2021 IEEE 37th International Conference on Data Engineering (ICDE) | 978-1-7281-9184-3/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICDE51399.2021.00287

IntelliTag: An Intelligent Cloud Customer Service System Based on Tag Recommendation

Minghui Yang, Shaosheng Cao*, Binbin Hu, Xianling Chen, Hengbin Cui,
Zhiqiang Zhang, Jun Zhou and Xiaolong Li
{minghui.ymh, shaosheng.css, bin.hbb, baizi.cxl, alexcui.chb, lingyao.zzq, jun.zhoujun, xl.li}@antfin.com
Ant Group, Hangzhou, China

***Abstract*—To reduce the customer service pressure of small and medium-sized enterprises, we propose an intelligent cloud customer service system, called *IntelliTag*. Unlike traditional customer service, a cloud service based system has difficulty in collecting user personal information. Therefore, we add a tag recommendation function to quickly capture the user's question intent by clicking on the tags. Specifically, IntelliTag is elaborately designed with the consideration of the following three aspects. First, how to mine high-quality tags is a challenging problem. Second, in the tag recommendation tasks, we have multifarious data types and relations that are used to build a sequential recommendation model. Finally, system implementation and deployment also need to be carefully designed to satisfy online service requirements. In this paper, we show the details of data construction, model designs, system implementation and deployment, and the empirical results compared with several state-of-the-art methods. Nowadays, our IntelliTag has already supported hundreds of thousands of enterprises and millions of users in our industrial production environment.**

***Index Terms*—intelligent customer service, chatbot, tag recommendation, graph neural network, sequential recommendation**

I. Introduction

In recent years, intelligent customer service has achieved great success in large enterprises, significantly reducing labor costs and improving response efficiencies, such as Microsoft's Superagent [11], Alibaba's AliMe [66], JD.com's JIMI [99] and etc. However, it is not an easy thing for most SMEs (Small and Medium-sized Enterprises), due to the limitation of their technical capabilities and capital reserves. As a result, manual customer service often brings them higher operating cost conversely.

The traditional intelligent customer service dialogue system is capable of collecting user personal information and historical behavior trajectories to understand the user's intention of questions [5], [11]. For example, in the area of E-commerce, a user has only one online purchase record in the past few weeks, and checked the logistics situation of the product before inquiring about customer service. Based on such information, intelligent customer service can easily infer that the user is most likely asking about the purchased items' logistics. In contrast, it is difficult for a cloud service based system, since assessing user's data in products is impossible due to data privacy reasons.

To address this problem, we aim to introduce a better interactive mode besides regular Q&A (Question and Answering) dialogue, which is able to take advantage of the information in a consultation session. Based on our investigation, several newly emerging chatbot assistants [47] have a tag feature, where a user gets answers by clicking recommended tags. In light of this idea, we add tag recommendation into our system, as illustrated in Fig. 1, where the shown tags are suspended in the input message dialogue box. By user's clicked tags, we can quickly speculate the user's intent of questions.

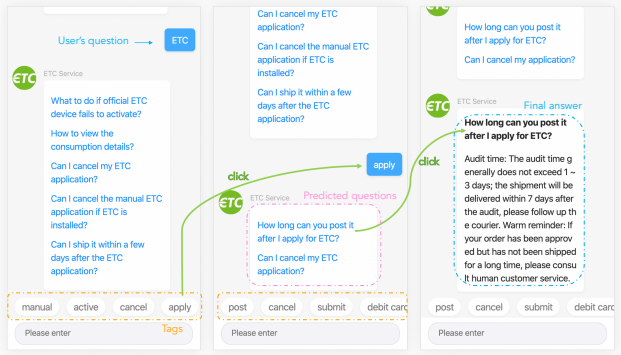


Fig. 1: The interface of our system based on tag recommendation. The left subfigure demonstrates the user's question and recommended tags. After clicking the second tag "apply", a new set of tags and predicted questions are recommended in terms of user's intent in the middle subfigure. And the final answer to the user's question is shown in the right subfigure after the user clicks on the predicted question.

In this scenario, we create an intelligent cloud customer service system with tag recommendation, known as *IntelliTag*. While SMEs, also called "tenants", can rent our customer service through cloud service at a meager rate. Without the user's personal data, we collect historical interactive behaviors in sessions, besides Q&A dialogues. The core of the first problem is how to mine useful and suitable tags efficiently and accurately.

On the task of tag mining, we aim to extract complete, representative and question-related tags, where a tag contains one word or multiple words. Therefore, two subtasks need to be solved: one is to distinguish tag boundary for its completeness, and the other is to measure tag's representative and relatedness to questions. Compared to two single-task

*Shaosheng Cao is the corresponding author.

978-1-7281-9184-3/21/$31.00 ©2021 IEEE
DOI 10.1109/ICDE51399.2021.00287

solutions, multi-task learning looks more promising due to their potential relationship of text semantics.

More important and difficult is TagRec (Tag Recommendation) task. Recently, session-based recommendation methods [85], [87] obtain satisfactory results to recommend products and movies. Moreover, sequence-based approaches [39], [73] are good at modeling sequences in the recommender system. However, it is still challenging to extract fruitful information from our various data and address a sequential recommendation problem. Besides, conventional sequential recommendation suffers from cold start issues.

From the model perspective, we collect multifarious data types and relations, where a heterogeneous graph ought to be constructed for reducing the impact of data sparsity in cold start. On the one hand, a heterogeneous graph can also alleviate the long tail problem to a certain extent, since it explores the topological information of unpopular tags in a network. On the other hand, the valuable sequential clicks' information also deserves careful consideration. We propose a novel hierarchical end-to-end model based on multiple attention mechanisms, which leverages node attention, metapath attention and contextual attention to respectively measure the relationship between different node neighbors, the importance of information transmission paths and the influence of sequential behaviors.

Further, system implementation and deployment are also crucial to real-world industrial customer service. Our online service, including response to answers, recommended tags and predicted questions, should be finished within tens of milliseconds. Nowadays, our deployed system in an industrial production environment has already supported hundreds thousands of SMEs and millions of users, where more than ten thousand customer questions are solved every day.

Our contributions can be summarized as follows:

- To the best of our knowledge, it is the first work to reveal model and system details of intelligent cloud customer service with tag recommendation deployed in the real world;
- An automatic way is introduced to collect Q&A pairs, and a new BERT-based multi-task learning model is proposed to mine valuable tags;
- We propose a novel model for tag recommendation, which significantly outperforms several state-of-the-art baseline algorithms both on offline and online evaluations;
- System implementation and deployment details are uncovered, which guarantees online service timely response with about 100 ms.

II. Related Work

In this section, we review related studies in Q&A chatbot, tag recommendation, graph neural network based recommendation and sequential recommendation.

A. Question and Answering Chatbot

Intelligent personal assistant, such as Microsoft's Cortana[1], Amazon's Alexa[2], Apple's Siri[3], Baidu's Xiaodu[4] and Alibaba's TmallGenie[5], provides new human-computer interaction experience to help people complete their daily affairs. As a comparison, intelligent customer service system, such as Microsoft's SuperAgent [11], Alibaba's AliMe [66], jd.com's JIMI [99] and Ant Financial's AntProphet [5], aims to solve the user's questions about products and services. Our IntelliTag helps SMEs solve their customer service issue through its cloud service. From a technical perspective, rule-based and template-based methods were first explored [82], [84]. Data-driven approaches have recently achieved promising performance, and two mainstreams are gradually formed, including information retrieve [38], [40], [90], [91] and sequence-to-sequence generation based techniques [48], [49], [71]. However, most previous studies focus on the dialogue system and seldom explore tag recommendations in this field.

B. Tag Recommendation

Tag recommendation, as a problem of recommendation area, are often divided into two main categories: user-centered and tag-centered cases. The former builds the foundation on the user's personal data to predict tags. Multilayer perceptron models were first applied to address the problems in recommender systems, such as Wide and Deep [9], auto-encoder [79], deep semantic neural networks [88], [89]. Factorization-based methods [28], [70] were later introduced and achieved better effectiveness. In contrast, the task of tag-centered recommendation mines the associated relationship between tags and documents. Besides efficient linear FastText [43], CNN (Convolutional Neural Network) was introduced to measure the semantic information [83]. Also, LSTM (Long Short-Term Memory) based models [51], [98] were adopted to characterize the sequential nature of the text, and more methods with attention were introduced to describe the importance of local information [26], [52], [58]. Our task locates in a tag-centered scenario, different from previous studies due to the integration of various information.

