Hubble: An Industrial System for Audience Expansion in Mobile Marketing

Chenyi Zhuang*, Ziqi Liu*, Zhiqiang Zhang, Yize Tan, Zhengwei Wu, Zhining Liu, Jianping Wei, Jinjie Gu, Guannan Zhang, Jun Zhou★, Yuan Qi
Ant Financial Services Group, Hangzhou, China
{chenyi.zcy,ziqiliu,lingyao.zzq,yize.tyz,zejun.wzw,eason.lzn,pinger.wjp,jinjie.gujj,zgn138592,jun.zhoujun,yuan.qi}@antfin.com

ABSTRACT

Recently, in order to take a preemptive opportunity in the mobile economy, the Internet companies conduct thousands of marketing campaigns every day, to promote their mobile products and services. In the *mobile marketing* scenario, one of the fundamental issues is the *audience expansion* task for marketing campaigns. Given a set of seed users, audience expansion aims to seek more users (audiences), who are similar to the seeds and will finish the business goal of the targeted campaign (*i.e.,* convert). However, the problem is challenging in three aspects. First, a company will run hundreds of campaigns to serve massive users every day. The requirements of *scalability and timeliness* make training model for each campaign extremely resource-consuming thus impractical. Therefore, we proposed to solve the problem in a two-stage manner, in which the offline stage employs heavyweight user representation learning and the online stage performs embedding-based lightweight audience expansion. Second, conventional two-stage audience expansion systems neglect the high-order user-campaign interactions and usually generate entangled user embeddings, thus fail to achieve *high-quality user representation*. Third, the seeds, which are usually provided by experts or collected from users' feedbacks, could be noisy and cannot cover the entire actual audiences, thus introduce *coverage bias*. Unfortunately, to our best knowledge, none of the related literatures tackle this crucial issue of audience expansion.

Addressing the above challenges, in this paper, we present the *Hubble System*, an industrial solution for audience expansion in mobile marketing scenario. Hubble System follows the hybrid online-offline architecture to satisfy the requirements of scalability and timeliness. Specifically, in the offline stage, we propose a novel *adaptive and disentangled graph neural network* (called AD-GNN), to adaptively explore the user-campaign graph and generate comprehensive user embedding in a disentangled manner. In the online stage, tackling the coverage bias issue, we develop a novel audience expansion model with *knowledge distillation mechanism* (called KD-AE), to absorb knowledge from the offline AD-GNN and alleviate the coverage bias. Finally, extensive offline experiments and online A/B testing demonstrate the superior performance of the proposed Hubble System, compared with other state-of-the-art methods.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; *Learning latent representations*; • **Applied computing** → **Marketing**; • **Information systems** → *Collaborative filtering*.

KEYWORDS

Mobile marketing; audience expansion; user representation learning; graph neural network; knowledge distillation

ACM Reference Format:
Chenyi Zhuang*, Ziqi Liu*, Zhiqiang Zhang, Yize Tan, Zhengwei Wu, Zhining Liu, Jianping Wei, Jinjie Gu, Guannan Zhang, Jun Zhou★, Yuan Qi. 2020. Hubble: An Industrial System for Audience Expansion in Mobile Marketing. In *26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '20), August 23–27, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3394486.3403295

1 INTRODUCTION

In recent years, with the rapid growth of mobile services and their users, the innovative mobile economy has formed a competitive market. In order to promote their mobile products and services, many Internet companies (e.g., Google, Facebook, Alibaba, Ant Financial, *etc.*) conduct thousands of marketing campaigns every day. One of the fundamental issues of running marketing campaigns is the *audience expansion* problem. Given a *campaign* for a certain promotion activity, as well as a set of users as *seeds*, the task of audience expansion is to target more users (called *audiences*) who are similar to the seeds, and are interested in the products or services promoted by the given campaign. The seeds may be generated in different ways (*e.g.,* selected by experts, retrieved according to the past relevant campaigns, *etc.*) and are aimed to guide the expansion. High-quality expanded audiences will benefit the companies by increasing the conversion population while reducing the operating costs for their marketing campaigns.

Most literatures about audience expansion focus on online advertising scenarios. Traditional *similarity-based* approaches [10, 14, 15, 17] suggest to design subtle pairwise user similarity measure for matching the potential audiences and the seeds. Recent *embedding-based* approaches [1, 11] propose to expand audiences based on user embeddings generated by a machine learning model. With the help of user representation learned from massive historical data, the embedding-based approaches achieve better performance. However, in the industrial mobile marketing scenarios, applying the above audience expansion techniques still suffers from serval challenges, which are summarized as the following three aspects.

* Equal Contributions.
★ Corresponding author.

KDD '20, August 23–27, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-7998-4/20/08…$15.00
https://doi.org/10.1145/3394486.3403295

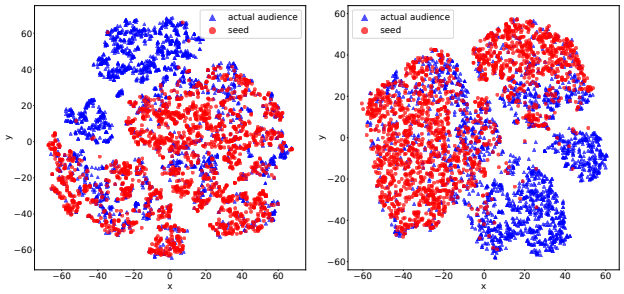
Scalability & Timeliness. Lots of companies conduct many marketing campaigns to promote their mobile products over massive users every day. Take Ant Financial [1] as an example, in order to promote the mobile financial services (*e.g.,* Alipay, Ant Insurance, Ant Credit, Ant Cash, etc), the marketing system in the company runs about 300 campaigns toward 1 billions of users every day, which also results in a massive amount of interactive data between users and campaigns. Instead of training an end-to-end model for each campaign, which is extremely resource-consuming (*i.e.,* CPU, memory, time) and thus impractical, we propose to decompose the industrial audience expansion tasks into two stages:

- **Offline Stage** to perform user representation learning based on the massive amount of historical interactions between users and campaigns.
- **Online Stage** to asynchronously train a lightweight model for each campaign with the user embedding from the offline stage.

High-quality user representation. One of the pivotal issues of such a two-stage approach is the quality of user representation (or user embedding). [1, 11] propose to learn user embedding by leveraging his/her historical interactive items (*i.e.,* Pins in [1] and contents in [11]) with a two-tower style DNN model. However, both of them neglect two crucial issues toward better user representation. The former is the intricate *high-order interactive information* between users and items (*i.e.,* campaigns). This is vital for the inactive users with limited historical campaigns, who happen to be the target of some marketing campaigns (campaigns that aim to enliven inactive users). The latter is the *entanglement of the user embedding*. In Ant Financial, the campaigns fall into different domains (*e.g.,* payment service, insurance, wealth management services, movies, local services, *etc.*), which implies a user interacts with campaigns from different domains for different reasons. Hence, a high-quality user embedding should have the ability to recognize and disentangle the heterogeneous latent aspects. Tackling this challenge, we propose an **Adaptive and Disentangled Graph Neural Network** (called **AD-GNN** for short) to adaptively capture the high-order interactive information of the bipartite graph formed by users and campaigns, and generate disentangled user embeddings.

