electronics

MDPI

Article

Boosting Social Spam Detection via Attention Mechanisms on Twitter

Hua Shen [1,2,3], Xinyue Liu [2] and Xianchao Zhang [2,*]

[1] Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian 116024, China; huashen.cn@gmail.com
[2] School of Software, Dalian University of Technology, Dalian 116620, China; xyliu@dlut.edu.cn
[3] College of Mathematics and Information Science, Anshan Normal University, Anshan 114007, China
* Correspondence: xczhang@dlut.edu.cn

Abstract: Twitter is one of the largest social networking platforms, which allows users to make friends, read the latest news, share personal ideas, and discuss social issues. The huge popularity of Twitter mean it attracts a lot of online spammers. Traditional spam detection approaches have shown the effectiveness for identifying Twitter spammers by extracting handcrafted features and training machine learning models. However, such models need knowledge from domain experts. Moreover, the behaviors of spammers can change according to the defense strategies of Twitter. These result in the ineffectiveness of the traditional feature-based approaches. Although deep-learning-based approaches have been proposed for detecting Twitter spammers, they all treat each tweet equally, and ignore the differences among them. To solve these issues, in this paper, we propose a new attention-based deep learning model to detect social spammers in Twitter. In particular, we first introduce the state-of-the-art pretraining model BERTweet for learning the representation of each tweet, and then use the proposed novel attention-based mechanism to learn the user representations by distinguishing the differences among tweets posted by each user. Moreover, we take social interactions into consideration and propose that a graph attention network is used to update the learned user representations, to further improve the accuracy of identifying spammers. Experiments on a publicly available, real-world Twitter dataset show the effectiveness of the proposed model, which is able to significantly enhance the performance.

Keywords: attention mechanism; graph attention network; BERTweet pretrained model; social spam detection

Citation: Shen, H.; Liu, X.; Zhang, X. Boosting Social Spam Detection via Attention Mechanisms on Twitter. *Electronics* **2022**, *11*, 1129. https://doi.org/10.3390/electronics11071129

Academic Editors: Andrea Prati, Luis Javier García Villalba and Vincent A. Cicirello

Received: 7 March 2022
Accepted: 31 March 2022
Published: 2 April 2022

1. Introduction

With the increasing popularity of online social platforms, such as Twitter and Facebook, more and more people use them to communicate with families and friends, read the latest news, share personal ideas with others, and discuss social issues. Twitter is one of the largest online social platforms. In the forth quarter report of Twitter in 2021, the number of global monetizable daily active users (mDAU) on Twitter amounted to 217 million users [1]. Due to the huge popularity, it has become a target of online spammers to disseminate fake news, online ads, and even illegitimate content. Thus, it is essential and urgent to automatically detect such illegitimate users and delete their posts quickly.

Towards this goal, many approaches have been proposed in recent decades, especially extracting handcrafted features to train a machine learning model [2–4]. Such approaches suffer from several drawbacks. For example, extracting relevant features usually requires the experiences of domain experts. Moreover, these features are fixed, which cannot work when the spammers change their spamming strategies. To address these issues, deep-learning-based approaches have been recently proposed [5–7], such as using convolutional neural networks (CNN) [7] and recurrent neural networks (RNN) [6]. One of the advantages

of incorporating deep learning techniques is to automatically learn feature representations from raw tweets, instead of extracting handcrafted features.

Recently, researchers also use advanced techniques, such as attention mechanisms working with deep learning models, to detect social spammers or socialbots [8,9]. TextSpamDetector [8] uses an attention mechanism to assign a weight to a word in a sentence to identify whether a tweet is spam or not, i.e., tweet-level spam detection, instead of user-level spam detection. We take Figure 1 as an example. The goal of TextSpamDetector is to assign a binary label to each tweet. However, for user-level spam detection, our goal is to determine whether the user "Designer Imposters" is a spammer or not. Thus, the setting of these two tasks are different.

DeepSBD [9] is the representative user-level spam detection work, which aims to aggregate different types of features via an attention mechanism, including profile, temporal behavior, activity sequence behavior, and content behavior. When learning the content feature, DeepSBD first extracts each tweet representation as a 2D word-embedding matrix using Glove vectors [10], then concatenates all the tweet matrices posted by a user as a 3D matrix, and finally, a CNN is applied through an attention layer to learn content-based, low-level contextual feature representation. Although this approach can improve the prediction performance via using an attention mechanism on low-level features learned by the CNN, it does not distinguish the importance of individual tweets when learning the content feature representation of each user. Next, an example is used to illustrate why we need to take the importance of individual tweets into consideration.

Figure 1. An example of a Twitter spam user. This user posted five tweets, but not all tweets contribute equally when we identify whether it is a spammer or a legitimate user.

Figure 1 shows an example of a Twitter spam user who published five tweets. Among all the five tweets, we can observe that there are three tweets marked in red color having the spamming behaviors, and the remaining two tweets are normal ones. When we infer whether the user "Designer Imposters" is a spammer or a legitimate user, the contribution of three tweets marked in red is much larger than that of the remaining two tweets. This example clearly demonstrates the necessity of learning different importance score for each tweet posted by each user.

In addition, one of the major characteristics of online social platforms is to allow users to interact with others. A common interaction is the following relation. To avoid being detected by platforms, spammers usually mimic the behaviors of legitimate users through following others and posting tweets, which can attract more followers who are legitimate users. Thus, only analyzing following relations among users may lead to the

failure of existing work to detect social spammers. To solve this issue, we follow our previous work [4] and use the interaction graph to capture the social relations among users. Intuitively, legitimate users seldom mention spammers or retweet their posts, but spammers frequently do these. The challenge here is how to use the social interaction graph and users' posts simultaneously for accurately identifying spammers.