C. Graph Neural Network based Recommendation

A graph is a kind of data structure to build a bridge between different types of entities. In recent years, GNN (Graph Neural Network) based approaches attract much attention. Several unsupervised learning based models [3], [15], [27], [64] were first proposed to generate node embeddings using the topological information of graph structure. With the full use of label information, graph convolutional network based approaches [30], [45], [61] achieved better performance. To characterize the fine-grained interaction in graphs, various attention based GNNs are proposed to locate important or

[1]https://www.microsoft.com/en-us/cortana
[2]https://developer.amazon.com/en-US/alexa
[3]https://www.apple.com/siri
[4]https://dumall.baidu.com
[5]https://en.wikipedia.org/wiki/Tmall_Genie

TABLE I: Illustrated examples for our mined tags.

Tags	RQs in Q&A pairs
change, password	How to **change password**?
apply for, ETC card	How can I **apply for ETC card**?
cancel, order	Where to **cancel** the **order**?
charge, phones	Can I **charge** multiple **phones** simultaneously?
initial VPN password	What is the **initial VPN password**?

relevant neighbors in an end-to-end manner [55], [78], [96]. Recently, several efforts [19], [21], [23], [35], [42], [80], [95] are also made to learn effective representations on heterogeneous graphs or knowledge graphs for recommendation. To solve session-based recommendation problems with GNNs, a series of approaches [8], [67], [81], [85], [87] are proposed and achieve considerable performance improvement. Further, recently proposed efficient graph databases [2], [14], [20], [29] and industrial-scale graph learning systems [7], [41], [92] provide effective support to various commercial scenarios in the real world [4], [53]. Nevertheless, most of them hardly measure sequential data.

D. Sequential Recommendation

In a session of our customer system, the interaction with a user is regarded as a sequential process, so we also investigate related literature in the area. Early algorithms were mainly based on Markov Chains, such as Markov decision processes [65] and factorizing personalized Markov Chains [69]. With the popularity of RNN (Recurrent Neural Network), LSTM and GRU (Gated Recurrent Units) become mainstream models for sequence recommendation, such as DREAM [94], user-based GRU [16], NARM [50] and GRU4Rec [39]. Following this line, a surge of variants are proposed to make more subtle feature extraction for sequences with memory network [36], [37], [57], reinforcement learning [86], self-supervised learning [97], ensemble learning [63] and so on. Most recently, BERT-based approaches [13], [73] achieved state-of-the-art results in sequential recommendation. These methods aim to measure the property of sequence but fail to leverage our task-related information.

III. Tag Mining from Knowledge Base Document Collection

In this section, we describe our proposed method of mining suitable tags on tag mining task. Different from the traditional social tagging tasks [25], [62], our goal is to mine representative and question-related words as tags from dialogue corpus.

A. Knowledge Base Document Collection

Consistent with traditional customer service, our IntelliTag supports Q&A dialogue feature. Following IR (Information Retrieve) -based approaches [90], we collect Q&A pairs and write them into KB (Knowledge Base) document warehouse, where each pair contains a question and its corresponding answer. When a user proposes a question, we first retrieve the most possible RQ (Representative Question)[6] from KB document and feedback the answer. More importantly, our recommended tags also originate from such document.

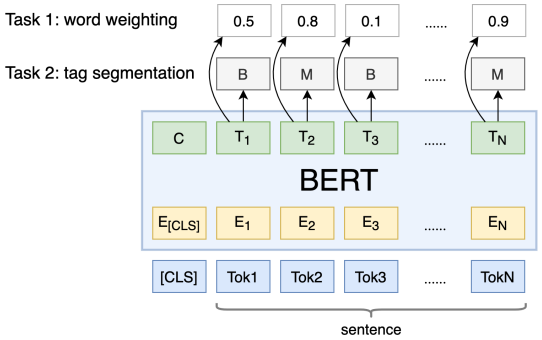


Fig. 2: The model of our BERT-based multi-task learning. "B" and "M" in gray rectangle indicate the word position at the Begin and in the Middle respectively.

Industrial customer service made by large enterprises can precipitate massive Q&A pairs, as for long-term experience accumulation by manual customer service. For each of our tenants, we provide an interface so as to upload their self-ordained frequent Q&A pairs. However, according to our statistics, only a small amount is added in this way, probably because it takes a lot of time and energy, but most SMEs do not have adequate energy for operations.

To address this problem, we design an automatic Q&A pairs collection way. First, we mix user's frequently proposed questions and RQs and utilize Transformers [77] to generate text embeddings which are subsequently fed into DBScan [18] to obtain question clusters. For each of the clusters, if there is not even an RQ, we randomly choose a user's question as a new one. Next, we select the user's high-rated content to questions replied by manual customer service, and apply machine reading comprehension [6], [31], [68] to find out suitable answers. Finally, new Q&A pairs will be continuously uploaded into our KB document warehouse.

B. Tag Mining from Knowledge Base Document

As we gradually accumulate massive Q&A pairs, we aim at mining tags that quickly capture the intent of the user's questions. As illustrated in Table I, tags could be a one-word or a multi-word term exploited from RQs, such as "change" and "apply for", which should be complete, representative and question-related. To mine suitable tags from massive RQs, we have to face two tasks, *i.e.,* tag segmentation and weighting. Tag segmentation task is to discover tag boundary, and tag weight is a defined metric to measure question representative and relatedness.

Intuitively, these tasks are highly related to text semantics, so we decide to adopt multi-task learning. We propose a novel

[6]To conveniently distinguish the questions in KB document and the questions proposed by users, we use "representative question" to denote the former, but we continue to use the common expression of "Q&A pair(s)".

BERT [13] -based multi-task model that jointly learns tag segmentation and word weighting simultaneously. As shown in Fig. 2, we feed the sentences of RQs as the input of the model, which outputs the predicted results of two tasks. Afterword weights are obtained, we average the values of word weights within a tag as tag weight. Finally, the tags with a weight greater than the preset threshold are retained. The task of tag mining is closely related to phrase mining, whose goal is to excavate suitable phrases from the documents. Different from traditional phrase mining methods [12], [17], [72], our BERT-based model can extract deep semantic information with its multiple neural network layers from the labeled sentences, and the knowledge is also fully exploited by the pretrained model with massive unlabeled corpora.

To further purify preliminary tags output by the model, we deeply consider several simple but effective rules as post-processing from the following aspects: (1) tag weight measures question representative and relatedness, which we obtain from the proposed model; (2) tag frequency indicates the importance of tag words that we calculate based on the whole KB document; (3) tag IDF (Inverse Document Frequency) is a critical information indicator to measure the influence of a tag; and (4) averaged PMI (Point-wise Mutual Information) [10] between any two words in a tag, reflects word's semantic consistency within a tag.[7]

In a consultation session, the user's successive clicked tags are composed as a query, which we use to retrieve a question list from RQs as predicted questions, as early demonstrated in Fig. 1. Since there is a lot of new daily questions, we need to perform the inference process of the model and generate tags every day. It is a remarkable fact it takes much time for BERT inference process. Motived by [32], we adopt knowledge distillation, which is much faster with little change in accuracy. We will report the effectiveness and efficiency results in Section VI.

IV. The Design Details of Our Model on Tag Recommendation Task

In this section, we construct a heterogeneous graph with a variety of types and relations, and elaborately analyze tag information transmission in the graph. Further, we propose a novel end-to-end hierarchical model, called TagRec model.

A. Heterogeneous Graph Construction and Metapath Design

We have already described the process of mining tags from RQs, where their association relation was naturally established. Based on our analysis, most data of low-frequency tenants is very sparse, so we aggregate RQs from different tenants. Until now, there are three different types: **T** (Tags), **Q** (RQs), **E** (tEnants). We then build four kinds of relations described as follows:

- **asc**: association relation between tags and RQs (T-Q), which represents inclusion relation between them;
- **crl**: correlation relation between RQs and tenants (Q-E), which describes mapping relation between them;
- **clk**: co-clicking relation between two tags (T-T), which indicates a user's two successively clicked tags in a session;
- **cst**: co-consulting relation between two RQs (Q-Q), which expresses two retrieved RQs with a user's two successively proposed questions.

Relations are built on a variety of information, where *asc* relation is obtained from text mining, *crl* relation is established between tenants and RQs, *clk* and *cst* relation are created by session-based interactive behaviors. We next build a heterogeneous graph to incorporate such types and relations, where a formal definition is given as follows:

Definition 1: **TagRec heterogeneous graph**. A TagRec heterogeneous graph is a graph denoted as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, consisting of a node set $\mathcal{V}$ and an edge set $\mathcal{E}$ on TagRec task. It is also associated with a node type mapping function $\phi : \mathcal{V} \rightarrow \mathcal{A}$ and an edge type mapping function $\varphi : \mathcal{E} \rightarrow \mathcal{R}$, where $\mathcal{A} = \{\text{T, Q, E}\}$ and $\mathcal{R} = \{\text{asc, crl, clk, cst}\}$.