Coverage bias induced by the seeds. In the setting of audience expansion, a set of seed users is given to guide the expansion process. Note that in the previous literature [1], the seeds are usually treated as positive examples of a classifier to distinguish seeds and other users. The seeds are generated in different ways, *e.g.,* provided by experts, collected from some past similar campaigns, or collected from users' feedbacks during the targeted campaign is running. Such a guidance usually induces bias *w.r.t.* the actual audiences because of various reasons (*e.g.,* the seeds given by experts could be noisy, the past similar campaigns could be different from the targeted one in some aspects, the feedbacks collected from the targeted campaign can not characterize the whole demographics at the very beginning of the promotion). In this paper, the term *coverage bias* is used to denote the gap between the seeds and the actual audiences. To our best knowledge, this is the first work to tackle this challenge in the audience expansion problem. Furthermore, the

[1] https://www.antfin.com/



(a) Campaign A (b) Campaign B

Figure 1: Visualization of the seed users (red cycles) and the actual audiences (blue triangles) in the embedding space (the embeddings are generated by AD-GNN and are visualized by t-SNE).

visualizations shown in Figure 1 validate the existing of the coverage bias, since the distribution of the seeds (*i.e.,* red cycles) cannot cover the distribution of the entire set of the actual audiences (*i.e.,* blue triangles) in the embedding space. Tackling this problem, a **Knowledge Distillation** mechanism is adopted within the online lightweight **A**udience **E**xpansion model (called **KD-AE** for short). By absorbing knowledge from the offline AD-GNN model, which explores the actual audiences of all historical campaigns, KD-AE can alleviate the coverage bias induced by the seeds of a certain campaign.

Our Approach. Addressing the challenges mentioned above, in this paper, we present an industrial system for audience expansion in mobile marketing scenario, called **Hubble System**. The architecture of Hubble System follows a hybrid online-offline manner. The core of the offline module is an industrial-scale AD-GNN model. By adaptively exploring the high-order interactive information of the user-campaign bipartite graph, AD-GNN generates user embedding (as well as campaign embedding) in a disentangled manner, which subtly captures users' heterogeneous latent intentions toward campaigns. Based on the user embedding learned from AD-GNN, the online module trains a lightweight KD-AE model to alleviate the coverage bias induced by the seeds and perform audience expansion for each upcoming campaign. Note that Hubble System suggests to retrain AD-GNN in a relatively long period (*e.g.,* weekly) and to train a lightweight KD-AE once the request of an audience expansion task is received. The whole online procedure for one campaign only takes several minutes and limited computational resources. Thus Hubble System has the ability to handle industrial-scale problems. Figure 2 demonstrates the complete architecture of Hubble System. The system overview and the algorithmic design are discussed in section 2 and section 3, respectively.

Hubble System has been deployed in the production environment of Ant Financial, to help hundreds of campaigns seek proper audiences from more than one billion users every day. Thanks to the hybrid online-offline two-stage architecture, Hubble System is able to finish an audience expansion task in several minutes and still outperforms the state-of-the-art end-to-end approaches, which takes hours for each task. Moreover, extensive offline experiments are conducted to verify the effectiveness and efficiency of the offline

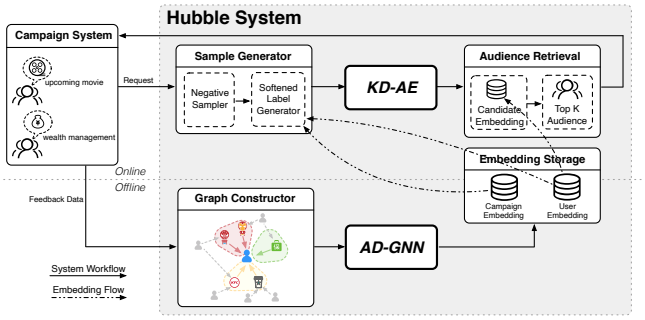


Figure 2: The architecture of Hubble System.

AD-GNN model, the online KD-AE model, as well as the whole Hubble System. We also perform online A/B testings in the production environment of Ant Financial. The experimental results demonstrate the superiority of the proposed Hubble System, which will be detailedly discussed in section 4.

2 SYSTEM OVERVIEW

In this section, we will describe the system overview of the proposed Hubble System, as well as some implementation details of the core components. As shown in Figure 2, Hubble System (in the gray rectangle) employs a hybrid online-offline architecture. Note that the *campaign system*, which is responsible for maintaining all campaigns in Ant Financial, is not included in Hubble System.

The offline module of our proposal contains a graph constructor and an AD-GNN model. The *graph constructor* collects the feedback data (*i.e.,* the interactions between users and campaigns) from the campaign system and then updates the user-campaign bipartite graph periodically (*i.e.,* daily in our implementation). Since it is implemented based on the powerful MaxCompute[2] platform from Alibaba Cloud, the graph constructor is able to handle the whole user-campaign graph with one billion users and tens of thousand historical campaigns, and maintain daily update in about one hour. The user-campaign graph will be defined in subsection 3.1. By adaptively exploring the large-scale user-campaign graph, an AD-GNN is trained to generate user embedding in a disentangled manner. As the learning objective of AD-GNN follows the link prediction setting (see the description in subsection 3.2), we implement AD-GNN based on the Ant Graph Learning system presented in [22], which is specially designed for the industrial-scale graph data. After finishing model training, the user and campaign embeddings are generated, and then stored into the embedding storage submodule (*i.e.,* HBase Services). Since users' intentions toward campaigns are relatively stable within a short period, the AD-GNN model is retrained once a week.

The online module of Hubble System contains a sample generator, an KD-AE model, and an audience retrieval submodule. After receiving the request of running audience expansion for a certain campaign from the campaign system, the *sample generator* first samples a list of users from the whole user set as negative examples for KD-AE. Then, based on the user/campaign embeddings generated by AD-GNN, the sample generator constructs the softened labels for all examples (see the description in subsection 3.3), which will be used to perform knowledge distillation mechanism in KD-AE. A KD-AE model is trained in several minutes and then deployed in the audience retrieval submodule. Based on the trained KD-AE, the *audience retrieval* submodule infers the affinity score for all candidate users *w.r.t.* the targeted campaign, then selects the top N users with higher score as the expanded result and returns it to the campaign system. Specially, the whole procedure of running an audience expansion task takes about 15 minutes and the training of KD-AE takes about 7 minutes, which is 5 times faster than a conventional end-to-end approach with much less computation and storage resources.

