BIC: Twitter Bot Detection with Text-Graph Interaction and Semantic Consistency

Anonymous ACL submission

Abstract

Twitter bot detection is an important and meaningful task. Existing bot detection methods use either text modality to detect bots with anomalies in tweet patterns or graph modality to detect bots with abnormal clustering information. They do not allow text and graph modalities to interact with each other, which fails to learn the relative importance of the two modalities. As a result, these methods struggle to detect bots comprehensively. Besides, existing methods ignore the potential consistency within users' semantic information. In this paper, we propose a novel model named BIC that makes the text and graph modalities interactive. BIC also detects semantic consistency within tweet content. Specifically, BIC contains a text propagation module to learn text information, a graph propagation module to learn neighborhood information, and a text-graph interactive module to make the two interact. Besides, BIC contains a semantic consistency detection module to learn semantic consistency information from tweets. Extensive experiments demonstrate that our framework outperforms competitive baselines on a comprehensive Twitter bot benchmark. We also prove the effectiveness of the proposed interaction and semantic consistency detection.

1 Introduction

Twitter is a popular social media platform with enormous registered users from all over the world. However, where there is prosperity, there is darkness. Millions of Twitter bots try to sneak into genuine users in disguise. Twitter bots are controlled by automated programs and manipulated to pursue malicious goals such as spreading misinformation (Cresci, 2020) and conducting extreme propaganda (Berger and Morgan, 2015). In such a case, great efforts have been devoted to counteracting the Twitter bots.

Early works in Twitter bot detection primarily focused on feature engineering. A variety of feature categories were taken into consideration with traditional machine learning algorithms: (i) features derived from user tweets (Cresci et al., 2016); (ii) user metadata features (Yang et al., 2020; Miller et al., 2014; D'Andrea et al., 2015); (iii) features extracted from neighborhood information (Yang et al., 2013). Due to the subjectivity of feature engineering methods, methods based on neural networks began to take the stage. Recurrent neural network was utilized to improve the performance, Wei and Nguyen (2019) adopted long short-term memory and Kudugunta and Ferrara (2018) combined feature engineering and neural network to detect bots with user's semantic information. Graph neural networks brought detection capability to a higher level. Ali Alhosseini et al. (2019) adopted graph convolutional networks for bot detection. Feng et al. (2021a) constructed a graph that leveraged relation and influence heterogeneity. Besides, heuristic methods were proposed, among which comprehensive anomaly detection (Miller et al., 2014) and

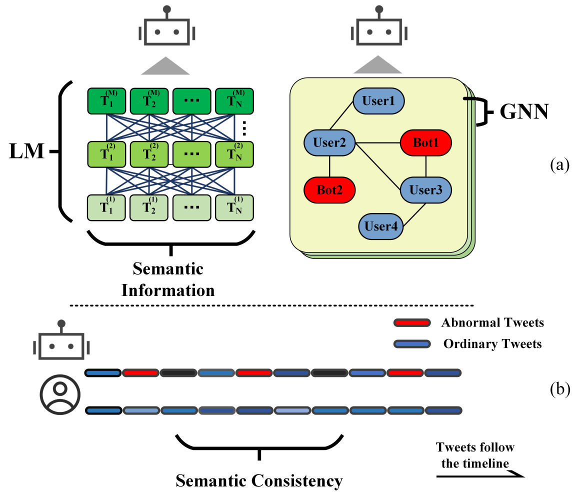

Figure 1: (a) Two kinds of defective models leveraging different modalities. T refers to tweet and *LM* refers to language model. (b) The semantic consistency characteristics of the Twitter bot and genuine human.

methods encoding tweets to string (Cresci et al., 2016) were representative. Self-supervised learning was also introduced to deal with bot evolution issue (Feng et al., 2021b).

Nevertheless, existing methods either leverage text modality or graph modality, which fails to consider both the important information from two modalities and make them interact rationally. As a result, they may be not enough in detecting all kinds of Twitter bots with anomalies in different aspects. The characteristics of two kinds of defective models leveraging different modalities are represented in Figure 1(a). Moreover, insufficient interaction such as simply averaging the two modalities may be unable to perceive the emphasis of two modalities for a user, therefore it could also not detect bots comprehensively.

Besides, existing methods fail to combat today's advanced Twitter bots which make all efforts to imitate humans by posting normal tweets similar to that of humans. However, these Twitter bots still attempt to send some malicious tweets for some purpose occasionally. Therefore, they lack some coherence and consistency within their overall tweet content, which has not been considered by existing methods. We illustrate the semantic consistency issue in Figure 1(b).

In this paper, we propose a framework BIC (Twitter **B**ot Detection with Text-Graph **I**nteraction and Semantic **C**onsistency) to leverage both information from two modalities with rational interaction and detect the coherence of tweet content. Specifically, BIC adopts text propagation module to tackle user's semantic information from tweets and description. BIC adopts graph propagation module to fully propagate user neighborhood information. BIC also contains an interactive module which generates deep link and fuses information between the two modules. The interactive module selects two interactive representations from two modalities to exchange information and is based on similarity to consider the relative importance of two modalities. Besides, BIC contains a semantic consistency detection module which can monitor user's abnormal tweet content leveraging attention weights for better detecting bots with inconsistent tweets. Finally, we aggregate all modules and conduct Twitter bot detection. Our main contributions are summarized as follows:

- We propose a model BIC, which leverages both text and graph modalities of users' information, with a similarity-based interactive model deeply linking the two modalities by interactive representations. BIC could learn the relative importance of modality and detect a bot more comprehensively.