Since the goal of our TagRec task is to extract plentiful tag information, we carefully analyze their transmission paths. Metapath is first introduced by [75] and successfully applied in many effective models [15], [34], [74], [80]. Based on our analysis, we deliberately predefine four metapaths to measure information transmission between two tags:

- **TT**: two connected tags are successively clicked by a user in a session;
- **TQT**: two connected tags are associated with the same RQ;
- **TQQT**: two connected tags are associated with two related RQs, retrieved by a user's successively proposed questions;
- **TQEQT**: two connected tags are mined from the KB document warehouse sourced from the same tenant.

We futher give the formal definition of TagRec metapath and TagRec metapath set as follows:

Definition 2: **TagRec Metapath and TagRec Metapath Set.** A TagRec metapath ρ is defined as an information transmission path starting and ending with tags, in the form of $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} ... \xrightarrow{R_l} A_{l+1}$, abbreviated as $A_1 A_2 ... A_{l+1}$, where $A_1 = A_{l+1} = T$. A TagRec metapath set is defined as a set $\mathcal{P}$ of TagRec metapaths, where $\mathcal{P} = \{\text{TT, TQT, TQQT, TQEQT}\}$.

For example, the composite relation of TagRec metapath *TQQT* can also be expressed as $T \xrightarrow{asc} Q \xrightarrow{cst} Q \xrightarrow{asc} T$.

B. Structural Information Excavation from TagRec Heterogeneous Graph

We will explore and exploit valuable structural information implied in the heterogeneous graph with neighbor attention and metapath attention. As shown in Fig. 3, we first respectively aggregate four types of neighbors to the target node t with neighbor attention, and then incorporate the results from different metapaths to generate tag embedding $\mathbf{z}_t$ of node t.

[7]We simply set the same weight for each rule.

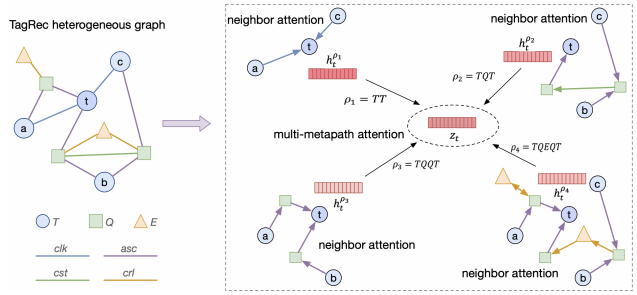


Fig. 3: An illustrated example of learning tag embedding with neighbor attention and metapath attention.

We aim to make the relatedness of two related tags higher than other unrelated pairs, which can be converted into the question of neighbor relationship analysis. Most intuitively, we can directly use edge information of the graph as tag neighbor. However, as illustrated in Section IV-A, two tags connected by a metapath can also be treated as a certain type of neighbor, where different metapaths mean different task implications. Actually, an edge-based neighbor is also a special case as the metapath is just *TT*.

Attention mechanism is successfully utilized in many areas, such as image recognition [22], [60], machine translation [1], [56] and graph neural networks [78], [80]. Motived by these previous works, we use attention mechanism for measuring the importance of each neighbor, called neighbor attention. Following [34], [80], we aggregate neighbors with respect to metapath ρ:

$$\mathbf{h}_t^\rho = \text{NeighborAttention}(\mathbf{x}_t; \rho) \tag{1}$$

where $\mathbf{x}_t$ is the feature vector of node t, and $\mathbf{h}_t^\rho$ represents target embedding of node t aggregated with neighbor attention via metapath ρ.

After measuring the relatedness with node neighbors, we next consider the influence of metapaths to target node, called metapath attention. Motived by [34], [80], we adaptively learn the importance of different metapaths. Given a target tag t, we can obtain a set of representations $\{\mathbf{h}_t^\rho\}_{\rho\in\mathcal{P}}$, where $\mathcal{P}$ is the set of TagRec metapaths as we mentioned before. We aim to fuse multifarious semantic information implied in multiple metapaths for more comprehensive tag representations, which is expressed abstractly as:

$$\mathbf{z}_t = \text{MetapathAttention}(\{h_t^\rho\}_{\rho\in\mathcal{P}}; \mathcal{P}) \tag{2}$$

where $\mathbf{z}_t$ denotes tag embedding of target node t aggregated by multi-metapath attention.

C. Sequential Modeling with Transformer Layers

With the help of hierarchical attention mechanism, *i.e.,* neighbor attention and metapath attention, for capturing rich semantics from the heterogeneous graph, we can obtain meaningful embedding for each tag with structural information. In TagRec task, it is essentially a sequential prediction problem. Specifically, given the user's clicked tags in a session, we aim to predict the next click and recommend tags by means of a ranking list. Formally, given a tag t $\in \mathcal{T}$, where tag set $\mathcal{T} \subseteq \mathcal{V}$, we collect N tags successively clicked by a user as context, denoted as $\mathcal{C} = \{t_1, t_2, ..., t_N\}$. Our goal is to predict the tag probabilities exhibited at $N+1$ time.

In the natural language processing area, modeling sentences is treated as a typical sequence problem, which has been widely explored in precious studies [24], [33], [77]. Particularly, Transformer has shown its effectiveness in many applications for measuring sequences [13], [73]. We add each tag embedding with corresponding position embedding and concatenate them together. The input consists of N clicked tag embeddings (*i.e.,* $\mathbf{z}_1 \sim \mathbf{z}_N$), and we use Transformer layers followed by a projection layer as the output layer to yield predicted tag probabilities at $N+1$ time:

$$\hat{\mathbf{y}} = \text{TransformerProjection}(\{\mathbf{z}_i\}_{i=1}^N) \tag{3}$$

where $\hat{\mathbf{y}}$ indicates tag probabilities at $N+1$ time.

Finally, we impose cross entropy[8] loss between predicted probabilities and ground truth (clicked tag at $N+1$ time), and backpropagate gradient errors. More technical details are described in Appendix Section.

D. End-to-End Training

Until now, we extract structural information from target tags with graph-based layers, and measure sequential information with sequence-based layers. In our design, the trainable parameters in the inner graph-based layer are shared, whose output (*i.e.,* tag embeddings) is fed as the input of outer sequence-based layers. Specifically, during the training process, we predict the results and calculate the loss of the model combined with label information, and backward update Transformer parameters. Different from the traditional step-by-step training pipeline, we adopt an end-to-end training mode. In other words, we further adjust the values of tag embeddings, and propagate gradient errors to sharable graph-based layers.

V. IntelliTag System Implementation and Deployment

In this section, we describe the implementation and deployment details of our IntelliTag system.

A. Implementation Details

Overall, the system has two main sections, including online real-time service and offline periodical training and inference. We provide two features for online service: Q&A dialogue and tag recommendation. Through the provided interface, users might input their questions and receive prompt responses, and they also can easily obtain the predicted questions and corresponding answers by clicking on recommended tags.

As shown in Fig. 4, **model server** and **ElasticSearch**[9] are two important components for supporting online service. As soon as a user proposes a question, the interface timely reports

[8]https://en.wikipedia.org/wiki/Cross_entropy
[9]https://www.elastic.co

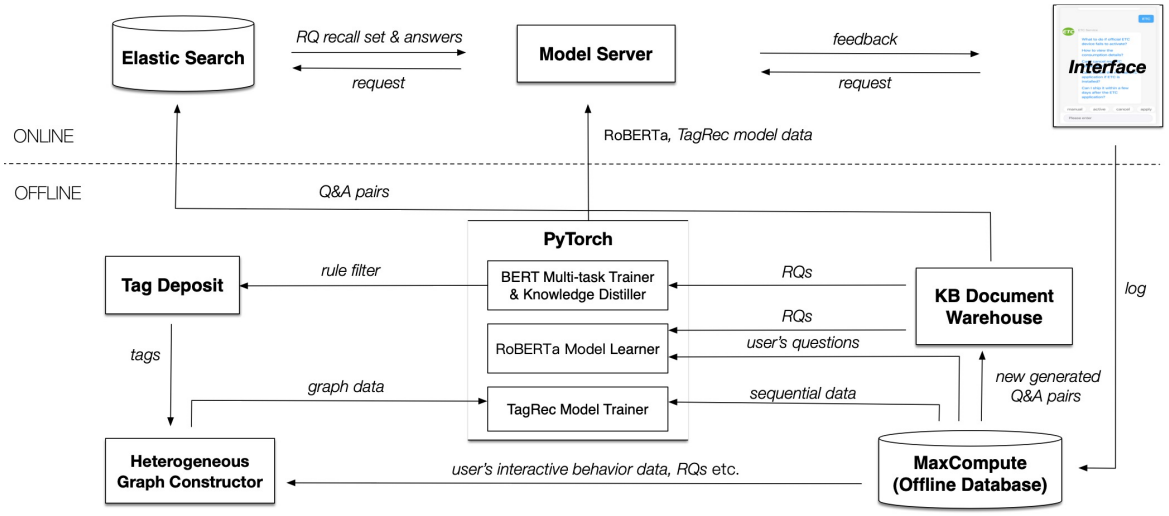


Fig. 4: The main components of our IntelliTag system.

to the model server. While the model server further sends a message to ElasticSearch to obtain a suitable RQs recall set (also including the corresponding answers). Based on the uploaded RoBERTa [54] model, the model server finds out the best matching RQ and passes on its answer to the interface, which is responsible for showing the final result. However, if a user clicks on a tag, TagRec task will be triggered. The model server first predicts a set of recommended tags, and combines historical clicked tags in the session as a query. The query is next sent to ElasticSearch, which will feedback an RQs recall set. After that, the model server sends the reranked RQs list to the interface, showing the user's most possible predicted questions.

