Real-time Relevant Recommendation Suggestion

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

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

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

Zhihong Yang
WeChat, Tencent
Beijing, China
harperyang@tencent.com

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

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

ABSTRACT

Users of recommendation systems usually focus on one topic at a time. When finishing reading an item, users may want to access more relevant items related to the last read one as extended reading. However, conventional recommendation systems are hard to provide the continuous extended reading function of these relevant items, since the main recommendation results should be diversified. In this paper, we propose a new task named recommendation suggestion, which aims to (1) predict whether users want extended reading, and (2) provide appropriate relevant items as suggestions. These recommended relevant items are arranged in a relevant box and instantly inserted below the clicked item in the main feed. The challenge of recommendation suggestion on relevant items is that it should further consider semantic relevance and information gain besides CTR-related factors. Moreover, the real-time relevant box insertion may also harm the overall performance when users do not want extended reading. To address these issues, we propose a novel Real-time relevant recommendation suggestion (R3S) framework, which consists of an Item recommender and a Box trigger. We extract features from multiple aspects including feature interaction, semantic similarity and information gain as different experts, and propose a new Multi-critic multi-gate mixture-of-experts (M3oE) strategy to jointly consider different experts with multi-head critics. In experiments, we conduct both offline and online evaluations on a real-world recommendation system with detailed ablation tests. The significant improvements in item/box related metrics verify the effectiveness of R3S. Moreover, we have deployed R3S on WeChat Top Stories, which affects millions of users. The source codes are in https://github.com/modriczhang/R3S.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

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

KEYWORDS

recommendation suggestion; recommender system; relevant recommendation

ACM Reference Format:
Ruobing Xie, Rui Wang, Shaoliang Zhang, Zhihong Yang, Feng Xia, and Leyu Lin. 2021. Real-time Relevant Recommendation Suggestion. In *Proceedings of the Fourteenth ACM International Conference on Web Search and Data Mining (WSDM ’21), March 8–12, 2021, Virtual Event, Israel.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3437963.3441733

1 INTRODUCTION

With the exponential growth of information, how to efficiently get useful information has attracted great attention. Search and recommendation are two complementary methods for active and passive information acquisition. Search helps users to actively get the exact information related to their queries [5]. When users do not know what they might be interested in, recommendation is used to predict user’s potential interests and provide appropriate items via user profiles and behaviors [24].

Users in recommendation usually focus on one topic (e.g., a football match or a news event) at a time. When finishing reading an item, users may want to actively receive more **relevant items** of the same topic if they are satisfied with the finished one. To achieve this, recommendation systems should smartly capture and emphasize user’s current preferences to provide relevant items *timely* and *explicitly*. However, in conventional recommendation system, users can only passively receive information, lacking effective and instant feedback mechanisms (e.g., queries in search) to actively interact with the system. Moreover, real-world recommendation item feeds should be diversified to alleviate homogenization, which makes it extremely inconvenient for continuous extended reading on relevant topics [27]. In this work, we attempt to implement the **extended reading** function of relevant items in recommendation, which could guide users to conduct high-quality deep reading and improve users’ experience and stickiness. The intuitive idea is to encode the activeness of search (actively searching for related contents) into the passive information acquisition of recommendation.

In search, query suggestion is proposed to predict user’s search intents and recommend relevant queries related to the entered query, which can be viewed as a query recommendation in search [12]. Similar to query suggestion, we propose a novel task named **recommendation suggestion** for extended reading in recommendation. It aims to predict user’s intents on extended reading and

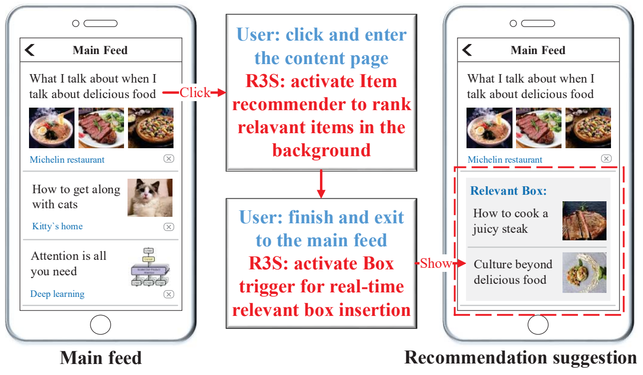
Main Feed
What I talk about when I talk about delicious food
Michelin restaurant
How to get along with cats
Kitty's home
Attention is all you need
Deep learning
Click
User: click and enter the content page
R3S: activate Item recommender to rank relavant items in the background
User: finish and exit to the main feed
R3S: activate Box trigger for real-time relevant box insertion
Show
Relevant Box:
How to cook a juicy steak
Culture beyond delicious food
Main feed
Recommendation suggestion

Figure 1: An example of recommendation suggestion for relevant items in WeChat Top Stories. The right screen shows the product form of real-time relevant recommendation suggestion. The relevant items are arranged in the relevant box, which is inserted instantly below the clicked item.

recommend appropriate relevant items related to the current clicked item, which can be viewed as a relevant item search in recommendation. The item a user has just clicked is regarded as the query of relevant recommendation in the user's subconscious. Fig. 1 shows an example of a real-world recommendation suggestion system. When a user clicks, reads and finishes an item, recommendation suggestion models calculate whether and what relevant items should be displayed to the user. In order to improve user's awareness of recommendation suggestion, we rank and arrange these relevant items in a **relevant box** and conduct a **real-time box insertion** attached to the clicked item in the main feed. In this case, recommendation suggestion can respond to the users' demands for extended reading instantly, explicitly and friendly, with well-structured relevant items listed in the relevant box.

Differing from conventional recommendation task, recommendation suggestion has two additional challenges: (1) recommendation suggestion should jointly consider multiple factors including CTR-oriented objectives, semantic relevance and information gain between clicked items and relevant candidates. It is not easy to balance these multi-aspect factors, since different users may have various preferences on these factors. (2) The real-time box insertion brings in opportunity costs, since the original recommended items in the main feed will be delayed and may not even be impressed to users (e.g., all items below the clicked item, including the cat and attention articles in Fig 1, are delayed by the real-time box insertion). Such opportunity costs (i.e., *delay costs*) should be carefully considered to improve the overall user experience.

To address these issues, we propose a novel **Real-time relevant recommendation suggestion (R3S)** framework, which aims to recommend relevant items of the clicked items with a real-time relevant box insertion. The R3S framework mainly consists of two modules. First, an *Item recommender* is used to retrieve and rank relevant items with the clicked item regarded as a "query". We propose a new **Multi-critic multi-gate mixture-of-experts (M3oE)** strategy to jointly consider multiple experts including feature interaction, semantic relevance and information gain from different critics' aspects. Next, a *Box trigger* is conducted to decide whether to display the relevant box with M3oE and extra features, working as a quality inspector to avoid too much negative disturbance for the main feed. Differing from the Item recommender, Box trigger should further consider user's satisfaction on clicked items and delay costs caused by real-time box insertion. The two-step architecture, multi-factor modeling and M3oE solve the challenges of multi-aspect factors and delay costs, which enable R3S to improve item-level, box-level and overall recommendation performances.