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

3 ALGORITHMIC DESIGN

In this section, we will elaborate the algorithmic design of Hubble System, mainly including the offline AD-GNN model and the online KD-AE model, as well as the connection between them. Let's start with some notations and definitions.

3.1 Notations and Definitions

In this section, we will introduce the formal definition of the user-campaign graph and the audience expansion problem in the mobile marketing scenario.

Definition 3.1. ***User-Campaign Graph.*** In our scenario, the user-campaign graph is an *attributed graph*, which can be denoted as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{X}\}$. Here the node set $\mathcal{V}$ consists of users $\mathcal{U}$ and campaigns C (*i.e.,* $\mathcal{V} = \mathcal{U} \cup C$), while the edge set $\mathcal{E}$ consists of the interactions between users and campaigns. We use an edge (u, c) to denote user u has interacted with campaign c. The node feature matrix $\mathbf{X} \in \mathbb{R}^{f \times |\mathcal{V}|}$ contains the vectorized attribute information of every nodes. Specifically, a column of $\mathbf{X}$, denoted as $\mathbf{x}_i \in \mathbb{R}^f$ represents the f-dimensional feature vector of node i. Obviously, $\mathcal{G}$ is a bipartite graph since the edges only exist between the user set $\mathcal{U}$ and the campaign set C.

For convenience, we put user features and campaign features into the same feature space with disjoint dimensions. Thus all node features are f-dimensional vectors in our setting. We take $\mathcal{G}$ as the input graph for AD-GNN, which will be discussed in the next section. Before that, we formalize the audience expansion problem as follows:

Definition 3.2. ***Audience Expansion.*** Given a set of candidate users $\mathcal{U}_c \subset \mathcal{U}$ and a set of seed users $\mathcal{S}_c \subset \mathcal{U}$ *w.r.t.* an upcoming campaign c, the audience expansion task requires to seek a larger amount of users from $\mathcal{U}_c$, denoted as $\mathcal{U}_c^* \subset \mathcal{U}_c$, which have the greatest potential to convert when c is conducted on them. Note that the size of $\mathcal{U}_c^*$ is usually given by the operator of c and is much larger than the size of $\mathcal{S}_c$, *i.e.,* $|\mathcal{U}_c^*| \gg |\mathcal{S}_c|$.

3.2 Offline Representation Learning

3.2.1 Data Preprocessing. The AD-GNN model is responsible for generating high-quality user embeddings representing users' heterogenous intentions toward different campaigns, as well as the campaign embeddings. To achieve this goal, we formalize the learning objective of AD-GNN in a graph learning based link prediction manner. More specifically, based on a historical user-campaign

graph, AD-GNN aims to predict whether a link exists between a user and a campaign in the future, *i.e.,* the user will convert when the campaign is conducted on him/her.

Following this setting, we first split the interactive data between users and campaigns into two parts according to the timeline. The former part is used to construct a user-campaign graph, while the latter part is used to build the training/validation/test set. In our mobile marketing scenario, the interaction between user and campaign falls into two types: *exposure* which means the campaign has been shown to the user in our APP and *conversion* which means the user attended the campaign and has finished its business goal (*e.g.,* purchase the fund or insurance promoted by the campaigns, click the campaign-relative ads to visit some pages). We only employ the conversion as an edge when constructing the user-campaign graph, while both exposure and conversion to build the training and validation datasets (*i.e.,* conversion as positive example and exposure without conversion as negative example).

Formally, the training examples of AD-GNN can be denoted as $\mathcal{D}_G = \{(u, c, y)\}$. The triple (u, c, y) represents that user $u \in \mathcal{U}$ have label y on campaign $c \in C$, where $y = 1$ denotes conversion and $y = 0$ denotes exposure without conversion.

3.2.2 Adaptive and Disentangled Graph Neural Network. The AD-GNN mainly tackles two pivotal challenges toward high-quality user representation, *i.e.,* the high-order interactive information between users and campaigns, and the entanglement of user embedding. Addressing the former challenge, we first construct the user-campaign graph $\mathcal{G}$ as mentioned in the previous subsection and then adopt graph neural network over $\mathcal{G}$ to explore the high-order interactions as well as the abundant attributes. Considering the fact that a user connects with different campaigns for different reasons, we employ the disentangled mechanism [13] in the proposed AD-GNN, to disentangle user embeddings into several channels *w.r.t.* their heterogeneous intentions toward different campaigns. Moreover, to further improve the quality of user embedding, we refine the disentangled mechanism by incorporating the well-designed adaptive breadth functions, which enable AD-GNN to eliminate some noisy neighbors connected by casual user-campaign interactions thus generate adaptive and disentangled embedding.

Specifically, given the feature vectors of a target node u and its neighborhood $\mathcal{N}_u = \{v : (u, v) \in \mathcal{E}\}$, denoted as $\{\mathbf{x}_i : i \in \{u\} \cup \mathcal{N}_u\}$, the goal is to adaptively generate node u's K-channel disentangled embedding, *i.e.,* $\mathbf{h}_u = [\mathbf{h}_u^1, \mathbf{h}_u^2, ..., \mathbf{h}_u^K] \in \mathbb{R}^d$, where $\mathbf{h}_u^k \in \mathbb{R}^{\frac{d}{K}}$ denotes node u's embedding in the k^{th} channel.

Projection. Similar to [13] , for all nodes in the set $\{u\} \cup \mathcal{N}_u$, the disentangled mechanism first projects their feature vectors into K different subspaces, which denote the K different aspects of users' heterogenous intentions toward campaigns. The projection in the k^{th} subspaces of node i, denoted as $\mathbf{z}_i^k \in \mathbb{R}^{\frac{d}{K}}$, is calculated as follows:

$$\mathbf{z}_i^k = \frac{\sigma(\mathbf{W}_P^k \mathbf{x}_i + \mathbf{b}_p^k)}{\|\sigma(\mathbf{W}_P^k \mathbf{x}_i + \mathbf{b}_p^k)\|_2}, \tag{1}$$

where σ is a nonlinear activation function, $\mathbf{W}_p^k$ and $\mathbf{b}_p^k$ are the learnable parameters of channel k, and $\|\cdot\|_2$ denote the l_2-norm operator.

Algorithm 1: The pseudocode of AD-GNN.

