Mixture of Graph Enhanced Expert Networks for Multi-task Recommendation

Binbin Hu, Bin Shen, Ruize Wu, Zhiqiang Zhang, Yuetian Cao, Yong He, Liang Zhang, Linjian Mo, and Jun Zhou

Ant Group
{bin.hbb, ringo.sb, kezhui,wrz, lingyao.zzq, yuetian.cyt, heyong.h, zhuyue.zl, linyi01, jun.zhoujun}@antfin.com

Abstract. Multi-task learning (MTL), which jointly tackles multiple tasks through information sharing, has been widely applied to many recommendation applications. Recently, current efforts targeted for recommendation focus on learning task relationships based on the Multi-gate Mixture-of-Experts (MMoE) architecture with shared input features (*i.e.,* subtle feature engineering for user-item interaction). Recent evidences suggest the Graph Neural Network (GNN) as a powerful component in characterizing deep interaction context for recommendation, greatly contributing to easing the data sparseness issue in online advertising services. Hence, we make the first attempt to explore the GNN towards multi-task recommendation, by designing Mixture of Graph enhanced Expert Networks (**MoGENet**). Specifically, we propose a novel multi-channel graph neural network to jointly model high-order information with the user-item bipartite graph as well as derived collaborative similarity graphs for users and items. On the top of the learned deep interaction context, a group of graph enhanced expert networks are introduced for contributing to the multi-task recommendation in a cooperative manner. Experimental results on three real-world datasets show that MoGENet consistently and significantly outperforms state-of-the-art baselines across all target tasks.

Keywords: Multi-task Learning, Graph Learning, Recommender System

1 Introduction

The recent integration of multi-task learning into recommender systems (*e.g.,* the Multi-gate Mixture-of-Experts (MMoE) architecture [13] and its variants [12, 19]) has demonstrated remarkable strength of comprehensively capturing users' inherent preferences by jointly tackling multiple target behaviors (*e.g.,* click and conversion) [14, 26, 3, 10, 8, 25]. Unfortunately, its success hinges on the subtle feature engineering and large amounts of labeled data, which are not always easily available (*i.e.,* the undesired data sparseness issue). Although knowledge transfer among multiple tasks (*i.e.,* click and conversion) could be effectively captured by current multi-task recommendation methods, it is still infeasible to

obtain enough training data and handcrafted features for long-tailed items in industrial settings. With the increasingly available historical user-item interaction records, characterizing the high-order user-item based connectivity and the user (item) based collaborative similarity sheds some lights on easing the data sparseness issue and facilitating multi-task recommendation.

While it is appealing to distill the connectivity and collaborative similarity from interaction history for the multi-task recommendation, the solution is non-trivial, with several challenges. (1) *How to flexibly extract the most relevant/important information for characterizing interaction?* The high-order user-item connectivity, as well as the collaborative similarity, is complicated. It is imperative to design a unified module to produce expressive representations by exploring and exploiting useful graph structural information in an automatic manner. (2) *How to effectively incorporate learned deep interaction context for contributing to multi-task recommendation?* The learned deep interaction context is composed of different view (*i.e.,* connectivity and similarity) based representations , which drives us to develop a new architecture to incorporate such information for the multi-task recommendation in a cooperative manner.

Recent evidences show that graph neural networks [28, 9, 21] have the excellent ability in structural feature learning and have been widely adopted in recommender systems for characterizing interaction context [24, 31] and side information [2, 23]. Inspired by these works, we make the first attempt to investigate the effectiveness of the graph neural network to multi-task recommendation by designing Mixture of Graph enhanced Expert Networks (**MoGENet**) . For simultaneously capturing the high-order user-item based connectivity and the user (item) based collaborative similarity, we propose a novel multi-channel graph neural network (MGNN) module to flexibly learn the most relevant/important information on the bipartite graph as well as derived collaborative similarity graphs for users and items. Based on the learned deep interaction context, we introduce graph enhanced expert networks, which are built upon the architecture of MMoE to flexibly incorporate such context for contributing to the multi-task recommendation. To well guide the learning process of MoGENet, we weigh the loss of each task in an automatic manner and balance multiple parallel expert networks for recommendation in a cooperative way. At last, we conduct extensive experiments on a benchmark and two industrial datasets respectively to show the superiorities of MoGENet.

