowledge transfer between hidden embeddings after MLP, while the KL divergence loss concentrates on distilling the knowledge of list-wise Q value distributions. Both knowledge distillation losses at intermediate and output levels can cooperate well with each other.

(3) In the third ablation group, we conduct an average pooling to replace the GRU in Eq. (3) and another average pooling to replace the self-attention in Eq. (1) for ablation tests. We find that the GRU and self-attention in DRL-Rec's network are necessary. Moreover, the discount factor $\gamma = 0$ equals that DRL-Rec does not consider any future rewards in list-wise recommendation. This ablation result indicates that considering long-term list-wise rewards is important in real-world scenarios.

Table 4: Ablation tests on LRec-1B dataset.

Task	AUC	RelaImpr
DRL-Rec	**0.7653**	**14.30%**
– Exploring and filtering	0.7622	12.97%
– Confidence in KD	0.7611	12.49%
– KL loss	0.7614	12.62%
– Hint loss	0.7629	13.27%
– GRU	0.7525	8.79%
– self-attention	0.7598	11.93%
– future rewards ($\gamma = 0$)	0.7600	12.02%

5.7 Model Analyses (RQ4)

We further conduct two model analyses to investigate the impacts of different key parameters in DRL-Rec.

5.7.1 Analysis on Compression Rate. We first evaluate DRL-Rec with different compression rates (i.e., student model size / teacher model size) to evaluate the robustness of DRL-Rec. Fig. 4 (a) shows the results, and we can find that: (1) in general, the AUC will first keep stable with the compression rate set from 87.5% to 50.0%, and then decrease sharply from 50.0% to 25.0%. It is natural that the AUC (and the computation cost) will decrease as the compression rate decreases, since the student has smaller parameter sizes. (2) In online and offline evaluations, we set the compression rate as 50.0%, which has a good balance between effectiveness and efficiency. (3) The robustness of DRL-Rec is also verified with different compression rates, which confirms the feasibility in online deployment.

5.7.2 Analysis on the Number of Filtered Exploration Items. In the Exploring and filtering module, the numbers of input items and filtered exploration candidates are also essential in DRL-Rec. We have tested various combinations of different input item and filtered exploration candidate sizes in the Exploring and filtering module. Fig. 4 (b) gives the parameter analysis of different filtered exploration item numbers. We can know that: (1) the AUC will first increase and then decrease as the number of top-k filtered items increases from 1 to 10. DRL-Rec achieves the best performance when we select top-3 filtered items. (2) It is because that too few filtered items will lead to an insufficient exploration of distillation on unlabelled data, while too many filtered items will bring in distillation noises. (3) We also evaluate DRL-Rec with different numbers of input exploration candidates. We find that too few inputs are insufficient, and choose 1,000 input candidates to balance effectiveness and efficiency.

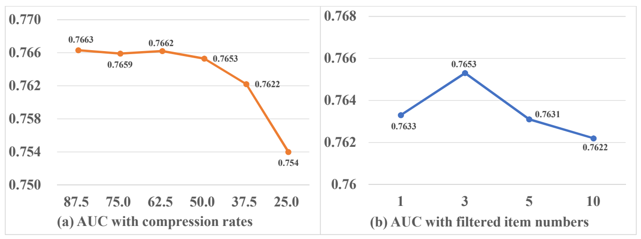

Figure 4: Results of two parameter analyses: (a) AUC with different compression rates. (b) AUC with different numbers of filtered exploration items.

6 CONCLUSION AND FUTURE WORK

In this work, we propose a novel DRL-Rec framework for knowledge distillation between RL-based models in list-wise recommendation. We design a novel Exploring and filtering module to learn what the teacher should teach from both teacher's and student's aspects. We also conduct a Confidence-guided distillation with both list-wise KL divergence and Hint losses to address how much the student should learn. DRL-Rec achieves significant offline and online improvements, and has been deployed on WeChat Top Stories.

In the future, we will continue to polish the compression rate and performances of DRL-Rec via more sophisticated knowledge distillation methods or paths between RL models. We will also explore deeper into the Exploring and Filtering module to discover more effective exploration and distillation directions.