On the other hand, an online interactive data log is periodically downloaded to **MaxCompute**[10], which is an offline database developed by Alibaba Cloud. First, as illustrated in Section III-A, MaxCompute obtains new RQs for a while, and employs machine reading understanding models to find out suitable answers, where the newly generated Q&A pairs will be temporarily stored in MaxCompute. Second, as described in Section III-B, **KB document warehouse** reads such generated pairs from MaxCompute, and sends all the RQs to BERT-based multi-task trainer. After the training process is finished, we start the inference process with knowledge distillation. Next, the output tags should again be filtered by our carefully designed rules, where **tag deposit** stores the mined tags. In addition, KB document warehouse ought to update the latest Q&A pairs to online ElasticSearch.

There exist two more important training models, *i.e.,* RoBERTa and TagRec model. RoBERTa model learner[11] can access the user's proposed questions from MaxCompute, and read RQs from KB document warehouse to train the model. At the same time, as mentioned in Section IV-A, **heterogeneous graph constructor** obtains tags from tag deposit and user's interactive behavior data, RQs and etc. from MaxCompute. Finally, TagRec model trainer is able to learn the model with structural and sequential data.

B. Deployment Details

As shown in Fig. 4, we have three models implemented by PyTorch in the offline section, where BERT-based multi-task model adopts offline training and inference. While the others will update the models to the online model server as soon as they finish training. In this way, the model server can support question answering, tag recommendation and question prediction with the latest model.

Especially for RoBERTa and TagRec models, we adopt the "T+1" mode to offline train, which indicates we train models every day in terms of the accumulated data from the last day. In some scenarios (*e.g.,* E-commence recommendation), a user's behavior might change quickly a few seconds ago, which has a significant impact on sequence recommendations, but our service is much more stable, so we use T+1 instead of online learning.

In practice, we first conduct model inference to generate tag node embeddings in the offline system shown in Section IV-B, after the training is finished. And then the embeddings are directly uploaded to online model servers, instead of online real-time GNN layers for inference. This strategy can greatly relieve the pressure of online service without loss of accuracy, since no real-time graph construction is required in the case. On the other hand, the parameters of sequence-based layers, shown in Section IV-C, are uploaded to online model servers for real-time sequence predictions. To solve the cold start issue, we recommend the most frequently clicked tags when a user opens the interface. But if a user proposes a question without any click behaviors, we will directly find out the best

[10]https://www.alibabacloud.com/product/maxcompute

[11]We do not describe the details in this paper, since RoBERTa is a common model in intelligent Q&A dialogue.

TABLE II: Statistics of our dataset.

Tag Mining	Tag Recommendation		
Sentence Label	Data Type	Relation	Session Information
training data: 49,093 test data: 5,167 KB doc: ∼2 million	T: 38,344 Q: 656,720 E: 446	asc: 194,116 clk: 25,390 cst: 137,784 crl: 656,720	sessions: 98,875 tag clicks: 286,802 average clicks: 2.9

TABLE III: Performance comparison on tag mining task (r: rule; d: knowledge distillation).

Training Mode	Precision	Recall	F1 Score	Inference Time
ST model	76.46%	79.07%	77.74%	-
MT model	79.10%	81.82%	80.44%	570 mins
MT model + r	81.59%	80.16%	80.87%	574 mins
MT model + d + r	80.77%	79.25%	80.00%	40 mins

matching RQ and recommend tags according to *asc* (*T-Q*) relation.

VI. EXPERIMENTS

To evaluate the effectiveness and efficiency of our IntelliTag system, we conduct a series of experiments on tag mining and recommendation tasks.

A. Experimental Setup

In this section, we introduce the details of our experimental settings.

1) Dataset Description: We collect the data from our intelligent cloud customer service system, where Table II illustrates the statistics information about our experimental dataset. For the tag mining task, we manually annotate approximately 54,260 RQ sentences with tag segmentation and word weighting information (a word weight is labeled as 1 if believed to be a part of a tag), where 49,093 and 5,167 sentences are respectively applied for training and test. Until now, our KB document already has more than 2 million Q&A pairs and the amount still keeps growing. That is why we need to use knowledge distillation to condensate model parameters for performing daily inference.

For offline evaluation of TagRec task, we randomly select 98,875 sessions consisting of 286,802 tag clicks. The average click number in sessions is 2.9, which indicates user's questions are addressed within 3 clicks on average. In order to reduce model pressure and filter noisy data, we only keep 30 days of active RQs retrieved by online user's questions from 2020/02/13 to 2020/03/13. Finally, 38,344 tags are mined from 656,720 RQs where there are 1,014,010 relations with 4 types. The whole data is split into training, validation and test dataset at the proportion of 80%, 10% and 10%, respectively.

2) Evaluation Metrics: Since our goal is to excavate as many and accurate tags as possible, we deliberately choose F1 score as our evaluation metric for tag mining task. Following [73], we use MRR (Mean Reciprocal Rank), NDCG@K (Normalized Discounted Cumulative Gain at rank K) and HR@K (Hit Ratio at rank K) to offline evaluate the model effectiveness of TagRec task. In our experiments, we set $K \in \{1, 5, 10\}$ for HR and NDCG. Following the common strategy [46], we randomly choose 49 tags from the same tenant as negative samples, and rank the list consisting of the above 50 tags. For all these metrics, the higher the value, the better the performance.

3) Model Settings: For the multi-task learning model, we use a 12-layer Transformer with 768 hidden units for BERT. We set the maximum sentence length as 512. The same weight is leveraged to add tag segmentation loss and word weighting loss. To distill knowledge, we use a 2-layer Transformer to construct a student BERT, and such a compact student BERT will be adopted for inference purposes. While for TagRec task, we generate 100-dimensional vectors as tag features, by learning semantic information from a text perspective. The number of multi-head is set as 4 for neighbor attention, metapath attention and contextual attention. The size of the input layer of the sequential model is 100, and we just use a 2-layer Transformer for the sake of model efficiency.

4) Baselines: For the task of tag mining, we simply show the improvement of our multi-task model versus single-task separated learning fashion, as well as knowledge distillation. We next compare our proposed model with recently state-of-art methods on TagRec task, where the same parameter setting is configured with our model for a fair comparison. To demonstrate the effectiveness of our end-to-end mode, we also prepare a variant of our IntelliTag (*i.e.,* IntelliTag$_{st}$). The baselines are given below:

- **GRU4Rec** [39]: It applies GRU to model sequence for recommendation with ranking-based loss;
- **SR-GNN** [85]: It is a graph-based recommendation model, which constructs a homogeneous graph in order of user's clicks and use GNN to learn item representations;
- **metapath2vec** [15]: It is a state-of-the-art unsupervised node representation learning method, fully exploiting metapath information of heterogeneous graph;
- **BERT4Rec** [73]: It employs BERT to model sequence, which reports the best results on many public sequential recommendation dataset;
- **IntelliTag**$_{st}$: It is a static variant that trains graph-based layers and Transformer layers separately.

All the methods run on a single NVIDIA Tesla P100 GPU with a batch size of 128, and we train each model using Adam [44] with learning rate of 0.001, weight decay of 0.01, and linear decay of the learning rate. Following [73], we set mask proportion as 0.2 for Bert4Rec and our models.

B. The Benefits to Our Multi-task Learning Model on Tag Mining Task

As illustrated in Table III, compared with the single-task learning model (*i.e.,* ST model), our designed multi-task learning model (*i.e.,* MT model) performs much better both on precision and recall metrics. We believe this improvement is conducive to the text semantic relevance of two tasks. After we add rules (*i.e.,* MT model + r), precision value obviously increases and recall has a certain decrease, since we further

TABLE IV: Offline evaluation results on TagRec task.

