# III. METHODOLOGY

## A. Overview

In this section, we propose a framework named DA-MRG for bot detection. We first construct multi-relational graphs from the initial user features and the origin graph for the task. And then, a user representation learning module, consisting of a series of graph embedding layers and semantic attention layers, is designed to obtain the representation for each user. Finally, we propose domain-aware classifiers to discriminate bots from humans. The overall architecture of our methods is shown in Fig. 2. Furthermore, we introduce a federated learning framework for DA-MRG to implement the joint training among multiple participants.

## B. Multi-Relational Graph Generator

In a social network, We can obtain a multi-relational graph *G* = {*V,X,E,Y* } based on the interactive behaviors between users, where *V* = {*v*1*,v*2*,...,vn*} is the set of user nodes, *X* is the initial features of all user nodes, and *n* is the number of users. *eri,j* ∈ *E* is an edge between *vi* and *vj* with a relation *r* ∈ {1*,...,R*}, indicating an interactive behavior between user *i* and user *j*. Such as user *i* follows user *j* or user *i* comments user *j*. *Y* is the set of labels for all users.

Then, We present a multi-relational graph generator to generate the relational graph *Gr* for the origin graph *G*. The generator consists of edge separating and feature learning.

*Edge Separating.* We first generate all relational graphs  by reserving the relation *r* between users in the whole graph. Thus, the set of edges in *Gr* is *Er*. We add the two nodes of each edge *eri,j* in *Er* to the nodes set *Vr*, as with nodes’ features and labels. The relational graph *Gr* can be denoted as

*Gr* = {*Vr,Xr,Er,Yr*}*.* (1)

*Feature Learning.* Following the assumption that the features of the same user have a different effect on different relational graphs, we learn the features for each relational graph independently:

*X*ˆ*r* = *σ*(*Wr* · *Xr* + *br*)*,* (2)

where *Xr* is the initial features of nodes in *Gr*, and *σ*(·) is non-linearity.

## C. User Representation Learning Module

In this module, we obtain the final high-level representation for each user by a series of graph embedding layers and semantic attention layers. Particularly, we first gain the representations for each node in all relational graphs via multiple GNN-based graph embedding layers. Then we aggregate the representations of each node for final embedding based on the semantic attention networks.

*Multi-Relational Graph Embedding Layer.* We first construct a GNN-based graph embedding layer to obtain the representation for a specific node in each relational graph *Gr* in this module, which is shown as below:



where  is the set of *vi*’s 1-hop neighborhood in *Gr*, and is the representation of *vj* in the (*l* − 1)-th layer of GNNs. And we use the node features *X*ˆ*r* as the initial representation in the 0-th layer. Then, we gain *vi*’s presentations in the *l*-th layer of GNNs as follows:

*,* (4)

where and  are learnable parameters. And *zr*(*vi*) is used as the final representation for *vi* in the GNN-based embedding layer.

*Semantic Attention Layer.* Each user node’s representations in multiple relational graphs are gained via multiple GNNbased embedding layers. Considering the diverse importance of relations, we adopt a semantic attention layer to fuse all representations of each user node.

Firstly, we introduce a relational preference vector *ar* ∈

′

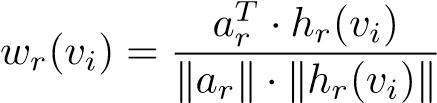
RR∗*d* for the relation *r*. For *vi*’s representation *zr*(*vi*) in the specific relation *r*, the weight assigned to *zr*(*vi*) for its contribution depends on the similarity between *ar* and *zr*(*vi*). To obtain the weight, we first transform *d*-dimension *zr*(*vi*)

′

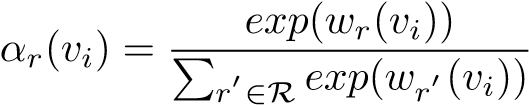
into *d* -dimension *hr*(*vi*):

*hr*(*vi*) = *σ*(*Wr* · *zk*(*vi*) + *br*)*,* (5)

where *σ*(·) is non-linearity and we use *tanh* in the paper. Then, we calculate the similarity between *ar* and *hr*(*vi*) as follows:

*,* (6)

where ∥·∥ is the *L*2 normalization of vectors. The weight assigned to relation *r* for node *vi* is normalized with *softmax* as follows:

*.* (7)

Finally, we fuse node *vi*’s representations in all relations:

*z*(*vi*) = X *αr*(*vi*) · *zr*(*vi*)*.* (8)

*r*∈R

|  |
| --- |
|  |

## D. Domain-Aware Classifiers

After the user representation learning module, we obtain the final high-level representation *zv* for each user node *v*. Generally, existing methods consider the detection task as a binary classification and the node’s representations are fed into a multiple fully connected neural network to gain the prediction:

*y*ˆ*v* = *σ*(*MLP*(*zv*))*,* (9)

where *σ*(·) denotes the activation function and *y*ˆ*v* is the predicted label of node *v*. Furthermore, Cross-Entropy is applied as the optimizer. Inspired by the observation that social bots in different domains have obvious differences, we propose the domain-aware classifiers to promote the detection performance. Specifically, we first train a bot classifier for each domain:

*Pd*(*v*) = *softmax*(*Wd* · *zv* + *bd*)*,* (10)

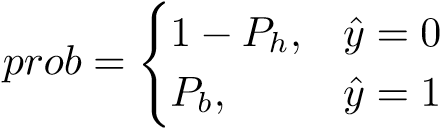
where *Pd*(*v*) denotes *v*’s bot probability in domain *d*. And then, we acquire the bot probability for *v* as follows:

*,* (11)

where *M* is the number of domains. Similarly, we train the human classifier:

*Ph*(*v*) = *softmax*(*Wh* · *zv* + *bh*)*,* (12)

where *Ph*(*v*) denotes *v*’s human probability. Thus, we obtain the predicted label as *y*ˆ = *argmax*([*Ph,Pb*]) and determine the final predicted bot probability as:

 (13)

## E. Federated learning

Due to the data privacy issues, we cannot collect data from multiple Social Network Services (SNSs) for centralized model training. Thus, we introduce a federated learning framework to address the problem. Each SNS, participating in the model training, downloads the global model from the server, trains with its own data, and uploads the trained model to the server, which aggregates models from all participants.

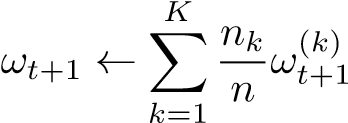
Specifically, suppose that *K* SNSs are contributing the federated learning in each round, the *k*-th participant calculates the local gradient of the model in round *t* according to Eq. 14.

*gk* = △*Fk*(*ωt*)*,* (14)

where *ωt* is the global parameters download from the server in the *t*-th round and each participant updates its own parameters locally as follows:

*.* (15)

Then, the server aggregates local parameters uploaded from all participants as Eq. 16:

*,* (16)

where *nk* is the data size of *k*-th participant, and *ωt*+1 is distributed to each participant in the (*t* + 1)-th round. The details are shown in Fig. 3.

In our work, the federated learning framework combined with the proposed model DA-MRG to implement the social bot detection across multiple social networks. We focus on exploring the influence of the number of participants and the amount of data in each participant. The overall process of our method is shown in Algorithm 1.