Learning to Build User-tag Profile in Recommendation System

Su Yan
WeiXin Group, Tencent Inc.
Beijing, China
suyan@tencent.com

Xin Chen
WeiXin Group, Tencent Inc.
Beijing, China
andrewxchen@tencent.com

Ran Huo
WeiXin Group, Tencent Inc.
Beijing, China
lavinhuo@tencent.com

Xu Zhang
WeiXin Group, Tencent Inc.
Beijing, China
xuonezhang@tencent.com

Leyu Lin
WeiXin Group, Tencent Inc.
Beijing, China
goshawklin@tencent.com

ABSTRACT

User profiling is one of the most important components in recommendation systems, where a user is profiled using demographic (e.g. gender, age, and location) and user behavior information (e.g. browsing and search history). Among different dimensions of user profiling, tagging is an explainable and widely-used representation of user interest. In this paper, we propose a user tag profiling model (UTPM) to study user-tag profiling as a multi-label classification task using deep neural networks. Different from the conventional model, our UTPM model is a multi-head attention mechanism with shared query vectors to learn sparse features across different fields. Besides, we introduce the improved FM-based cross feature layer, which outperforms many state-of-the-art cross feature methods and further enhances model performance. Meanwhile, we design a novel joint method to learn the preference of different tags from a single clicked news article in recommendation systems. Furthermore, our UTPM model is deployed in the WeChat "Top Stories" recommender system, where both online and offline experiments demonstrate the superiority of the proposed model over baseline models.

CCS CONCEPTS

• **Information systems** → **Personalization**.

KEYWORDS

personalization, user profiling, neural networks, recommendation systems

ACM Reference Format:
Su Yan, Xin Chen, Ran Huo, Xu Zhang, and Leyu Lin. 2020. Learning to Build User-tag Profile in Recommendation System. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20), October 19–23, 2020, Virtual Event, Ireland.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3340531.3412719

CIKM '20, October 19–23, 2020, Virtual Event, Ireland

ACM ISBN 978-1-4503-6859-9/20/10…$15.00
https://doi.org/10.1145/3340531.3412719

1 INTRODUCTION

In recent years, personalization techniques have been extensively applied in large-scale recommendation platforms, such as YouTube and Amazon. Widely used personalization techniques, such as CTR (click-through-rate) prediction, evaluate a user's preference for an item based on user and article features. Therefore, an accurate selection of user representation from over-loaded user information is the backbone for satisfactory recommendation results. Consequently, user profiling is one way to enhance the performance of recommendation systems by building accurate user representation that improves personalization [12].

Figure 1 briefly shows how user profiling works in the WeChat "Top Stories" news recommendation system. The architecture of the recommendation system consists of four parts: news profile layer, user profile layer, recall layer, and rank layer. In the news profile layer, basic properties of the news article, such as its WeChat subscription account, are recorded, while semantic information, such as tag and news category, is extracted through natural language processing techniques. As for the user profile layer, user's demographic information, such as gender and age, and behavioral information, such as reading history, are collected. User profiling utilizes both behavior information and news profile, transmitting news properties to users to reflect users' interest in certain kinds of articles. Among these properties, tagging is an interpretable and widely used representation of users' interests in recommendation systems [2]. In this study, we introduce how to build a user-tag profile using the WeChat "Top Stories" recommender system as an example. In this paper, "clicked tags" refer to tags within news articles that the user clicked, and "un-clicked tags" refer to tags within news articles that the user browsed but has not clicked. Note that an article is associated with various tags related to its content. In the recall layer, news articles related to the user's interest are recalled through several recall strategies, most of which take the user-tag profile as an important feature input. Major recall strategies are tag-based recall, where news articles containing user's interest tags are recalled, cf-based recall, which includes content-based cf and user-based cf, and other model-based recalls. After the aforementioned steps, the rank layer ranks the recalled news articles based on user and news profiles. Finally, top-ranked news articles are shown to users. Due to the steep decline in the number of candidate news articles from the recall layer to the rank layer, the rank layer can use a more complex model with a higher number of feature input than the recall layer without compromising on

efficiency. It is noteworthy that the user profile is highly connected to strategies and models used in the recall and rank layers; it is, therefore, necessary to build an accurate user profile for online news recommendation systems.

The process to build a user-tag profile is demonstrated in the blue dashed rectangle in Figure 1. Firstly, we collect data and process it in the feature generation and label organization steps (the process is detailed in the Experiment Setup section). Then, for each user, all "clicked tags" (tags within news articles that the user clicked) in the user's reading history are collected in the candidate selection step, and the user's preference on these tags is calculated in the model training step. One user can have thousands of clicked tags. Besides, when predicting a user's preferences on unseen tags, candidates could consist of a large number of tags.

While building the user-tag profile, it is essential to select features and learn feature interaction efficiently since the model input consists of numerous sparse features from multiple fields, as shown on the left part of Figure 1. In news recommendation systems, articles clicked are taken as positive samples; articles browsed but not clicked are taken as negative samples. However, directly applying this method to tagging leads to treating tags within clicked news articles as positive samples, which can be problematic, since the user who clicks an article may be interested in only one of its tags. Based on the aforementioned consideration, we aim to answer the following two questions:

- **RQ1**: How to automatically select useful features and learn the interaction between features within and among different fields ?
- **RQ2**: How to learn user's preference over different tags from each clicked news article ?

