# Methodology

## Overview

Figure 2 presents an overview of our proposed graph-based and heterogeneity-aware Twitter bot detector. Specifically, we firstly construct a heterogeneous information network with diversified relations to represent the Twittersphere. We then learn node representations under each relation with our proposed relational graph transformers. After that, we take a global view of the graph and dynamically aggregates representations across relations with semantic attention networks. Finally, we classify Twitter users into bots or genuine users and learn model parameters.

## Graph Construction

We construct a heterogeneous information network (HIN) to represent the Twittersphere, which takes the relation heterogeneity into account and leverages diversified interactions between users. Specifically, we take Twitter users as nodes in the graph and we connect them with different types of edges, representing diversified relations on Twitter. We denote the set of relations in the HIN as *R* while our framework supports any relation settings.

Since this paper focuses on leveraging relation and influence heterogeneity to improve bot detection, we follow the same user information encoding procedure in the state-ofthe-art approach (Feng et al. 2021d) for fairness. We denote user *i*’s feature vector as *xi* and transform it with a fully connected layer to serve as initial features in the GNNs, *i.e.*,

(0)

*xi* = *σ*(*WI* · *xi* + *bI*) (1)

where *WI* and *bI* are learnable parameters, *σ* denotes nonlinearity and we use leaky-relu as *σ* without further notice.

## Relational Graph Transformers

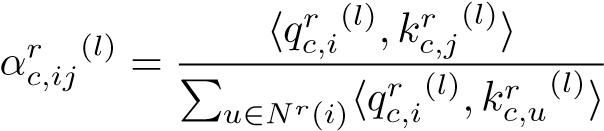
Inspired by Transformers (Vaswani et al. 2017) and its success in natural language processing, we propose relational graph transformers, a GNN architecture that incorporates transformers and operates on HINs. We firstly obtain query, key and value for the *c*-th attention head with regard to relation *r* and node *i*, formulated as

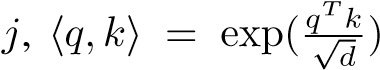
*qc,ir* (*l*) = *Wc,qr* (*l*) · *x*(*il*−1) + *brc,q*(*l*)*,*

*kc,jr* (*l*) = *Wc,kr* (*l*) · *x*(*jl*−1) + *brc,k*(*l*)*,* (2)

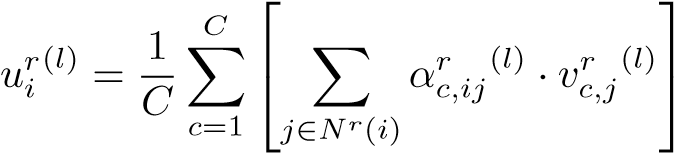
*vc,jr* (*l*) = *Wc,vr* (*l*) · *x*(*jl*−1) + *brc,v*(*l*)*,*

where *q*, *k* and *v* are query, key and value of the attention mechanism, (*l*) denotes the *l*-th layer of GNNs, all *W* and *b* are learnable parameters with regard to different relations and attention heads. We then model influence heterogeneity by calculating attention weights between different nodes by

*,* (3)

where *αc,ijr* (*l*) denotes the attention weight between nodes *i* and is the exponential scale dotproduct function where *d* is the hidden size of each attention head, *Nr*(*i*) denotes node *i*’s neighborhood with regard to relation *r*. We then aggregate over node neighborhood and attention heads to obtain node representation under relation

*r*, *i.e.*,

 *,* (4)

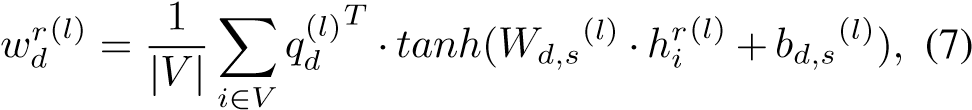
where *uri*(*l*) is the hidden representations of node *i* in *l*-th layer for relation *r*, *C* is the number of attention heads. We then apply the gate mechanism to obtained results to ensure smooth representation learning. We firstly obtain the gate level as follows

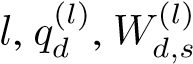
*,* (5)

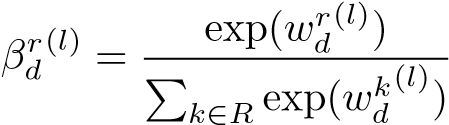
where [·*,*·] is the concatenation operation, *WA* and *bA* are learnable parameters. We then apply the gate mechanism to learned representation *uri*(*l*) and input *xri*(*l*) by *hri*(*l*) = *tanh*(*uri*(*l*)) ⊙ *zir*(*l*) + *xri*(*l*) ⊙ (1 − *zir*(*l*))*,* (6) where ⊙ denotes the Hadamard product operation and *hri*(*l*) is the learned representation of node *i* with regard to relation *r* in the *l*-th layer.

## Semantic Attention Networks

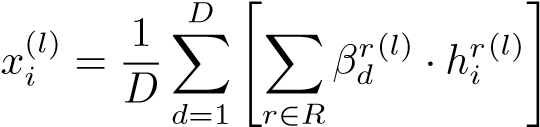
After analyzing the HIN while separating different relations, we use semantic attention networks to aggregate node representations across relations while preserving the relation heterogeneity entailed in the Twitter HIN. Firstly, we obtain the importance of each relation by taking a global view of all nodes in the HIN, *i.e.*,



where *wdr*(*l*) denotes the weight of relation *r* at the *d*-th attention head, *V* denotes the set of nodes in HIN, is the semantic attention vector at the *d*-th attention head in layer and are learnable parameters of the semantic attention network. We normalize the importance of each relation with softmax, formulated by

*,* (8)

where *βdr*(*l*) denotes the weight of relation *r*. We then fuse node representations under different relations with these weights as follows

 *,* (9)

(*l*) *r*(*l*) denotes the where *xi* denotes the output of layer *l*, *hi* results of relational graph transformers and *D* is the number of attention heads in the semantic attention network.

## Learning and Optimization

Each layer of GNN in our model contains a relational graph transformer and a semantic attention network. After *L* layers of GNNs, we obtain the final node representations *x*(*L*). We transform them with an output layer and a softmax layer for Twitter bot detection, *i.e.*,



where *y*ˆ*i* is our model’s prediction of user *i*, all *W* and *b* are learnable parameters. We then train our bot detector with supervised annotations and a regularization term, formulated as

*Loss* = −X[*yi* log(ˆ*yi*) + (1 − *yi*)log(1 − *y*ˆ*i*)]+*λ* X *w*2*,*

*i*∈*Y w*∈*θ*

(11)

where *Y* is the annotated user set, *yi* is the ground-truth labels, *θ* denotes all trainable parameters in the model and *λ* is a hyperparameter. To sum up, Algorithm 1 presents the overall training schema of our proposed graph-based and heterogeneity-aware bot detection framework, with time complexity of *O*(|*E*|) for each layer where *E* denotes the edge set, assuming embedding dimension and the number of relations are constants.