Input: disentangled channel K, depth L, routing iteration T, original node features $\mathbf{H}^{(0)} = \mathbf{X} = \{\mathbf{x}_u : u \in \mathcal{V}\}$, user-campaign graph $\mathcal{G}$.
Param: $\Theta_G = \{\mathbf{W}_P^{k,(l)}, \mathbf{b}_P^{k,(l)}, \mathbf{W}_B^{(l)}, \mathbf{b}_B^{(l)}, \mathbf{v}_B^{(l)}\}_{k=1,l=1}^{K,L}$
Output: node embeddings of the L^{th} GNN layer $\mathbf{H}^{(L)}$

```
while not converged do
    for l = 1 to L do
        // the l^th AD-GNN layer
        for each node u ∈ V do
            calculate {α_{u,v}^{(l)}}_{v∈N_u} w.r.t. Equation 4
            // projection
            for i ∈ {u} ∪ N_u do
                calculate {z_i^{k,(l)}}_{k=1}^K w.r.t. Equation 1
            end
            // neighborhood routing
            h_u^{k,(l)} ← z_u^{k,(l)}, ∀k = 1, 2, ..., K
            for routing iteration t = 1, 2, ..., T do
                for each neighbor v ∈ N_u do
                    calculate {p_{u,v}^{k,(l)}}_{k=1}^K w.r.t. Equation 2
                end
                for channel k = 1, 2, ..., K do
                    construct h_u^{k,(l)} w.r.t. Equation 5
                end
            end
            h_u^{(l)} ← [h_u^{1,(l)}, h_u^{2,(l)}, ..., h_u^{K,(l)}]
        end
    end
    backpropagation w.r.t. L_G, i.e., Equation 6
end
return H^{(L)}
```

Neighborhood Routing. In order to construct each disentangled component $\mathbf{h}^k$, the disentangled mechanism performs *neighborhood routing* iteratively. It starts by initializing $\mathbf{h}^k$ as $\mathbf{z}^k$, then each iteration of the neighborhood routing takes two step: first calculate the probability $p_{u,v}^k$ that neighbor v can be used to construct $\mathbf{h}_u^k$:

$$p_{u,v}^k = \frac{\exp(\mathbf{z}_v^{k\mathrm{T}} \mathbf{h}_u^k / \tau)}{\sum_{k'=1}^K \exp(\mathbf{z}_v^{k'\mathrm{T}} \mathbf{h}_u^{k'} / \tau)}, \tag{2}$$

and second construct each $\mathbf{h}_u^k$ as follows:

$$\mathbf{h}_u^k = \frac{\mathbf{z}_u^k + \sum_{v \in \mathcal{N}_u} p_{u,v}^k \mathbf{z}_{u,v}^k}{\|\mathbf{z}_u^k + \sum_{v \in \mathcal{N}_u} p_{u,v}^k \mathbf{z}_{u,v}^k\|_2}. \tag{3}$$

Here, τ is a hyper-parameter that controls the hardness of the assignment in the second step. The neighborhood routing is iterated for T times and output the final disentangled embedding.

However, the above disentangled mechanism neglects that there are some noisy edges caused by the casual conversion between users and campaigns, which are common in the mobile marketing

scenario (*e.g.,* click some campaign-relative ads casually). Tackling this problem, we refine the disentangled mechanism by incorporating the adaptive breath function to enhance the propagation along the important edges thus eliminate the noise. Specifically, before the disentangled mechanism, the normalized adaptive breadth score $\alpha_{u,v}$ for node u and its neighbor v is calculated as:

$$\alpha_{u,v} = \frac{\exp(\mathbf{v}_B^{\mathrm{T}} \tanh(\mathbf{W}_B[\mathbf{x}_u, \mathbf{x}_v] + \mathbf{b}_B))}{\sum_{v' \in \mathcal{N}_u} \exp(\mathbf{v}_B^{\mathrm{T}} \tanh(\mathbf{W}_B[\mathbf{x}_u, \mathbf{x}_{v'}] + \mathbf{b}_B))}, \tag{4}$$

where $\mathbf{W}_B$, $\mathbf{b}_B$ and $\mathbf{v}_B$ are learnable parameters. Then we refine Equation 3 by introducing the normalized adaptive breadth score α as follows:

$$\mathbf{h}_u^k = \frac{\mathbf{z}_u^k + \sum_{v \in \mathcal{N}_u} (\alpha_{u,v} \cdot p_{u,v}^k)\mathbf{z}_{u,v}^k}{\|\mathbf{z}_u^k + \sum_{v \in \mathcal{N}_u} (\alpha_{u,v} \cdot p_{u,v}^k)\mathbf{z}_{u,v}^k\|_2}. \tag{5}$$

Note that $\alpha_{u,v} \cdot p_{u,v}^k$ can be treated as the adjusted weight which tends to pay more attention to the important neighbors when constructing disentangled embedding. Such an adaptive and disentangled layer can be stacked L times iteratively to capture information of L-hops neighborhood and generate the final disentangled embedding, denoted as $\mathbf{h}_u^{(L)}$ for node u (*i.e.,* $\mathbf{h}_u^{(0)} = \mathbf{x}_u$).

As mentioned above, the learning objective of AD-GNN follows the link prediction setting. Specifically, given the training set $\mathcal{D}_G$, we employ cross entropy loss as follows:

$$\mathcal{L}_G = -\frac{1}{|\mathcal{D}_G|}\left\{\sum_{(u,c,y)\in\mathcal{D}_G} y\log(\hat{y}_G) + (1-y)\log(1-\hat{y}_G)\right\} + \frac{\lambda_G}{2}\|\Theta_G\|_2, \quad \text{where} \quad \hat{y}_G = \sigma({\mathbf{h}_u^{(L)}}^{\mathrm{T}}\mathbf{h}_c^{(L)}). \tag{6}$$

In Equation 6, the last term is a l_2 normalization to regularize all learnable parameters of AD-GNN, where Θ_G denotes the set of learnable parameters in AD-GNN and λ_G is a hyper-parameter to control the impact of the regularizer. Note that the predicted label $\hat{y}_G$ is calculated by a sigmoid function σ of the dot product of the final embedding of u and c. The algorithm 1 demonstrates the entire procedure of AD-GNN.

Here, we have the adaptive and disentangled embeddings for all users (as well as all past campaigns) implied users' intentions toward campaigns. The following online stage is to perform audience expansion for a given campaign.

3.3 Online Audience Expansion with Knowledge Distillation

As mentioned in section 2, the online lightweight KD-AE model is responsible for performing audience expansion *w.r.t.* a given upcoming campaign c. Since the only information about c is the set of seed users $\mathcal{S}_c$, the task of audience expansion is to take $\mathcal{S}_c$ as input, and to find out a much larger expanded user set $\mathcal{U}_c^*$ in which the users are similar to the seeds and have potential to convert when c is conducted on them. Obviously, the seed users are treated as positive examples, while the negative examples are generated by a simple sampler which randomly selects a certain amount of users from $\mathcal{U} - \mathcal{S}_c$, denoted as $\mathcal{U}_n \subset \mathcal{U} - \mathcal{S}_c$.