Model	MRR	NDCG			HR	
		1	5	10	5	10
GRU4Rec	0.204	0.087	0.192	0.235	0.292	0.427
SR-GNN	0.246	0.111	0.228	0.299	0.348	0.570
metapath2vec	0.299	0.177	0.300	0.340	0.415	0.539
BERT4Rec	0.349	0.201	0.364	0.412	0.517	0.663
IntelliTag$_{st}$	0.395	0.239	0.413	0.464	0.574	0.730
IntelliTag	**0.404**	**0.247**	**0.424**	**0.475**	**0.589**	**0.745**

filter the result that model yields. Nevertheless, rules still gain a performance improvement of 0.43% on final F1 score. However, the original MT model spends too much time for inference. We then leverage knowledge distillation (*i.e.,* MT model + d + r), which is 14 times faster than the original fashion. Although F1 score decreases 0.87%, it is still much better than ST model. By the way, all the reported results are based on the pretrained BERT-Base model[12] [76].

C. Offline Evaluation Results on TagRec Task

As shown in Table IV, for two sequential recommendation algorithms, BERT4Rec significantly outperforms GRU4Rec, consistent with the conclusion in [73]. SR-GNN, as a graph-based method, is able to learn structural information of the graph. But it can not handle multifarious node types and connected relations, that is why it is not as good as metapath2vec. As a comparison, metapath2vec achieves relatively promising performance, due to its certain capability to incorporate heterogeneous information. However, as an unsupervised learning method, it fails to impose label information and attention mechanism. BERT4Rec is the strongest competitor due to its effectively sequential modeling capacity.

Overall, our IntelliTag$_{st}$ improves BERT4Rec by 4.6% on MRR, probably because our graph-based layers can take full use of various types and relations information with neighbor attention and metapath attention. It is interesting that the larger K is, the much better IntelliTag$_{st}$ outperforms BERT4Rec both on NDCG and HR. We think it might because our model is able to aggregate information from different tenants, which helps to learn better representations of low-frequency tags. Finally, compared with our variant, IntelliTag further boosts the performance by approximate 1% on MRR and NDCG, and 1.5% on HR. It proves end-to-end training mode is better than step-by-step on our task, since the former can disseminate information across graph-based and sequence-based layers if sufficient data is provided.

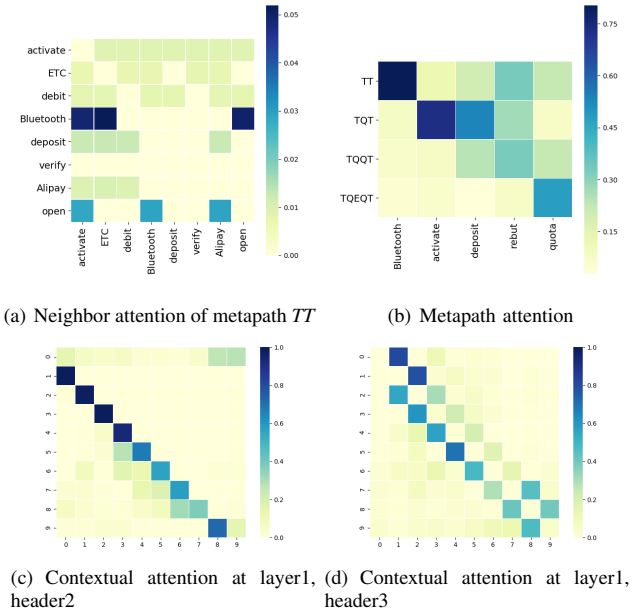
D. Case Study of Multiple Attention on TagRec Task

One of our contributions is to employ a multi-attention mechanism for learning useful information from graph structure and sequential model. As illustrated in Table V, we respectively remove neighbor attention, metapath attention and contextual attention to obtain three variants. Not surprisingly, the complete IntelliTag consistently achieves the best performance on all metrics. More importantly, the influence of contextual attention is more considerable than the others, since the performance of IntelliTag$_{w/o\ ca}$ deteriorates more rapidly.

[12]https://github.com/google-research/bert

TABLE V: The influence of each attention (na: neighbor attention; ma: metapath attention; ca: contextual attention).

Model	MRR	NDCG			HR	
		1	5	10	5	10
IntelliTag$_{w/o\ na}$	0.382	0.223	0.401	0.453	0.567	0.726
IntelliTag$_{w/o\ ma}$	0.379	0.220	0.398	0.451	0.564	0.728
IntelliTag$_{w/o\ ca}$	0.272	0.123	0.261	0.337	0.399	0.635
IntelliTag	**0.404**	**0.247**	**0.424**	**0.475**	**0.589**	**0.745**



(a) Neighbor attention of metapath *TT*

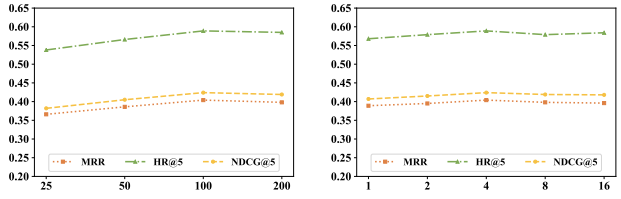
(b) Metapath attention

(c) Contextual attention at layer1, header2

(d) Contextual attention at layer1, header3

Fig. 5: Heat maps of attention weights by means of different types of attentions.

To visualize how multiple attention mechanism works, we present a series of case study experiments. Fig. 5 (a) shows mutual influence between two tags connected with the metapath TT. It is easy to find "open", "activate" and "ETC" attain maximum values, since they are often clicked followed by the tag "Bluetooth". Fig. 5 (b) shows the preference towards multiple metapaths for different tags. The influence of metapath*TT* is the strongest for "Bluetooth", because users are used to clicking other tags, such as "activate", "open" and etc. While "quota" is very related to specific scenarios, so it is easy to be connected with the other tags (*e.g.,* credit card or debit card) in the same tenant (*i.e.,* a bank) through metapath *TQEQT*. Fig. 5 (c)(d) present the attention values learned by contextual attention. We have two insights from the results: a user's click is most influenced by the last click; contextual attention is capable of capturing bidirectional information implied in a sequence.



(a) The effectiveness versus the dimension size of the learned tag node embeddings

(b) The effectiveness versus the numbers of attention heads

Fig. 6: The effectiveness versus the dimension size of the learned tag node embeddings and the numbers of attention heads.

E. Hyperparameter Sensitivity

We further perform a hyperparameter sensitivity analysis on our TagRec task. The dimension size of the learned embeddings is an important parameter, which influences the amount of heterogeneous information extraction from GNN layers. As illustrated in Fig. 6 (a), 100 is the best dimension over different evaluation metrics, since the information of the heterogeneous graph is not well exploited when the dimension is too small, while the results probably overfit when it is too large. In addition, we vary the numbers of our three attention heads[13], *i.e.,* neighbor attention, metapath attention and contextual attention, to examine its influence. As shown in Fig. 6 (b), compared to the parameter of dimension size, the number of attention heads is not sensitive under different evaluation metrics.

F. Online Performance

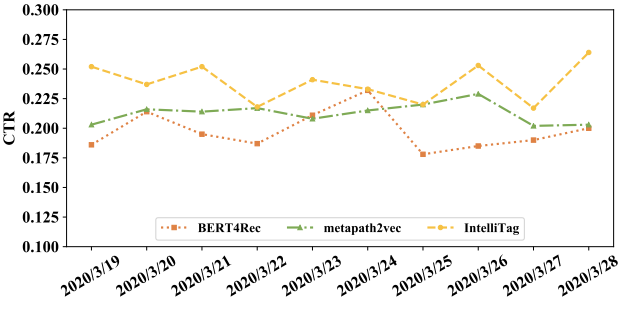


Fig. 7: Online CTR performance of our IntelliTag versus BERT4Rec and metapath2vec.

To further evaluate our IntelliTag, we conduct a series of experiments for online service. Based on the method of A/B testing [93], we divide two extra traffic buckets to test two competitive baseline algorithms (*i.e.,* metapath2vec and Bert4Rec). CTR (Click-Through Rate), as a key indicator of online performance, is under monitored from 2020/3/19 to 2020/3/28. As shown in Fig. 7, our IntelliTag keeps the best performance consistent with offline evaluation. It is worth noting that BERT4Rec online performance is worse than metapath2vec conversely. Based on our analysis, metapath2vec performs better when a tenant has few Q&A pairs, due to its superiority of aggregating information from other tenants. For online evaluation, we pay attention to SMEs with little operation capacity (with few Q&A pairs), so the macro average is applied to the CTR metric. That is, BERT4Rec's results variance of different tenants is very large, which leads to worse results.

[13]The values of head numbers for different attentions are set same.

TABLE VI: Online HIR and response latency comparison.