In experiments, we conduct extensive evaluations to verify the effectiveness of R3S on both item- and box- level metrics. Moreover, we also deploy R3S on a widely-used recommendation system WeChat Top Stories, and conduct an online A/B test to evaluate R3S via millions of users. R3S achieves significant improvements in all evaluations compared with competitive baselines. We further conduct detailed ablation tests to better understand different components in R3S. The main contributions are concluded as follows:

- We propose a novel recommendation suggestion task, which recommends relevant items related to the clicked items and conducts a real-time box insertion for instant and explicit display. It enables the extended reading function in practice.
- We creatively design an R3S framework containing the Item recommender and the Box trigger. Specifically, we design two neural networks specially for modeling semantic similarity and information gain. We also propose the Multi-critic multi-gate mixture-of-experts strategy to consider multi-aspect factors including feature interaction, semantic relevance and information gain with multiple critics.
- Both offline and online evaluations show that our R3S system achieves the best performances on item-level, box-level and overall recommendation metrics. The ablation tests verify the effectiveness of different components in R3S.
- We have deployed R3S on a well-known recommendation system named WeChat Top Stories, which affects millions of users. The practical usefulness has been verified.

2 RELATED WORKS

Recommendation systems. Classical recommendation methods often utilize Matrix factorization (MF) [11] or Collaborative filtering (CF) [3, 20] to capture user interests from user-item interactions or item similarities. To prevent data sparsity and feature engineering, Factorization machine (FM) [18] is proposed to model second-order feature interactions with latent vectors, which has been widely used in industry. To better model feature interactions, FNN [35] replaces manual features with neural networks. Wide&Deep [6] creatively uses both Wide and Deep components simultaneously for memorization and generalization abilities. DeepFM [8], AFM [28], NFM [9], AFN [7] and DFN [29] combine neural FM layers with DNN and attention layers. DCN [25] and xDeepFM [13] are designed to capture high-order interactions. AutoInt [23] and ICAN [30] further bring in self-attention for comprehensive feature interactions. AutoFIS [15] automatically identify essential feature interactions for FM based models. DIN [36] and BERT4Rec [24] also bring attention mechanism to session-based recommendation. Inspired by the great successes, R3S also relies on deep neural models to extract different types of feature interactions.

Semantics in Recommendation. Semantic relevance and information gain are two essential factors for relevant recommendation. Some efforts have explored the usage of semantic similarities in recommendation. Capelle et al. [4] proposes several WordNet-based semantic similarities for news recommendation. Al-Hassan et al. [1] and [19] implement semantic-enhanced hybrid recommender systems with ontology-based semantic similarity. [31] models user-item similarities with their tag-based semantic relationships. [16] considers heterogeneous interactions including tag-related information. [26] also models news- and word-level attention in news recommendation. The "more like this“ features have also been studied in search [14, 22]. The information gain is usually considered in diversified recommendation in the form of the novelty [34] or the increments of taxonomy [37]. Multi-objective optimization and multi-task learning are also effective methods that can be used to jointly consider semantic relevance and clicks [2, 21]. However, R3S mainly focuses on CTR-oriented objectives, where semantic relevance and information gain are regarded as features (not optimization objectives). Hence, it is not very suitable to directly adapt Multi-objective optimization to recommendation suggestion task. In R3S, we propose two neural networks to highlight both semantic relevance and information gain as experts. We do not use sentence matching models [10] as baselines, since recommendation suggestion should further consider user preferences and information gain beyond similarity.

3 METHODOLOGY

We propose an R3S framework for recommendation suggestion. In this section, we first define the goals, notions and architecture of R3S, and then introduce both Item recommender and Box trigger.

3.1 Problem Formulation

Recommendation suggestion is a new recommendation task, which aims to provide user-interested, informative, and relevant items related to user's current clicked item for extended reading. We first define the notions used in our R3S framework as follows:

- **Seed**. The current clicked item is regarded as the seed for each recommendation suggestion request.
- **Relevant item**. The relevant items are the items that share related topics/events with the seed item, which are the candidates of recommendation suggestion.
- **Relevant box**. The recommended relevant items are arranged in a relevant box shown in Fig. 1. The relevant box is inserted instantly below the clicked seed when users finish reading items and exit to the main feed.
- **Delay cost**. The real-time box insertion will inevitably delay all items below the seed and reduce their impression, which may harm the overall performance. The potential opportunity cost is regarded as the delay cost.

The R3S framework aims to (1) predict whether users want extended reading of *relevant items* when they have read and finished the *seed* item (solved by the Box trigger in Sec. 3.4), and (2) provide high-quality *relevant items* related to the *seed* and arrange them in the *relevant box* for real-time relevant box insertion in the main feed (solved by the Item recommender in Sec. 3.3).

3.2 Overall Architecture

R3S mainly contains two modules, namely the Item recommender and the Box trigger. The Item recommender aims to provide relevant items of the seed to fill the relevant box. It first conducts a fast retrieval to generate a relevant item candidate pool, and then builds three sub-networks as experts to model feature interactions with self-attention, semantic relevance and information gain. The M3oE strategy is also proposed to jointly consider multiple experts with multi-head critics. Next, the Box trigger works as a quality inspector to determine whether it is beneficial to insert the relevant box (containing recommended relevant items) into the main feed instantly and explicitly. User's satisfaction on seeds and the potential delay costs caused by all delayed item below the relevant box are further considered in Box trigger. Finally, we conduct different objectives for Item recommender and Box trigger.

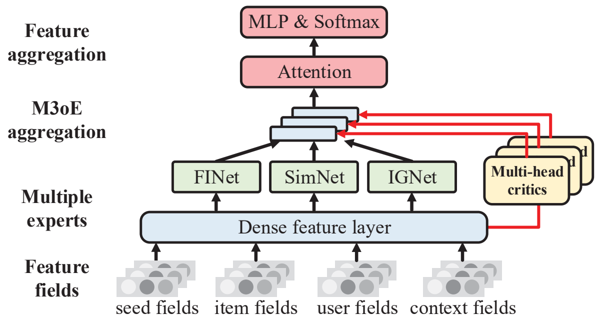

Figure 2: Overall architecture of the Multi-critic multi-gate mixture-of-experts (M3oE) strategy in R3S.