To solve this challenge, in this paper, we propose a new spam detection model, called `ASpamDetector`, which uses novel attention mechanisms to automatically learn different weights for different tweets and learn user representations via graph attention network when detecting spammers. Additionally, we introduce a state-of-the-art pretraining model with a large number of Twitter datasets, named BERTweet [11], to learn the feature representation for each tweet. In particular, `ASpamDetector` first learns feature representations for tweets with the pretrained BERTweet. `ASpamDetector` then aggregates all the learned tweet representations to obtain the user representation with the proposed attention mechanism. To incorporate the social interactions among users, the learned user representations are then taken as the inputs of graph attention network [12]. Specifically, we consider the frequency between a pair of users to learn graph attention weights. Finally, the updated user representation is used to predict whether this user is a normal user or a spammer.

In summary, the contributions of this paper are listed as follows:

- To the best of our knowledge, we are the first to distinguish the importance of different tweets when conducting spam detection.
- We propose a new model, named `ASpamDetector`, which incorporates the pretraining model BERTweet. Moreover, we design an attention mechanism to automatically learn the weight for each tweet and propose a frequency-based graph attention network to learn accurate structure-based user representations.
- We conduct experiments on a publicly available and real-world Twitter dataset to show the effectiveness of the proposed model compared with state-of-the-art baselines.

The rest of this paper is structured as follows: Section 2 briefly discusses the related works on spam detection in Twitter. Section 3 gives the details about the proposed model `ASpamDetector`. The experimental evaluation of the method is discussed in Section 4. Section 5 concludes the paper by summarizing the contributions and future work.

2. Related Work

In this section, we briefly summarize the work in social spam detection and divide the work in two categories: feature-based approaches and deep-learning-based approaches.

2.1. Feature-Based Social Spam Detection

Traditional social spam detection approaches mainly use handcrafted features, extracted based on expert knowledge from user tweets and following relations [13]. The features include text-based features, URL/hashtag features, user profile features, and so on, which are used to train a classifier using traditional supervised machine learning approaches, such as decision tree, logistic regression, random forest, and support vector machines. For example, Zheng et al. [14] extracted features from text messages and the social behavior of users and used the SVM algorithm with the labeled dataset to detect spammers. In [2], Zhu et al. proposed a supervised matrix factorization method for spammer detection, which exploited both social activities as well as users' social relations. Hu et al. [3] proposed an optimization formulation that models the social network and content information in a unified framework to perform social spammer detection in microblogging. In [15], Sohrabi et al. designed an spam filtering system to detect and filter malicious content, which exploits some exploration methods and optimization algorithms with the extracted features (i.e., the posts and comments). In our previous work [4], we proposed a generalized classification framework (MVSD) by jointly integrating multiple-view information, (i.e., text, URL, and hashtag) to cope with different and variable strategies used by spammers.

2.2. Deep-Learning-Based Social Spam Detection

Existing deep-learning-based approaches mainly focus on detecting tweet content or tweet-level spam, instead of detecting spammer accounts, and they all directly apply existing models to this task, such as wording embedding [16], long short-term memory (LSTM) networks [17], and text convolutional neural networks (TextCNN) [18]. In [5], the authors proposed to use paragraph vector modeling technique [16] to learn a Doc2Vec for each tweet and then train a classifier based on fully connected layers with the softmax function. To capture the sequential characteristics of tweets, Ban et al. [6] proposed the use of Bi-LSTM (bi-directional long short-term memory), and Alom et al. [7] proposed the use of TextCNN, that is proposed in [18]. TextSpamDetector [8] is a recent work that uses an attention mechanism to detect tweet-level spam. Although TextSpamDetector uses an attention mechanism, it assigns an attention weight to each word within the tweet. In our proposed approach, we use BERTweet, which uses self-attention mechanism to learn word-level attention weights. In addition, we propose the use of an attention mechanism to learn weights to different tweets posted by the user. In fact, our approach can be considered as a hierarchical attention mechanism.

For user-level social spam detection approaches via deep learning, DeepSBD [9] is a representative attention-based mechanism to detect socialbots. This approach aims to aggregate different types of features via an attention mechanism, including profile, temporal behavior, activity sequence behavior, and content behavior. Learning the content behavior is similar to our proposed content-based user representation learning. However, this approach uses a CNN model to extract a 3D matrix as the content feature and then applies an attention mechanism to assign an attention weight to each low-level feature learned by CNN. In our proposed approach, we apply pretrained Twitter language model BERTweet to learn tweet representations instead of Doc2Vec, Bi-LSTM, or CNN. Moreover, we design an attention mechanism to distinguish the importance of different tweets and propose the use of a state-of-the-art graph attention mechanism to learn structure-based user representations on user interaction graph by considering interaction frequency. Thus, our work is significantly different from all the existing work.

3. Methodology

In this section, we first introduce the dataset used in this paper and then describe the overview of the proposed `ASpamDetector`, and the pretraining model BERTweet. Next, we introduce the proposed attention mechanism to learn content-based user representations and present the frequency-based graph attention network to learn structure-based user presentations. Finally, a detector is used to predict the label of each user.

3.1. Dataset

In this paper, we used a real-world Twitter dataset, which is constructed by two datasets, i.e., Twitter social honeypot dataset [19] and the Kwak's dataset [20]. The Twitter social honeypot dataset provides the ground truth label for each user, and the Kwak's dataset contains user tweets. When constructing the dataset, we first identified common users in these two datasets and then extracted their corresponding tweets. Finally, we filtered out non-English tweets. If the number of remaining tweets was larger than 1, then we kept the user; otherwise, we removed this user from the dataset.