Conventional audience expansion techniques suggest to train a classifier (*e.g.,* LR, GBDT, DNN, *etc.*) with positive $\mathcal{S}_c$ and negative $\mathcal{U}_n$. Such methodology neglects the fact that the seeds are usually biased thus fails to cover the entire actual audiences. Intuitively, to eliminate such coverage bias, one feasible approach is to seek help from the offline AD-GNN model, which has learned knowledge from all historical interactions between users and past campaigns. Inspired by knowledge distillation mechanism [5], we employ such a teacher-student style in KD-AE, to absorb knowledge from the offline trained heavyweight teacher model (*i.e.,* AD-GNN) and to guide the training of a lightweight student model.

The connection between a teacher model and a student model is the softened label predicted by the teacher model. However, most upcoming campaigns are new in our scenario. Thus it's hard for AD-GNN to make predictions for the users in $\mathcal{S}_c \cup \mathcal{U}_n$ *w.r.t.* the new campaign c. Therefore, the first thing to do is to construct softened labels for users in $\mathcal{S}_c \cup \mathcal{U}_n$.

3.3.1 Softened Label Construction. Although AD-GNN is unavailable to directly predict for a new campaign c, there still exists a certain amount of past campaigns which are similar to c. If we find out these similar past campaigns, the softened label for a certain user could be constructed according to the average predictions of AD-GNN on these campaigns. Given the seed users $\mathcal{S}_c$, to find out the past campaigns similar to c, a straightforward method is to first calculates the similarity scores between all seed embeddings $\{\mathbf{h}_u^{(L)}\}_{u\in\mathcal{S}_c}$ and all past campaign embeddings $\{\mathbf{h}_c^{(L)}\}_{c\in C}$, with the prediction model in Equation 6. Then, we select the top k campaigns with the highest average similarity as results. However, this method suffers a complexity of $O(|\mathcal{S}_c| \cdot |C|)$. As $|\mathcal{S}|$ could be several millions and $|C|$ could be tens of thousand in the industrial scenario, this method is time consuming thus impractical.

Considering the fact that some users may have similar preferences toward campaigns, we propose to divide the seeds into several groups according to their preference, which is encoded in user embedding, and calculate the similarity between user groups and past campaigns. Specifically, we run a k-means clustering method based on seed embeddings $\{\mathbf{h}_u^{(L)}\}_{u\in\mathcal{S}_c}$, and then represent each user cluster as the average embedding of users in it (called cluster embedding). For each cluster, we find the most similar past campaign based on the similarity between its cluster embedding and all past campaign embeddings. Finally, for k user clusters, we obtain k most similar campaigns, *i.e.,* $\{\tilde{c}_i\}_{i=1}^k$. The softened label y_s of user $u \in \mathcal{S}_c \cup \mathcal{U}_n$ can be defined as $y_s = \frac{\sum_{i=1}^k \sigma({\mathbf{h}_u^{(L)}}^{\mathrm{T}}\mathbf{h}_{\tilde{c}_i}^{(L)})}{k}$. The time complexity of the whole procedure is $O(|\mathcal{S}_c| \cdot k \cdot N + |C| \cdot k)$, where N is the number of iterations in k-means.

3.3.2 Learning Objective. Given the hard label y_h ($y_h = 1$ indicates seed user otherwise $y_h = 0$) and the softened label y_s, the training set of KD-AE can be denoted as $\mathcal{D}_A = \{(u, y_h, y_s)\}$, where $u \in \mathcal{S}_c \cup \mathcal{U}_n$. Moreover, taking the adaptive and disentangled embedding $\mathbf{h}_u^{(L)}$ as inputs, a simple mulit-layer perceptron with two fully-connected layers is applied as the predicted model, *i.e.,* $\hat{y}_A = \mathrm{MLP}(\mathbf{h}_u^{(L)})$. Similar to [5], we formalize the learning objective

of KD-AE as follows:

$$\begin{aligned}\mathcal{L}_A = &- \frac{1}{|\mathcal{D}_A|} \sum_{(u,y_h,y_s)\in\mathcal{D}_A} y_h \log(\hat{y}_A) + (1-y_h)\log(1-\hat{y}_A) \\ &- \frac{\gamma}{|\mathcal{D}_A|} \sum_{(u,y_h,y_s)\in\mathcal{D}_A} y_s \log(\hat{y}_A) + (1-y_s)\log(1-\hat{y}_A) \\ &+ \frac{\lambda_A}{2}\|\Theta_A\|_2,\end{aligned} \tag{7}$$

where γ is a hyper-parameter to control the impact of the second term, λ_A is another hyper-parameter to control the impact of the last term, and Θ_A denotes the set of all learnable parameters of KD-AE.

Note that in Equation 7, the first cross-entropy loss denotes the hard loss which is used to distinguish the positive and negative examples. It can be treated as a classical binary classification loss. The second cross-entropy loss denotes the soft loss which tends to restrict the predictions to approximate the softened labels. Since the softened labels indicate the predictions made by the teacher model, the second term is responsible for alleviating the coverage bias by absorbing knowledge from the teacher model. Last, the third term is a l_2 normalization to regularize all learnable parameters of KD-AE.

The trained KD-AE is deployed in the audience retrieval submodule and then makes predictions for all candidate users with their embeddings. Finally, the audience retrieval submodule will select top $|\mathcal{U}_c^*|$ users with the highest predicted score to form the expanded audience set $\mathcal{U}_c^*$, and send them back to the Campaign System.

4 EXPERIMENTS

[3] In this section, we conduct extensive offline and online experiments on the proposed Hubble System, to demonstrate its effectiveness and efficiency. In summary, our experiments aim to answer the following three questions:

- **Q1**: Does our proposed Hubble System outperform other state-of-the-art approaches in the audience expansion task?
- **Q2**: Does the proposed AD-GNN learn high-quality user representation?
- **Q3**: Does the proposed KD-AE alleviate the coverage bias induced by the seed users?

Before diving into the experimental results, we first introduce the experiment settings, including datasets, comparison methods and evaluation metrics.

4.1 Experiment Settings

Datasets. Since this paper mainly focuses on the industrial problem of audience expansion in the mobile marketing scenario, we employ the real-world industrial datasets which are collected from the mobile marketing scenario of Alipay. Specifically, for the offline task of user representation learning, we randomly select a portion of users, as well as their interacted campaigns exhibited in a certain page of Alipay APP. All the interactions occur in one week in January, 2020. The former 70% of them are used to construct the user-campaign graph, while the rest is used to build the training/validation/test sets in a 60%/10%/30% manner, respectively. For the online audience expansion task, we select three campaigns (*i.e.,* payment service, travel and fund service), which are conducted in the next week, as the targeted campaigns for evaluation. The seed users, actual audiences, and candidate users of the three selected campaigns are collected and form three datasets for the audience expansion task, respectively. Note that the original user features consist of the discretization of abundant user profiles and behaviors in Alipay, while the original campaign features consist of the one-hot encoding of the campaign id. Detailed statistics of our datasets are shown in Figure 3.