3.3 Item Recommender

Item recommender focuses on providing relevant items of the seeds that users may be interested in. The inputs are categorized into four groups as F^S, F^I, F^U, F^C, indicating the *seed*, *target item*, *user*, and *recommendation contexts*. Following [8], we group similar types of features into *feature fields*, where each field belongs to either continuous field F_{con} (e.g., age) or categorical field F_{cat} (e.g., tag/topic). We use a trainable lookup function $\mathbf{f}_i = L(f_i)$ to project a sparse feature f_i to a d-dimensional dense feature vector $\mathbf{f}_i$. Finally, we have four feature group vectors $\mathbf{f}^S$, $\mathbf{f}^I$, $\mathbf{f}^U$ and $\mathbf{f}^C$ for seed, target item, user and contexts respectively, where each feature group contains k feature fields. For example, $\mathbf{f}^S$ is the concatenation of all seed-related feature fields $\{\mathbf{f}_1^S, \cdots, \mathbf{f}_k^S\}$ defined as follows:

$$\mathbf{f}^S = \text{Concat}(\mathbf{f}_1^S, \cdots, \mathbf{f}_k^S). \tag{1}$$

The largest difference between conventional recommendation and recommendation suggestion is that the seed and recommended item should be relevant. We argue that besides classical feature interaction for CTR-oriented objectives, the semantic relevance and information gain brought by new recommended relevant items should also be considered. Moreover, the feature interactions between seed and target item should be further highlighted. Hence, we propose a novel **Multi-critic multi-gate mixture-of-experts (M3oE)** strategy, which (1) builds three sub-networks as experts to

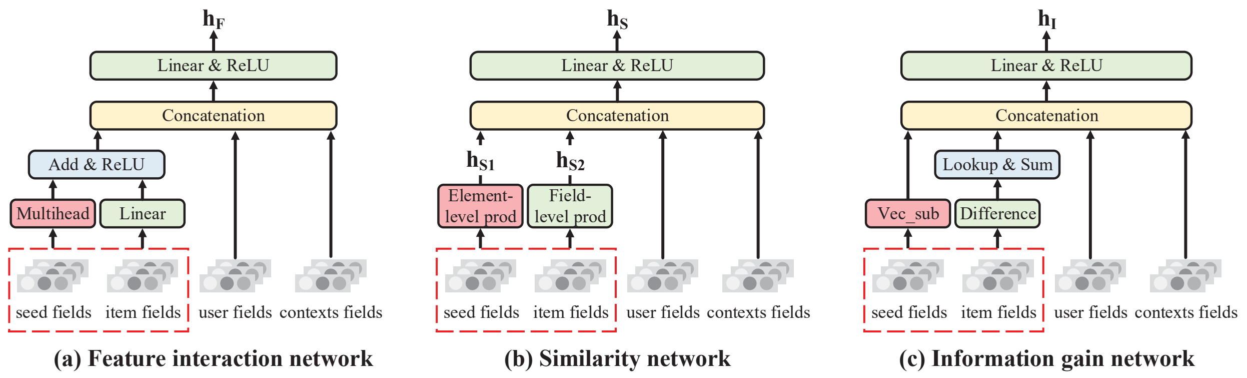

Figure 3: Three experts in M3oE: (a) Feature interaction network, (b) Similarity network, and (c) Information gain network.

capture multi-aspect of features between all feature groups (especially between the seed and target item), and (2) introduces multi-head critics to jointly consider multiple experts for recommendation. Fig 2 shows the overall architecture of M3oE.

3.3.1 Feature Interaction Network (FINet). We first model the general feature interactions between different feature fields with multi-head self-attention also used in [23]. In Fig. 3 (a), the main input feature field matrix $\mathbf{F} = \{\mathbf{f}_1, \cdots, \mathbf{f}_{2k}\}$ consists of the seed and item feature groups F^S and F^I. We conduct a h-head self-attention, and the j-th query, key, value of $\mathbf{F}$ are represented as follows:

$$\mathbf{Q}_j = \mathbf{W}_j^Q\mathbf{F}, \quad \mathbf{K}_j = \mathbf{W}_j^K\mathbf{F}, \quad \mathbf{V}_j = \mathbf{W}_j^V\mathbf{F}, \tag{2}$$

where $\mathbf{W}_j^Q, \mathbf{W}_j^K, \mathbf{W}_j^V \in \mathbb{R}^{d'\times d}$ are projection matrices for the j-th query, key and value. d and $d' = d/h$ are the dimension of field and query. The j-th head's output head$_j$ is as:

$$\text{head}_j = \text{Softmax}(\mathbf{Q}_j^\top \cdot \mathbf{K}_j)\mathbf{V}_j. \tag{3}$$

We then concatenate all heads to generate the output of multi-head self-attention $\hat{\mathbf{F}} = \{\hat{\mathbf{f}}_1, \cdots, \hat{\mathbf{f}}_{2k}\}$ as follows:

$$\hat{\mathbf{F}} = \text{MultiHead}(\mathbf{F}) = \text{Concat}(\text{head}_1, \cdots, \text{head}_h)\mathbf{W}_F^O. \tag{4}$$

We also add a short pass of the raw feature $\mathbf{f}_i$ to generate the feature embedding $\hat{\mathbf{h}}_i \in \mathbb{R}^d$ of the i-th field as:

$$\hat{\mathbf{h}}_i = \text{ReLU}(\hat{\mathbf{f}}_i + \mathbf{W}_F^S \cdot \mathbf{f}_i). \tag{5}$$

$\mathbf{W}_F^O, \mathbf{W}_F^S \in \mathbb{R}^{d\times d}$ are the weighting matrices. ReLU$(\cdot)$ is the non-linear activation function and Concat$(\cdot)$ is the concatenation operation. Finally, we combine all interacted fields with the feature vectors of user $\mathbf{f}^U$ and context $\mathbf{f}^C$ to form the final output feature $\mathbf{h}_F$ of the FINet as follows:

$$\mathbf{h}_F = \text{ReLU}(\mathbf{W}_F^H \cdot \text{Concat}(\hat{\mathbf{h}}_1, \cdots, \hat{\mathbf{h}}_{2k}, \mathbf{f}^U, \mathbf{f}^C)), \tag{6}$$

in which $\mathbf{W}_F^H \in \mathbb{R}^{d_1\times(4kd)}$ is a weighting matrix. $\mathbf{h}_F \in \mathbb{R}^{d_1}$ captures the feature interactions between feature fields.