After obtaining the Twitter dataset, we then extracted the interaction graph among these users. In particular, if a user, u_i, mentioned or retweeted the tweets of another user, u_j, then there exists a directed interaction link from u_i to u_j. We also counted the frequency of interactions from u_i to u_j, which is used in the proposed frequency-based graph attention network (fGAT) in Section 3.4.

Next, we use an example to demonstrate the procedure of extracting the interaction graph. Table 1 lists four tweets posted by two users. From the first tweet, we can extract a interaction link between *ChipotleTweets* and *CashApp*, and we denote the interaction frequency as 1. We ignore some tweets, such as the second tweet, since there is no interac-

tion. From the third tweet, we can extract another interaction link from the user *NEFF303* to *ChipotleTweets*. This user also retweeted the first tweet, then the interaction frequency between *NEFF303* and *ChipotleTweets* is 2. Figure 2 shows the interaction graph extracted from the example listed in Table 1, and the graph visualization of the whole dataset is shown in Figure 3.

Table 1. Example of extracting interaction graph.

TID	User ID	Tweet Content
1	ChipotleTweets	Guac Mode is back until 3/31 + we are doing surprise $25K $GuacMode drops w/**@CashApp** all week!
2	ChipotleTweets	It's time to switch up that order you've had since 2005
3	NEFF303	Holy moly guacamole this is amazing, pretty much just made my entire week, thank you **@ChipotleTweets**
4	NEFF303	RT **@ChipotleTweets** Guac Mode is back until 3/31 + we are doing surprise $25K $GuacMode drops w/**@CashApp** all week!

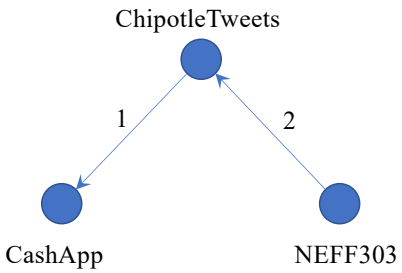

Figure 2. The extracted interaction graph from the data in Table 1. The numbers on the edges denote the interaction frequency between a pair of users.

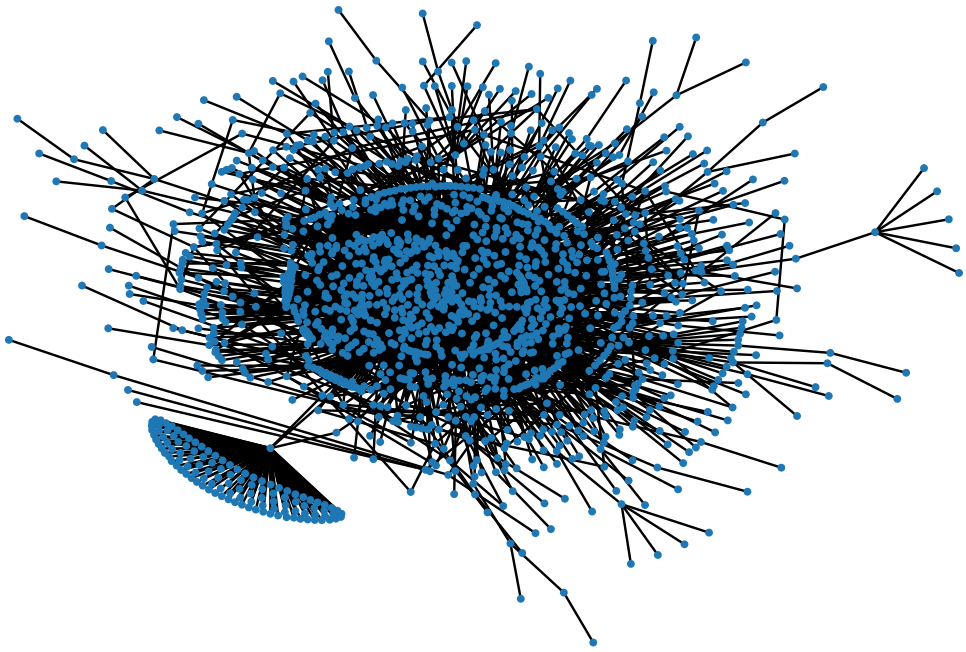

Figure 3. Interaction graph among Twitter users. Each node represents a Twitter account, and each edge represents the interaction between a pair of users. Note that the edge is directed and has a associated frequency-based weight.

3.2. Overview

Figure 4 shows the overview of the proposed ASpamDetector. Each tweet t_n posted by user u_j contains N words. We first learn a tweet representation $\mathbf{t}_n$ using the pretraining model BERTweet, which will be introduced in Section 3.3.1. After obtaining each tweet-level representation $\mathbf{t}_n$ ($n \in [1, N]$), we then generate the **content-based user representation,** $\mathbf{u}_j$, with the proposed attention mechanism, where each tweet will be assigned a learned weight, $\alpha_{j,n}$, according to its importance. The proposed attention mechanism is introduced in Section 3.3.2. Additionally, we propose the use of a graph attention network [12] to model the social interactions and the corresponding interaction frequency to further learn accurate **structure-based user representations,** $\mathbf{u}'_j$, which are introduced in Section 3.4. Finally, a spam **detector** is used to classify $\mathbf{u}'_j$ as a normal user or a spammer based on the learned $\mathbf{u}_j$ and $\mathbf{u}'_j$. Next, we introduce the details of each component.