- We propose a semantic consistency detection module which can dig deeply into user semantic information and monitor discordance and inconsistency within massive tweets, therefore can detect advanced bots that imitates humans.

- We conduct extensive experiments on a comprehensive dataset. The results demonstrate that our model outperforms state-of-the-art methods. Further analysis also bears out the effectiveness of our proposed interactive model and semantic consistency detection model.

2 Related Work

2.1 Twitter-bot Detection

Feature-based Methods. Traditional methods mainly focused on feature engineering and adopted classifiers of machine learning methods. A diversity of features were leveraged to detect bots, including user tweet features (Cresci et al., 2016), user profile features (Yang et al., 2020), and other features extracted from metadata (Miller et al., 2014).

Text-based Methods. With the boom in neural network, bot detection methods based on deep learning sprang up. Wei and Nguyen (2019) adopted recurrent neural network to capture tweet features. Kudugunta and Ferrara (2018) applied LSTM to features in different levels. Stanton and Irissappane (2019) proposed to leverage generative adversarial networks to detect spam bots. Feng et al. (2021b) construct a self-supervised representation learning task by learning on a sequence of user features and conduct bot detection with fine-tuning.

Graph-based Methods. Apart from text-based neural networks, graph neural networks are also utilized to improve the Twitter bot detectors. Ali Alhosseini et al. (2019) used convolutional graph networks for bot detection. Feng et al. (2021d) constructed a heterogeneous graph network for bot detection while Feng et al. (2021a) improved the heterogeneity with additional relations.

In this paper, we build on these works and propose a modality-interactive bot detector which leverage both advantages of two structures. We also propose to detect anomalous behaviour pattern in a user's semantic information.

2.2 Text-Graph Interaction in NLP

Text-graph interaction was widely used in the area of NLP. In the problem of knowledge-guided question answering, it is necessary to leverage both the text modality and knowledge graph modality. Early works simply aggregated the two modalities without interaction (Mihaylov and Frank, 2018). Later works only allowed two modalities to interact in a shallow way. Typically, they put one modality to another modality as an add-on (Feng et al., 2020; Wang et al., 2019; Lin et al., 2019; Yang et al., 2019a; Lv et al., 2020), learned implicit modality information from another one (Bosselut et al., 2019; Petroni et al., 2019; Hwang et al., 2020), or jointly learned information from two modalities with GNN (Yasunaga et al., 2021). Recently, GreaseLM (Zhang et al., 2022) proposed a model to interact two modalities between layers by interactive nodes, in which truly deep interaction was achieved. In this paper, We propose a framework inspired by GreaseLM and conduct bot detection with the help of modality interaction.

3 Problem Definition

In the task of bot detection, we might get multiple information from a user. In this paper, we leverage a user's description and tweets for text modules. $B = \{b_i\}_{i=1}^{L}$ denotes a user's description with L words. $S = \{s_i\}_{i=1}^{T}$ denotes a users tweets with each tweet $s_i = \{w_1^i, \cdots, w_{Q_i}^i\}$ containing Q_i words. For graph modules, user metadata feature sets are taken into account: numerical and categorical features, where $P = \{P^{num}, P^{cat}\}$ denote a user's numerical and categorical user property sets. A user's neighbor set with J neighbors is denoted by $N = \{n_1, \cdots, n_J\}$. We feed these user information B, S, P, N into our model and derive the prediction labels.

4 Methodology

Figure 2 displays an overview of our proposed framework named BIC. BIC consists of M layers, where each layer has two components: (i) modality interact, (ii) semantic consistency detection, while the first component contains three modules : text propagation module, graph propagation module, and text-graph interactive module.

4.1 Modality Interact

4.1.1 Text Propagation Module

For each layer in text propagation module, we feed text representations to a language model which will learn the semantic information and update the interacted information from interactive representations to other representations, *i.e.*,

$$\{\tilde{h}_{int}^{(l)}, \tilde{h}_1^{(l)}, \cdots, \tilde{h}_T^{(l)}\} = \mathrm{LM}(\{h_{int}^{(l-1)}, h_1^{(l-1)}, \cdots, h_T^{(l-1)}\}),$$
$$for\ l = 1, \cdots, M, \tag{1}$$

where LM refers to the language model, $h_{int}^{(l-1)}$ denotes interactive representation of text modality in the (l-1)-th layer, and $\{h_i^{(l-1)}\}_{i=1}^{T}$ denotes other representations of text modality in the (l-1)-th layer. To thoroughly mine the user's content information, we adopt transformer (Vaswani et al., 2017) to serve as text module for each layer.

For the first layer, we firstly use RoBERTa (Liu et al., 2019) pre-trained encoding model to encode user description B and gain a user's initial embedding for description $h_{int}^{(0)}$ which will be used as interactive representation. We then send the user tweets $S = \{s_i\}_{i=1}^{T}$ respectively to RoBERTa encoding model and derive the initial tweets embedding sets $\{h_i^{(0)}\}_{i=1}^{T}$. Finally, the initial representations are fed into the first text propagation module.