Evaluation Metric	metapath2vec	BERT4Rec	IntelliTag
HIR	0.218	0.214	0.212
Response latency	50.8 ms	106.2 ms	109.8 ms

The goal of intelligent customer service is to solve user's questions through artificial intelligence, at least as soon as possible to ease the pressure of manual customer service. Therefore, we also monitor the indicator HIR (Human Intervention Rate) in a period. As illustrated in Table VI, the HIR of IntelliTag is the smallest, which demonstrates our system can solve most of the user's questions. Since we have already supported millions of users, a tiny change of HIR is remarkable in our scenario, leading to a significant reduction of SMEs' labor costs. The other important indicator is response latency, which is related to user experience. In practice, 3.6 ms superiority of BERT4Rec compared with our IntelliTag is hard to be perceived by users. Obviously, metapath2vec is much faster for its online service, since it only depends on the last clicked tag. In fact, we directly upload the closest tags of each tag from the offline calculation in advance. So there is no need to online measure relatedness between two tags because metapath2vec does not originally support sequential modeling. Nevertheless, approximately 100 ms is totally acceptable for our IntelliTag from the user's perspective.

VII. Conclusion and Future Work

We have already introduced our IntelliTag system, including data construction, model design, system implementation and experiment evaluation, which is capable to widely inspire real-world customer service developers. Also, our proposed hierarchical end-to-end model combining with inner graph-based and outer sequence-based layers can be applied in more tasks. In the future, we will try more Q&A pairs collection ways for tag mining and cache high-frequency data to decrease system latency, aiming to provide better cloud service and faster response. Further, we will also introduce our proposed technical solutions into IVR (Interactive Voice Response) services to replace manually preset voice prompt phrase.

Acknowledgment

The authors thank the anonymous reviewers for their constructive and valuable advice, Zark team for their suggestions on online deployment, as well as Shuang Peng, Xiaojun Chen, Weifeng Lou and Ze Zhan's help on data preparation and system implementation.

REFERENCES

[1] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*, 2015.

[2] C. Buragohain, K. M. Risvik, P. Brett, M. Castro, W. Cho, J. Cowhig, N. Gloy, K. Kalyanaraman, R. Khanna, J. Pao, et al. A1: A distributed in-memory graph database. In *Proceedings of SIGMOD*, pages 329–344, 2020.

[3] S. Cao, W. Lu, and Q. Xu. Grarep: Learning graph representations with global structural information. In *Proceedings of CIKM*, pages 891–900, 2015.

[4] S. Cao, X. Yang, C. Chen, J. Zhou, X. Li, and Y. Qi. Titant: online real-time transaction fraud detection in ant financial. *Proceedings of VLDB*, 2019.

[5] C. Chen, X. Zhang, S. Ju, C. Fu, C. Tang, J. Zhou, and X. Li. Antprophet: an intention mining system behind alipay's intelligent customer service bot. *Proceedings of IJCAI*, 8:6497–6499, 2019.

[6] D. Chen, A. Fisch, J. Weston, and A. Bordes. Reading wikipedia to answer open-domain questions. In *Proceedings of ACL*, volume 1, pages 1870–1879, 2017.

[7] H. Chen, H. Yin, T. Chen, Q. V. H. Nguyen, W.-C. Peng, and X. Li. Exploiting centrality information with graph convolutions for network representation learning. In *Proceedings of ICDE*, pages 590–601. IEEE, 2019.

[8] T. Chen and R. C.-W. Wong. Handling information loss of graph neural networks for session-based recommendation. In *Proceedings of SIGKDD*, pages 1172–1180, 2020.

[9] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, et al. Wide & deep learning for recommender systems. In *Proceedings of RecSys*, pages 7–10, 2016.

[10] K. W. Church and P. Hanks. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29, 1990.

[11] L. Cui, S. Huang, F. Wei, C. Tan, C. Duan, and M. Zhou. Superagent: A customer service chatbot for e-commerce websites. In *Proceedings of ACL*, pages 97–102, 2017.

[12] P. Deepak, A. Dey, and D. Majumdar. Fast mining of interesting phrases from subsets of text corpora. In *EDBT*, pages 193–204, 2014.

[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[14] M. Ding and S. Chen. Efficient partitioning and query processing of spatio-temporal graphs with trillion edges. In *Proceedings of ICDE*, pages 1714–1717. IEEE, 2019.

[15] Y. Dong, N. V. Chawla, and A. Swami. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of SIGKDD*, pages 135–144, 2017.

[16] T. Donkers, B. Loepp, and J. Ziegler. Sequential user-based recurrent neural network recommendations. In *Proceedings of RecSys*, pages 152–160, 2017.

[17] A. El-Kishky, Y. Song, C. Wang, C. Voss, and J. Han. Scalable topical phrase mining from text corpora. *VLDB*, 2014.

[18] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of KDD*, pages 226–231, 1996.

[19] S. Fan, J. Zhu, X. Han, C. Shi, L. Hu, B. Ma, and Y. Li. Metapath-guided heterogeneous graph neural network for intent recommendation. In *Proceedings of SIGKDD*, pages 2478–2486, 2019.

[20] W. Fan, W. Yu, J. Xu, J. Zhou, X. Luo, Q. Yin, P. Lu, Y. Cao, and R. Xu. Parallelizing sequential graph computations. *TODS*, 43(4):1–39, 2018.

[21] Y. Feng, B. Hu, F. Lv, Q. Liu, Z. Zhang, and W. Ou. Atbrg: Adaptive target-behavior relational graph network for effective recommendation. In *Proceedings of SIGIR*, pages 2231–2240, 2020.

[22] J. Fu, H. Zheng, and T. Mei. Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In *Proceedings of CVPR*, pages 4476–4484, 2017.

[23] X. Fu, J. Zhang, Z. Meng, and I. King. Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding. In *Proceedings of WWW*, pages 2331–2341, 2020.

[24] F. A. Gers, J. Schmidhuber, and F. Cummins. Learning to forget: Continual prediction with lstm. 1999.

[25] S. A. Golder and B. A. Huberman. Usage patterns of collaborative tagging systems. *Journal of information science*, 32(2):198–208, 2006.

[26] Y. Gong and Q. Zhang. Hashtag recommendation using attention-based convolutional neural network. In *Proceedings of IJCAI*, pages 2782–2788, 2016.

[27] A. Grover and J. Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of SIGKDD*, pages 855–864, 2016.

[28] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He. Deepfm: a factorization-machine based neural network for ctr prediction. *arXiv preprint arXiv:1703.04247*, 2017.

[29] A. Gustavo, C. Binnig, I. Pandis, K. Salem, J. Skrzypczak, R. Stutsman, L. Thostrup, T. Wang, Z. Wang, and T. Ziegler. Dpi: the data processing interface for modern networks. *Proceedings of CIDR*, 2019.

[30] M. Henaff, J. Bruna, and Y. LeCun. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*, 2015.

[31] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom. Teaching machines to read and comprehend. In *Proceedings of NeurIPS*, pages 1693–1701, 2015.

[32] G. E. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[33] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[34] B. Hu, Z. Zhang, C. Shi, J. Zhou, X. Li, and Y. Qi. Cash-out user detection based on attributed heterogeneous information network with a hierarchical attention mechanism. *Proceedings of AAAI*, 33(1):946–953, 2019.

[35] Z. Hu, Y. Dong, K. Wang, and Y. Sun. Heterogeneous graph transformer. In *Proceedings of WWW*, pages 2704–2710, 2020.

[36] J. Huang, Z. Ren, W. X. Zhao, G. He, J.-R. Wen, and D. Dong. Taxonomy-aware multi-hop reasoning networks for sequential recommendation. In *Proceedings of WSDM*, pages 573–581, 2019.

[37] J. Huang, W. X. Zhao, H. Dou, J.-R. Wen, and E. Y. Chang. Improving sequential recommendation with knowledge-enhanced memory networks. In *Proceedings of SIGIR*, pages 505–514, 2018.

[38] C. L. Isbell, M. Kearns, D. Kormann, S. Singh, and P. Stone. Cobot in lambdamoo: A social statistics agent. In *Proceedings of AAAI*, pages 36–41, 2000.

[39] D. Jannach and M. Ludewig. When recurrent neural networks meet the neighborhood for session-based recommendation. In *Proceedings of RecSys*, pages 306–310, 2017.

[40] Z. Ji, Z. Lu, and H. Li. An information retrieval approach to short text conversation. *arXiv preprint arXiv:1408.6988*, 2014.

[41] J. Jiang, P. Xiao, L. Yu, X. Li, J. Cheng, X. Miao, Z. Zhang, and B. Cui. Psgraph: How tencent trains extremely large-scale graphs with spark? In *Proceedings of ICDE*, pages 1549–1557. IEEE, 2020.

[42] J. Jin, J. Qin, Y. Fang, K. Du, W. Zhang, Y. Yu, Z. Zhang, and A. J. Smola. An efficient neighborhood-based interaction model for recommendation on heterogeneous graph. In *Proceedings of SIGKDD*, pages 75–84, 2020.