In recent years, deep neural networks have been widely used in recommendation systems, where user and news article features are utilized and their interactions are learned. One widely used deep recommendation model is the Youtube model proposed by Google [4]. However, we have identified two weaknesses of this model that we aim to improve. Firstly, the YouTube model uses the average pooling layer to merge multiple input feature embeddings, failing to consider that useful and useless features should be assigned different weights. Furthermore, the YouTube model uses concatenation to merge features across different fields and feed the merged output into the upper layer through MLP(multilayer perceptron). In experiments, we observed that weights of some fields are underestimated, especially when these fields are not highly related to labels, which hinders feature fusion across fields.

To avoid the pitfall of the Youtube model and inspired by the attention mechanism [20], which captures useful word and sentence embedding for doc classification, we design an attention fusion layer within and across each feature field. In the original attention mechanism, there is only one query vector to determine feature usefulness. We believe that multi-head attention helps reserve more useful features from multi-aspects, and therefore our model uses two query vectors and shares the query vector with each head of attention units.

Furthermore, we propose a cross feature layer to enhance model performance. FM-based feature interaction methods like AFM [18] and NFM [7] are widely used, where the Hadamard product of pair-wise hidden vectors is summed into a vector with the same size of the hidden vector. In the user profiling task, the aforementioned methods might lead to the loss of user information. It is therefore advisable to output all inner product values of each pair-wise hidden vectors. It helps to learn a user's multiple interests and leads to higher performance for multi-labels classification.

The contributions of this paper are as follows: 1) We propose a user-tag profiling model (UTPM), which could make use of multiple fields of user information and is suitable for other user profiling tasks. 2) In this model, we introduce a multi-head attention mechanism with shared query vectors to capture the important attributes within each field and merge multiple fields by assigning each field a reasonable weight. 3) We propose a specially designed FM-based cross feature layer to promote user profiling where all crossed values are fed as a dense vector to the next layer along with linear values to generate the final user embedding. 4) Particularly for the user-tag profiling task where each news article contains several tags, we design joint loss to learn each user-tag preference, which is proved to achieve better performance compared with the separate training.

2 RELATED WORK

Tag recommendation system. In recent years, tag recommendation techniques have received more and more attention. An adaptation of user-based collaborative filtering and graph-based recommender is proposed for tag recommendation system [8]. Meanwhile, Vig introduces a tag-based explainable recommendation system [16] where he studied two key components: tag-relevance and user-tag preference during recommendation, both of which improve effectiveness. Researchers at Sina Weibo designed an integrated recommendation algorithm to collectively explore the social relationship among users, the co-occurrence relationship and semantic relationship among tags [19]. So far, the methods applied to the tag recommendation problem have been mainly collaborative algorithms that are based simply on the co-occurrence among users and tags. But in reality, these methods cannot fully utilize multi-field user information which could help discover users' interests. It is necessary to apply state-of-the-art deep models to enhance the performance of user profiling tasks.

Attention Mechanism. Attention mechanism originates from Neural Machine Translation(NMT) [1], where words are assigned with different weights in different contexts. Attention has been used successfully not only in a variety of NLP tasks including reading comprehension, abstractive summarization, and textual entailment [10], but also in recommendation systems [3, 21]. One type of self-attention [7] learns the words and sentences normalized weight for a certain classification task where only one query vector is learned. Liu [11] utilizes the self-attention mechanism to fuse features among fields and achieves better performance than concatenation and MLP for online news recommendation task. Another kind of self-attention [15] studies the inner correlation between words, where each word is assigned with a query vector. Song [14] utilizes this self-attention mechanism to design a stack of cross networks called AutoInt, which can learn high-order feature interaction among different fields.