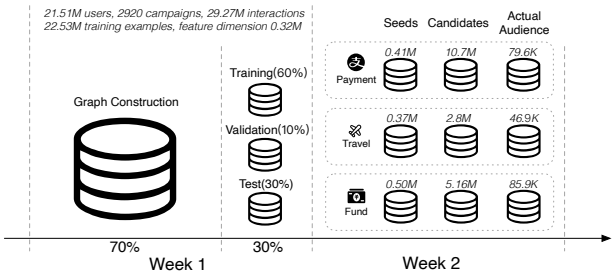


Figure 3: The statistics of Alipay dataset.

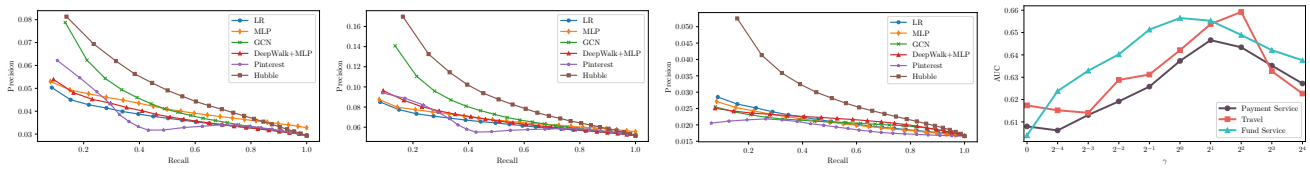
Compared Methods. Several state-of-the-art methods are employed in the experiments. They fall into two categories: end-to-end model and two-stage model.

- **LR**: an end-to-end **L**ogistic **R**egression (LR) classifier based on the raw features of users.
- **MLP**: an end-to-end **M**ulti-**L**ayer **P**erceptron (MLP) based on the raw features of users.
- **GCN**: an end-to-end **G**raph **C**onvolutional **N**etwork (GCN) [8] based on the user-campaign graph.
- **DeepWalk+MLP**: a two-stage approach. In the offline stage, a DeepWalk [16] model is trained based on the user-campaign graph. In the online stage, a MLP is trained based on the concatenation of user embedding generated by DeepWalk and the raw features of users.
- **Pinterest**: a two-stage approach presented by Pinterest [1]. In the offline stage, a global user embedding model is trained to generate user embedding. In the online stage, an embedding-based scoring model is employed to assign the affinity score for each user *w.r.t.* a given campaign.

The former three end-to-end methods take positive $\mathcal{S}_c$ and negative $\mathcal{U}_n$ as input to train a binary classifier. For the latter two two-stage methods, their offline stages generate user embedding based on the user-campaign interactions, while their online stages build embedding-based audience expansion model with $\mathcal{S}_c$ and $\mathcal{U}_n$.

For all two-stage methods (*i.e.,* DeepWalk+MLP, Pinterest and Hubble), the embedding sizes of users and campaigns are both set to 64. For all the neural network-based methods (i.e., MLP, GCN, DeepWalk+MLP, Pinterest and Hubble), the learning optimizer is Adam with a learning rate of 1e-4, and the batch size is set to 256. All the models are fine-tuned, and their specific settings are as follows. In LR, we employ the L1 regularization and set its weight to 1.0. In

[3] A standalone version of AD-GNN and KD-AE will be open-source after this paper is published. We also consider to release the relative datasets after passing the security check of the company.



(a) Payment service campaign (b) Travel campaign (c) Fund service campaign (d) Impact of γ

Figure 4: Subfigure (a), (b) and (c) show the P-R@m% curves of compared methods in three audience expansion tasks. Here, we set m ranging from 0.05 to 1.0 with 0.05 step size, each marker in the curves indicates a m value. Subfigure (d) shows the test AUC values of KD-AE with different γ.

MLP, the ReLU activation function is employed, and the dimension of each layer is set to 512, 128, 32 and 2, respectively. In GCN, we use two convolutional layers with the number of hidden units setting to 64 and 32, respectively. In DeepWalk+MLP, for each node in a graph, 20 paths are selected by random walk, each of which has a length of 10 hops. After concatenating the DeepWalk user embeddings with original user features, a followed MLP classifier is devised, which has 4 hidden layers. The dimensions of the hidden layers are 512, 256, 128 and 2, respectively. In Pinterest, we first implement their embedding method [20] to generate user embeddings. Then, in the step of audience scoring [1], we split the whole user embedding space into 10 random hyperplanes and repeat the entire process of determining random projections for 10 times. In Hubble, for the four hyper-parameters in AD-GNN Algorithm 1, we set $K = 4$, $L = 2$, $T = 5$ and $\tau = 1.0$. For the KD-AE, k is set to 5 in k-means for constructing softened labels, and the γ in Equation 7 is set to 2.0. Since the proposed knowledge distillation mechanism dramatically improves the audience expansion performance, we will further discuss the impact of γ in subsection 4.4.

Metrics. Either the audience expansion task or the intermediate AD-GNN model follows the binary classification manner (to predict whether a user is actual audience or whether a conversion interaction exists between a user and a campaign, respectively). Thus, we employ the commonly-used binary classification metric in our experiments, *i.e.,* **A**rea **U**nder the **C**urve (AUC). Moreover, to intuitively demonstrate the performance of the audience expansion task, we employ two novel metrics called **P**recision/**R**ecall at top m percent candidates (P@m% or R@m% for short). Specifically, we use $\mathcal{U}_{at}$ to denotes the set of actual audiences *w.r.t.* a certain campaign, and $\mathcal{U}_{cdd,m}$ to denote the set of m% candidate users with higher score predicted by the audience expansion model *w.r.t.* the certain campaign. Then we calculate P@m% and R@m% as follows:

$$P@m\% = \frac{|\mathcal{U}_{at}| \cap |\mathcal{U}_{cdd,m}|}{|\mathcal{U}_{cdd,m}|}, \quad R@m\% = \frac{|\mathcal{U}_{at}| \cap |\mathcal{U}_{cdd,m}|}{|\mathcal{U}_{at}|}. \quad (8)$$

These two metrics evaluate the effectiveness of audience expansion at different expanded levels (*i.e.,* different m in Equation 8).

4.2 Effectiveness of End-to-end Audience Expansion Task

In this experiment, we analyze the end-to-end performance of the audience expansion task for all compared methods. By treating the actual audiences as positive examples while other candidate users as negative examples, we calculate the AUC values of all compared methods in three audience expansion tasks, and report the results in Table 1. From Table 1, we have the following observations and analyses:

Table 1: Test AUC values of three audience expansion tasks.

Methods	Audience Expansion Tasks		
	Payment Service	Travel	Fund Service
LR	0.576	0.572	0.576
MLP	0.572	0.565	0.570
GCN	0.603	0.606	0.583
DeepWalk+MLP	0.581	0.582	0.594
Pinterest	0.567	0.563	0.544
Hubble	**0.647**	**0.654**	**0.655**

Among three end-to-end baselines, GCN outperforms MLP or LR in all three tasks, which implies that involving user-campaign graph can improve the performance of audience expansion. However, the time complexity of GCN is much higher than MLP or LR, thus GCN is impractical for this industrial scenario.