4.1.2 Graph Propagation Module

For each layer in graph module, graph representations are firstly fed into a GNN layer to disseminate information between a user and its neighbors, where interacted information is also updated for the neighbors, *i.e.*,

$$\{\hat{g}_{int}^{(l)}, \hat{g}_1^{(l-1)}, \cdots, \hat{g}_J^{(l)}\} = \mathrm{GNN}(\{g_{int}^{(l-1)}, g_1^{(l-1)}, \cdots, g_J^{(l-1)}\}),$$
$$for\ l = 1, \cdots, M, \tag{2}$$

where $g_{int}^{(l-1)}$ denotes a user's interactive representations of graph modality in the (l-1)-th layer, while $\{g_i^{(l-1)}\}_{i=1}^{J}$ denotes neighbor representations in the (l-1)-th layer. Since GCN (Kipf and Welling, 2016) is a widely used model with excellent performance, we use GCN as the graph neural network. A multi-head attention layer is then used for updating its neighborhood information with attention weights for the user, *i.e.*,

$$\{\tilde{g}_{int}^{(l)}, \tilde{g}_1^{(l)}, \cdots, \tilde{g}_J^{(l)}\} = \mathrm{Att}(\{\hat{g}_{int}^{(l)}, \hat{g}_1^{(l)}, \cdots, \hat{g}_J^{(l)}\}), \tag{3}$$

for $l = 1, \cdots, M$, where Att denotes multi-head attention.

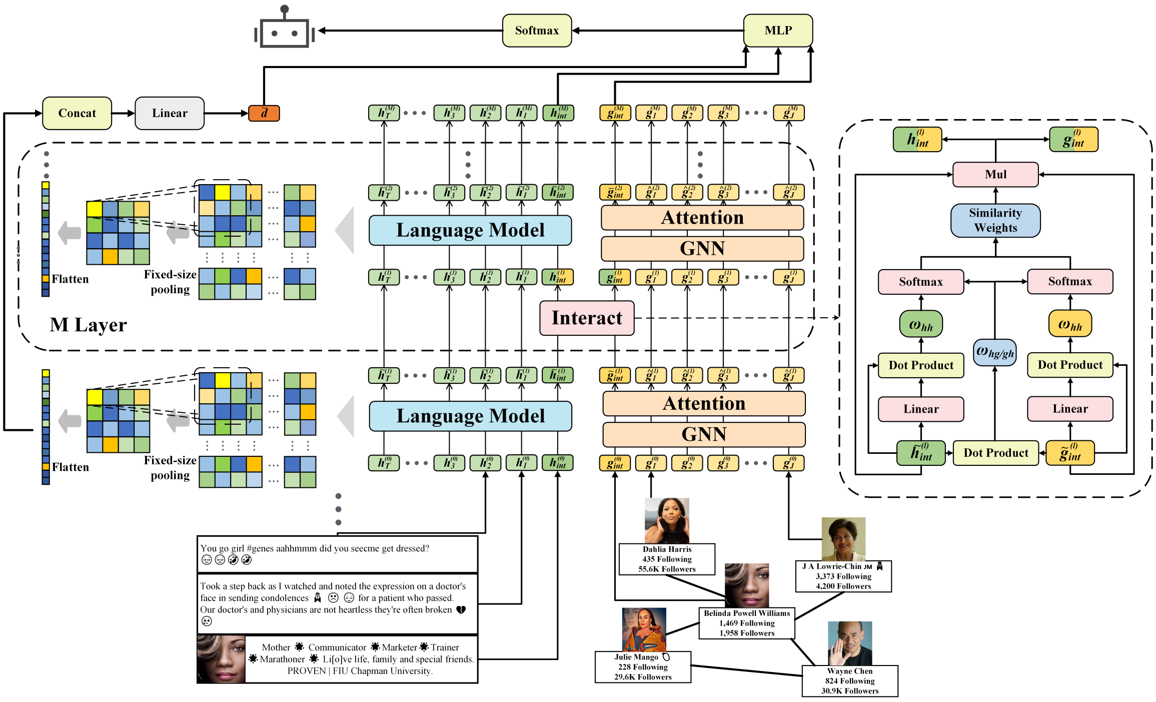

Figure 2: Overview of our proposed framework BIC. In the consistency matrix, brighter colors. *Mul* refers to the operation in Equation (7).

For the first layer, we adopt the same user feature encoding procedures as Feng et al. (2021d). We conduct z-score normalization and obtain representation of a user's numerical properties P^{num} and categorical properties P^{cat}. We aggregate all diversities of user feature representations to generate user initial embeddings $\{g_{int}^{(0)}, g_1^{(0)}, \cdots, g_J^{(0)}\}$ for graph modality, where $g_{int}^{(0)}$ denotes a user's initial feature embedding which will be used as interactive representations, while $\{g_i^{(0)}\}_{i=1}^{J}$ are initial feature embeddings of the user's J neighbors. The initial representations are then fed into the first graph module.

4.1.3 Text-Graph Interactive Module

Text-graph interactive module learns the relative importance of the two modalities and fuses information between two specially selected interactive representations, *i.e.*,

$$(g_{int}^{(l)}, h_{int}^{(l)}) = \mathrm{Int}(\tilde{g}_{int}^{(l)}, \tilde{h}_{int}^{(l)}), \tag{4}$$

where Int refers to interactive function. Specifically, we firstly calculate the similarity coefficient between the two representations and themselves. In this paper, we adopt dot product to serve as similarity function, *i.e.*,

$$\begin{cases} w_{hh} = \tilde{h}_{int}^{(l)} \otimes (\theta_1 \cdot \tilde{h}_{int}^{(l)}), \\ w_{hg} = \tilde{h}_{int}^{(l)} \otimes \tilde{g}_{int}^{(l)}, \\ w_{gg} = \tilde{g}_{int}^{(l)} \otimes (\theta_2 \cdot \tilde{g}_{int}^{(l)}), \\ w_{gh} = \tilde{g}_{int}^{(l)} \otimes \tilde{h}_{int}^{(l)}, \end{cases} \tag{5}$$

where θ_1 and θ_2 are learnable parameters and $\otimes$ denotes dot product. We then apply softmax and derive final similarity weights, *i.e.*,

$$\begin{cases} \tilde{w}_{hh}, \tilde{w}_{hg} = \mathrm{softmax}(w_{hh}, w_{hg}), \\ \tilde{w}_{gg}, \tilde{w}_{gh} = \mathrm{softmax}(w_{gg}, w_{gh}). \end{cases} \tag{6}$$