[43] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

[44] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[45] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[46] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of SIGKDD*, pages 426–434, 2008.

[47] F.-L. Li, M. Qiu, H. Chen, X. Wang, X. Gao, J. Huang, J. Ren, Z. Zhao, W. Zhao, L. Wang, et al. Alime assist: An intelligent assistant for creating an innovative e-commerce experience. In *Proceedings of CIKM*, pages 2495–2498, 2017.

[48] J. Li, M. Galley, C. Brockett, J. Gao, and B. Dolan. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*, 2015.

[49] J. Li, W. Monroe, A. Ritter, M. Galley, J. Gao, and D. Jurafsky. Deep reinforcement learning for dialogue generation. *arXiv preprint arXiv:1606.01541*, 2016.

[50] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma. Neural attentive session-based recommendation. In *Proceedings of CIKM*, pages 1419–1428, 2017.

[51] J. Li, H. Xu, X. He, J. Deng, and X. Sun. Tweet modeling with lstm recurrent neural networks for hashtag recommendation. In *Proceedings of IJCNN*, pages 1570–1577. IEEE, 2016.

[52] Y. Li, T. Liu, J. Jiang, and L. Zhang. Hashtag recommendation with topical attention-based lstm. In *Proceedings of COLING*, 2016.

[53] Z. Li, X. Shen, Y. Jiao, X. Pan, P. Zou, X. Meng, C. Yao, and J. Bu. Hierarchical bipartite graph neural networks: Towards large-scale e-commerce applications. In *Proceedings of ICDE*, pages 1677–1688. IEEE, 2020.
[54] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.
[55] Z. Liu, C. Chen, L. Li, J. Zhou, X. Li, L. Song, and Y. Qi. Geniepath: Graph neural networks with adaptive receptive paths. In *Proceedings of AAAI*, pages 4424–4431, 2019.
[56] M.-T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of EMNLP*, pages 1412–1421, 2015.
[57] C. Ma, L. Ma, Y. Zhang, J. Sun, X. Liu, and M. Coates. Memory augmented graph neural networks for sequential recommendation. *arXiv preprint arXiv:1912.11730*, 2019.
[58] R. Ma, X. Qiu, Q. Zhang, X. Hu, Y.-G. Jiang, and X. Huang. Co-attention memory network for multimodal microblog's hashtag recommendation. *IEEE TKDE*, 2019.
[59] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of ICML*, volume 30, page 3, 2013.
[60] V. Mnih, N. Heess, A. Graves, and koray kavukcuoglu. Recurrent models of visual attention. In *Proceedings of NeurIPS*, pages 2204–2212, 2014.
[61] M. Niepert, M. Ahmed, and K. Kutzkov. Learning convolutional neural networks for graphs. In *Proceedings of ICML*, pages 2014–2023, 2016.
[62] M. G. Noll and C. Meinel. Web search personalization via social bookmarking and tagging. In *The semantic web*, pages 367–380. Springer, 2007.
[63] R. Otunba, R. A. Rufai, and J. Lin. Deep stacked ensemble recommender. In *Proceedings of SSDBM*, pages 197–201, 2019.
[64] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: Online learning of social representations. In *Proceedings of SIGKDD*, pages 701–710, 2014.
[65] M. L. Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.
[66] M. Qiu, F.-L. Li, S. Wang, X. Gao, Y. Chen, W. Zhao, H. Chen, J. Huang, and W. Chu. Alime chat: A sequence to sequence and rerank based chatbot engine. In *Proceedings of ACL*, pages 498–503, 2017.
[67] R. Qiu, H. Yin, Z. Huang, and T. Chen. Gag: Global attributed graph neural network for streaming session-based recommendation. In *Proceedings of SIGIR*, pages 669–678, 2020.
[68] P. Rajpurkar, R. Jia, and P. Liang. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*, 2018.
[69] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of WWW*, pages 811–820, 2010.
[70] S. Rendle and L. Schmidt-Thieme. Pairwise interaction tensor factorization for personalized tag recommendation. In *Proceedings of WSDM*, pages 81–90, 2010.
[71] I. V. Serban, A. Sordoni, Y. Bengio, A. Courville, and J. Pineau. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of AAAI*, 2016.
[72] J. Shang, J. Liu, M. Jiang, X. Ren, C. R. Voss, and J. Han. Automated phrase mining from massive text corpora. *TKDE*, 30(10):1825–1837, 2018.
[73] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of CIKM*, pages 1441–1450, 2019.
[74] Y. Sun and J. Han. Mining heterogeneous information networks: principles and methodologies. *Synthesis Lectures on Data Mining and Knowledge Discovery*, 3(2):1–159, 2012.
[75] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *Proceedings of VLDB*, 4(11):992–1003, 2011.
[76] I. Turc, M.-W. Chang, K. Lee, and K. Toutanova. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962*, 2019.
[77] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Proceedings of NeurIPS*, pages 5998–6008, 2017.
[78] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks. In *Proceedings of ICLR*, 2018.
[79] H. Wang, X. Shi, and D.-Y. Yeung. Relational stacked denoising autoencoder for tag recommendation. In *Proceedings of AAAI*, 2015.
[80] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu. Heterogeneous graph attention network. In *Proceedings of WWW*, pages 2022–2032, 2019.
[81] Z. Wang, W. Wei, G. Cong, X.-L. Li, X.-L. Mao, and M. Qiu. Global context enhanced graph neural networks for session-based recommendation. In *Proceedings of SIGIR*, pages 169–178, 2020.
[82] T.-H. Wen, D. Vandyke, N. Mrksic, M. Gasic, L. M. Rojas-Barahona, P.-H. Su, S. Ultes, and S. Young. A network-based end-to-end trainable task-oriented dialogue system. *arXiv preprint arXiv:1604.04562*, 2016.
[83] J. Weston, S. Chopra, and K. Adams. # tagspace: Semantic embeddings from hashtags. In *Proceedings of EMNLP*, pages 1822–1827, 2014.
[84] J. D. Williams and G. Zweig. End-to-end lstm-based dialog control optimized with supervised and reinforcement learning. *arXiv preprint arXiv:1606.01269*, 2016.
[85] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan. Session-based recommendation with graph neural networks. In *Proceedings of AAAI*, volume 33, pages 346–353, 2019.
[86] X. Xin, A. Karatzoglou, I. Arapakis, and J. M. Jose. Self-supervised reinforcement learning for recommender systems. In *Proceedings of SIGIR*, pages 931–940, 2020.
[87] C. Xu, P. Zhao, Y. Liu, V. S. Sheng, J. Xu, F. Zhuang, J. Fang, and X. Zhou. Graph contextualized self-attention network for session-based recommendation. In *Proceedings of IJCAI*, pages 3940–3946, 2019.
[88] Z. Xu, C. Chen, T. Lukasiewicz, Y. Miao, and X. Meng. Tag-aware personalized recommendation using a deep-semantic similarity model with negative sampling. In *Proceedings of CIKM*, pages 1921–1924, 2016.
[89] Z. Xu, T. Lukasiewicz, C. Chen, Y. Miao, and X. Meng. Tag-aware personalized recommendation using a hybrid deep model. In *Proceedings of IJCAI*, 2017.
[90] R. Yan, Y. Song, and H. Wu. Learning to respond with deep neural networks for retrieval-based human-computer conversation system. In *Proceedings of SIGIR*, pages 55–64, 2016.
[91] Z. Yan, N. Duan, J. Bao, P. Chen, M. Zhou, Z. Li, and J. Zhou. Docchat: An information retrieval approach for chatbot engines using unstructured documents. In *Proceedings of ACL*, pages 516–525, 2016.
[92] H. Yang. Aligraph: A comprehensive graph neural network platform. In *Proceedings of SIGKDD*, pages 3165–3166, 2019.
[93] S. W. Young. Improving library user experience with a/b testing: Principles and process. *Weave: Journal of Library User Experience*, 1(1), 2014.
[94] F. Yu, Q. Liu, S. Wu, L. Wang, and T. Tan. A dynamic recurrent model for next basket recommendation. In *Proceedings of SIGIR*, pages 729–732, 2016.
[95] S. Yun, M. Jeong, R. Kim, J. Kang, and H. J. Kim. Graph transformer networks. In *Proceedings of NeurIPS*, pages 11983–11993, 2019.
[96] J. Zhang, X. Shi, J. Xie, H. Ma, I. King, and D.-Y. Yeung. Gaan: Gated attention networks for learning on large and spatiotemporal graphs. *arXiv preprint arXiv:1803.07294*, 2018.
[97] K. Zhou, H. Wang, W. X. Zhao, Y. Zhu, S. Wang, F. Zhang, Z. Wang, and J.-R. Wen. Sˆ 3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of CIKM*, 2020.
[98] R. Zhu, D. Yang, and Y. Li. Learning improved semantic representations with tree-structured lstm for hashtag recommendation: An experimental study. *Information*, 10(4):127, 2019.
[99] X. Zhu. Case ii (part a): Jimi's growth path: Artificial intelligence has redefined the customer service of jd. com. In *Emerging Champions in the Digital Economy*, pages 91–103. Springer, 2019.