2 The Proposed Method

In this section, we present MoGENet, a novel multi-task recommendation method that leverages the deep interaction context with graph learning. The overall architecture of MoGENet is illustrated in Fig. 1. We start with the construction of the bipartite graph and derived collaborative similarity graphs based on historical user-item interaction records with the user set $\mathcal{U}$ and the item set $\mathcal{I}$. Here, each $\langle u, i\rangle$ is assigned K related labels (*e.g.,* click and conversion in e-commerce scenarios), denoted as $y_{u,i}$. Formally, we define the user-item **Bipartite Graph**

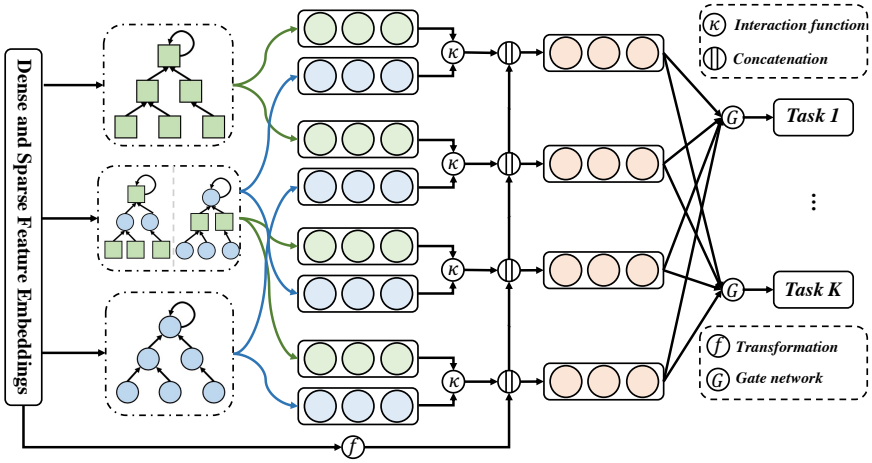

Fig. 1. The overall architecture of MoGENet, which consists of multi-channel graph neural network module and graph enhance expert networks.

as $\mathcal{G} = \{\mathcal{U}, \mathcal{I}, \mathcal{E}\}$ with the user set $\mathcal{U}$ and the item set $\mathcal{I}$. Moreover, $\mathcal{E} \subset \mathcal{U} \times \mathcal{I}$ is the edge set between $\mathcal{U}$ and $\mathcal{I}$ where $e_{u,i} = e_{i,u} = 1$ if and only if $\sum_{k=1}^{K} y_{u,i}^{k} \geq 1$.

In order to effectively exploit deep interaction context for recommendation, we further define collaborative similarity for users (items) with the similar click (clicked) and purchase (purchased) history. Hence, we define the **User based Collaborative Similarity Graph** as $\mathcal{G}^{\mathcal{U}} = \{\mathcal{U}, \mathcal{E}^{\mathcal{U}}\}$ with the user set $\mathcal{U}$ and edge set $\mathcal{E}^{\mathcal{U}} \subset \mathcal{U} \times \mathcal{U}$. Here, $e_{u,v} = e_{v,u} = sim(u, v)$ calculated by the common interaction history of the users [27]. Analogously, we also denote the **Item based Collaborative Similarity Graph** as $\mathcal{G}^{\mathcal{I}}$.

2.1 Deep Interaction Context Exploitation with Multi-channel Graph Neural Network

In order to take full advantage of the high-order connectivity in $\mathcal{G}$ and the collaborative similarity in $\mathcal{G}^{\mathcal{U}}$ and $\mathcal{G}^{\mathcal{I}}$, we propose a Multi-channel Graph Neural Network (MGNN) to effectively aggregate most informative neighbors in each channel (*i.e.,* graph). To avoid the over-parameterized issue, the proposed MGNN applies graph learning on each channel with shared initial representations for users and items. Specifically, we encode the feature vector (*i.e.,* $\mathbf{x}_u, \mathbf{x}_i$) of user u and item i into $\boldsymbol{p}_u, \boldsymbol{q}_i \in \mathbb{R}^d$, which are shared in the following propagations and aggregations with bipartite and collaborative similarity graphs.

MGNN Module In general, the underlying reason for the interaction between target user u and item i is fairly complicated, which may be contributed by not only the historical interaction records (*i.e.,* $\mathcal{G}$), but also the deep collaborative similarity between users (*i.e.,* $\mathcal{G}^{\mathcal{U}}$) and items (*i.e.,* $\mathcal{G}^{\mathcal{I}}$). Formally, let $\boldsymbol{h}_u^0 = \boldsymbol{h}_u^{\mathcal{U},0} =$

$\boldsymbol{p}_u$ and $\boldsymbol{h}_i^0 = \boldsymbol{h}_i^{\mathcal{I},0} = \boldsymbol{q}_i$ be the first input layers for the MGNN module with shared encoding for users and items. Subsequently, MGNN layers in each channel can recursively propagate the connectivity or collaborative similarity in different graphs by neighborhood aggregation. Here, we zoom into the detailed process as below.