A state-of-the-art two-stage method, *i.e.,* Pinterest, performs slightly worse than MLP, which accords with the online A/B testing result reported by [1]. Another observation is that a simple two-stage graph learning baseline, *i.e.,* DeepWalk+MLP, achieves better performance against Pinterest. This further verifies the superiority of modelling high-order user-campaign interaction for audience expansion. However, this simple baseline still performs worse against MLP, thus we propose Hubble System for further improvement.

Our proposed Hubble System outperforms other audience expansion methods in all three tasks. On one hand, the offline part of Hubble System utilizes AD-GNN to adaptively explore the user-campaign graph and generate disentangled embedding, which describe users in a more comprehensive way. On the other hand, the online part employs KD-AE to alleviate the coverage bias introduced by seeds, which other methods fail to tackle. By combing these two advantages together, Hubble System achieves the best performance compared to both end-to-end classifiers and two-stage methods.

Furthermore, to intuitively demonstrate the performance of audience expansion, we calculate P@m% and R@m% with different m and plot a curve with P@m% as y-axis and R@m% as x-axis (P-R@m% curve) for each methods in Figure 4. Figure 4 further demonstrates the detailed results of audience expansion. Note that in most cases of our scenario, the amount of expanded audiences

accounts for at most 50% of the candidate users. That means the left part of a P-R@m% curve is more crucial than the right part in evaluating the performance of audience expansion. From Figure 4, we observe that our proposal has significant improvement in the left part of the P-R@m% curves, which is much more meaningful to our scenario.

4.3 Effectiveness of User Representation Learning

In this experiment, we analyze the performances of different user representation learning methods and illustrate the superiority of our proposed AD-GNN.

In order to comprehensively evaluate the user representation learning, we employ two extra public datasets, *i.e.,* TaobaoAD[4] and Tmall[5]. Although these two datasets are not collected from the mobile marketing scenario, they all contain user-item interactions. If we treat their items (*i.e.,* the categories of advertisement and the merchant, respectively) as campaigns, they still qualify for this experiment. Similarly, for the two datasets, we separate the interactions into two parts regarding the timeline. The former 70% is used for graph construction, while the rest to build the training/validation/test sets in a 60%/10%/30% manner, respectively. The original user features consist of discretized user profiles (TaobaoAD) or the one-hot encoding of users' purchased merchants (Tmall), while the original campaign features consist of the one-hot encoding of the campaign id.

Four state-of-the-art methods are employed as baselines, including DSSM [6], GCN [8], GAT [19] and DisenGCN [13].

For the latter three graph-based methods, we implement them with a link prediction learning objective, similar to AD-GNN (see Equation 6). All the compared methods are trained with the training set, and do hyper-parameter tuning with the validation set, and calculate the link prediction AUC values in the test set. We report the experimental results in Table 2.

Table 2: Test AUC values of the link prediction task.

Methods	Alipay	TaobaoAD	Tmall
DSSM	0.6981	0.5710	0.6496
GCN	0.7015	0.5421	0.5132
GAT	0.7437	0.5730	0.6654
DisenGCN	0.7546	0.5798	0.6623
AD-GNN	**0.7554**	**0.5833**	**0.6703**

From Table 2, we observe that three graph-based methods,*i.e.,* GAT, DisenGCN and AD-GNN, perform better than DSSM. This observation again verifies the idea that graph-based methods have the ability to capture high-order interactions between users and campaigns thus generate better user representation. We also observe that these three graph-based methods outperform the conventional GCN, which implies that both the adaptive aggregator and the disentangled mechanism are more suitable in our scenario. At last, our proposed AD-GNN combines the adaptive aggregator and the disentangled mechanism together. It is able to eliminate the noisy neighbors and disentangle the user embedding at the same time, thus achieve the best performance among all compared methods on all three datasets. Note that GAT and DisenGCN can be viewed as two simplified versions of the proposed AD-GNN, *i.e.,* GAT is the adaptive but not disentangled version, while DisenGCN is the disentangled but not adaptive version.

[4]https://tianchi.aliyun.com/dataset/dataDetail?dataId=56
[5]https://tianchi.aliyun.com/dataset/dataDetail?dataId=42

4.4 Impact of Knowledge Distillation toward Coverage Bias

In this experiment, we analyze the impact of the proposed knowledge distillation mechanism in KD-AE. As mentioned in section 1, in the scenario of audience expansion, there exists a gap between the seed users and the actual audiences, *i.e.,* coverage bias. The proposed KD-AE employs a novel knowledge distillation mechanism to absorb knowledge from the offline AD-GNN model and hopes to alleviate the coverage bias. To verify the effectiveness of this idea, we tune the hyper-parameter γ in Equation 7 ranging from 0 to 2^4, and report the variation of AUC values of the three audience expansion tasks in Figure 4d. Note that when $\gamma = 0$, the KD-AE degenerate to a MLP classifier with positive examples $\mathcal{S}_c$ and negative examples $\mathcal{U}_n$.

From Figure 4d, we observe that the three test AUC curves roughly follow the single-peaked style with a peak at around 2^0 to 2^2. Compared with the cases of $\gamma = 0$, which is a MLP classifier, the KD-AE model with a applicable γ (*e.g.,* $\gamma = 2$) achieves significant improvement. It verifies the superiority of the proposed novel knowledge distillation mechanism. By absorbing knowledge from offline AD-GNN, KD-AE alleviates the coverage bias thus performs better in the audience expansion tasks.

Table 3: Online A/B testing results.

Campaigns	Gain # exposure	Gain # conversion	Gain CVR	Training Time
Fund	+0.7%	+18.8%	+18.0%	5x speed-up (5min)
Insurance	+0.6%	+23.8%	+23.1%	5x speed-up (7min)

4.5 Online A/B Testing

The Hubble System has already been deployed in the production environment of Alipay to serve a billion users and conduct tens of thousand campaigns. Here, we conduct online A/B testing experiments, to demonstrate the performance of Hubble System in real traffic. Specifically, we perform audience expansion for two campaigns, *i.e.,* a fund-related campaign and an insurance-related campaign. For each of them, we randomly split all candidates into two buckets with the same size. The control group here is an end-to-end MLP classifier as mentioned in subsection 4.1, while the treatment group is the proposed Hubble System. We allow them to select the same amount of top-scored users from their buckets, respectively. Then we conduct the campaigns on the selected users and report the results based on the following three metrics: **# exposure**: the amount of users who have been exposed by the campaign, **# conversion**: the amount of users who have converted in the campaign, **CVR**: the conversion rate of the campaign. Note that

the audience expansion task is expected to find users who are most likely to convert in the current campaign. Therefore, a higher CVR indicates a higher quality of selected audiences. Due to the data security policy of Alipay, we will not reveal the specific amount of these metrics mentioned above. Instead, we report the gains of the treatment group against the control group in Table 3.