APPENDIX
THE TECHNICAL DETAILS OF THE MODEL

In this section, we reveal the formal derivation of our TagRec model.

A. Structural Information Excavation from TagRec Heterogeneous Graph

1) Aggregating Node Information with Neighbor Attention: Given a metapath ρ, we collect neighbors for target tag t,

which can be denoted as $\mathcal{N}_t^\rho$. Formally, for each neighbor $t' \in \mathcal{N}_t^\rho$, we calculate its importance to tag t as follows:

$$\alpha_{tt'}^\rho = \text{LeakyReLU}(\mathbf{W}_n^T[\mathbf{x}_t||\mathbf{x}_{t'}]) \tag{4}$$

where $\mathbf{W}_n \in \mathbb{R}^{2d\times 1}$ is the weight matrix for neighbor attention, and $[\mathbf{x}_t||\mathbf{x}_{t'}]$ indicates vector concatenation of $\mathbf{x}_t$ and $\mathbf{x}_{t'}$, which are respectively feature vectors of node t and node t'. $\alpha_{tt'}^\rho$ indicates attention value of target node t with neighbor node t' transmitted by metapath ρ, where LeakyReLU activation function [59] is applied. Motived by [77], [80], we aggregate neighbors with respect to metapath ρ by weighted sum, and adopt a multi-head attention to reduce the high variance:

$$\mathbf{h}_t^\rho = \|_{m=1}^M \sigma\Big(\sum_{t'\in\mathcal{N}_t^\rho} \text{softmax}(\alpha_{tt'}^\rho)\mathbf{x}_{t'}\Big) \tag{5}$$

where the term $\sum_{t'\in\mathcal{N}_t^\rho} \text{softmax}(\alpha_{tt'}^\rho)\mathbf{x}_{t'}$ is the node feature average weighted by neighbor attention, where softmax layer is used to normalize $\alpha_{tt'}^\rho$. M denotes the number of attention heads, and $\|_{m=1}^M(\cdot)$ indicates the vector concatenation of each multi-head. $\sigma(\cdot)$ is sigmoid activation function, and $\mathbf{h}_t^\rho \in \mathbb{R}^{Md\times 1}$ represents target embedding of node t aggregated with neighbor attention via metapath ρ.

2) Learning Tag Embedding with Metapath Attention: Given a target tag t, we can obtain a set of representations $\{\mathbf{h}_t^\rho\}_{\rho\in\mathcal{P}}$. Concretely, we calculate the attention weight β_t^ρ of metapath ρ for tag t as follows:

$$\beta_t^\rho = \mathbf{v}_p^T \tanh(\mathbf{W}_p\mathbf{h}_t^\rho + \mathbf{b}_p) \tag{6}$$

where $\{\mathbf{v}_p \in \mathbb{R}^{Md\times 1}, \mathbf{W}_p \in \mathbb{R}^{Md\times Md}, \mathbf{b}_p \in \mathbb{R}^{Md\times 1}\}$ is the parameter set, and we adopt tanh as activation function. Similarly, a softmax is applied for normalization and we employ weighted sum for multiple metapath fusion with dimensionality reduction, as follows:

$$\mathbf{z}_t = \mathbf{W}_l\Big(\sum_{\rho\in\mathcal{P}} \text{softmax}(\beta_t^\rho)\mathbf{h}_t^\rho\Big) + \mathbf{b}_l \tag{7}$$

where $\mathbf{W}_l \in \mathbb{R}^{d\times Md}$ and $\mathbf{b}_l \in \mathbb{R}^{d\times 1}$ represents the weight matrix and the bias term for linear transformation, respectively. And $\mathbf{z}_t$ denotes tag embedding of target node t aggregated by multiple metapath attention.

B. Sequential Modeling with Transformer Layers

As illustrated in Fig. 8, our sequential model consists of three main layers: an input layer, a Transformers layer (including attention and feed-forward part) and an output layer. In particular, multiple Transformer layers are stacked for comprehensively capturing sequential information.

For the input layer, we add each tag embedding with corresponding position embedding and concatenate them together [77]. The input consists of N clicked tag embeddings (*i.e.,* $\mathbf{z}_1 \sim \mathbf{z}_N$) and our prediction goal (*i.e.,* $\mathbf{z}_{mask}$), which is marked as "mask" in the input layer.

$$\mathbf{X}^{(1)} = [\mathbf{z}_1+\mathbf{p}_1; \mathbf{z}_2+\mathbf{p}_2; \ldots; \mathbf{z}_N+\mathbf{p}_N; \mathbf{z}_{mask}+\mathbf{p}_{N+1}] \tag{8}$$

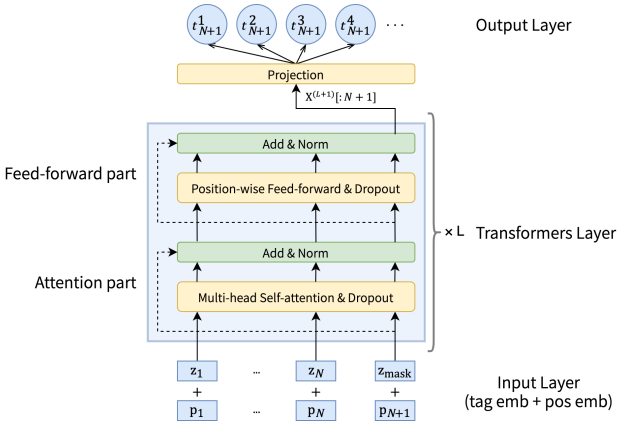


Fig. 8: The illustration of sequential model.

where we use $\mathbf{X}^{(1)}$ to denote the output of this layer, which will be fed into the first Transformer layer. Specifically, each Transformer layer is comprised of an attention part and a feed-forward part. Formally, given the input of l^{th} Transformer layer, the attention part can be described as:

$$\mathbf{A}^{(l)} = \text{Norm}(\mathbf{X}^{(l)} + \text{Dropout}(\text{MultiHead}(\mathbf{X}^{(l)}))) \tag{9}$$

We purposely omit the detail of MultiHead self-attention, Dropout and Norm operators as we continue to use the same calculation in [77]. Next, we calculate feed-forward part in terms of the output of attention part as follows:

$$\mathbf{X}^{(l+1)} = \text{Norm}(\mathbf{A}^{(l)} + \text{Dropout}(\text{FNN}(\mathbf{A}^{(l)}))) \tag{10}$$

where $\mathbf{X}^{(l+1)}$ is the output of the l^{th} Transformer layer and also the input of $l+1^{th}$ layer, since multiple Transformers are stacked to take full advantage of sequential information. Similarly, the details of FNN operator is also omitted as for same implementation in [77]. We stack L Transformers and finally use a projection layer as the output layer to yield predicted tags probabilities:

$$\begin{aligned}\hat{\mathbf{y}} &= \text{Projection}(\mathbf{X}^{(L+1)}[:N+1]) \\ &= \text{softmax}(\mathbf{W}_t\mathbf{X}^{(L+1)}[:N+1] + \mathbf{b}_t)\end{aligned} \tag{11}$$

where $\hat{\mathbf{y}}$ is the predicted tags probability distribution at $N+1$ time. $\mathbf{X}^{(L+1)}[:N+1]$ is the last row of $\mathbf{X}^{(L+1)}$, which represents the output of the mask embedding in the $L+1^{th}$ layer. And $\mathbf{W}_t \in \mathbb{R}^{|\mathcal{T}|\times d}$ and $\mathbf{b}_t \in \mathbb{R}^{|\mathcal{T}|\times 1}$ are respectively projection weight matrix and bias term, where $|\mathcal{T}|$ is the number of tags. Finally, we impose cross entropy predicted probabilities and ground truth to calculate the loss:

$$\mathcal{L}(\hat{\mathbf{y}}) = -\sum_{i=1}^{|\mathcal{T}|} \mathbf{y}_i \log(\hat{\mathbf{y}}_i) + (1-\mathbf{y}_i)\log(1-\hat{\mathbf{y}}_i) \tag{12}$$

where $\mathcal{L}(\hat{\mathbf{y}})$ is the loss function, $\mathbf{y}$ denotes the one-hot encoding vector of the ground truth tags, and $\hat{\mathbf{y}}_i$ indicates the predicted probability of i-th tag.