MGNN to capture the user-item collaborative connectivity in $\mathcal{G}$. Given the l-th layer input $\boldsymbol{h}_u^l$ and $\boldsymbol{h}_i^l$ for each user u and item i respectively, we update the user and item representation $\boldsymbol{h}_u^{l+1}$ and $\boldsymbol{h}_i^{l+1}$ at the $(l+1)$-th layer with $\mathcal{G}$ as follows:

$$\begin{aligned} \boldsymbol{h}_u^{l+1} &= f(\boldsymbol{h}_u^l, \sigma(\mathbf{W} \sum_{i \in \mathcal{N}(u)} \alpha_{u \leftarrow i} \boldsymbol{h}_i^l)), \\ \boldsymbol{h}_i^{l+1} &= f(\boldsymbol{h}_i^l, \sigma(\mathbf{W} \sum_{u \in \mathcal{N}(i)} \alpha_{i \leftarrow u} \boldsymbol{h}_u^l)), \end{aligned} \tag{1}$$

where $\sigma(\cdot)$ is the LeakyReLU activation function, $\mathbf{W} \in \mathbb{R}^{d \times d}$ is the weight matrix for graph $\mathcal{G}$, $\mathcal{N}(u)$ and $\mathcal{N}(i)$ is the neighbor set of user u and item i in $\mathcal{G}$, respectively. $\alpha_{u \leftarrow i}$ and $\alpha_{i \leftarrow u}$ are implemented by the emerging attention mechanism [20], indicating how much information being propagated through the connectivity $u \leftarrow i$ and $i \leftarrow u$ in $\mathcal{G}$, respectively. $f(\cdot)$ aims to aggregate the messages propagated from corresponding neighbors, which is implemented as the LSTM-like operator, inspired by [29].

MGNN to capture the user (item) based collaborative similarity in $\mathcal{G}^U(\mathcal{G}^I)$. For each user u (item i), with its l-th layer representation $\boldsymbol{h}_u^{\mathcal{U},l}$ ($\boldsymbol{h}_i^{\mathcal{I},l}$), the updated representation $\boldsymbol{h}_u^{\mathcal{U},l+1}$ ($\boldsymbol{h}_i^{\mathcal{I},l+1}$) in $\mathcal{G}^{\mathcal{U}}$ ($\mathcal{G}^{\mathcal{I}}$) can be calculated as follows:

$$\begin{aligned} \boldsymbol{h}_u^{\mathcal{U},l+1} &= f(\boldsymbol{h}_u^{\mathcal{U},l}, \sigma(\mathbf{W}^{\mathcal{U}} \sum_{u \in \mathcal{N}^{\mathcal{U}}(u)} \alpha_{u \leftarrow u} \boldsymbol{h}_u^{\mathcal{U},l})), \\ \boldsymbol{h}_i^{\mathcal{I},l+1} &= f(\boldsymbol{h}_i^{\mathcal{I},l}, \sigma(\mathbf{W}^{\mathcal{I}} \sum_{i \in \mathcal{N}^{\mathcal{I}}(i)} \alpha_{i \leftarrow i} \boldsymbol{h}_i^{\mathcal{I},l})), \end{aligned} \tag{2}$$

where $\mathbf{W}^{\mathcal{U}}, \mathbf{W}^{\mathcal{I}} \in \mathbb{R}^{d \times d}$ is the weight matrix for $\mathcal{G}^{\mathcal{U}}$ and $\mathcal{G}^{\mathcal{I}}$, $\mathcal{N}^{\mathcal{U}}(u)$ and $\mathcal{N}^{\mathcal{I}}(i)$ is the set of collaborative similar neighbors of user u and item i respectively, and $\alpha_{u \leftarrow u}$ and $\alpha_{i \leftarrow i}$ controls the information propagation in $\mathcal{G}^{\mathcal{U}}$ and $\mathcal{G}^{\mathcal{I}}$ respectively.

Through propagation processes, MGNN refines the representations in each channel (*i.e.,* $\boldsymbol{h}_u^{l+1}, \boldsymbol{h}_i^{l+1}, \boldsymbol{h}_u^{\mathcal{U},l+1}, \boldsymbol{h}_i^{\mathcal{I},l+1}$) by aggregating most related information from different graphs at the l-th layer.

2.2 Graph Enhanced Expert Network

In this part, we investigate into the integration of learned graph representation for multi-task recommendation. As proved in previous works [13, 10, 17], the design of expert networks guarantees the soft-parameter sharing for modeling task relations and conflicts. Therefore, we build upon the architecture of multi-gate of mixture-of-experts (MMoE) to incorporate graph learning for facilitating multi-task recommendation.

Intuitively, the interactions between users and items are facilitated with not only the bipartite graph, but also corresponding collaborative similarity graphs. Given a user-item pair $\langle u, i\rangle$, after L layers' propagations, we obtain the final graph representation list $\{\boldsymbol{h}_u^L, \boldsymbol{h}_i^L, \boldsymbol{h}_u^{\mathcal{U},L}, \boldsymbol{h}_i^{\mathcal{I},L}\}$, which jointly captures user preferences and item attributes in different graphs. In order to take full advantage of information derived from different graphs, we propose to apply the graph representation interaction between user u and item i in a pair-wise manner.

$$\mathcal{Z} = \{\kappa(\boldsymbol{h}_u^L, \boldsymbol{h}_i^L), \kappa(\boldsymbol{h}_u^L, \boldsymbol{h}_i^{\mathcal{I},L}), \kappa(\boldsymbol{h}_u^{\mathcal{U},L}, \boldsymbol{h}_i^L), \kappa(\boldsymbol{h}_u^{\mathcal{U},L}, \boldsymbol{h}_i^{\mathcal{I},L})\}, \quad (3)$$

where $\kappa(\cdot,\cdot)$ is the interaction function, which can be set as element-wise product, concatenation or multi-layer perceptron, and the corresponding studies are detailed in the experiment part.