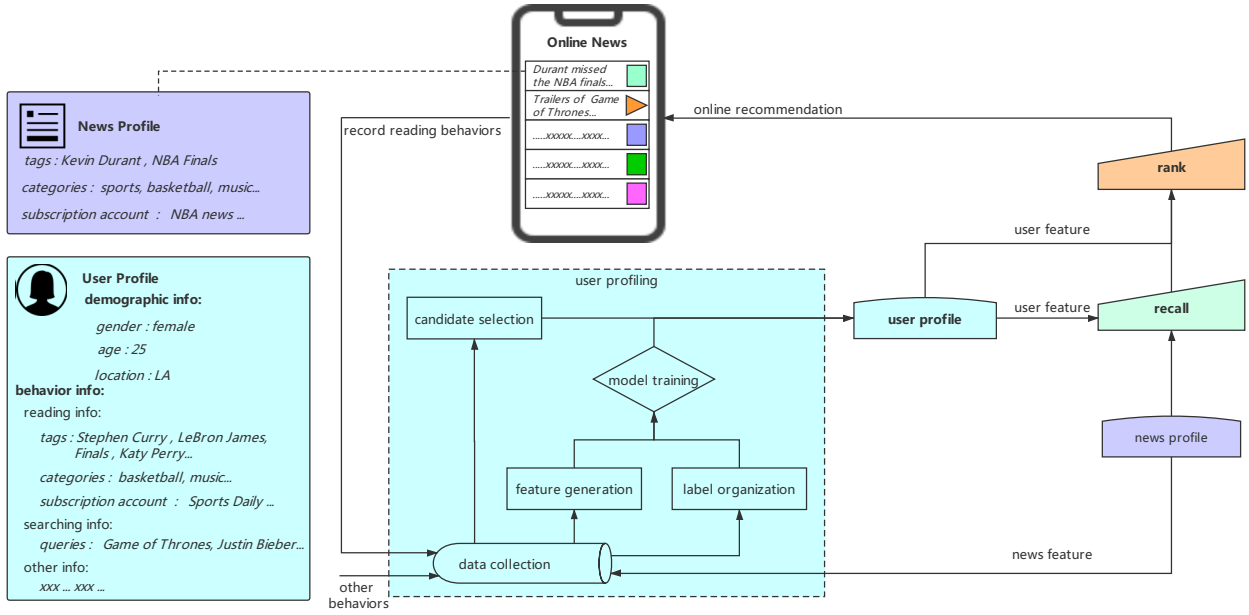

Figure 1: Structure of Online News Recommendation System and User Profile, where user profile serves as the most important feature input in recall and rank layer. For example, a 25-year-old girl who is interested in *Stephen Curry* might find some related news articles like *Kevin Durant* on her online news streaming.

Cross Feature. Research has shown high-order feature interaction helps improve model performance. FM [13] is a famous second-order feature interaction model where each sparse feature correlation is learnt. DeepFM [5] combines the traditional factorization machine and deep neural network, where linear, second-order, and deep parts are fed into the predicting layer. NFM [7] learns the second-order feature interactions through the Hadamard product of hidden vectors. AFM [18] adds attention pooling layer to learn second-order feature interactions. These interacted feature values or crossed hidden vectors are summed up as the final prediction, which might lead to the loss of a user's multiple interests. DCN [17] introduces a novel iterative cross network on dense features without explicitly calculating the pair-wise crossed feature value. XDeepFM [9] builds compressed cross networks to learn vector-wise feature interaction from different feature fields.

Click Rate Predicting. CTR prediction is a binary classification task and has been widely studied in recent years, with many networks and structures proposed to learn user and item feature interaction. Among these methods, one popular example is item-wise models, where a user can have different representations for different items. For instance, in DIN [23], a model deployed in *TaoBao* recommendation, candidate ads are used to merge different behaviors, thus a user is assigned with different feature weights for different ads. Likewise, DIEN [22] and ATrank [21] use item id as the key to merging user features. Different from the aforementioned item-wise models, another type of models like DeepFM and DCN mix raw input features of users and items from the bottom layer, so that upper layers of the network learn user-item feature interaction. All these methods require user-item pairs during training and predicting. However, these structures are not suitable for user-tag profiling tasks due to the efficiency constraint. User-tag profiling engine incorporates more user behaviors and usually hundreds of times more tag candidates, but less computing resources than ranking engine of CTR models. It is, therefore, reasonable to model user-tag profiling as a multi-label classification task, where user-tag preference is calculated through the inner product of user feature vector and tag feature vector. Additionally, to make use of these network structures, we can simply feed user features into these CTR networks to learn user feature interaction rather than mixing user and item features.

3 OUR MODEL

In this section, we introduce our user-tag profiling model, a multi-head attention mechanism with shared query vectors, improved FM-based cross feature mechanism and a novel joint method to learn the tags of items.

Feature-input layer. In the first layer, user information from multiple fields is fed into the network. The information mainly consists of two parts corresponding to Figure 1: 1) reading-related information, such as clicked tag, clicked category, and WeChat subscription account field 2) demographic information, such as gender and age field. All input variables are categorical and each feature is mapped to a vector of the same length E ($E = 128$ in our experiment) through the embedding look-up for each field. Note that different fields contain various numbers of unique features, for instance, the clicked tag field has more than one million unique tags, the location field has more than hundreds of cities, category and age fields more than dozens of features. Thus we keep E big enough to fit the vector space of million numbers of features.