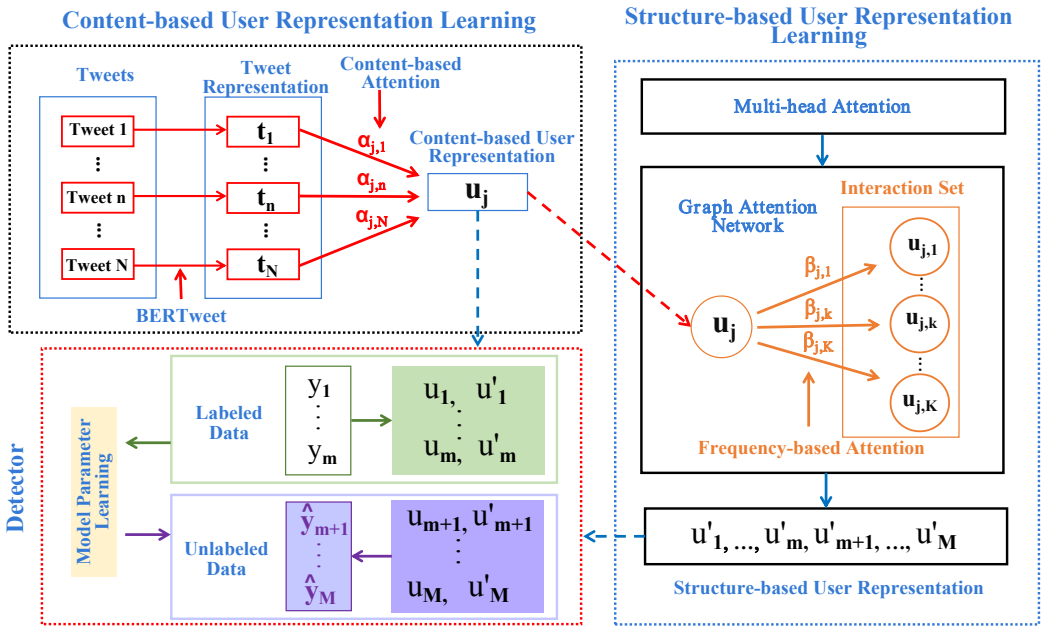

Figure 4. Overview of the proposed ASpamDetector, which contains three key components: content-based user representation learning, structure-based user representation learning, and detector.

3.3. Content-Based User Representation Learning

In this paper, we directly use the raw or original tweets as the input to learn content-based user representation via the pretraining language model using Twitter data and an attention mechanism.

3.3.1. BERTweet

BERTweet [11] is the first public large-scale pretrained language model for English tweets, which has the same architecture as $\text{BERT}_{\text{base}}$ [21] and is trained using the RoBERTa pretraining procedure [22]. Given a tweet, t_n, containing N words, i.e., $t_n = [w_1, w_2, \cdots, w_N]$, we first embed each word, w_i, to a latent vector, $\mathbf{w}_i$, using the pretrained word embeddings such as Word2Vec [23] or GloVe [10]. Then, a positional embedding, $\mathbf{p}_i$, is introduced using wave frequency to capture positional information. The concatenation of $\mathbf{w}_i$ and $\mathbf{p}_i$ is the input of BERTweet. Note that there is a special symbol, $[cls]$, which represents the whole tweet. Thus, the embedding of $[cls]$ outputted by BERTweet can be considered as the representation of the tweet, t_n, i.e., $\mathbf{t}_n \in \mathbb{R}^{d_t}$.

3.3.2. Content-Based Attention Mechanism

As discussed in Section 1, each tweet may contribute differently when we identify spammers. To learn such differences, we propose the use of an attention mechanism to automatically assign a weight, $\alpha_{j,n}$, to each tweet, $\mathbf{t}_n$, as follows:

$$\alpha_{j,n} = \mathbf{w}_a^\top (\tanh(\mathbf{W}_t \mathbf{t}_n + \mathbf{b}_t)) + b_a, \tag{1}$$

where $\mathbf{w}_a \in \mathbb{R}^{d_a}$, $\mathbf{W}_t \in \mathbb{R}^{d_a \times d_t}$, $\mathbf{b}_t \in \mathbb{R}_a^d$, $b_a \in \mathbb{R}$ are learnable parameters. Using Equation (1), `ASpamDetector` can assign a weight to each tweet, and we can obtain a weight vector, $\boldsymbol{\alpha}_j = [\alpha_{j,1}, \cdots, \alpha_{j,n}, \cdots, \alpha_{j,N}]$. We then normalize this vector using the softmax function, as follows:

$$\alpha'_{j,n} = \frac{\exp(\alpha_{j,n})}{\sum_{c=1}^{N} \exp(\alpha_{j,c})}. \tag{2}$$

Based on the normalized attention weights $\boldsymbol{\alpha}'_j = [\alpha'_{j,1}, \cdots, \alpha'_{j,n}, \cdots, \alpha'_{j,N}]$, we can learn the content-based user representation for user u_j, as follows:

$$\mathbf{u}_j = \sum_{n=1}^{N} \alpha'_{j,n} \mathbf{t}_n. \tag{3}$$

where $\mathbf{u}_j \in \mathbb{R}^{d_t}$. Based on the learned user representations, a natural way is to train a classifier to predict the corresponding label. However, such a naive approach ignores the importance of social interactions. Next, we introduce how to use social interactions to update user representations.

3.4. Structure-Based User Representation Learning

Our previous work has shown the effectiveness of incorporating social interactions [4]. However, it uses optimization-based techniques to model the social relations as a regularization term. Differently, in this paper, we propose the use of a more advanced technique to model social interactions, which is graph neural network [24,25]. However, it is challenging to directly apply state-of-the-art graph neural network models, such as graph convolutional networks (GCNs) [26] and graph attention networks [12].