In our study, for convenience, we let $|\mathcal{Z}| = N$, which is also the number of expert networks in our model. We denote our expert networks as $\{f^n(\cdot)\}_{n=1}^N$, and equip each task k with a separate gating network $g^k(\cdot)$. The output of task k is calculated as follows:

$$\boldsymbol{o}_{u,i}^k = \boldsymbol{v}^T \sum_{n=1}^{N} g^k([\mathbf{x}_u \mid\mid \mathbf{x}_i])_n [f^n([\mathbf{x}_u \mid\mid \mathbf{x}_i]) \mid\mid \mathcal{Z}^n], \quad (4)$$

where "||" is the concatenation operator and $\boldsymbol{v}$ is the weight vector for output. Following the common strategies in previous related works [13, 10, 17], the n-th expert network $f^n(\cdot)$ is implemented as the identical multi-layer perceptron with the LeakyReLU activation function, while the k-th gating network $g^k(\cdot)$ is implemented as the simple linear transformation of the input with a softmax layer.

2.3 Model Learning

We guide the learning process of MoGENet by jointly optimizing multiple related tasks. For the k-th task, we adopt the commonly used binary cross entropy as the loss function. Hence, the main loss function can be formulated as follows:

$$\mathcal{L}_{main} = \sum_{k=1}^{K} \mathcal{L}^k = \sum_{k=1}^{K} \sum_{\langle u,i\rangle \in \mathcal{H}} \mathcal{C}(\hat{y}_{u,i}^k, y_{u,i}^k), \quad (5)$$

where $\mathcal{C}(\cdot,\cdot)$ is the cross entropy function. Inspired by the idea of homoscedastic uncertainty [6], we adaptively weigh the loss of each task during model training. Specifically, following [7, 22], we learn a relative weight for each task with the task-dependent uncertainty, and thus rewrite the above loss function as follows:

$$\begin{aligned} \mathcal{L}_{main}(\Theta, \Sigma) &= \sum_{k=1}^{K} \mathcal{L}^k(\theta^k, \sigma^k) \\ &\approx \sum_{k=1}^{K} \sum_{\langle u,i\rangle \in \mathcal{H}} \left(\tfrac{1}{{\sigma^k}^2} \mathcal{C}(\hat{y}_{u,i}^k, y_{u,i}^k) + \log({\sigma^k}^2) \right), \end{aligned} \quad (6)$$

Algorithm 1 Model training for MoGENet

Require: Bipartite Graph $\mathcal{G} = \{\mathcal{U}, \mathcal{I}, \mathcal{E}\}$, historical user-item interaction records $\mathcal{H} = \{u, i, y_{u,i} | u \in \mathcal{U}, i \in \mathcal{I}, y_{u,i} \in \{0,1\}^K\}$, user features $\{\mathbf{x}_u | u \in \mathcal{U}\}$ and item features $\{\mathbf{x}_i | i \in \mathcal{I}\}$.

```
Obtain G^U and G^I from G through collaborative similarity.
Initialize {P, Q} and other global parameters (e.g., W^U and W^I) for the MGNN module, and initialize {Θ, Σ} for graph enhanced expert networks.
while Not coverage do
    Sample a batch of interaction records H^B from H.
    /**MGNN module**/
    Collect L-hops neighbors for each users (or items) in H^B over G, G^U and G^I, respectively.
    Obtain initial representations for users and items in H^B with shared encoding in Eq. 2.
    for l = 0; l < L do
        Perform propagation in each channel (i.e., Eq. 3 ∼ 4).
    end for
    /**Graph enhanced expert networks.**/
    Apply the graph representation interaction as Eq. 5.
    for k = 1; k ≤ K do
        Calculate ŷ^k_{u,i} according to Eq. 6.
    end for
    Perform back propagation w.r.t. L, i.e., Eq. 10.
end while
```

where $\Theta = \{\theta^k\}_{k=1}^K$, $\Sigma = \{\sigma^k\}_{k=1}^K$ is the parameter set and task uncertainty set of MoGENet, respectively. Noting that $\log({\sigma^k}^2)$ serves as a regularizer to avoid overfitting.

As mentioned above, MoGENet contains multiple parallel expert networks, which is hoped to be beneficial to the final prediction in a cooperative manner [5, 16]. Hence, we propose an auxiliary cooperative loss to balance the load of each parallel expert network.

$$\mathcal{L}_{co} = -\sum_{\langle u,i \rangle \in \mathcal{H}} \sum_{k=1}^{K} \sum_{n=1}^{N} \sum_{m=n+1}^{N} g^k(\hat{\mathbf{x}}_{u,i})_n g^k(\hat{\mathbf{x}}_{u,i})_m. \tag{7}$$

By incorporating both of above losses together, the overall loss function for our model is defined as follows:

$$\mathcal{L} = \mathcal{L}_{main} + \beta \mathcal{L}_{co} + \lambda ||\Theta||^2, \tag{8}$$

where $\beta, \lambda > 0$ controls the cooperative loss and regularization term, respectively.