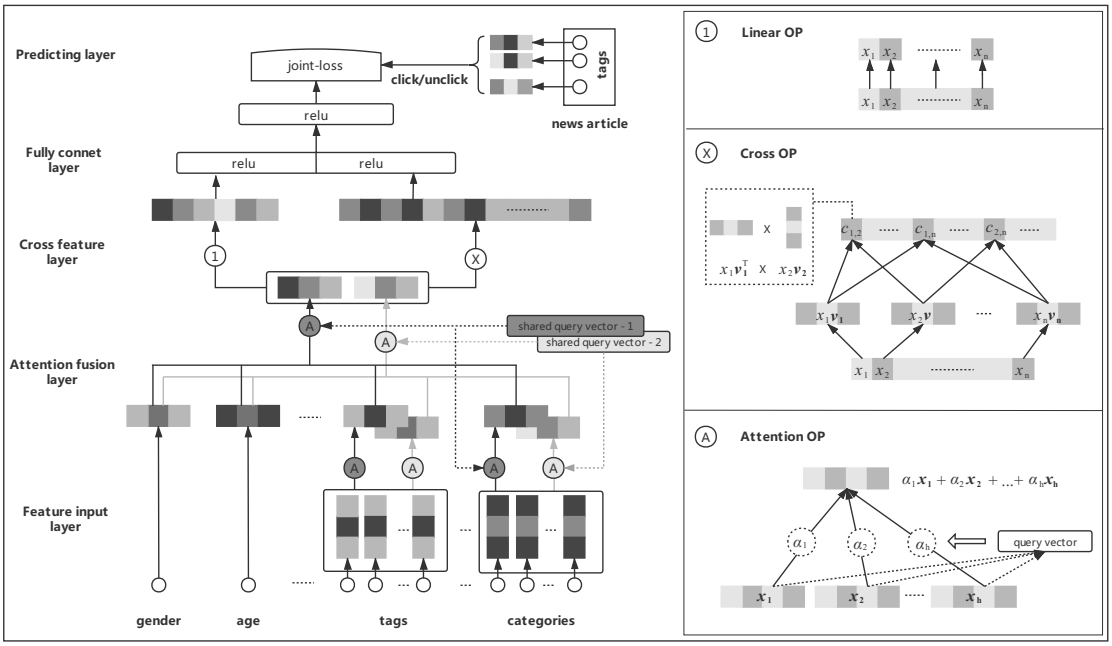

Figure 2: User-tag Profiling Model

Attention-fusion layer. Here we try to solve the problem mentioned in RQ1 based on a multi-head attention fusion method. Note that we get several feature fields including gender, age, and clicked tags. Some fields contain more than one categorical feature. First, features within each field are merged through a multi-head attention mechanism where each feature is assigned an attention weight. Note that we use two-head attention, where every single attention is based on the self-attention proposed by Yang [20].

Take clicked tags for instance, assuming the k-th feature field contains H tag features. Vector $\boldsymbol{t}_i \in \mathbb{R}^E$ represents the i-th of the H embedded tag vectors. For the first attention, each $\boldsymbol{t}_i$ is summed by the weight $\alpha_{i,1}$ which is calculated through a soft-max function given the first query vector $\boldsymbol{q}_1 \in \mathbb{R}^T$, transformation matrix $W_t \in \mathbb{R}^{E\times T}$, and bias $\boldsymbol{b}_k \in \mathbb{R}^T$ for tag field.

$$\alpha_{i,1} = \frac{e^{\boldsymbol{q}_1^T \text{relu}(\boldsymbol{W}_{k,1}^T \boldsymbol{t}_i + \boldsymbol{b}_k)}}{\sum_{j=1}^{H} e^{\boldsymbol{q}_1^T \text{relu}(\boldsymbol{W}_{k,1}^T \boldsymbol{t}_j + \boldsymbol{b}_k)}} \tag{1}$$

As in Equation 1, $\boldsymbol{t}_i$ is transformed into the first attention space through nonlinear function $\text{relu}(\boldsymbol{W}_{k,1}^T \boldsymbol{t}_i + \boldsymbol{b}_t)$. The dot product of the transformed vector with query vector $\boldsymbol{q}_1^T$, represents the importance of the tag i to the classification task, and is normalized by a soft-max function. After that, all embedding will be merged into one embedding $\boldsymbol{f}_1 \in \mathbb{R}^E$ by Equation 2 and fed into the next layer. Useful tag embedding has a higher weight. Note that for the first head of attention, each input field shares the same query vector $\boldsymbol{q}_1$, but has different transformation matrix $\boldsymbol{W}_{k,1}$.

$$\boldsymbol{f}_1 = \sum_{i=1}^{H} \alpha_{i,1} \boldsymbol{t}_i \tag{2}$$

After that, we also use a similar attention mechanism to merge these embedding from M fields, as shown in the upper part of the attention fusion layer in Figure 2. It is worth mentioning that while merging these features in the first head of attention, the same query vector $\boldsymbol{q}_1$ is shared in all attention units within and across fields, thus the same evaluation criterion of feature usefulness is learned together in all attention units. We argue that learning from features of different fields with a shared query vector helps build a more robust and accurate selection criterion than learning from each separate field with different query vectors, especially for fields with sparse features. Likewise, we can fuse the features through the second head of attention to learn features from different aspects as proved in our experiments. Finally, two merged embeddings from the two-head attention mechanism are concatenated together to the upper layer.

Cross-feature layer. To further solve the problem mentioned in RQ1, we make feature interactions based on output embedding $\boldsymbol{x} \in \mathbb{R}^E$ of the attention-fusion layer. We firstly send the input feature vector $\boldsymbol{x}$ as the linear part directly to the next layer in Figure 2. Meanwhile, we try to make interactions between each pair of dimensions within $\boldsymbol{x}$ through our designed cross feature method. Each dimension of feature is also mapped into the hidden vector space $\mathbb{R}^C$. In Equation 3, for each pair of dimensions like i and j, $\boldsymbol{v}_i^T \boldsymbol{v}_j$ represents the feature correlation weight and is multiplied with their feature value x_i and x_j as the cross value $c_{i,j}$. Thus this layer outputs a linear vector whose length is E and a cross vector whose length is $\frac{E(E-1)}{2}$.

$$\begin{aligned} c_{i,j} &= (\boldsymbol{v}_i^T \boldsymbol{v}_j) x_i x_j \\ \boldsymbol{c} &= [c_{1,2}; c_{1,3}; ...c_{i,j}; ...](i < j \leq E) \end{aligned} \tag{3}$$

It is worth mentioning that in our model, all interacted values are kept instead of being summed up as in traditional FM. Our model is also different from NFM [7]and AFM [18], where the weighted sum is calculated from the Hadamard product of hidden vectors. Thus

NFM and AFM generate a cross vector of size C. Because C is much smaller than $\frac{E(E-1)}{2}$, our cross feature layer has more potential to keep user interests that are useful for multi-label classification. Besides, we also incorporate the linear part with L2-normalization to make it comparable with the cross part.

Fully-connect layer. Two fully-connect layers are used to generate the final user embedding $\boldsymbol{u} \in \mathbb{R}^U$.