Intuitively, spammers aim to mimic the behaviors of legitimate users, and they will frequently post tweets, mention other users, and retweet posts published by others, especially legitimate and verified users who have a huge number of followers. However, legitimate users seldom mention spammers and retweet their posts. Moreover, spammers collude with their accomplices to construct the criminal communities for hiding themselves, and many legitimate users may follow spammers out of courtesy [27]. In our dataset, about 7% of legitimate users (217 among 3012) follow spammers back. Thus, the social interactions are asymmetric in Twitter, which motives us to consider the direction of interaction relation between a pair of users.

In addition, the frequency of interactions between a pair of users also influences the user representation learning. In general, one user, u_j, mentions an other user, u_k, multiple times, or retweets their tweets frequently, which indicates that user u_j's behaviors are significantly influenced by those of user u_k. Thus, when `ASpamDetector` learns the representation of user u_j, it is reasonable to assign a larger weight to user u_k.

Based on these motivations, we propose a new frequency-based graph attention network (fGAT), which takes interaction frequency into consideration when learning the attention weight between a pair of users who have social interactions. In particular, let $\mathcal{N}_j = \{u_{j,1}, \cdots, u_{j,k}, \cdots, u_{j,K}\}$ represent the set of users that are mentioned by user u_j or whose tweets are retweeted by user u_j, where the size of directly interacted or first-order neighbors is K. Note that $u_j \in \mathcal{N}_j$. We then follow the graph attention network to learn attention weight for each neighbor. However, we also consider the interaction frequency between them. Let $\mathbf{a}_j = [a_{j,1}, \cdots, a_{j,k}, \cdots, a_{j,K}]$ denote the frequency vector,

where $\sum_{k=1}^{K} a_{j,k} = 1$ and $a_{j,k}$ is the relative frequency, by normalizing the natural logarithm of the real frequency.

fGAT first maps the content-based user representations $\mathbf{u}_j$ and $\mathbf{u}_{j,k} \in \mathcal{N}_j$ to latent representations as follows:

$$\mathbf{h}_j = \mathbf{W}_h \mathbf{u}_j, \ \mathbf{h}_{j,k} = \mathbf{W}_h \mathbf{u}_{j,k}, \tag{4}$$

where $\mathbf{W}_h \in \mathbb{R}^{d_h \times d_t}$ is the parameter, and $\mathbf{u}_{j,k}$ is the content-based neighbor user representation learned by Equation (3). To incorporate the social interaction into the proposed model, we then learn the attention coefficient using the structural information by concatenating the two latent representations using Equation (4) as follows:

$$e_{j,k} = \mathbf{w}_e^\top [\mathbf{h}_j || \mathbf{h}_{j,k}], \tag{5}$$

where $\mathbf{w}_e \in \mathbb{R}^{2d_h}$ is the parameter, and $||$ denotes the concatenation operation. After obtaining the attention coefficients, we need to normalize them by applying the LeakyReLU nonlinearity (with negative input slope as 0.2), as follows:

$$e'_{j,k} = \frac{\exp(\text{LeakyReLU}(e_{j,k}))}{\sum_{c \in \mathcal{N}_j} \exp(\text{LeakyReLU}(e_{j,c}))}. \tag{6}$$

The proposed fGAT considers not only the learned attention coefficient but also the interaction frequency to update user representations. Then, we propose to calibrate the learned attention weights using the relative frequency $\mathbf{a}_j$, as follows:

$$\beta_{j,k} = \frac{e'_{j,k} * a_{j,k}}{\sum_{c \in \mathcal{N}_j} e'_{j,c} * a_{j,c}}. \tag{7}$$

Based on the frequency-based attention weights, we can obtain the structure-based user representation, as follows:

$$\mathbf{u}'_j = \sigma(\sum_{k=1}^{K} \beta_{j,k} \mathbf{h}_{j,k}) = \sigma(\sum_{k=1}^{K} \beta_{j,k} \mathbf{W}_h \mathbf{u}_{j,k}), \tag{8}$$

where $\sigma()$ is the nonlinear function, and we use sigmoid in this paper.

Note that, following [12], we also use multi-head attention mechanism [28] to boost the learning of structure-based user representations. Specifically, P-independent attention mechanisms execute the transformation of Equation (8), and then their features are concatenated, resulting in the following output feature representation:

$$\mathbf{u}'_j = ||_{p=1}^{P} \sigma(\sum_{k=1}^{K} \beta_{j,k}^p \mathbf{W}_h^p \mathbf{u}_{j,k}) \tag{9}$$

Since the concatenation operation for the final (prediction) layer is not sensible, we then average all the representations as follows:

$$\mathbf{u}'_j = \sigma(\frac{1}{P} \sum_{p=1}^{P} \sum_{k=1}^{K} \beta_{j,k}^p \mathbf{W}_h^p \mathbf{u}_{j,k}). \tag{10}$$

3.5. Social Spam Detector

The outputs from the proposed fGAT are the structure-based user representations $[\mathbf{u}'_1, \cdots, \mathbf{u}_m, \mathbf{u}_{m+1}, \cdots, \mathbf{u}_M]$, where M is the total number of users, and m denotes the number of labeled users. For each user, we can predict a label distribution, as follows:

$$\hat{\mathbf{y}}_j = \textbf{softmax}(\mathbf{W}_y \sigma(\mathbf{W}_p [\mathbf{W}_o \mathbf{u}'_j || \mathbf{W}_q \mathbf{u}_j] + \mathbf{b}_p) + \mathbf{b}_y), \tag{11}$$

where $\mathbf{W}_y \in \mathbb{R}^{2\times d_y}$, $\mathbf{b}_y \in \mathbb{R}^2$, $\mathbf{W}_p \in \mathbb{R}^{d_y \times d_e}$, $\mathbf{b}_p \in \mathbb{R}^{d_y}$, $\mathbf{W}_o \in \mathbb{R}^{d_e \times d_h}$, and $\mathbf{W}_q \in \mathbb{R}^{d_e \times d_t}$ are parameters. Note that, in the prediction stage, we use both the learned content-based user representation, $\mathbf{u}_j$, and the structure-based user representation, $\mathbf{u}'_j$, by mapping them to the same space first and then concatenating them together. Finally, a softmax function is used upon the nonlinear transformation to learn the label distribution $\hat{\mathbf{y}}_j$.