REFERENCES

[1] Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. 2017. Learning efficient object detection models with knowledge distillation. In *Proceedings of NLPS.*

[2] Xu Chen, Yongfeng Zhang, Hongteng Xu, Zheng Qin, and Hongyuan Zha. 2018. Adversarial distillation for efficient recommendation with external knowledge. *TOIS* (2018).

[3] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the DLRS workshop.*

[4] Weiyu Cheng, Yanyan Shen, and Linpeng Huang. 2020. Adaptive Factorization Network: Learning Adaptive-Order Feature Interactions. In *Proceedings of AAAI.*

[5] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of RecSys.*

[6] Wojciech Marian Czarnecki, Siddhant M Jayakumar, Max Jaderberg, Leonard Hasenclever, Yee Whye Teh, Simon Osindero, Nicolas Heess, and Razvan Pascanu. 2018. Mix&match-agent curricula for reinforcement learning. In *Proceedings of ICML.*

[7] Wojciech Marian Czarnecki, Razvan Pascanu, Simon Osindero, Siddhant M Jayakumar, Grzegorz Swirszcz, and Max Jaderberg. 2019. Distilling policy distillation. *Proceedings of AISTATS.*

[8] Takashi Fukuda, Masayuki Suzuki, Gakuto Kurata, Samuel Thomas, Jia Cui, and Bhuvana Ramabhadran. 2017. Efficient Knowledge Distillation from an Ensemble of Teachers.. In *Proceedings of Interspeech.*

[9] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. In *Proceedings of IJCAI.*

[10] Xiaobo Hao, Yudan Liu, Ruobing Xie, Kaikai Ge, Linyao Tang, Xu Zhang, and Leyu Lin. 2021. Adversarial Feature Translation for Multi-domain Recommendation. In *Proceedings of KDD.*

[11] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *Proceedings of SIGIR.*

[12] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2014. Distilling the knowledge in a neural network. In *Proceedings of NIPS.*

[13] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling bert for natural language understanding. In *Findings of EMNLP.*

[14] SeongKu Kang, Junyoung Hwang, Wonbin Kweon, and Hwanjo Yu. 2020. DE-RRD: A Knowledge Distillation Framework for Recommender System. In *Proceedings of CIKM.*

[15] Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR.*

[16] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2016. Continuous control with deep reinforcement learning. In *Proceedings of ICLR.*

[17] Dugang Liu, Pengxiang Cheng, Zhenhua Dong, Xiuqiang He, Weike Pan, and Zhong Ming. 2020. A General Knowledge Distillation Framework for Counterfactual Recommendation via Uniform Data. In *Proceedings of SIGIR.*

[18] Pengpeng Liu, Irwin King, Michael R Lyu, and Jia Xu. 2019. Ddflow: Learning optical flow with unlabeled data distillation. In *Proceedings of AAAI.*

[19] Xi Liu, Li Li, Ping-Chun Hsieh, Muhe Xie, Yong Ge, and Rui Chen. 2020. Developing Multi-Task Recommendations with Long-Term Rewards via Policy Distilled Reinforcement Learning. *arXiv preprint arXiv:2001.09595* (2020).

[20] Fei Mi, Xiaoyu Lin, and Boi Faltings. 2020. Ader: Adaptively distilled exemplar replay towards continual learning for session-based recommendation. In *Proceedings of RecSys.*

[21] Yiteng Pan, Fazhi He, and Haiping Yu. 2019. A novel enhanced collaborative autoencoder with knowledge distillation for top-N recommender systems. *Neurocomputing* (2019).

[22] Steffen Rendle. 2010. Factorization machines. In *Proceedings of ICDM.*

[23] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2015. Fitnets: Hints for thin deep nets. In *Proceedings of ICLR.*

[24] Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. 2015. Policy distillation. *arXiv preprint arXiv:1511.06295* (2015).