2.4 Discussion

Flexibility of MoGENet The proposed MoGENet is a flexible framework to leverage graph structure for promoting multi-task recommendation. If we ignore

the deep interaction context learned by the MGNN module, MoGENet degrades into the typical MMoE model [13] without graph learning. On the other hand, MoGENet can also naturally degrade into neural graph collaborative filtering model [24] when the collaborative similarity and graph enhanced expert network are overlooked. Moreover, MoGENet is also a general framework which can be easily extended to incorporated more complicated graph structure (*e.g.,* social graphs and item knowledge graphs).

Efficient training for industrial application For efficient numerical computation in the MGNN module, we follow the trick in [11] to perform attention calculation in each channel. Hence, the complexity and storage are both linear with the number of edges in each corresponding graph. In sum, the upper bound of complexity for the MGNN module is $\mathcal{O}(|\mathcal{E}|+|\mathcal{E}^{\mathcal{U}}|+|\mathcal{E}^{\mathcal{I}}|)$. In addition, we follow the similar strategy in [4] to randomly collect multi-hop (*i.e., L*) neighbors for each $\langle u, i\rangle$ pair in batch. Instead of the full graph, we feed the sub-graph organized by the sampled multi-hop neighbors into the MGNN module for training, which endows MoGENet with excellent scalability for large-scale data in real-world applications. The pseudocode of the training procedure for MoGENet is detailed in Algorithm 1. *Actually, MoGENet has been deployed in industrial advertising systems to support user-item interaction graphs, consisting of hundreds of millions of nodes and edges.*

3 Experiments

In this section, we conduct a series of experiments to evaluate the effectiveness of MoGENet on industrial and public datasets, respectively. To sum up, we aim to answer the following research questions:

- **RQ1**: Does our proposed model outperforms other state-of-the-art methods on both industrial and public datasets.
- **RQ2**: How does our proposed MGNN module improve existing multi-task recommendation methods.
- **RQ3**: How about the impact of key designs of MoGENet (*i.e.,* interaction function, loss function and graph enhanced experts).

Dataset and evaluation metrics We evaluate MoGENet on an industrial **Advertising dataset** (extracted as **Adv. A** and **Adv. B**) and a public **IJCAI CUP 2015 dataset** [1]. The detailed descriptions of the three datasets are summarized in Table 1. Following [33, 26], we adopt **GAUC** to measure the recommendation performance, which is a more fine-grained metric and proved to be more relevant to online performance. We also report **RelaImpr** [30, 18] to measure the relative improvement of MoGENet over the best baseline. It is also worthwhile to note **0.1%** improvement *w.r.t.* GAUC is remarkable in industrial scenarios.

[1] https://ijcai-15.org/index.php/repeat-buyers-prediction-competition

	Adv. A	Adv. B	IJCAI CUP 2015
#Train	2.67×10^6	2.18×10^6	3.81×10^6
#Test	1.45×10^5	1.72×10^5	9.69×10^5
#Nodes / #Edges / #Attr.	3.01×10^6 / 9.31×10^7 / 93		1.07×10^6 / 3.94×10^6 / N.A.
Tasks	Click/Conversion		Favorite/Purchase

Table 1. Statistics of datasets used in experiments.

Dataset	Adv. A		Adv. B		IJCAI CUP 2015	
Task	Click	Conv.	Click	Conv.	Favorite	Purchase
MLP	0.7340	0.7818	0.7138	0.8018	0.6049	0.7449
NGCF	<u>0.7383</u>	0.7811	<u>0.7207</u>	0.8027	<u>0.7431</u>	<u>0.7466</u>
Shared-Bottom	0.7288	0.7857	0.7051	0.8015	0.6044	0.7462
Cross-Stitch	0.7314	0.7857	0.7108	<u>0.8063</u>	0.6049	0.7445
ESMM	0.7261	0.7862	0.7009	0.7970	0.6055	0.7448
MMoE	0.7288	0.7860	0.7092	0.8033	0.6013	0.7369
SNR-trans	0.7327	<u>0.7889</u>	0.7159	0.8059	0.6040	0.7416
CGC	0.7307	0.7862	0.7084	0.8012	0.6030	0.7402
MoGENet	**0.7428****	**0.8051****	**0.7232****	**0.8092***	**0.7442***	**0.7477***
RelaImpr *v.s.* Best	**1.89%**	**5.61%**	**1.13%**	**0.42%**	**0.41%**	**0.44%**

Table 2. Overall performance comparison *w.r.t.* GAUC (We <u>underline</u> the best performance from the baselines for each comparison. We use "**(or *)" to indicate that improvement of MoGENet over the best performance from the baselines based on paired t-tests at the significance level of 0.01 (or 0.05).). "Conv." is short for "Conversion".