Predicting layer. With the user embedding $\boldsymbol{u}$ and training labels (K clicked news articles and "un-clicked news articles", referring to news articles browsed but not clicked), we design a joint loss which can formulate the preference of different tags within a single news article to solve RQ2. Note that we map tags of a news article through embedding look-up into the same vector space of user embedding. As Equation 4 shows, $\hat{y_k} \in \{0,1\}$ is the k-th training label. For the N tags of the news article, the sum of user preference $\boldsymbol{u} \cdot \boldsymbol{t}_i$ is taken into a sigmoid function to determine whether the user clicks the news article.

$$\mathcal{L} = -\frac{1}{K}\sum_{k=1}^{K}\left(\hat{y_k}\log y_k + (1-\hat{y_k})\log(1-y_k)\right) \tag{4}$$

where $y_k = sigmoid(\sum_{i=1}^{N} \boldsymbol{u} \cdot \boldsymbol{t}_i)$.

It seems plausible to choose tags from clicked news articles as positive labels, and tags from news articles that have not been clicked as negative labels and train them separately. However, the underlying assumption behind this method is that when a user clicks a news article, he is interested in all tags associated with this news article. However, in reality, while clicking a news article, the user might be interested only in one or two tags among its N tags. Consequently, using joint loss is more advisable, since each user-tag preference $\boldsymbol{u} \cdot \boldsymbol{t}_i$ for a news article is learned jointly, thus tags that a user is more interested in will have a higher value of $\boldsymbol{u} \cdot \boldsymbol{t}_i$.

4 EXPERIMENTS

In this section, we present our experiments in detail. Based on the YouTube model, we conduct three groups of experiments and offer one explainable case as following:

1) **Multi-head attention mechanism.** Here we design experiments to evaluate the performance of the multi-head attention-fusion layer. In practice, we use a two-head attention mechanism with two query vectors. (More heads of attention will enlarge the embedding size fed into the cross feature layer and cause exponential growth in the calculation). We test the superiority of the single-head attention fusion layer over the YouTube model. And then we evaluated the performance of two-head attention over single-head attention.

2) **Cross feature.** In this part, we prove our cross feature layer further enhances the model capability, compared with several state-of-the-art cross feature methods. To contrast, we replace our cross feature layer with the cross feature methods in DCN, AFM, and NFM respectively. We also compare our cross feature networks with AutoInt, which learns feature interaction by self-attention. Because AutoInt incorporates dense features from different fields, we feed features into AutoInt after the fusion within each feature field in the attention fusion layer. All these baseline models can be incorporated into our networks to learn user representation independently. However, some other methods, such as DIN, DIEN, and ATrank, as mentioned in the related work section, require item-wise training and predicting, thus are not efficient for user-tag profiling tasks.

3) **Joint loss** We test our joint loss method over the aforementioned separate loss. Note that each of these experiments is based on the previous step.

4) **Explainable cases** Besides, we provide some explainable cases to prove that with improved model structure, useful information from multiple fields can be discovered.

4.1 Experiment Setup

Data Sets. Our model is firstly built on the WeChat "Top Stories" Recommender, and then further tested on an open dataset. We used log data of 30 days from the "Top Stories" recommendation system. Note that all log data is prepossessed by data masking to protect the user privacy. From the daily log, we collect a user's clicked tags and un-clicked tags on that day. Besides, we can get a user's demographic information and behavioral information including tag and category. Each training sample is composed of user features and positive or negative labels. There are 149,076,658 training samples and 1,176,273 tags as labels. To guarantee model accuracy, the training set has eliminated users whose features and labels are too sparse. For testing, we use the next day's logs to generate a validation dataset of 50,000 randomly sampled user behavior data. Note that all experiment results are calculated on the validation set.

To consolidate our result, we also used the widely-used movie rating dataset MovieLens 20M dataset [6], which consists of movie id, user id, movie tag, category, rating, and timestamp. We split users in a roughly 80-20 ratio, using ratings from 112,485 users for training and ratings from 26,000 users for testing. There are 1,042 movie tags and 19 categories. Rating equals or below 1.5 is taken as a negative sample and rating equals or above 3.5 as a positive sample. For each user, we sort his ratings in ascending order of time and use his rating behavior in the top 80 percent record, together with the tag and category of the movie, to build tag field and category field to predict tags he might be interested in.

Metrics. Generally, loss function defined in Equation 4 is a standard way to evaluate model performance. Besides, we introduce a metric called "precision at K", marked as Prec@K and defined in Equation 5, denoting the proportion of the top K recommended tags for a user that hits his clicked tags.

$$Prec@K = \frac{1}{N}\sum_{i}^{N}\frac{size(P_i@K \bigcap C_i)}{min(K, size(C_i))} \tag{5}$$

where K denotes the number of tags to predict. Note that a tag can appear in multiple news articles and thus can be browsed and clicked multiple times. $P_i@K$ denotes the top K unique tags in user i's recommended tags ranked by the predicted score in descending order, C_i denotes the number of unique tags that user i has clicked, N represents the number of users in the test set. Notice that some users may click fewer than K tags, leading to an abnormally low Prec@K, so we use the smaller value between K and size of C_i. In the experiment, we compare models by Prec@K with K in {1, 5, 10, 20, 50} in the WeChat "Top Stories" recommender system dataset and K in {1, 2, 3} in MovieLens dataset.