3.3.2 Similarity Network (SimNet). We build a similarity network as the second expert to explicitly highlight the relevance between the seed and the target item. As shown in Fig. 3 (b), the main inputs of SimNet are the k-field feature groups of the seed F^S and the target item F^I. Intuitively, we calculate the element-level similarity-based feature $\mathbf{h}_{S1}$ between their corresponding feature group vectors with the element-wise product $\odot$ as follows:

$$\mathbf{h}_{S1} = \mathbf{f}^S \odot \mathbf{f}^I, \quad \mathbf{h}_{S1} \in \mathbb{R}^{kd}. \tag{7}$$

$\mathbf{h}_{S1}$ is regarded as the element-level similarity feature between seed and item. We also calculate the field-level inner product via the corresponding feature field embeddings in F^S and F^I as:

$$\mathbf{h}_{S2} = \{\mathbf{f}_1^S \cdot \mathbf{f}_1^I, \cdots, \mathbf{f}_k^S \cdot \mathbf{f}_k^I\}, \quad \mathbf{h}_{S2} \in \mathbb{R}^k, \tag{8}$$

The i-th element in $\mathbf{h}_{S2}$ indicates the i-th field's inner product of the seed and target item, which is viewed as the field-level similarity. Finally, we combine these two similarity features also with user and context features as:

$$\mathbf{h}_S = \text{ReLU}(\mathbf{W}_S^H \cdot \text{Concat}(\mathbf{h}_{S1}, \mathbf{h}_{S2}, \mathbf{f}^U, \mathbf{f}^C)). \tag{9}$$

$\mathbf{W}_S^H \in \mathbb{R}^{d_1\times(3kd+k)}$ is a weighting matrix and $\mathbf{h}_S \in \mathbb{R}^{d_1}$ is the final output of the Similarity network.

3.3.3 Information Gain Network (IGNet). Query suggestion always needs to ensure that the generated query and user intent are completely consistent. In contrast, recommendation suggestion hopes to recommend relevant items that are **related to but not exactly the same** as clicked items. Hence, we design a new information gain network as the third expert to highlight the additional information brought by target items. The i-th feature field in seed and item indicates the same element. Similar to SimNet, we consider F^S and F^I as main inputs, and define a function IG$(\mathbf{f}_i^I, \mathbf{f}_i^S)$ to measure the information gain from seed to target item in the i-th fields. Precisely, we calculate IG$(\mathbf{f}_i^I, \mathbf{f}_i^S)$ in two ways that varies according to the field type (categorical F_{cat} or continuous F_{con}) as:

$$\mathbf{g}_i = \text{IG}(\mathbf{f}_i^I, \mathbf{f}_i^S) = \begin{cases} \text{Sum}(L(F_i^I - F_i^S)), & f_i^I \in F_{cat}. \\ \mathbf{f}_i^I - \mathbf{f}_i^S, & f_i^I \in F_{con}. \end{cases} \tag{10}$$

F_i^I is the sparse feature set of the i-th field in the target item, and F_i^S is that in the seed. $L(F)$ is the lookup function for all sparse features in set F to dense features, and Sum$(\cdot)$ is a vector addition. For a categorical field, we first calculate the difference set of F_i^I and F_i^S, and then project every sparse feature in the difference set to their dense features. The sum of these dense features is viewed as the information gain from the seed to the target item. Intuitively, for a tag field with $F_i^S = \{Apple, IPhone\}$ and $F_i^I = \{Apple, MacBook\}$, the dense feature of *MacBook* is regarded as the additional information brought by the target item, which should be highlighted in IGNet. As for the continuous fields, we intuitively use the subtraction between their dense features as the additional information. We replace the field-level similarity operation in SimNet with the field-level subtraction function $\mathrm{IG}(\mathbf{f}_i^I, \mathbf{f}_i^S)$ as displayed in Fig. 3 (c) , and generate the output of IGNet $\mathbf{h}_I$ as follows:

$$\mathbf{h}_I = \mathrm{ReLU}(\mathbf{W}_I^H \cdot \mathrm{Concat}(\mathbf{g}_1, \cdots, \mathbf{g}_k, \mathbf{f}^U, \mathbf{f}^C)). \tag{11}$$

$\mathbf{W}_I^H \in \mathbb{R}^{d_1 \times (3kd)}$ is a weighting matrix. $\mathbf{h}_I \in \mathbb{R}^{d_1}$ highlights the additional diversified information brought by the target item, which is also essential in relevant recommendation.

3.3.4 Multi-critic Multi-gate Mixture-of-experts (M3oE). The feature interaction, semantic relevance and information gain networks calculate essential factors for relevant recommendation, while their importances may vary with different scenarios. For example, users usually have different preferences on extended or cursory reading, and some topics (e.g., news event) concern more about certain objectives (e.g., semantic relevance). To jointly incorporate opinions from different experts, we propose a novel Multi-critic multi-gate mixture-of-experts (M3oE) framework, which is inspired by the classical multi-task learning framework MMoE [17]. Different from MMoE that builds a task-specific gate for each task separately, M3oE constructs multi-head gates as critics to judge the importances of different experts for one task from multiple aspects. It then jointly considers all critics to fuse opinions from different experts. Specifically, we build d_c gates (i.e., critics) with multi-head strategy to judge the importances of d_e experts ($d_e = 3$ for three networks). We assume that each gate is controlled by the user, seed and context. The gate query $\mathbf{x}_j$ of the j-th head in M3oE is built according to the feature group vectors of user $\mathbf{f}^U$, seed $\mathbf{f}^S$ and context $\mathbf{f}^C$ as:

$$\mathbf{x}_j = \mathbf{W}_j^X \cdot \mathrm{Concat}(\mathbf{f}^U, \mathbf{f}^S, \mathbf{f}^C). \tag{12}$$

We simply use a softmax layer to jointly combine multiple experts. The j-th gate for three experts is then calculated as:

$$g^j(\mathbf{x}_j) = \mathrm{Softmax}(\mathbf{W}_j^G \mathbf{x}_j). \tag{13}$$

$\mathbf{W}_j^X$ and $\mathbf{W}_j^G$ are two weighting matrices. $g^j(\mathbf{x}_j) \in \mathbb{R}^{d_e}$ is the gating vector where each element represents the importance of each expert. Next, the j-th critic will generate the aggregated feature vector $\mathbf{c}_j \in \mathbb{R}^{d_1}$ with the gating vector $g^j(\mathbf{x}_j)$ as:

$$\mathbf{c}_j = g_1^j(\mathbf{x}_j)\mathbf{h}_F + g_2^j(\mathbf{x}_j)\mathbf{h}_S + g_3^j(\mathbf{x}_j)\mathbf{h}_I. \tag{14}$$

$\mathbf{c}_j$ combines the outputs of feature interaction, similarity and information gain networks with the j-th critic's preference on different experts. Finally, we combine d_c critics with an attention via $\mathbf{f}_a = \mathrm{Concat}(\mathbf{f}^U, \mathbf{f}^S, \mathbf{f}^I, \mathbf{f}^C)$ as follows:

$$\mathbf{h}_0 = \sum_{i=1}^{d_c} \alpha_i \mathbf{c}_i, \quad \alpha_i = \frac{\exp(\mathbf{c}_i \mathbf{W}^M \mathbf{f}_a)}{\sum_{j=1}^{d_c} \exp(\mathbf{c}_j \mathbf{W}^M \mathbf{f}_a)}. \tag{15}$$

$\mathbf{h}_0 \in \mathbb{R}^{d_1}$ is the feature extracted with the collaboration of multiple experts and critics. We conduct a 2-layer multilayer perceptron $\mathbf{h}_f = \mathrm{MLP}(\mathbf{h}_0)$ to generate the final output feature $\mathbf{h}_f$ of M3oE for downstream learning objectives.