Compared Methods We mainly consider two kinds of representative prediction methods and corresponding variants:

- **Single-task methods**, which only contribute to one task with shared features (*i.e.,* **MLP**) or graph structure (*i.e.,* **NGCF**[2] [24]).
- **Multi-task methods**, which learn multiple task relationships (*i.e.,* **Shared-Bottom** [1], **Cross-Stitch** [3] [15], **ESMM** [4] [14], **MMoE** [5] [13], **SNR-trans** [6] [12], **CGC** [7] [19]).

[2] https://github.com/xiangwang1223/neural$_g raph_c ollaborative_f iltering$.
[3] https://github.com/helloyide/Cross-stitch -Networks-for-Multi-task-Learning.
[4] https://github.com/qiaoguan/deep-ctr-predicti- on/tree/master/ESMM
[5] https://github.com/drawbridge/keras-mmoe
[6] https://github.com/tomtang110/Multitask
[7] https://github.com/tomtang110/Multitask

Dataset	Adv. A		Adv. B		IJCAI CUP 2015	
Task	Click	Conv.	Click	Conv.	Favorite	Purchase
Shared-Bottom	0.7288	0.7857	0.7051	0.8015	0.6044	0.7462
+ MGNN	0.7333↑	0.7872↑	0.7189↑	0.8079↑	0.7439↑	0.7351↑
Cross-Stitch	0.7314	0.7857	0.7108	0.8063	0.6049	0.7445
+ MGNN	0.7309	0.7866↑	0.7175↑	0.8027	0.7378↑	0.7457↑
ESMM	0.7261	0.7862	0.7009	0.7970	0.6055	0.7448
+ MGNN	0.7363↑	0.7879↑	0.7177↑	0.8067↑	0.7417↑	0.7438
MMoE	0.7288	0.7860	0.7092	0.8033	0.6013	0.7369
+ MGNN	0.7370↑	0.7880↑	0.7140↑	0.8035↑	0.7314↑	0.7329
SNR-trans	0.7327	0.7889	0.7159	0.8059	0.6040	0.7416
+ MGNN	0.7352↑	0.7863	0.7185↑	0.8021	0.7366↑	0.7385
CGC	0.7307	0.7862	0.7084	0.8012	0.6030	0.7402
+ MGNN	0.7342↑	0.7862	0.7147↑	0.8046↑	0.7373↑	0.7314

Table 3. Performance comparison of different methods and their improved variants(*i.e.,* + MGNN) on tree datasets. ↑ means that variants achieve the better performance.

Dataset	Adv. A		Adv. B	
Task	Click	Conversion	Click	Conversion
EP	0.7428	**0.8051**	0.7232	0.8092
CON	0.7400	0.7961	0.7194	0.8121
ADD	**0.7436**	0.7954	0.7190	**0.8126**
MLP	0.7430	0.7986	**0.7250**	0.8072

Table 4. Study of the interaction function $\kappa(\cdot,\cdot)$. EP: element-wise product; CON: concatenation; ADD: Addition; MLP: multi-layer perceptron.

Implementation Details We implement all models in our experiments on parameter server based distributed learning systems [32], with the aims of scaling up to large-scale datasets adopted in the paper. For fair comparison, we set learning rate = 1e-4, regularizer = 1e-3, batch size = 256, embedding size = 64 and select Adam as optimizer for all models. Moreover, for MLP and ESMM, we set the architecture as [512, 256, 128]. For NGCF, we follow the optimal architecture reported in the original paper with 2 layers graph neural network. For Shared-Bottom, Cross-stitch, MMoE, SNR-trans and CGC, we set the number of expert networks = 4, and set the architecture of each expert network as [32, 32] and each gate network as [16, 16]. For MoGENet, we follow same architectures of expert networks and gate networks as MMoE, and set $L = 2$, $\beta =$ 1e-6, $\lambda =$ 1e-3. It is worthwhile to that all parameters for the comparison methods are optimized by using 10% training data as the validation set.

Dataset	Adv. A		Adv. B	
Task	Click	Conversion	Click	Conversion
i	0.7408	0.7940	0.7169	0.8098
i+ii	0.7424	0.7966	0.7180	**0.8110**
i+iii	0.7381	0.7877	0.7190	0.8084
i+ii+iii	**0.7428**	**0.8051**	**0.7232**	0.8092

Table 5. Study of the loss function. i: the original loss function; ii: the uncertain loss function; iii: the auxiliary cooperative loss.

3.1 Performance Comparison (RQ1)

We show the overall performance of the compared methods on industrial and public datasets in Table 2. Here, we have the following key observations:

- The proposed MoGENet model consistently and significantly achieves the best performance over baselines in all cases. This result validates the effectiveness of MoGENet, benefiting from incorporating the deep interaction context for multi-task recommendation with graph learning.
- Among baselines, multi-task methods outperform single-task methods in most cases, which indicates the usefulness of learning task relations during model training. This phenomenon is fairly obvious in the industrial dataset, since the relation between *Click* and *Conversion* maybe stronger than *Favorite* and *Purchase*.
- It is worthwhile to note that NGCF, as a single-task method, works well among these baselines. It indicates that graph structure plays a vital role in recommendation task, which further inspires the development of MoGENet.