Table 1: Parameter settings of our model

Para.	Industrial	MovieLens	Description
E	128	16	embedding dimension of feature input layer
H	100	10	max number of input categorical feature
T	32	8	size of query vector
M	8	2	number of feature fields
F	32	8	context dimension of AF layer
C	10	4	dimension of hidden vector in CF layer
U	128	16	dimension of final user embedding of CF layer
K	30	10	max number of training labels each user
N	5	3	max number of tags of each news article

Parameter Settings. We define the model parameters of each layer as shown in Table 1. These parameters are set according to a large amount of off-line experiments and this combination of parameters achieves the best performance. We fix these settings during our following experiments.

4.2 The improvements of attention mechanism

As for the YouTube model, it uses average-pooling to merge features within fields and then concatenates these features across fields. We firstly replace it with a one-head attention fusion mechanism but with separate query vectors named UTPM(AF-1head-unshared), which means there will be M attention units for M feature fields and another attention unit to merge all these fields. Then we keep all attention units with a shared query vector (AF-1head in Table 2). We observe that attention units with a shared query vector perform better. Note that the query vector decides the importance of features on this user-tag profiling task. With a shared query vector, features within fields and across fields are learned together and interact with each other, which in turn helps find a better query vector. After that, we add another head of attention called UTPM(AF-2head). Table 2 shows that two-head attention performs better than single-head attention. We argue that 2-head attention helps discover useful features from multiples aspects. As shown in Figure 3, we take a real user case and present the input tag feature fields. The first-head attention focus on the Chinese historical figures and the second-head attention on other interest like dogs and cars and football.

Tag (in English)	Tag (in Chinese)	Head1 Weight	Head2 Weight
Diabetes	糖尿病	0.002918885	0.118066988
Insulin	胰岛素	0.006801179	0.077416592
Automatic gear	自动挡	0.012230576	0.091979842
HAVAL, car brand	哈弗	0.002254583	0.062001974
Dog	狗狗	0.019319911	0.093495606
Golden retriever dog	金毛	0.0151551	0.06948731
Elkeson, football player	埃尔克森	0.057462535	0.123426514
National football player	国脚	0.037131165	0.106170456
Yi Jianlian, baseketball player	易建联	0.016758948	0.07442898
Warriors, baseketball team	勇士	0.01005347	0.062880735
Military rank	军衔	0.065956278	0.009978651
Founding marshal	开国元帅	0.073384985	0.015956701
People's Republic of China	新中国	0.072028226	0.00768709
Commander	司令员	0.079540594	0.015855259
Huang Yongsheng, general	黄永胜	0.084913066	0.020760404
People's Liberation Army	解放军	0.075850793	0.008306353
Xu Xianggian, marshal	徐向前	0.086728867	0.009187418
Su Yu, senior general	粟裕	0.088124182	0.010594392
Peng Dehuai, marshal	彭德怀	0.092335952	0.014220834
President	主席	0.101051215	0.008101121

Figure 3: Multi-head attention weight distribution for input tag features

4.3 The improvements of cross feature

Based on UTPM (AF-2head), we add cross feature layer (CF) to enhance the model capability, meanwhile, we also compare several other methods like AFM, NFM, DCN, AutoInt with the optimal parameters. As Table 3 shows, our method outperforms baseline models on both industrial and public datasets. Note that our cross feature networks actually learn and store each pair-wise feature interaction explicitly within a dense feature vector. The hidden vectors in our cross feature layer only decide how two features are correlated through inner product, while AFM and NFM use it as another feature representation through the Hadamard product. Experiments show that keeping all pair-wise interacted values, as in our cross feature layer, works better than summing up pair-wise interacted hidden vectors in AFM and NFM. DCN and AutoInt use a stack of cross networks that might work well on CTR tasks, where item-wise training mix user and item feature at the beginning. But for user profiling tasks, they are slightly weaker than AFM, which proves the effectiveness of FM-style cross feature methods.

4.4 The improvements of joint loss

Based on UTPM (AF-2head-CF), we test whether joint-loss training works or not. The previous group of experiments is based on separate training, where each tag of the clicked news article is trained as a single positive label. Here we replace it with the joint-loss learning aforementioned. As Table 4 shows, joint-loss promotes model performance because it learns each of user-tag preference jointly for a single clicked/un-clicked news article instead of treating them equally and separately as a positive or negative label. Note that in Table 3 and Table 2, we evaluate the performance of attention fusion layer and cross feature layer with separate loss. The conclusions still hold if we change to joint loss.

4.5 The performance of our model in online recommendation

Furthermore, to verify the performance of our model in an online environment, we conduct a series of online A/B tests on the WeChat "Top Stories" recommendation system. There are two important evaluation metrics in online experiments, named QC (quantity of click tags), and DCTR (de-duplicated click-through rate of tags).

From daily browsing history, we collect all users' clicked and un-clicked tags. As we aim at evaluating our user-tag profiling model, we only keep clicked and un-clicked tags that are within the user-tag profile we built while calculating the following metrics. QC is defined as the number of unique clicked tags. A higher QC indicates a higher coverage of user interests. DCTR is defined as the number of distinct clicked tags divided by the number of distinct browsed tags, which represents the accuracy of tag profiling. We de-duplicate all browsed tags because we care more about whether a user clicks the tag than the number of clicks for the tag.