In the model parameter learning, we only calculate the cross entropy (CE) loss on the training data, as follows:

$$\mathcal{L} = \frac{1}{m}\sum_{j=1}^{m} \mathbf{CE}(\mathbf{y}_j, \hat{\mathbf{y}}_j), \tag{12}$$

where $\mathbf{y}_j$ ($j \in [1, m]$) is the one-hot ground truth vector. After training all the data, we can directly estimate the labels for unlabeled data, $[u_{m+1}, \cdots, u_M]$, based on the learned $[\hat{\mathbf{y}}_{m+1} \cdots, \hat{\mathbf{y}}_M]$.

4. Experiments

In this section, we first describe the experimental settings and then show the experimental results compared with baselines.

4.1. Experiment Settings

We first introduce the dataset that is used in the experiments, then describe state-of-the-art baselines for social spam detection, provide the implementation details of the proposed `ASpamDetector`, and finally present the evaluation metrics.

4.1.1. Data Statistics

In Section 3.1, we introduced how we create the dataset. We finally obtain 7450 users in our dataset and randomly split the dataset into training and testing sets with the ratio 7:3. The statistics of our dataset are shown in Table 2.

Table 2. Statistics of our dataset.

Item	Number
Number of spammers	4438
Number of legitimate users	3012
Number of tweets	15,025
Maximum number of tweets posted by a user	85
Minimum number of tokens in a tweet	3
Maximum number of tokens in a tweet	46

4.1.2. Baselines

The major goal of this work was to automatically identify social spammers by analyzing Twitter data. Towards this goal, we used the following approach as a baseline compared with the proposed `ASpamDetector`. In particular, we divided the baselines into two categories. One comprises the traditional social spam detection approaches, which need to use handcrafted features as the inputs. The other comprises deep-learning-based approaches and takes raw data as the inputs.

For feature-based approaches, we need to extract the feature matrix $\mathbf{X}$ for labeled users and $\mathbf{X}'$ for unlabeled users first from tweets via the tf-idf vectorizer in the scikit-learn package, where the *min_df* parameter is set as 20. Finally, we obtain 4144 features. We use the following four feature-based approaches as baselines:

- **SVM**: We directly train a classifier on the content matrix $\mathbf{X}$ using support vector machines (SVM) and then apply it on the unlabeled data $\mathbf{X}'$ to make predictions.
- **SMFSR** [2]: This approach uses the matrix factorization technique on the concatenated $\mathbf{X}$ and $\mathbf{X}'$, which is guided by the social relationship graph and the label information. Through jointly optimizing matrix factorization and social regularization term, we can train a classifier, which is further used to predict the unlabeled users.

- **SSDM** [3]: SSDM employs a directed Laplacian formulation to model the social network and then integrates the network information into a sparse supervised formulation for the modeling of content information. After obtaining the classifier, SSDM makes prediction on $\mathbf{X}'$.
- **MVSD** [4]: MVSD is our previous work, which uses multiple types of information for detecting spammers. In this experiments, besides of using the content matrix $\mathbf{X}$, we also use URL and hashtag information as the inputs.

Existing deep-learning-based approaches mainly focus on detecting tweet-level spam instead of user-level spam. To make the comparison fair, we modify existing approaches by aggregating each tweet representation via averaging to learn a user representation, which is used to train a classifier and then make predictions. We use the following four approaches as baselines:

- **Doc2Vec** [5]: This approach applies the paragraph vector modelling technique [16] to learn a tweet-level representation. We then averaged all the tweet representations published by the same user to train a user-level classifier.
- **Deep-learnt** [6]: This approach first uses Word2Vec [23] to learn embeddings for words and then applies Bi-LSTM (bi-directional long short term memory [17]) to learn tweet-level representations. After obtaining the representations, we use the same way as Doc2Vec to train the classifier and make predictions.
- **TextCNN** [7]: This approach proposes to convolutional neural network (CNN) to learn the representation of each tweet, following [18]. Then, the averaging tweet representations treated as user representations are used to learn the classifier and predict labels.
- **DeepSBD** [9]: This approach uses the Glove vectors to form a 2D matrix to represent each tweet, and all the tweet posted by a user can be represented by a 3D matrix. A CNN model is applied on this 3D matrix with low-level attention mechanism to learn a user representation, which is used to predict the corresponding label.

4.1.3. Implementation Details

For all the deep-learning-based models, we implement them in PyTorch [29] and train them on an Ubuntu 20.04 with 384 GB memory and an NVIDIA RTX A6000 GPU. The batch size is set to 16 for all the methods. The parameter $d_t = 768$, which is the same as BERTweet. We set $P = 4$ in the experiments, i.e., we use four attention layers to learn structure-based user representations. We use grid search technique to determine the values of other parameters, and finally, we have $d_h = 256$, $d_y = 64$, and $d_e = 128$. The dropout technique [30] is used in the final prediction layer, and the dropout rate is set to 0.5. AdamW [31] is used to optimize the proposed `ASpamDetector`, and the learning rate is set to $1.e - 5$.