3.3.5 Optimization Objective. Conventional ranking models usually use Click-through-rate (CTR) as the training objective. However, CTR may be cheated by clickbait or spammer with exaggerated/untrue titles and cover images. Moreover, we hope to encourage extended reading, where the reading time is a better evaluation metric. Inspired by [33], we enhance the original CTR-oriented objectives with the item-level **dwell time (DT)**, which has been widely used in industry to find what users are really interested in. The item-level dwell time metric indicates how much time a user spends on an item, which could be regarded as a more precise and quantitative CTR metric. Precisely, we consider the training of Item recommender as a regression task after DT logarithm and discretization, and conduct a discretized mean squared error (MSE) loss as follows:

$$L_{RR} = \frac{1}{N} \sum_{N_a} (y - \mathbf{w}_{RR}^\top \mathbf{h}_f)^2, \tag{16}$$

where y is the discretized DT of the user-item pair. N_a is the overall instance set and $\mathbf{w}_{RR}$ is a weighting vector.

3.4 Box Trigger

Box trigger is one of the major differences in R3S, which aims to decide whether the system should conduct real-time relevant box insertion considering the overall performance. Box trigger is indispensable in R3S to prevent too many relevant boxes, since the real-time box insertion will delay other items below the clicked seed. These delayed items may lose chances to be impressed to users, which may harm user experience in the main feed if the user does not want extended reading. A straightforward solution is to set a threshold according to the predicted DT given by the Item recommender with some rule-based heuristic strategies. However, it fails to consider both user satisfaction on the seed and the delay costs caused by the delayed items, which are essential in Box trigger.

3.4.1 M3oE Feature Extraction. Similar to the feature extraction and aggregation in Item recommender, we also conduct M3oE with the feature interaction, semantic similarity and information gain networks regarded as experts. The inputs contain F^U, F^S, F^I, F^C also used in Item recommender. We build the item feature F^I as the combination of all relevant items' features selected by the Item recommender. Similarly, the outputs of three experts are represented as $\mathbf{h}'_F$, $\mathbf{h}'_S$ and $\mathbf{h}'_I$ respectively, which are calculated the same as Eq. (2) to Eq. (11) with different trainable parameters.

In M3oE, besides $\mathbf{f}^U$, $\mathbf{f}^S$ and $\mathbf{f}^C$, the multi-head critics further consider (1) the interacting feature $\mathbf{f}^{US}$ between user and seed (e.g., user's reading time on the seed) that models user satisfaction, and (2) the aggregated feature $\mathbf{f}^D$ of top delayed items in the main feed. The j-th gate query $\mathbf{x}'_j$ is built similar to Eq. (12), combining all features stated above that may affect critics in Box trigger as:

$$\mathbf{x}'_j = \mathbf{W}'^X_j \cdot \mathrm{Concat}(\mathbf{f}^U, \mathbf{f}^S, \mathbf{f}^C, \mathbf{f}^D, \mathbf{f}^{US}). \tag{17}$$

Next, we follow Eq. (13) to Eq. (15) and get the aggregated feature $\mathbf{h}'_0$. We concatenate $\mathbf{h}'_0$, $\mathbf{f}^D$ and $\mathbf{f}^{US}$, and feed them into a 2-layer MLP to get the final output $\mathbf{h}'_f$ as:

$$\mathbf{h}'_f = \mathrm{MLP}(\mathrm{Concat}(\mathbf{h}'_0, \mathbf{f}^D, \mathbf{f}^{US})). \tag{18}$$

$\mathbf{h}'_f$ decides whether to explicitly insert the relevant box instantly below the clicked seed in the main feed.

3.4.2 Optimization Objectives. In Box trigger, box-level CTR is considered as the most important rewards, since R3S hopes to enhance users' awareness and cultivate users' habit of using relevant reading. Therefore, we turn back to the CTR-based objective in box level. Specifically, we define the predicted click probability of relevant box $p(x)$ with $\mathbf{h}'_f$ as follows:

$$p(x) = \sigma(\mathbf{w}_{BT}^{\top}\mathbf{h}'_f). \tag{19}$$

$\mathbf{w}_{BT}$ is a weighting vector, and $\sigma(\cdot)$ is the sigmoid function. Different from conventional CTR-oriented loss functions, the Box trigger should further consider the delay costs brought by delayed items in the main feed. Therefore, we add a penalty factor to avoid providing too much relevant boxes when users actually want to move on to other topics. The loss function of Box trigger L_{BT} contains three parts, which is defined as follows:

$$\begin{aligned} L_{BT} = -\frac{1}{N}(\lambda_p \sum_{N_p} \log p(x) + \lambda_n \sum_{N_n} \log(1 - p(x)) \\ + \lambda_d \sum_{N_d} \log(1 - p(x))). \end{aligned} \tag{20}$$

N_d is a new negative set to measure delay costs, where the relevant box has not been clicked but other items below the seed has been clicked, which is exactly what we want to avoid. N_p and N_n indicate the classical positive and negative sets where the relevant box has or has not been clicked. λ_p, λ_n, λ_d are hyper-parameters of loss weights. The overall loss of R3S is the addition of L_{RR} and L_{BT}.

4 ONLINE DEPLOYMENT

In this section, we will give a detailed discussion on the online system, real-time serving and product form of R3S.