Before conducting the online A/B test, we test models offline and choose the best model-structure and hyper-parameters. Due to limited resources, we only compare two versions of our model structure. In our first A/B test, we compare UTPM (AF-2head+CF) with the Youtube model. The result shows that both QC and DCTR increases, proving the effectiveness of the attention fusion and cross feature layer. After that, we replace separate training loss with joint-loss. Note that for separate training loss, user-tag preferences could be learned because different tags have different distributions of positive and negative samples. But for each click, separate loss

Table 2: The improvements of attention mechanism

versions	industrial data					movielens-20M		
	Prec@1	Prec@5	Prec@10	Prec@20	Prec@50	Prec@1	Prec@2	Prec@3
YouTube	0.4312	0.3384	0.2832	0.2450	0.2041	0.7485	0.7255	0.7010
UTPM (AF-1head-unshared)	0.4440	0.3390	0.2850	0.2456	0.2050	0.7545	0.7304	0.7056
UTPM (AF-1head)	0.4651	0.3412	0.2882	0.2456	0.2058	0.7554	0.7307	0.7070
UTPM (AF-2head)	0.4715	0.3439	0.2908	0.2476	0.2070	0.7576	0.7314	0.7103

Table 3: The improvements of cross feature

versions	industrial data					movielens-20M		
	Prec@1	Prec@5	Prec@10	Prec@20	Prec@50	Prec@1	Prec@2	Prec@3
UTPM (AF-2head)	0.4715	0.3439	0.2908	0.2476	0.2070	0.7576	0.7314	0.7103
+CF	0.4818	0.3546	0.2985	0.2532	0.2094	0.7614	0.7345	0.7092
+AFM	0.4787	0.3504	0.2957	0.2515	0.2094	0.7610	0.7339	0.7090
+NFM	0.4748	0.3491	0.2947	0.2509	0.2095	0.7590	0.7321	0.7072
+DCN	0.4723	0.3486	0.2942	0.2505	0.2088	0.7608	0.7351	0.7089
+AUTOINT	0.4739	0.3487	0.2945	0.2505	0.2090	0.7596	0.7323	0.7073

Table 4: The improvements of joint loss

versions	industrial data					movielens-20M		
	Prec@1	Prec@5	Prec@10	Prec@20	Prec@50	Prec@1	Prec@2	Prec@3
UTPM(AF-2head+CF)	0.4818	0.3546	0.2985	0.2532	0.2094	0.7614	0.7345	0.7092
+joint-loss	0.4957	0.3552	0.3018	0.2596	0.2193	0.7623	0.7345	0.7096

Table 5: Explainable cases translated from Chinese.

Feaure	Value	Top predicted video tags of V2	Top predicted video tags of V5
News Tag	iPartment (a situation comedy, 爱情公寓)	Stephen Chow	Stephen Chow
	Jiong He (TV show host, 何炅)	Mo lei tau	Mo lei tau
	Dying to Survive (movie, 我不是药神)	Classical Hong Kong films (经典香港电影)	Japanese anime
	Zheng Xu (director and actor of Dying to Survive, 徐峥)	Movie trailers	Classical Hong Kong films
Video Tag	Japanese anime (日漫)	Japanese anime	Movie trailers
	Stephen Chow (HK comedy actor, 周星驰)	Jinjiang Huang (Hong Kong comedy actor, 黄锦江)	The Negotiator (a TV show, 王牌对王牌)
	Mo lei tau (comedy type of nonsensical humor, 无厘头)	Flirting Scholar (film acted by Stephen Chow , 唐伯虎点秋香)	Crayon Shin-chan (Japanese anime character, 蜡笔小新)
	Movie trailers (电影片花)	Ng Man-tat (film patenter of Stephen Chow, 吴孟达)	Zheng Xu
	Spirited Away (a Japanese anime, 千与千寻)	Fight Back to School (film acted by Stephen Chow, 逃学威龙)	Ling Jia (TV show actress, 贾玲)
	Pokemon (a Japanese anime, 口袋妖怪)	Infernal Affairs (famous HK film, 无间行者)	Running Man China (reality TV show, 奔跑吧兄弟)
Gender	Male	The Deer and the Cauldron (film acted by Stephen Chow, 鹿鼎记)	Jacky Wu (TV show host, 吴宗宪)
Age	20	Slam Dunk (a Japanese anime about basketball sports, 灌篮高手)	Digital Monster (a Japanese anime, 数码宝贝)

Tags related to Japanese anime Tags related to Hong Kongese comedy actor Stephen Chow Tags related to popular TV show and latest movie

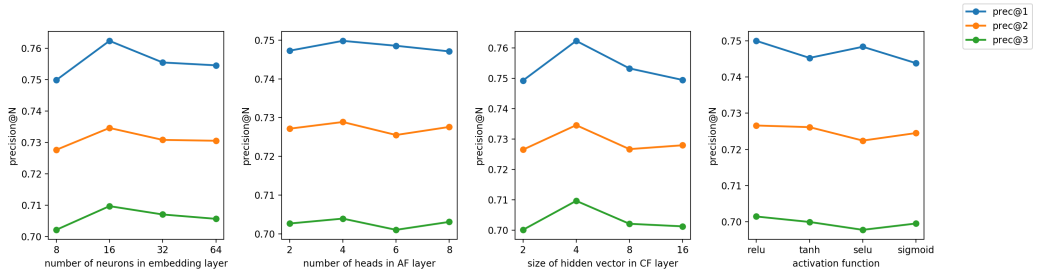

Figure 4: Evaluation of hyper-parameters

Table 6: The metric increase on online recommendation

Versions	QC	DCTR
UTPM (AF-2head+CF) vs Youtube	1.56%	0.52%
UTPM (AF-2head+CF+joint-loss) vs UTPM (AF-2head+CF)	1.36%	1.31%

treats all clicked tags equally, while joint-loss could learn different user-tag preferences. The experiment shows the superiority of joint-loss proved by higher QC and DCTR. Till now, our model has been used in the WeChat "Top Stories" platform for nealry 5 months.