3.2 Effect of the MGNN module

To demonstrate the effectiveness of graph representations learned by our MGNN module, we prepare a series of variants of above multi-task methods, which simply take the output of the MGNN module as input.

As shown in Table 3, the variants of multi-task methods (*i.e.,* + MGNN) yield better performance than the base by a relatively large margin in most cases, which is attributed to the useful graph structure learned by the MGNN module. Nevertheless, our MoGENet provides a more principled mechanism for combining graph learning and multi-task recommendation, and thus still outperforms these variants.

3.3 Study of MoGENet

In this section, we conduct a series of ablation studies on MoGENet to investigate the impact and rationality of the interaction function, loss function and graph enhanced experts.

Ablation study 1: interaction function. First of all, we investigate the impact of the interaction function $\kappa(\cdot,\cdot)$ in Eq. 3. In particular, we equip our MoGENet model with different interaction functions (*i.e.,* element-wise product (**EP**), concatenation (**CON**), addition (**ADD**) and multi-layer perceptron (**MLP**)) and study the corresponding performance.

As shown in Table. 4, we find that MoGENet yields relatively superior performance when the relatively simple interaction functions (*i.e.,* element-wise product and addition) are applied. A possible reason is that such an interaction function seems enough to capture affinity between users and items, and also increases the model representation ability. This observation is consistent with previous findings in [24].

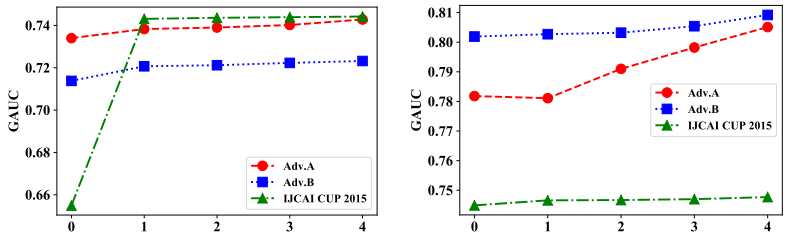

Fig. 2. Study of the graph enhanced experts.

Ablation study 2: loss function. As mentioned above, we improve the (i) original multi-task loss function (See Eq. 7) by (ii) incorporating task-dependent uncertainty loss (See Eq. 8) and (iii) auxiliary cooperative loss (See Eq. 9), with the aims of effectively learning relatedness and differences across multiple tasks.

As shown in Table. 5, we observe that the complete MoGENet (*i.e.,* "i+ii+iii") achieves the best performance in most cases compared with other variants. It demonstrates the rationality and effectiveness of the design of loss function, which provides a more reasonable way to learn knowledge transfer among multiple tasks.

Ablation study 3: graph enhanced experts. Finally, we analysis the impact of graph enhanced experts on the recommendation performance through gradually incorporating them into the proposed model (See Eq. 3).

As illustrated in Fig. 2, we observe that the performance of MoGENet overall improves with the incorporation of graph enhanced experts (as shown in Fig. 1 (b), we add experts from left to right). And the performance will drop a lot without graph enhanced experts in most cases. These observations demonstrate the effectiveness of graph learning for easing the data sparseness issue.

4 Conclusion

In this paper, we proposed MoGENet, a novel graph learning enhanced multi-gate mixture-of-experts framework for recommendation. We elaborately developed a MGNN module to jointly model the high-order connectivity and the collaborative similarity. On the top of learned structural information, a group of graph enhanced expert networks was introduced for contributing to multi-task recommendation during end-to-end training. Extensive experimental results have demonstrated the superiority of our model. As for future work, we will investigate into incorporating more complicated graph structures (*e.g.,* knowledge graphs and heterogeneous graphs) into our framework.