We then interact the two representations with the help of derived similarity weights, *i.e.*,

$$\begin{cases} h_{int}^{(l)} = \tilde{w}_{hh}\tilde{h}_{int}^{(l)} + \tilde{w}_{hg}\tilde{g}_{int}^{(l)}, \\ g_{int}^{(l)} = \tilde{w}_{gg}\tilde{g}_{int}^{(l)} + \tilde{w}_{gh}\tilde{h}_{int}^{(l)}. \end{cases} \tag{7}$$

By similarity weights, the relative importance of text and graph modalities is learned. After an interactive module layer, the interactive representations $h_{int}^{(l)}$ and $g_{int}^{(l)}$ concatenated respectively with user tweets $\{h_i^{(l)}\}_{i=1}^{T}$ and neighbor nodes $\{g_i^{(l)}\}_{i=1}^{J}$ are fed into the next layer, where $h_i^{(l)} = \tilde{h}_i^{(l)}$ and $g_i^{(l)} = \hat{g}_i^{(l)}$ are not involved in the interactive layer.

The text propagation module and graph propagation module right after the interactive layer allow information received by the interactive representation from one modality to another modality, making interaction between the two modalities deeper.

4.2 Semantic Consistency Detection

Since attention weights from transformer could indicate the correlations and consistency between tweets, we adopt attention weights for semantic consistency detection. From each model layer, we pull out the attention weights from the transformer in language model to construct a matrix $\mathcal{M} \in \mathbb{R}^{(T+1)\times(T+1)}$ which stores the coherence information between tweets of a user. When a user posts tweets with abnormal patterns and inconsistent content, some anomaly will display in the matrix composed of attention weights. To leverage the consistency information in a better way, we firstly implement max-pooling to derive a fixed-size matrix, *i.e.*,

$$\tilde{\mathcal{M}} = \text{Fixed-size-pooling}(\mathcal{M}),\ \tilde{\mathcal{M}} \in \mathbb{R}^{K\times K}, \tag{8}$$

where K is a hyperparameter; Fixed-size-pooling denotes pooling with fixed-size matrix as result. For better usage of neural networks in the following part, we then flatten the matrix to generate consistency vector, *i.e.*,

$$d = \text{Flatten}(\tilde{\mathcal{M}}),\ d \in \mathbb{R}^{K^2\times 1}. \tag{9}$$

For each layer, we have one consistency vector and aggregate them to derive the final consistency vector, denoted by

$$\tilde{d} = \sigma(W_D \cdot \text{Concat}(\{d_i\}_{i=1}^{M}) + b_D),\ \tilde{d} \in \mathbb{R}^{D\times 1},$$

4.3 Training and Inference

We aggregate outputs from text propagation module and graph propagation module of the final layer and consistency vector to derive the final feature representation of a single user by

$$\tilde{d} = W_D \cdot \text{Concat}(\tilde{d}, h^{(M)}, g^{(M)}) + b_D. \tag{10}$$

The final feature representation is later sent into a MLP and softmax layer to derive the prediction score $\hat{y}$, *i.e.*,

$$\hat{y} = \text{softmax}(W_O \cdot \tilde{d} + b_O). \tag{11}$$

We optimize the model end to end using the Cross Entropy Loss between prediction score and the ground truth label, *i.e.*,

$$\text{Loss} = -\textstyle\sum_{i\in \mathrm{Y}}[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)] + \lambda \sum_{\omega\in\theta}\omega^2,$$

where Y denotes all users in the training set, θ denotes all training parameters, y_i denotes the ground-truth label and λ is a regular coefficient. At inference time, we predict the most possible label according to the prediction score.

We present BIC implementation details and hyperparameter settings in Appendix A.

5 Experiments

In the experiment section, we conduct comprehensive experiments with in-depth analysis.

5.1 Experiments Settings

5.1.1 Dataset

In this paper, we conduct our experiments on TwiBot-20 (Feng et al., 2021c), a comprehensive Twitter bot detection benchmark which includes 229,580 Twitter users, 33,488,192 tweets, and 8,723,736 user property items. This dataset almost covers all varieties of bots in social networks and is the only high-quality Twitter bot dataset with graph information, thus our methods can be proved to be applicable to diverse social bots. We follow the same splits provided in the benchmark so that the results are directly comparable with previous works. More datasets Cresci-17 (Cresci et al., 2017), botometer-feedback-19 (Yang et al., 2019b) are also adopted for evaluation in previous works, they do not provide the graph information to support our approach and state-of-the-art baselines.

5.1.2 Baselines

We compare BIC with the following methods:

- **Lee *et al.*** (Lee et al., 2011) adopt random forest classifier with several Twitter user features. e.g. the longevity of the account.
- **Miller *et al.*** (Miller et al., 2014) extract 107 features from a user's tweet and metadata and conduct Twitter bot detection as anomaly detection.
- **Cresci *et al.*** (Cresci et al., 2016) utilize strings to encode the sequence of a user online activity.
- **Botometer** (Davis et al., 2016) is a publicly available service for bot detection that leverages more than one thousand features.
- **SATAR** (Feng et al., 2021b) constructs a self-supervised representation learning task by jointly learning on a range of user features. It then classifies bots with fine-tuning.