[25] Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. 2019. Autoint: Automatic feature interaction learning via self-attentive neural networks. In *Proceedings of CIKM.*

[26] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *Proceedings of CIKM.*

[27] Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. Mobilebert: a compact task-agnostic bert for resource-limited devices. In *Proceedings of ACL.*

[28] Jiaxi Tang and Ke Wang. 2018. Ranking distillation: Learning compact ranking models with high performance for recommender system. In *Proceedings of KDD.*

[29] Yee Teh, Victor Bapst, Wojciech M Czarnecki, John Quan, James Kirkpatrick, Raia Hadsell, Nicolas Heess, and Razvan Pascanu. 2017. Distral: Robust multitask reinforcement learning. In *Proceedings of NIPS.*

[30] Hado Van Hasselt, Arthur Guez, and David Silver. 2016. Deep reinforcement learning with double q-learning. In *Proceedings of AAAI.*

[31] Xiting Wang, Yiru Chen, Jie Yang, Le Wu, Zhengtao Wu, and Xing Xie. 2018. A reinforcement learning framework for explainable recommendation. In *Proceedings of ICDM.*

[32] Xiaojie Wang, Rui Zhang, Yu Sun, and Jianzhong Qi. 2018. Kdgan: Knowledge distillation with generative adversarial networks. In *Proceedings of NLPS.*

[33] Chaojun Xiao, Ruobing Xie, Yuan Yao, Zhiyuan Liu, Maosong Sun, Xu Zhang, and Leyu Lin. 2021. UPRec: User-Aware Pre-training for Recommender Systems. *arXiv preprint arXiv:2102.10989* (2021).

[34] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional factorization machines: Learning the weight of feature interactions via attention networks. In *Proceedings of IJCAI.*

[35] Ruobing Xie, Cheng Ling, Yalong Wang, Rui Wang, Feng Xia, and Leyu Lin. 2020. Deep Feedback Network for Recommendation.. In *Proceedings of IJCAI.*

[36] Ruobing Xie, Yanlei Liu, Shaoliang Zhang, Rui Wang, Feng Xia, and Leyu Lin. 2021. Personalized Approximate Pareto-Efficient Recommendation. In *Proceedings of WWW.*

[37] Ruobing Xie, Zhijie Qiu, Jun Rao, Yi Liu, Bo Zhang, and Leyu Lin. 2020. Internal and Contextual Attention Network for Cold-start Multi-channel Matching in Recommendation. In *Proceedings of IJCAI.*

[38] Ruobing Xie, Shaoliang Zhang, Rui Wang, Feng Xia, and Leyu Lin. 2021. Hierarchical Reinforcement Learning for Integrated Recommendation. In *Proceedings of AAAI.*

[39] Ling Yan, Wu-Jun Li, Gui-Rong Xue, and Dingyi Han. 2014. Coupled group lasso for web-scale ctr prediction in display advertising. In *Proceedings of ICML.*

[40] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. 2017. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *Proceedings of CVPR.*

[41] Zheni Zeng, Chaojun Xiao, Yuan Yao, Ruobing Xie, Zhiyuan Liu, Fen Lin, Leyu Lin, and Maosong Sun. 2021. Knowledge transfer via pre-training for recommendation: A review and prospect. *Frontiers in big Data* (2021).

[42] Dongyang Zhao, Liang Zhang, Bo Zhang, Lizhou Zheng, Yongjun Bao, and Weipeng Yan. 2020. MaHRL: Multi-goals Abstraction Based Deep Hierarchical Reinforcement Learning for Recommendations. In *Proceedings of SIGIR.*

[43] Xiangyu Zhao, Long Xia, Liang Zhang, Zhuoye Ding, Dawei Yin, and Jiliang Tang. 2018. Deep reinforcement learning for page-wise recommendations. In *Proceedings of RecSys.*

[44] Guanjie Zheng, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhenhui Li. 2018. DRN: A deep reinforcement learning framework for news recommendation. In *Proceedings of WWW.*

[45] Guorui Zhou, Ying Fan, Runpeng Cui, Weijie Bian, Xiaoqiang Zhu, and Kun Gai. 2018. Rocket launching: A universal and efficient framework for training well-performing light net. In *Proceedings of AAAI.*

[46] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of KDD.*

[47] Lixin Zou, Long Xia, Zhuoye Ding, Jiaxing Song, Weidong Liu, and Dawei Yin. 2019. Reinforcement Learning to Optimize Long-term User Engagement in Recommender Systems. In *Proceedings of KDD.*