4.1.4. Evaluation Metrics

Following existing work [4–6], we use precision, recall, and F1 scores as the evaluation metrics. In particular, we run each baseline and the proposed `ASpamDetector` five times and report the averaging values.

4.2. Performance Evaluation

To thoroughly evaluate the proposed `ASpamDetector`, we aim to answer the following research questions in our experiments:

- **RQ1**: Does the proposed `ASpamDetector` outperform state-of-the-art baselines?
- **RQ2**: Is the proposed content-based attention in the content-based user representation learning component useful for spam detection?
- **RQ3**: In the structure-based user representation learning component, is using the interaction graph helpful for improving the performance? Is considering the frequency of user interactions helpful for improving the performance?

- **RQ4**: Is concatenating the two kinds of user representations essential in the detector component?

4.2.1. Performance Comparison (RQ1)

Table 3 shows the performance of different approaches on the Twitter dataset. We can observe that the proposed `ASpamDetector` achieves the best performance in terms of precision, recall, and F1 score. Among the four feature-based baselines, we can observe that MVSD achieves the best results. The reason is that MVSD utilizes extra information to learn accurate latent user features. Moreover, all the remaining three methods perform worse than deep learning baselines. These results show that deep-learning-based models are more suitable for social spam detection.

Table 3. Performance evaluation on the Twitter dataset. We report the average scores under five runs.

Category	Method	Precision	Recall	F1
Feature-based	SVM	0.644	0.802	0.714
	SSDM	0.684	0.883	0.771
	SMFSR	0.741	0.891	0.809
	MVSD	0.801	0.847	0.823
Deep Learning	Doc2Vec	0.767	0.861	0.811
	Deep-learnt	0.798	0.852	0.824
	TextCNN	0.809	0.832	0.820
	DeepSBD	0.845	0.863	0.854
	`ASpamDetector`	**0.886**	**0.908**	**0.897**

For deep-learning-based approaches, Doc2Vec only uses a naive approach to learn user representations and treats each word in a tweet as independent with others. However, Deep-learnt uses Bi-LSTM and TextCNN uses the convolutional operation to capture the inner relationships among words within each tweet. Thus, they can achieve better performance compared with Doc2Vec. DeepSBD achieves the best performance among all the baselines, which also applies CNN to extract tweet representations and then uses attention techniques to learn better user representations. However, its performance is still worse than that of the proposed `ASpamDetector` in terms of all the metrics. The proposed `ASpamDetector` not only uses an attention mechanism to distinguish the importance of different tweets but also utilizes the frequency-based graph attention network to learn structure-based user representations. These more advanced techniques help the proposed `ASpamDetector` significantly improve the performance.

The experimental results shown in Table 3 can clearly validate the effectiveness of the proposed approach `ASpamDetector` for detecting social spam. Next, we use ablation studies to demonstrate the importance and contributions of attention mechanisms used in `ASpamDetector`.

4.2.2. Content-Based Attention Evaluation (RQ2)

One of the key motivations is that the importance of different tweets contributing to the user representation should be different. To validate this assumption, we conduct the following experiments and use BERTweet$_{avg}$ and `ASpamDetector`$_{con-avg}$ as the baseline.

- BERTweet$_{avg}$: For each tweet, we first learn its representation, $\mathbf{t}_n$, and then average all the tweet representations from the same user as the user representation, i.e., $\alpha'_{j,n} = \frac{1}{N}$ in Equation (3). The learned user representations are directly used for training the classifier.
- `ASpamDetector`$_{con-avg}$: We first learned the content-based user representations using BERTweet$_{avg}$, but the other parts are the same as `ASpamDetector`.

Table 4 shows the performance comparison between BERTweet$_{avg}$, `ASpamDetector`$_{con-avg}$ and `ASpamDetector`. We can observe that the performance

of BERTweet$_{avg}$ is the worst, but it is better than other deep-learning-based approaches, as shown in Table 3, which demonstrates the effectiveness of applying BERTweet for detecting spammers on Twitter. Compared to ASpamDetector, removing the content-based attention mechanism indeed reduces the performance. Thus, the proposed content-based attention mechanism is useful. However, the comparison between ASpamDetector$_{con-avg}$ and ASpamDetector$_{con}$ in Table 5 confirms that using the interaction graph is more helpful compared with distinguishing the importance of tweets.

Table 4. Validating the importance of the proposed content-based attention mechanism.

Method	Precision	Recall	F1
BERTweet$_{avg}$	0.813	0.847	0.830
ASpamDetector$_{con-avg}$	0.854	0.889	0.871
ASpamDetector	**0.886**	**0.908**	**0.897**

In addition, we used a case study to show the learned weights by the proposed ASpamDetector on the data shown in Figure 1. The ground truth label of this user is spammer. There are five tweets, and we allocate an ID to each tweet according to the post date in a ascending order. The first tweet was posted on 23 July 2009, and the fifth tweet was posted on 11 August 2009. Figure 5 shows the learned weight of each tweet. We can see that the weights of the first, second, and forth tweets are greater than those of the third and fifth ones, especially the weights of the second and forth tweets. This case study validates that the proposed content-based attention mechanism can automatically assign accurate weights to the corresponding tweets.