Table 1: Bot detection performance on TwiBot-20 benchmark. For each method except for Cresci *et al.* and Botometer which have fixed results, we run 5 times to derive averaged metrics and the corresponding standard deviations.

Method	Text	Graph	Modality-Int	Accuracy	F1-score	Precision	Recall
Lee *et al.*				77.36 (±0.53)	79.98 (±0.50)	76.60 (±0.37)	83.66 (±0.69)
Yang *et al.*				81.64 (±0.46)	84.89 (±0.42)	76.40 (±0.40)	94.91 (±0.69)
Cresci *et al.*				47.76	13.69	7.66	64.47
Miller *et al.*				64.50 (±0.35)	74.81 (±0.26)	60.71 (±0.20)	**97.44** (±0.47)
Botometer				53.09	55.13	55.67	50.82
SATAR	✓			84.02 (±0.85)	86.07 (±0.70)	81.50 (±1.45)	91.22 (±1.82)
Kudugunta *et al.*	✓			59.59 (±0.65)	47.26 (±1.35)	80.40 (±0.60)	33.47 (±1.30)
Wei *et al.*	✓			70.23 (±0.10)	53.61 (±0.10)	62.74 (±0.10)	46.83 (±0.20)
BotRGCN		✓		83.27 (±0.57)	85.26 (±0.38)	81.39 (±1.18)	89.53 (±0.88)
Alhossini *et al.*		✓		59.92 (±0.68)	72.09 (±0.54)	57.83 (±0.49)	95.72 (±2.16)
RGT		✓		86.57 (±0.41)	88.01 (±0.41)	85.15 (±0.28)	91.06 (±0.80)
BIC text-only	✓			77.63 (±0.62)	79.00 (±0.57)	78.01 (±1.94)	80.41 (±1.08)
BIC graph-only		✓		85.46 (±0.85)	87.27 (±0.56)	83.64 (±3.33)	91.51 (±4.03)
BIC	✓	✓	✓	**87.37** (±0.18)	**88.83** (±0.33)	**85.20** (±1.43)	92.84 (±2.26)

- **Kugugunta *et al.*** (Kudugunta and Ferrara, 2018) propose an architecture that jointly leverages a user's tweets and property information.
- **Wei *et al.*** (Wei and Nguyen, 2019) propose a bot detection model with a three-layer BiLSTM to encode tweets.
- **Alhosseini *et al.*** (Ali Alhosseini et al., 2019) utilize graph convolutional network to learn user representations and classify bots.
- **BotRGCN** (Feng et al., 2021d) constructs a framework based on relational graph convolutional network jointly leveraging user tweets and three kinds of metadata.
- **Yang *et al.*** (Yang et al., 2020) adopt random forest with account metadata for bot detection.
- **RGT** (Feng et al., 2021a) leverages relation and influence heterogeneous graph network to conduct bot detection.

5.2 Experiment Results

Overall Model Analysis. To better grasp the difference of each method and show the innovation points of our model, we firstly evaluate each method by modalities which they use. The evaluation details are presented in Appendix B. We then present the performance of each method on the benchmark of Twibot-20. The result in Table 1 demonstrates that:

- BIC consistently outperforms all methods including state-of-art methods RGT (Feng et al., 2021a) with relative 0.9% improvement of accuracy and f1-score on the comprehensive and representative dataset TwiBot-20.
- BIC outperforms state-of-art text modality-based model SATAR (Feng et al., 2021b) by 4.0% and state-of-art graph modality-based model RGT (Feng et al., 2021b) by 0.9% in accuracy, which bears out the effectiveness of both leveraging two modalities.

In Table 1, *Text*, *Graph*, *Modality-Int* respectively denote whether method leverages text modality, graph modality and modality interaction.

Modality Effectiveness Study. To further verify the effectiveness of the aggregation of both text modality and graphic modality, we remove one of them and conduct bot detection with the rest of the model. In Table 1, *BIC text-only* and *BIC graph-only* respectively denote models with only text modality and only graph modality. The results illustrate that both modalities perform worse than our proposed model. To be specific, removing either text modality or graph modality indeed hampers the model's ability to consider all kinds of information, thus declining the overall performance. We also find that model with only text modality performs worse than model with only graph modality by approximately 9%. One reason may be that more information can be learned in the graph modality and graph modality plays a more important role for detecting most of the bots.

Table 2: Performance of model with different interactive strategies.

Strategy	Accuracy	F1-score	Precision	Recall
Ours	**87.37**	**88.83**	85.20	**92.84**
No Interact	85.97	87.42	84.85	90.16
Average	86.64	88.15	84.72	91.87
MLP	86.98	88.44	85.12	92.03
Text	78.53	79.52	82.17	77.03
Graph	86.30	87.65	**85.56**	89.84

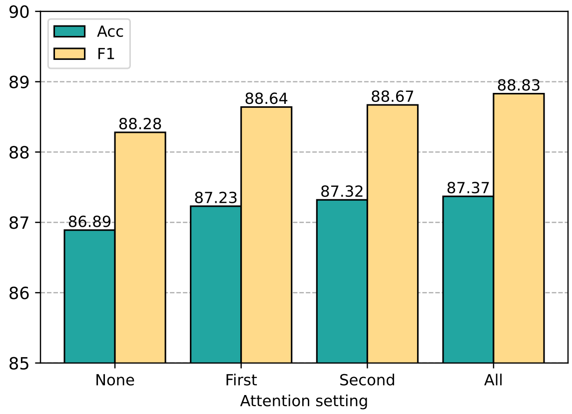

Figure 3: Performance of our model with different settings of considering semantic consistency. *None* means no semantic consistency detection, *First* and *Second* mean adopting consistency detection in the first and the second model layers.