From Table 3, we can observe that the proposed Hubble System maintains a great improvement on the number of conversion and CVR, as well as slightly improvement in the number of exposure, against the MLP classifier. Note that a 18% to 24% gain of CVR in the industrial scenario is a very significant improvement and will result in great economic benefit to the company. Moreover, in terms of training efficiency, our proposal is 5 times faster than a MLP classifier. Specifically, the proposed KD-AE takes the 64-dimensional user embeddings as input, which is more than 4600 times smaller against the input of MLP, *i.e.,* the 0.3 million dimensional original user feature. This huge gap results in the great improvement of training time, as well as the great reduction of computation resources (CPU and memory). In summary, the proposed Hubble System performs more effectively and efficiently in the real-world online A/B testing, thus is more suitable for the industry.

5 RELATED WORK

Audience Expansion. Referring to audience expansion, the related work can be categorized into two lines: similarity-based methods [10, 14, 15, 17], and embedding-based methods [1, 7, 11]. Methods in the latter category propose to expand audiences based on user embeddings generated by a machine learning model, and have achieved success in some companies (*e.g.,* Linkedin [7], Pinterest [1] and Tencent [11]). However, the two issues, *i.e.,* how to obtain high quality embeddings and how to reduce coverage bias of the seeds, have not been well investigated yet.

Graph Learning. Recent literature has successfully developed various types of graph learning methods. This includes graph embeddings [3, 16, 18], and graph neural networks (GNNs) [21]. The methods in the latter category study the learning of node representation by aggregating the structural and attributed information of their neighbors [4, 8, 12, 13, 19]. The proposed AD-GNN is inspired by [19] and [13]. By combining the adaptivity and disentanglement together with a well-designed aggregator, AD-GNN has the ability of adaptively exploring graph data and then generating disentangled representation.

Knowledge distillation. The knowledge distillation mechanism is first proposed in [5], which aims to transfer the generalization ability of a cumbersome teacher model to a lightweight student model. Such teacher-enhanced student models have achieved success in various settings, like supervised [2] and semi-supervised [9] scenarios. In our case, since the teacher AD-GNN model cannot directly transfer its knowledge to the student model *w.r.t.* a new campaign, we devise a novel teacher-student structure to reduce the coverage bias in seeds.

6 CONCLUSION

In this paper, we proposed an innovative industrial system for audience expansion in mobile marketing, called Hubble. To better balance the effectiveness and efficiency, our proposal employs two stages: offline user representation learning and online audience expansion. For the offline stage, we propose a novel GNN model, called AD-GNN, to adaptively explore the user-campaign graph and generate high-quality user embedding in a disentangled manner. For the online stage, we propose a novel KD-AE model, which employs a well-designed knowledge distillation mechanism, to absorb knowledge from AD-GNN and eliminate the coverage bias introduced by the given seed users. By combining these two stages, Hubble System demonstrates its superior effectiveness and efficiency in both offline experiments and online A/B testing, compared to other state-of-the-art approaches. The Hubble System has been deployed in the production environment of Alipay to serve a billion users, and is achieving great economic benefit for the company. In the future, we will further explore how to integrate other heterogenous interactive data to enrich the user representation. As our system has the ability to better understand a user, it offers a chance of applying it into other industrial scenarios, such as recommendation, financial risk analysis, wealth management and so on.

REFERENCES

[1] Stephanie deWet and Jiafan Ou. 2019. Finding Users Who Act Alike: Transfer Learning for Expanding Advertiser Audiences. In *SIGKDD*. 2251–2259.

[2] Krzysztof J Geras, Abdel-rahman Mohamed, Rich Caruana, Gregor Urban, Shengjie Wang, Ozlem Aslan, Matthai Philipose, Matthew Richardson, and Charles Sutton. 2015. Blending lstms into cnns. *arXiv preprint arXiv:1511.06433* (2015).

[3] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *SIGKDD*. 855–864.

[4] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *NIPS*. 1024–1034.

[5] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).

[6] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry P. Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *CIKM*. 2333–2338.

[7] Yuchin Juan, Damien Lefortier, and Olivier Chapelle. 2017. Field-aware Factorization Machines in a Real-world Online Advertising System. In *WWW*. 680–688.

[8] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.

[9] Jinyu Li, Rui Zhao, Jui-Ting Huang, and Yifan Gong. 2014. Learning small-size DNN with output-distribution-based criteria. In *ISCA*. 1910–1914.

[10] Haishan Liu, David Pardoe, Kun Liu, Manoj Thakur, Frank Cao, and Chongzhe Li. 2016. Audience Expansion for Online Social Network Advertising. In *SIGKDD*.

[11] Yudan Liu, Kaikai Ge, Xu Zhang, and Leyu Lin. 2019. Real-time Attention Based Look-alike Model for Recommender System. In *SIGKDD*.

[12] Ziqi Liu, Chaochao Chen, Longfei Li, Jun Zhou, Xiaolong Li, Le Song, and Yuan Qi. 2019. GeniePath: Graph Neural Networks with Adaptive Receptive Paths. In *AAAI*. 4424–4431.

[13] Jianxin Ma, Peng Cui, Kun Kuang, Xin Wang, and Wenwu Zhu. 2019. Disentangled Graph Convolutional Networks. In *ICML*. 4212–4221.

[14] Qiang Ma, Eeshan Wagh, Jiayi Wen, Zhen Xia, Róbert Ormándi, and Datong Chen. 2016. Score Look-Alike Audiences. In *ICDM*. 647–654.

[15] Qiang Ma, Musen Wen, Zhen Xia, and Datong Chen. 2016. A Sub-linear, Massive-scale Look-alike Audience Extension System. In *BigMine*. 51–67.

[16] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: online learning of social representations. In *SIGKDD*. 701–710.

[17] Jianqiang Shen, Sahin Cem Geyik, and Ali Dasdan. 2015. Effective Audience Extension in Online Advertising. In *SIGKDD*. 2099–2108.

[18] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *WWW*. 1067–1077.

[19] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR*.

[20] Ledell Yu Wu, Adam Fisch, Sumit Chopra, Keith Adams, Antoine Bordes, and Jason Weston. 2018. Starspace: Embed all the things!. In *AAAI*.

[21] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. 2019. A comprehensive survey on graph neural networks. *arXiv preprint arXiv:1901.00596* (2019).

[22] Dalong Zhang, Xin Huang, Ziqi Liu, Jun Zhou, Zhiyang Hu, Xianzheng Song, Zhibang Ge, Lin Wang, Zhiqiang Zhang, and Yuan Qi. 2020. AGL: A Scalable System for Industrial-purpose Graph Machine Learning. In *VLDB*.