4.6 Explainable Case

In an industrial news recommendation system, the recommendation list consists of both news articles and videos, so users can either read news articles or watch videos. News articles and videos have different tagging systems, so each user has both user-tag profile for

news and user-video-tag profile for video. We aim to understand whether incorporating user's news reading behavior can facilitate user-video-tag profiling.

User's demographic and behavioral information is used as an input feature to train the YouTube model and UTPM model and build a user-video-tag profile. We compare the top predicted tags and find that UTPM can recommend video tags that reflect the user's preference for news articles, while the YouTube model fails to incorporate reading behaviors.

As Table 5 shows, both models can predict video tags related to input video-tag features, but they differ greatly in terms of video tags approximating news-tag features. UTPM can predict video tags related to the news-tag field, as it discovers the famous TV shows *Running Man China* corresponding to user's interest in TV shows in news-tag and predicts *Zheng Xu*, which is also consistent with his input news-tag feature. However, the user's interests in the news-tag field are neglected in the YouTube model. For example, the user's news tags demonstrated his interest in TV shows and the latest films, but the YouTube model failed to output any related result. We argue that attention and cross feature mechanisms help to discover and fuse useful features across multiple fields.

4.7 The evaluation of hyper-parameters

We study the impact of hyper-parameters on UTPM. As Figure 4 shows, we evaluate four parts of hyper-parameters on the MovieLens dataset.

Number of neurons for the input feature layer. Precision firstly increases, and then decreases as the number of neurons becomes larger. As in Table 1, best performance is attained when $E = 16$. Note that tag embeddings in predicting layers and feature input layers have the same size.

Number of heads in attention fusion layer. Here we observe that the four-head attention mechanism achieves the best performance. An additional increase in the number of the head leads to over-fitting and exponential growth in computing time.

Number of heads in attention fusion layer. The hidden vector is the key component in the cross feature layer, determining the weight of pair-wise feature interaction. As in Table 1, we mark it as C. We get the best performance when $C = 4$.

Activation function. We evaluate four commonly-used activation functions including tanh, relu, selu and sigmoid and conclude that relu is the most suitable for our model, while sigmoid gets the worst performance at prec@1 and selu gets the worst performance at prec@2 and prec@3.

In this paper, we proposed a UTPM model that highlights a multi-head attention mechanism with shared query vectors, specially-designed cross feature layer, and joint loss function. In our model, such attention mechanism captures the important attributes within fields and assigns each field a reasonable weight, consequently outperforming both its YouTube model counterpart and single-head attention mechanism. In the specially-designed cross feature layer, crossed values along with linear values are fed to the next layer to generate final user embedding. Joint loss function caters to user-tag profiling task and it outperforms separate training in this scenario. In general, we have made meaningful attempts in building a user-tag profile with deep learning methods.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).
[2] Fabiano M Belém, Jussara M Almeida, and Marcos A Gonçalves. 2017. A survey on tag recommendation methods. *Journal of the Association for Information Science and Technology* 68, 4 (2017), 830–844.
[3] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 335–344.
[4] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 191–198.
[5] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. Deepfm: a factorization-machine based neural network for ctr prediction. *arXiv preprint arXiv:1703.04247* (2017).
[6] F Maxwell Harper and Joseph A Konstan. 2016. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2016), 19.
[7] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 355–364.
[8] Robert Jäschke, Leandro Marinho, Andreas Hotho, Lars Schmidt-Thieme, and Gerd Stumme. 2007. Tag recommendations in folksonomies. In *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer, 506–514.
[9] 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 the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1754–1763.
[10] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130* (2017).
[11] Yudan Liu, Kaikai Ge, Xu Zhang, and Leyu Lin. 2019. Real-time Attention Based Look-alike Model for Recommender System. *arXiv preprint arXiv:1906.05022* (2019).
[12] Wanvimol Nadee. 2016. *Modelling user profiles for recommender systems*. Ph.D. Dissertation. Queensland University of Technology.
[13] Steffen Rendle. 2010. Factorization machines. In *2010 IEEE International Conference on Data Mining*. IEEE, 995–1000.
[14] 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 the 28th ACM International Conference on Information and Knowledge Management*. 1161–1170.
[15] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.
[16] Jesse Vig, Shilad Sen, and John Riedl. 2009. Tagsplanations: explaining recommendations using tags. In *Proceedings of the 14th international conference on Intelligent user interfaces*. ACM, 47–56.
[17] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17*. ACM, 12.
[18] 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. *arXiv preprint arXiv:1708.04617* (2017).
[19] Deqing Yang, Yanghua Xiao, Hanghang Tong, Junjun Zhang, and Wei Wang. 2015. An integrated tag recommendation algorithm towards weibo user profiling. In *International Conference on Database Systems for Advanced Applications*. Springer, 353–373.
[20] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*. 1480–1489.
[21] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao. 2018. ATRank: An attention-based user behavior modeling framework for recommendation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
[22] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep interest evolution network for click-through rate prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 5941–5948.
[23] 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 the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1059–1068.