5.3 Text-Graph Interaction Study

In this paper, we propose a similarity-based modality interactive model which has been elaborated in Section 4.1.3. To verify the effectiveness of our similarity-based modality interactive model, we compare it with methods without modality interaction and other possible models with different modality interactive strategies: Average, MLP, Text, and Graph. These modality interactive strategies are presented in details in Appendix C. It is illustrated in Table 2 that:

- Our similarity-based modality interactive strategy outperforms others all, which well confirmed the efficacy of our proposed module, indicating that it can learn the relative importance of modalities.
- Text modality interactive strategy performs worse than other strategies, while it performs better than model with only text modality, which is probably because roughly replacing the graph interactive embedding with text interactive embedding hampers the previously learned neighborhood information but retains the semantic information. In contrast, Graph modality interactive strategy performs better, which indicates the higher importance of the graph modality for detecting most of bots.
- MLP modality interactive strategy performs better than Average modality interactive strategy, which indicates the neural network-based strategy can learn a little emphasis on modalities.

5.4 Semantic Consistency Study

To prove the effectiveness of leveraging semantic consistency, we experiment with three different settings of two-layer model: no semantic consistency detection, semantic consistency detection only in the first model layer, and semantic consistency detection only in the second model layer. The result is shown in the Figure 3. It is illustrated that no consistency detection performs the worst, while consistency detection in the first layer and second layer neck and neck, indicating the inconsistency of bots' tweet content can be detected in both layers. All of them are inferior to the whole model, in which the effectiveness of semantic consistency detection is proved.

To better comprehend the effectiveness of the semantic consistency of users, we select one typical Twitter bot and typical genuine user and visualize their consistency matrix consisting of attention weights of the first and the second layer. From the Figure 4, in the first layer, although there is still slight discordance in human tweet content, the bot's tweets show much greater inconsistency. And in the second layer, except for the first line of the graph, more inconsistency exists in bot. And the anomaly in the first line may be the result of modality interaction.

5.5 Model Layer Study

To find out how many layers of our model have the best performance, We conduct experiments with different model layers. The results in Figure 5 demonstrate that the two-layer model performs the best over other layer settings. When the number of layers increases, the performance decline gradually, which may be caused by higher complexity increasing the training difficulty. Besides, the two-layer model has less time and memory cost, which makes it the best selection. Specifically, it outperforms the second-best three-layer model in accuracy by

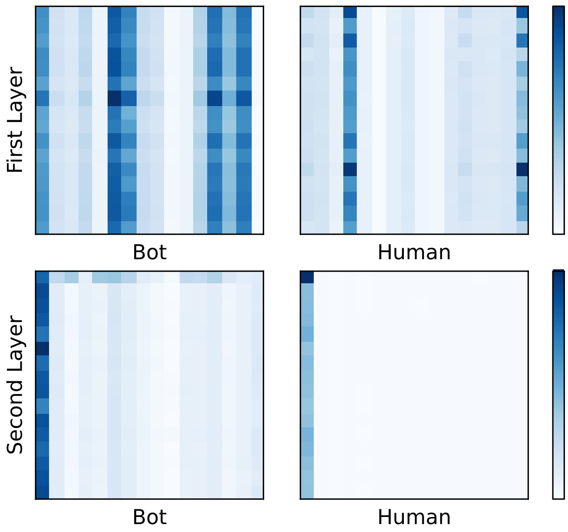

Figure 4: Consistency matrix composed of attention weights from the first and the second layers of typical bot and human.

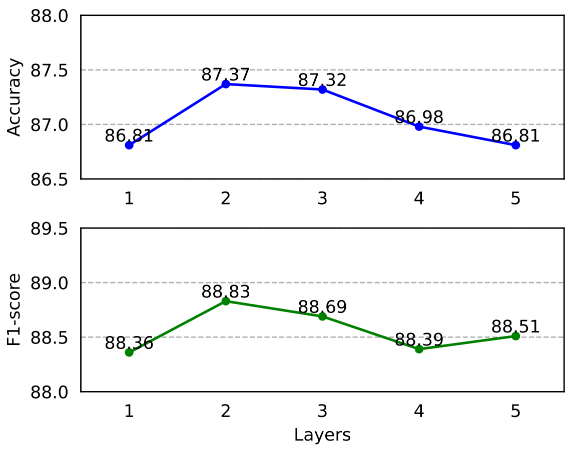

Figure 5: Performance of different model layers.

0.06%, and f1-score by 0.16%, while it costs much less time by 37% and less GPU memory by 11%.