4.1 Online System and Real-time Serving

We have deployed R3S on a well-known recommendation system named WeChat Top Stories. WeChat is the most popular instant messaging APP in China, which has approximately 1.2 billion active users per month. WeChat Top Stories is responsible for dispatching million-level items for users, which has approximately billion-level daily views. In online serving, the matching and ranking modules first generate a recommended item list in the main feed. Next, the R3S system works as in Fig. 1. Precisely, the Item recommender module of R3S is activated to generate top-k relevant items when a user clicks an item and enters the content page. In preprocessing, a fast retrieval component is firstly conducted to generate a small subset of relevant item candidates (top 200 in R3S) of the seed, with the help of the textual and taxonomy information (e.g., inverted indices of titles and tags). It could greatly accelerate the following user-item pair-wise ranking process with M3oE. We then rank these item candidates via M3oE, and select up to three relevant items according to an empirical threshold with online rule-based strategies. Once the user has finished reading the clicked item and exits to the main feed, the Box trigger module is activated to decide whether to insert the relevant box instantly below the clicked item. In this case, R3S can effectively and efficiently handle recommendation suggestion, which enables real-time relevant box insertion. For efficiency, we use the parameter-server architecture of Tensorflow to provide a distributed training, which contains 100 workers equipped with 4 CPU cores and 4GB memory. We spend nearly 1 hour on daily model updates of R3S. It is also convenient to adopt R3S on other recommendation feed applications.

4.2 Discussion on Real-time Box Insertion

There are mainly three product forms for the relevant recommendation: (1) we can simply mix the relevant items with other recommended candidates and rerank them in the main feed. However, it will reduce users' awareness of the relevant recommendation function. Users who want extended reading may find it challenging to effectively access relevant items, since they may not be ranked next to the clicked items. On the contrary, users who do not want extended reading may be annoyed by these homogeneous contents. (2) We can insert the relevant box in the content page of the seed. This is a typical product form of relevant recommendation widely used in Youtube and WeChat Top Stories. However, users sometimes do not slide down to the bottom of the content page, so it is hard to guarantee that they will be impressed to users. (3) We can conduct real-time relevant box insertion in the main feed once users exit the content page, as we have introduced in R3S. R3S can significantly improve users' awareness of the relevant recommendation function. The explicit relevant box and dynamic insertion can also enhance user experience and alleviate too much disturbance to other topics. Comprehensively considering the pros and cons of three product forms of relevant recommendation with online experiments and investigation, we finally choose the real-time box insertion (i.e., the current R3S version) in online deployment.

5 EXPERIMENTS

In experiments, we conduct extensive offline and online evaluations on a real-world recommendation system widely used by hundreds of millions of users. Specifically, R3S is mainly evaluated from the following three aspects: (1) the *dwell time metric of relevant items*, which reflects the real user satisfaction on relevant items and the capability of R3S in guiding users to conduct deep extended reading (see Sec. 5.4). (2) The *CTR metric of relevant boxes*, which implies the users' awareness and satisfaction on the overall extended reading function (see Sec. 5.5). And (3) the *overall performance in the main feed*, which indicates that R3S could achieve competitive or even better performances in the overall system, eliminating the side effects caused by the delay costs (see Sec. 5.6). We also conduct ablation tests (see Sec. 5.7) to verify the effectiveness of different components in R3S for recommendation suggestion.

5.1 Datasets

We construct a new dataset RS-331M for relevant recommendation suggestion from WeChat Top Stories since there is no open dataset for this novel task. Precisely, we randomly select 21 million users and collect their 332 million impression instances of the relevant box. In total, we extract nearly 43 million box clicks and 47 million item clicks related to the relevant recommendation suggestion scenario. For each item click, its dwell time (DT) is also recorded for training and evaluation. We split the dataset into a train set and a test set using the chronological order, resulting in 232 million train instances and 100 million test instances. All data including user profiles and user behaviors are preprocessed via data masking to protect user privacy. Table 1 shows the detailed statistics of RS-331M.

Table 1: Statistics of the RS-331M dataset.

#user	#item	#item click	#instance
20,802,239	3,073,826	46,918,912	331,838,667

5.2 Experimental Settings

In Item recommender and Box trigger, we use four feature groups including user, seed, target item and recommendation contexts, each of which contains five feature fields. The dimension of each feature field is 16, while the dimensions of three sub-networks' outputs are set as 120. In M3oE, we use 16 critics to judge different sub-networks. The loss weights λ_p, λ_n and λ_d in Box trigger is 1.0, 1.0, 0.1, and the batch size is set as 128. We conduct a grid search for parameter selection. In online deployment, we set the maximum of relevant items in relevant box as 3.

5.3 Competitors

We implement several competitive ranking models as baselines for both item and box evaluation. We have:

- **FM [18].** Factorization machine (FM) models all interactions between features using factorized parameters. FM is considered as the base model in evaluation.
- **Wide&Deep [6].** Wide&Deep consists of a Wide part for raw feature and a Deep part for feature interaction.
- **NFM [9].** NFM proposes a bi-interaction layer before DNN layers for feature interaction.
- **AFM [28].** AFM brings in attention over feature interactions from the bi-interaction layer.
- **DeepFM [8].** It combines FM with DNN in parallel to model raw features and high-order interactions.
- **AutoInt [23].** AutoInt introduces self-attentive neural network for raw feature interactions.
- **AFN [7].** AFN is a recently proposed model which learns adaptive-order feature interactions for ranking.

Note that we do not use conventional query suggestion models [12] or sentence matching models [10] as baselines, for they are designed for different tasks where the query-title or sentence-sentence similarity is the most essential objective.

All baselines follow the two-step architecture of R3S, which consists of both Item recommender and Box trigger versions. The Item recommenders of baselines are optimized under the DT-based discretized MSE objective in Eq. (16), while the Box triggers are updated with the CTR-based cross entropy objective in Eq. (20) (without the delay loss). The Item recommender is used for Item DT prediction (all CTR-based baselines perform worse than DT-based baselines in Item DT prediction), while the Box trigger is used for Box CTR prediction. All models and R3S share the same features (if needed) and experimental settings in evaluation.

5.4 Item DT Prediction

5.4.1 Evaluation Protocols. The Item dwell time (DT) prediction task attempts to predict how long users will spend on the items in the relevant box [33]. The dwell time metric could be regarded as an enhanced CTR-related metric since it further considers user reading time beyond clicks, which reflects user's true satisfaction. We rank all results with the predicted DT of Eq. (16) in Item recommender, and use the classical AUC metric for evaluation [8, 23]. We also bring in RelaImpr [32] to measure the relative improvements over the base model (i.e., FM in our evaluation).

Table 2: Results of Item DT prediction.

model	AUC	RelaImpr
FM (Rendle 2010)	0.6949	0.00%
AFM (Xiao et al. 2017)	0.7002	2.72%
NFM (He and Chua 2017)	0.7012	3.23%
Wide&Deep (Cheng et al. 2016)	0.7191	12.42%
DeepFM (Guo et al. 2017)	0.7248	15.43%
AutoInt (Song et al. 2019)	0.7220	13.90%
AFN (Cheng et al. 2020)	0.7294	17.70%
R3S	**0.7419**	**24.11%**

5.4.2 Experimental Results. Table 2 shows the results of item DT prediction. We can observe that:

(1) R3S achieves the best performance compared to all baselines. The 1.25% AUC improvement compared to AFN is significant and encouraging. We also conduct a significance test to verify that the significance level $\alpha = 0.01$. Note that all baselines are competitive baselines widely-used in real-world systems that are also enhanced with the same DT-based optimization objective. It indicates that R3S can well capture multi-aspect factors for relevant recommendation. Moreover, it also implies the advantages of FINet, SimNet and IGNet in capturing informative messages for DT prediction.