References

1. Caruana, R.: Multitask learning. Machine learning **28**(1), 41–75 (1997)
2. Fan, W., Ma, Y., Li, Q., He, Y., Zhao, E., Tang, J., Yin, D.: Graph neural networks for social recommendation. In: WWW. pp. 417–426 (2019)
3. Gao, C., He, X., Gan, D., Chen, X., Feng, F., Li, Y., Chua, T.S., Jin, D.: Neural multi-task recommendation from multi-behavior data. In: ICDE. pp. 1554–1557 (2019)
4. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: NIPS. pp. 1024–1034 (2017)
5. Hsu, W.T., Lin, C.K., Lee, M.Y., Min, K., Tang, J., Sun, M.: A unified model for extractive and abstractive summarization using inconsistency loss. In: ACL. pp. 132–141 (2018)
6. Kendall, A., Gal, Y.: What uncertainties do we need in bayesian deep learning for computer vision? In: NIPS. pp. 5574–5584 (2017)
7. Kendall, A., Gal, Y., Cipolla, R.: Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In: CVPR. pp. 7482–7491 (2018)
8. Kim, S., Ahn, N., Sohn, K.A.: Restoring spatially-heterogeneous distortions using mixture of experts network. In: ACCV (2020)
9. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: ICLR (2017)
10. Li, D., Li, X., Wang, J., Li, P.: Video recommendation with multi-gate mixture of experts soft actor critic. In: SIGIR. pp. 1553–1556 (2020)
11. Liu, Z., Chen, C., Li, L., Zhou, J., Li, X., Song, L., Qi, Y.: Geniepath: Graph neural networks with adaptive receptive paths. In: AAAI. pp. 4424–4431 (2019)
12. Ma, J., Zhao, Z., Chen, J., Li, A., Hong, L., Chi, E.H.: Snr: Sub-network routing for flexible parameter sharing in multi-task learning. In: AAAI. pp. 216–223 (2019)
13. Ma, J., Zhao, Z., Yi, X., Chen, J., Hong, L., Chi, E.H.: Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In: SIGKDD. pp. 1930–1939 (2018)
14. Ma, X., Zhao, L., Huang, G., Wang, Z., Hu, Z., Zhu, X., Gai, K.: Entire space multi-task model: An effective approach for estimating post-click conversion rate. In: SIGIR. pp. 1137–1140 (2018)
15. Misra, I., Shrivastava, A., Gupta, A., Hebert, M.: Cross-stitch networks for multi-task learning. In: CVPR. pp. 3994–4003 (2016)

16. Niu, X., Li, B., Li, C., Tan, J., Xiao, R., Deng, H.: Heterogeneous graph augmented multi-scenario sharing recommendation with tree-guided expert networks. In: WSDM. pp. 1038–1046 (2021)
17. Qin, Z., Cheng, Y., Zhao, Z., Chen, Z., Metzler, D., Qin, J.: Multitask mixture of sequential experts for user activity streams. In: SIGKDD. pp. 3083–3091 (2020)
18. Shen, Q., Tao, W., Zhang, J., Wen, H., Chen, Z., Lu, Q.: Sar-net: A scenario-aware ranking network for personalized fair recommendation in hundreds of travel scenarios. In: CIKM. pp. 4094–4103 (2021)
19. Tang, H., Liu, J., Zhao, M., Gong, X.: Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations. In: RecSys. pp. 269–278 (2020)
20. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: NIPS. pp. 5998–6008 (2017)
21. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. In: ICLR (2018)
22. Wang, M., Lin, Y., Lin, G., Yang, K., Wu, X.m.: M2grl: A multi-task multi-view graph representation learning framework for web-scale recommender systems. In: SIGKDD. pp. 2349–2358 (2020)
23. Wang, X., He, X., Cao, Y., Liu, M., Chua, T.S.: Kgat: Knowledge graph attention network for recommendation. In: SIGKDD. pp. 950–958 (2019)
24. Wang, X., He, X., Wang, M., Feng, F., Chua, T.S.: Neural graph collaborative filtering. In: SIGIR. pp. 165–174 (2019)
25. Wang, X., Yu, F., Dunlap, L., Ma, Y.A., Wang, R., Mirhoseini, A., Darrell, T., Gonzalez, J.E.: Deep mixture of experts via shallow embedding. In: UAI. pp. 552–562 (2020)
26. Wen, H., Zhang, J., Wang, Y., Lv, F., Bao, W., Lin, Q., Yang, K.: Entire space multi-task modeling via post-click behavior decomposition for conversion rate prediction. In: SIGIR. pp. 2377–2386 (2020)
27. Wu, Q., Zhang, H., Gao, X., He, P., Weng, P., Gao, H., Chen, G.: Dual graph attention networks for deep latent representation of multifaceted social effects in recommender systems. In: WWW. pp. 2091–2102 (2019)
28. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive survey on graph neural networks. IEEE Transactions on Neural Networks and Learning Systems (2020)
29. Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K.i., Jegelka, S.: Representation learning on graphs with jumping knowledge networks. In: ICML. pp. 5449–5458 (2018)
30. Yan, L., Li, W.j., Xue, G.R., Han, D.: Coupled group lasso for web-scale ctr prediction in display advertising. In: ICML. pp. 802–810 (2014)
31. Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W.L., Leskovec, J.: Graph convolutional neural networks for web-scale recommender systems. In: SIGKDD. pp. 974–983 (2018)
32. Zhou, J., Li, X., Zhao, P., Chen, C., Li, L., Yang, X., Cui, Q., Yu, J., Chen, X., Ding, Y., et al.: Kunpeng: Parameter server based distributed learning systems and its applications in alibaba and ant financial. In: SIGKDD. pp. 1693–1702 (2017)
33. Zhu, H., Jin, J., Tan, C., Pan, F., Zeng, Y., Li, H., Gai, K.: Optimized cost per click in taobao display advertising. In: SIGKDD. pp. 2191–2200 (2017)