5.6 Case Study

To further analyze how our proposed model leverages information from two modalities to identify bots, we study a specific case from bot sets. We firstly find some tweets and neighbors of the bot with attention weights. The attention weights from multiple layers are averaged to one. We then retrieve similarity weights in Equation (5) to quantitatively analyze it. From the detailed user information displayed in the Figure 6, we discovered that, neighborhood information is learned more, due to more difference in attention weights of the selected bot's bot neighbors and human neighbors

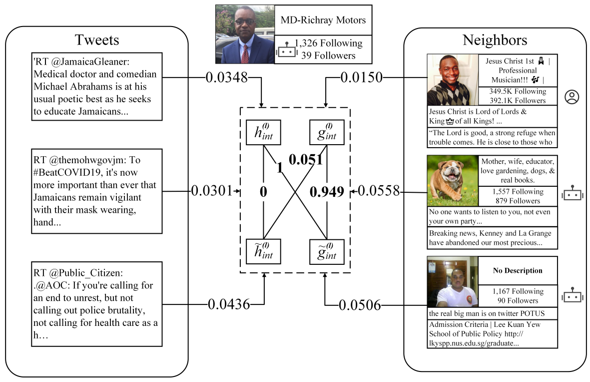

Figure 6: A sample bot with its similarity weights inside the box in the middle. On the left are its tweets with attention weights from transformer in text propagation module, on the right are its neighbors with attention weights from multi-head attention in graph propagation module.

than attention weights of tweets. The conclusion is also reflected in similarity weights. The similarity weights of original interactive representation from text modality are 0 and 0.051, while the similarity weights of original interactive representation from graph modality are 1 and 0.049. The results further display the effectiveness of similarity-based interaction that it indeed learns the emphasis on modalities.

6 Conclusion

Twitter bot detection is a challenging task with increasing importance. To conduct a more comprehensive bot detection, we proposed a bot-detection model named BIC based on both graph and text modalities, which leverages the graph neural network and language model in parallel. BIC also contains a similarity-based interactive module leveraging two interactive representations, which learns the relative importance of two modalities. BIC also adopts attention weights from language model to create consistency vectors for semantic consistency detection. We conducted extensive experiments on a comprehensive benchmark to demonstrate the efficacy of our model in comparison to competitive baselines. Further experiments also bear out the effectiveness of modality interaction and semantic consistency detection. In the future, we plan to explore better interactive approaches to conduct a more comprehensive bot detection.

7 Limitations

In this paper, we proposed a model named BIC make the text modality and graph modality interact and detect the semantic consistency. However, our proposed model has two minor limitations:

- We only leverage semantic and graph modalities. However, other diversities of useful modalities are not taken into consideration, within which user image might greatly promote the ability to detect Twitter bots.
- Since Twibot-20 is the only high-quality Twitter bot dataset with graph information, we conduct experiments only on this benchmark. However, experiments on more datasets might be more pervasive, which might be possible in the future.

References

Seyed Ali Alhosseini, Raad Bin Tareaf, Pejman Najafi, and Christoph Meinel. 2019. Detect me if you can: Spam bot detection using inductive representation learning. In *Companion Proceedings of The 2019 World Wide Web Conference*, pages 148–153.

Jonathon M Berger and Jonathon Morgan. 2015. The isis twitter census: Defining and describing the population of isis supporters on twitter.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction. *arXiv preprint arXiv:1906.05317*.

Stefano Cresci. 2020. A decade of social bot detection. *Communications of the ACM*, 63(10):72–83.

Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. 2016. Dna-inspired online behavioral modeling and its application to spambot detection. *IEEE Intelligent Systems*, 31(5):58–64.

Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. 2017. The paradigm-shift of social spambots: Evidence, theories, and tools for the arms race. In *Proceedings of the 26th international conference on world wide web companion*, pages 963–972.

Eleonora D'Andrea, Pietro Ducange, Beatrice Lazzerini, and Francesco Marcelloni. 2015. Real-time detection of traffic from twitter stream analysis. *IEEE transactions on intelligent transportation systems*, 16(4):2269–2283.

Clayton Allen Davis, Onur Varol, Emilio Ferrara, Alessandro Flammini, and Filippo Menczer. 2016. Botornot: A system to evaluate social bots. In *Proceedings of the 25th international conference companion on world wide web*, pages 273–274.

Shangbin Feng, Zhaoxuan Tan, Rui Li, and Minnan Luo. 2021a. Heterogeneity-aware twitter bot detection with relational graph transformers. *arXiv preprint arXiv:2109.02927*.

Shangbin Feng, Herun Wan, Ningnan Wang, Jundong Li, and Minnan Luo. 2021b. Satar: A self-supervised approach to twitter account representation learning and its application in bot detection. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 3808–3817.

Shangbin Feng, Herun Wan, Ningnan Wang, Jundong Li, and Minnan Luo. 2021c. Twibot-20: A comprehensive twitter bot detection benchmark. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 4485–4494.

Shangbin Feng, Herun Wan, Ningnan Wang, and Minnan Luo. 2021d. Botrgcn: Twitter bot detection with relational graph convolutional networks. In *Proceedings of the 2021 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 236–239.

Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. 2020. Scalable multi-hop relational reasoning for knowledge-aware question answering. *arXiv preprint arXiv:2005.00646*.

Matthias Fey and Jan Eric Lenssen. 2019. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*.

Jena D Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2020. Comet-atomic 2020: On symbolic and neural commonsense knowledge graphs. *arXiv preprint arXiv:2010.05953*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Sneha Kudugunta and Emilio Ferrara. 2018. Deep neural networks for bot detection. *Information Sciences*, 467:312–322.

K Lee, BD Eoff, and J Caverlee. 2011. A long-term study of content polluters on twitter. *ICWSM, seven months with the devils*.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. Kagnet: Knowledge-aware graph networks for commonsense reasoning. *arXiv preprint arXiv:1909.02151*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Shangwen Lv, Daya Guo, Jingjing Xu, Duyu Tang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, and Songlin Hu. 2020. Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8449–8456.