(2) The improvements mainly come from two aspects including the three experts (i.e., sub-networks) and the Multi-critic multi-gate mixture-of-experts strategy. First, the feature interaction network, similarity network and information gain network consider different aspects of feature interactions between the seed and the target item. In this case, R3S could consider self-attention based feature interactions, seed-item semantic similarity and information gain simultaneously, which are essential in the relevant recommendation scenario. Second, the M3oE strategy smartly combines three experts with multiple critics that have different concentrations on these interactions, which further improves the Item DT prediction performance. We conduct detailed analyses on different experts and M3oE aggregation in R3S with ablation tests in Sec. 5.7.

5.5 Box CTR Prediction

5.5.1 Evaluation Protocols. The Box CTR prediction aims to predict whether a relevant box will be clicked, which is judged by the Box trigger module. In R3S, we hope to cultivate users' habit of using our relevant recommendation function. Therefore, we encourage users to click more relevant boxes instantly inserted below their clicked items, and thus CTR is viewed as the main evaluation metric for the Box trigger. We follow the same metrics of AUC and RelaImpr also used in Item DT prediction, and rank test instances with the predicted box click probability in Eq. (19) for evaluation.

Table 3: Results of Box CTR prediction.

model	AUC	RelaImpr
FM (Rendle 2010)	0.7658	0.00%
AFM (Xiao et al. 2017)	0.7704	1.73%
NFM (He and Chua 2017)	0.7724	2.48%
Wide&Deep (Cheng et al. 2016)	0.7866	7.83%
DeepFM (Guo et al. 2017)	0.7901	9.14%
AutoInt (Song et al. 2019)	0.7899	9.07%
AFN (Cheng et al. 2020)	0.7982	12.19%
R3S	**0.8101**	**16.67%**

5.5.2 Experimental Results. Table 3 shows the results of all models on Box CTR prediction, from which we can know that:

(1) R3S significantly outperforms all baselines in Box CTR prediction with the significance level $\alpha = 0.01$. It confirms that R3S is capable of deciding whether to conduct an instant box insertion when users have finished items. Our M3oE strategy with three experts is effective in not only Item DT prediction, but also Box CTR prediction. The Box trigger is designed to control the frequency of relevant box, which is essential for the overall recommendation performance. In Sec. 5.6, we will give detailed analyses on both box-level and overall online performances in a real-world recommendation suggestion system of WeChat Top Stories.

(2) Box CTR prediction and Item DT prediction are two similar tasks, while they still have some differences. Item DT prediction mainly evaluates the Item recommender. We enhance the classical CTR-oriented objective with dwell time to measure user's true preferences and reduce the influence of clickbait. In contrast, Box CTR prediction mainly evaluates the Box trigger. It still uses clicks for training, since we regard the clicks of relevant box as the most important rewards in Box trigger as discussed in Sec. 3.4 for recommendation suggestion. Moreover, Box trigger should consider the side effects and impacts of real-time box insertion on the overall system beyond the box-level performance. Therefore, Box trigger further models user satisfaction on seed and regards delay cost as a penalty to balance both overall and box-level performances. The AUC of Item DT is lower than the AUC of Box CTR, since DT prediction is more fine-grained and challenging.

5.6 Online Evaluation

The real-world performances of R3S (especially of the Box trigger and overall system) are hard to be thoroughly evaluated in offline. Therefore, we further conduct an online A/B test to evaluate R3S on both recommendation suggestion and the overall system in practice.

5.6.1 Evaluation Protocols. We deploy R3S on the recommendation suggestion module in WeChat Top Stories following Sec. 4, with matching and ranking modules unchanged. The online baseline model uses DeepFM to recommend relevant items and several rule-based strategies to control the frequency of relevant box. We first replace DeepFM with the Item recommender in R3S, and then replace the rule-based strategies with the Box trigger in R3S, noted as R3S (Item) and R3S (Item+Box) respectively.

In online evaluation, we focus on four classical metrics to measure both box-level and overall performances in practice. The metrics include: (1) Average dwell time in the overall system (DT), (2) Box-level CTR (BCTR), (3) Box-level user has-click rate (BUHR), and (4) Box-level item views (BIV). DT focuses on the overall performance, while the rest three metrics reflect box-level performances. We conduct the online A/B tests for 7 days.

Table 4: Online A/B tests on WeChat Top Stories.

model	DT	BCTR	BUHR	BIV
R3S (Item)	+1.68%	-7.62%	+3.14%	+10.68%
R3S (Item+Box)	+1.64%	+9.90%	+16.62%	+24.26%

5.6.2 Experimental Results. We report the improvement percentages of R3S compared to the online DeepFM with rule-based strategies baseline in Table 4. We can find that:

(1) R3S (Item+Box) achieves significant improvements on all overall and box-level metrics with the significance level $\alpha = 0.01$, which verifies the effectiveness of R3S in real-world scenarios. The improvement in DT indicates that users are more satisfied with and willing to spend more time on the recommended results of the overall online system. The improvements on three box-level metrics also demonstrate that R3S could smartly control the frequency of relevant box. Therefore, the quality of relevant boxes is enhanced (BCTR), more users are willing to conduct continuous extended reading via relevant box (BUHR), and thus R3S achieves more interactions from users on recommendation suggestion (BIV).

(2) R3S (Item) also outperforms the DeepFM model on three metrics except for BCTR. It is natural since the baseline DeepFM is learned with a CTR-oriented objective. Converting the training objective from CTR to DT will improve the overall dwell time (which is the most essential metric in the system), while it inevitably harms the box-level CTR. Nevertheless, the improvements on BUHR and BIV show that more users will use our recommendation suggestion function, and more relevant items will be clicked.

(3) The decrease in BCTR brought by R3S (Item) reveals the serious problem of overexposure in relevant box. Moreover, the overall CTR also has a slight decline. To address this problem, we introduce the Box trigger that can better consider the delay costs to balance relevant boxes and items in the main feed. All box-level metrics have impressive improvements compared to R3S (Item). The box-level CTR achieves 17.52% improvement compared to R3S (Item) in Table 4. We also find that even the overall CTR has 0.58% improvements over R3S (Item). It reconfirms the significance of Box trigger in the overall online performances.