Todor Mihaylov and Anette Frank. 2018. Knowledgeable reader: Enhancing cloze-style reading comprehension with external commonsense knowledge. *arXiv preprint arXiv:1805.07858*.

Zachary Miller, Brian Dickinson, William Deitrick, Wei Hu, and Alex Hai Wang. 2014. Twitter spammer detection using data stream clustering. *Information Sciences*, 260:64–73.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*.

Gray Stanton and Athirai A Irissappane. 2019. Gans for semi-supervised opinion spam detection. *arXiv preprint arXiv:1903.08289*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Xiaoyan Wang, Pavan Kapanipathi, Ryan Musa, Mo Yu, Kartik Talamadupula, Ibrahim Abdelaziz, Maria Chang, Achille Fokoue, Bassem Makni, Nicholas Mattei, et al. 2019. Improving natural language inference using external knowledge in the science questions domain. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7208–7215.

Feng Wei and Uyen Trang Nguyen. 2019. Twitter bot detection using bidirectional long short-term memory neural networks and word embeddings. In *2019 First IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, pages 101–109. IEEE.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

An Yang, Quan Wang, Jing Liu, Kai Liu, Yajuan Lyu, Hua Wu, Qiaoqiao She, and Sujian Li. 2019a. Enhancing pre-trained language representations with rich knowledge for machine reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2346–2357.

Chao Yang, Robert Harkreader, and Guofei Gu. 2013. Empirical evaluation and new design for fighting evolving twitter spammers. *IEEE Transactions on Information Forensics and Security*, 8(8):1280–1293.

Kai-Cheng Yang, Onur Varol, Clayton A Davis, Emilio Ferrara, Alessandro Flammini, and Filippo Menczer. 2019b. Arming the public with artificial intelligence to counter social bots. *Human Behavior and Emerging Technologies*, 1(1):48–61.

Kai-Cheng Yang, Onur Varol, Pik-Mai Hui, and Filippo Menczer. 2020. Scalable and generalizable social bot detection through data selection. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 1096–1103.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. Qagnn: Reasoning with language models and knowledge graphs for question answering. *arXiv preprint arXiv:2104.06378*.

Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D Manning, and Jure Leskovec. 2022. Greaselm: Graph reasoning enhanced language models for question answering. *arXiv preprint arXiv:2201.08860*.

A Implementation Details

We implement our framework with pytorch (Paszke et al., 2019), pytorch geometric (Fey and Lenssen, 2019), and the transformer library from huggingface (Wolf et al., 2019). We limit each user's tweet number to 200, and for those who have posted fewer tweets, we bring their initial embeddings up to full strength with vectors made up of all zeros. Our implementation is publicly available on GitHub[1].

A.1 Hyperparamter Setting

Table 3 presents the hyperparameter settings of BIC. For early stopping, we utilize the package provided by Bjarten[2].

Table 3: Hyperparameter settings of our model.

Hyperparameter	**Value**
model layer count M	2
graph module input size	768
graph module hidden size	768
text module input size	768
text module hidden size	768
epoch	30
early stop epoch	10
batch size	64
dropout	0.5
learning rate	1e-4
L2 regularization	1e-5
lr_scheduler_patience	5
lr_scheduler_step	0.1
Optimizer	RAdamW

A.2 Computation

Our proposed method totally has 4.2M learnable parameters and 0.92 FLOPs[3] with hyperparameters presented in Table 3. Our implementation is trained on an NVIDIA GeForce RTX 3090 GPU with 24GB memory, which takes approximately 0.06 GPU hours for training an epoch.

B Evaluation Details

We elaborate the evaluation of our baselines here. For methods without semantic and graph modalities. Lee et al. (2011) adopt random forest classifier with Twitter bot features. Yang et al. (2020) adopt random forest with minimal account metadata. Miller et al. (2014) extract 107 features from a user's tweet and metadata. Cresci et al. (2016) encode the sequence of a user online activity with strings. Botometer (Davis et al., 2016) leverages more than one thousand features. All of them extract Twitter bot features, without dealing with these features in graph modality or text modality.

For methods with only text modality, SATAR (Feng et al., 2021b) leverages LSTM for its tweet-semantic sub-network. Kudugunta and Ferrara (2018) adopt deep neural networks for tackling user tweets. Wei and Nguyen (2019) propose a model with a three-layer BiLSTM. All of them deal with user information in text modalities.

For methods with only graph modality, BotRGCN (Feng et al., 2021d) utilizes relational graph convolutional network in its proposed framework. Ali Alhosseini et al. (2019) adopt graph convolutional network to learn user representations and classify bots. RGT (Feng et al., 2021a) leverages heterogeneous graph network to conduct bot detection. All of them deal with user information in graph modalities.

C Modality Interactive Strategy

Different modality interactive strategies are itemized here:

- **Average Modality Interactive Strategy** computes the average of two interactive embeddings to derive two new interactive embeddings.

- **MLP Modality Interactive Strategy** concatenates two interactive embeddings and feeds the intermediate into one MLP layer. The result is then split into two new interactive embeddings.

- **Text Modality Interactive Strategy** feeds the interactive embedding from text modality into two different Linear layers to generate new interactive embeddings of both modalities.

- **Graph Modality Interactive Strategy** feeds the interactive embedding from graph modality into two different Linear layers to generate new interactive embeddings of both modalities.

[1] https://anonymous.4open.science/r/BIC-FB63/
[2] https://github.com/Bjarten/early-stopping-pytorch
[3] https://github.com/Lyken17/pytorch-OpCounter