5.7 Ablation Tests

To verify the effectiveness of different components of R3S, we further conduct several ablation tests in Table 5. We can find that: (1) all three experts including feature interaction network, similarity network and information gain network are significantly beneficial for improving both item-level DT and box-level CTR. It reconfirms the effectiveness of all experts. (2) Among these experts, FINet contributes more to Item DT, while SimNet and IGNet contribute more to Box CTR. It confirms that the semantic relevance and information gain are more essential factors for guiding users to conduct extended readings, while the general feature interactions capture other useful information for predicting dwell time of items. (3) The multi-critic attention also helps to learn a better R3S model. We replace M3oE with an average pooling for feature aggregation from different experts. It implies that the combination of multiple experts should be carefully customized with M3oE in R3S.

Table 5: Ablation tests for R3S.

Ablation version	Item AUC	Box AUC
R3S	**0.7419**	**0.8101**
– Feature interaction network	0.7336	0.8057
– Similarity network	0.7367	0.8040
– Information gain network	0.7371	0.8037
– Multi-critic attention	0.7348	0.8038

6 CONCLUSION AND FUTURE WORK

In this work, we propose a novel task named recommendation suggestion for relevant items, and design R3S for real-time relevant box insertion. R3S consists of an Item recommender and a Box trigger, which uses an M3oE strategy to jointly combine multi-aspect factors including feature interaction, semantic relevance and information gain. The improvements in offline and online evaluations verify the effectiveness of R3S in relevant recommendation.

In the future, we will utilize more types of feature interactions between seed and target items, and conduct more sophisticated models to model the semantic relevance and information gain. We will also explore the joint training of recommendation suggestion and overall recommendation to better model delay costs. Other feedback mechanisms and product forms of recommendation suggestion should be investigated to further improve user's activeness in recommendation. We will also explore the possibility of applying R3S (or its M3oE, SimNet, IGNet modules) to other tasks.

REFERENCES

[1] Malak Al-Hassan, Haiyan Lu, and Jie Lu. 2015. A semantic enhanced hybrid recommendation approach: A case study of e-Government tourism service recommendation system. *Decision Support Systems* (2015).
[2] Trapit Bansal, David Belanger, and Andrew McCallum. 2016. Ask the gru: Multi-task learning for deep text recommendations. In *Proceedings of RecSys*.
[3] John S Breese, David Heckerman, and Carl Kadie. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of UAI*.
[4] Michel Capelle, Flavius Frasincar, Marnix Moerland, and Frederik Hogenboom. 2012. Semantics-based news recommendation. In *Proceedings of WIMS*.
[5] Wanyu Chen, Fei Cai, Honghui Chen, and Maarten De Rijke. 2020. Personalized query suggestion diversification in information retrieval. *Frontiers of Computer Science* (2020).
[6] 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*.
[7] Weiyu Cheng, Yanyan Shen, and Linpeng Huang. 2020. Adaptive Factorization Network: Learning Adaptive-Order Feature Interactions. In *Proceedings of AAAI*.
[8] 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*.
[9] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *Proceedings of SIGIR*.
[10] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Proceedings of NIPS*.
[11] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* (2009).
[12] Ruirui Li, Liangda Li, Xian Wu, Yunhong Zhou, and Wei Wang. 2019. Click feedback-aware query recommendation using adversarial examples. In *Proceedings of WWW*.
[13] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In *Proceedings of KDD*.
[14] Jimmy Lin and Mark D Smucker. 2008. How do users find things with PubMed? Towards automatic utility evaluation with user simulations. In *Proceedings of SIGIR*.
[15] Bin Liu, Chenxu Zhu, Guilin Li, Weinan Zhang, Jincai Lai, Ruiming Tang, Xiuqiang He, Zhenguo Li, and Yong Yu. 2020. AutoFIS: Automatic Feature Interaction Selection in Factorization Models for Click-Through Rate Prediction. (2020).
[16] Qi Liu, Ruobing Xie, Lei Chen, Shukai Liu, Ke Tu, Peng Cui, Bo Zhang, and Leyu Lin. 2020. Graph Neural Network for Tag Ranking in Tag-enhanced Video Recommendation. In *Proceedings of CIKM*.
[17] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of KDD*.
[18] Steffen Rendle. 2010. Factorization machines. In *Proceedings of ICDM*.
[19] Masoumeh Riyahi and Mohammad Karim Sohrabi. 2020. Providing effective recommendations in discussion groups using a new hybrid recommender system based on implicit ratings and semantic similarity. *Electronic Commerce Research and Applications* (2020).
[20] Badrul Munir Sarwar, George Karypis, Joseph A Konstan, John Riedl, et al. 2001. Item-based collaborative filtering recommendation algorithms.. In *Proceedings of WWW*.
[21] Ozan Sener and Vladlen Koltun. 2018. Multi-task learning as multi-objective optimization. In *Proceedings of NIPS*.
[22] Mark D Smucker and James Allan. 2006. Find-similar: similarity browsing as a search tool. In *Proceedings of SIGIR*.
[23] 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*.
[24] 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*.
[25] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & cross network for ad click predictions. In *Proceedings of ADKDD*.
[26] Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. Npa: Neural news recommendation with personalized attention. In *Proceedings of KDD*.
[27] Qiong Wu, Yong Liu, Chunyan Miao, Binqiang Zhao, Yin Zhao, and Lu Guan. 2019. PD-GAN: Adversarial Learning for Personalized Diversity-Promoting Recommendation. In *Proceedings of IJCAI*.
[28] 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*.
[29] Ruobing Xie, Cheng Ling, Yalong Wang, Rui Wang, Feng Xia, and Leyu Lin. 2020. Deep Feedback Network for Recommendation. In *Proceedings of IJCAI*.
[30] 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*.
[31] Zhenghua Xu, Cheng Chen, Thomas Lukasiewicz, Yishu Miao, and Xiangwu Meng. 2016. Tag-aware personalized recommendation using a deep-semantic similarity model with negative sampling. In *Proceedings of CIKM*.
[32] 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*.
[33] Xing Yi, Liangjie Hong, Erheng Zhong, Nanthan Nan Liu, and Suju Rajan. 2014. Beyond clicks: dwell time for personalization. In *Proceedings of RecSys*.
[34] Mi Zhang and Neil Hurley. 2008. Avoiding monotony: improving the diversity of recommendation lists. In *Proceedings of RecSys*.
[35] Weinan Zhang, Tianming Du, and Jun Wang. 2016. Deep learning over multi-field categorical data. In *European conference on information retrieval*.
[36] 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*.
[37] Cai-Nicolas Ziegler, Sean M McNee, Joseph A Konstan, and Georg Lausen. 2005. Improving recommendation lists through topic diversification. In *Proceedings of WWW*.