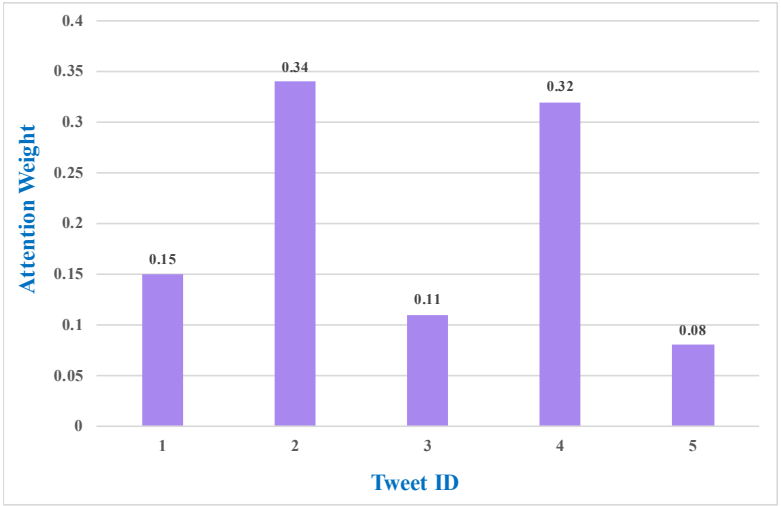

Figure 5. Attention weights learned by the proposed ASpamDetector for the example in Figure 1.

4.2.3. Structure-Based User Representation Learning Evaluation (RQ3)

(1) Interaction graph evaluation: Different from most existing deep-learning-based work, we propose the use of the interaction graph among users to boost the detection performance. To validate the usefulness of this graph, we conduct the following experiment. We first use BERTweet to learn each tweet's representation and then use the content-based attention mechanism to learn user representations, which are directly used as the input of the social spam detector to train the model. This baseline is denoted as ASpamDetector*con*, and Table 5 lists the results. We can observe that discarding the structure-based user representation learning module (Section 3.4), the values of all the metrics drop (precision—4.2%; recall—3.6%; F1 score—3.9%). These results show that introducing the structure information into spam detection models is essential. Additionally, we can observe that ASpamDetector$_{con}$ outperforms DeepSBD, as shown in Table 3, which demonstrates that

using BERTweet and the content-based attention mechanism is better than applying the Glove embeddings and CNN-based attention mechanism for social spam detection task.

Table 5. Validating the importance of using the interaction graph.

Method	Precision	Recall	F1
ASpamDetector$_{con}$	0.849	0.875	0.862
ASpamDetector	**0.886**	**0.908**	**0.897**

(2) Frequency-based graph attention evaluation: Another advantage of the proposed ASpamDetector is to take the interaction frequency into consideration to calibrate the attention coefficients with Equation (6). Now, to evaluate the importance of the interaction frequency factor, we set the relative frequency to $a_{j,k} = \frac{1}{K}$, i.e., all the interaction edges have the same weights. This baseline is denoted as ASpamDetector$_{fre-avg}$. Table 6 lists the results, and we can see that the performance drops, which confirms the effectiveness of the proposed frequency-based attention in graph attention network. Comparing ASpamDetector$_{fre-avg}$ with ASpamDetector$_{con-avg}$, we can find that the content-based attention may be more helpful for boosting the prediction performance since removing the content-based attention makes F1 score drop about 2.9%, but the F1 score drops 2.0% when removing the frequency-based attention from the proposed ASpamDetector.

Table 6. Validating the importance of the proposed frequency-based attention mechanism in the graph attention network.

Method	Precision	Recall	F1
ASpamDetector$_{fre-avg}$	0.865	0.894	0.879
ASpamDetector	**0.886**	**0.908**	**0.897**

4.2.4. Evaluation of User Representation Concatenation in Detector (RQ4)

In the proposed ASpamDetector, we propose that the learned content-based and structure-based user representations are concatenated together to make predictions. To validate this benefit of such a design, we use ASpamDetector$_{str}$ as a baseline, which only uses the learned structure-based user representations. The results are shown in Table 7. We can still observe the performance drop compared with ASpamDetector. However, the performance drop is sightly comparable with ASpamDetector$_{con-avg}$ and ASpamDetector$_{fre-avg}$, which again confirms the reasonableness of the proposed two novel attention mechanisms.

Table 7. Validating the importance of the concatenation of the learned content-based and structure-based user representations.

Method	Precision	Recall	F1
ASpamDetector$_{str}$	0.878	0.903	0.890
ASpamDetector	**0.886**	**0.908**	**0.897**

5. Conclusions

In this paper, we propose that BERTweet is applied to learn tweet representations, and we propose that attention mechanisms are used to distinguish the importance of tweets posted by the same user. Then, the learned tweet representation and the corresponding weights are used to generate the content-based user representations. Additionally, we consider to use the social interactions among users to learn structure-based user representations. Specifically, we propose the use of the frequency of interactions between a pair of users to calibrate the learned attention coefficients by the graph attention network. Finally, the proposed ASpamDetector takes both the learned content-based and structure-based user representation as the inputs to make predictions. We conducted several experiments

on a real-world Twitter dataset, and the experimental results clearly demonstrate the effectiveness of the proposed `ASpamDetector` for the social spam detection task. Our experimental results show that the use of the interaction graph is the most dominant factor in boosting the prediction performance, and the content-based attention is more helpful than the frequency-based attention in this task.

In this work, we did not take URLs in each tweet into consideration, which were proven to be a key factor in detecting social spammers in our previous work [4]. Additionally, when learning the structure-based user representation learning using Equation (5), we ignored the users' types (i.e., spammer or legitimate user) and treated all the neighbors equally. In fact, if user u_j and its neighbor $u_{j,k}$ have the same label, then the attention weight between them may be larger than the neighbors with different labels. In the future, we will consider multiple types of information and design more advanced attention mechanisms and graph learning techniques to further improve the detection performance.

Author Contributions: Conceptualization, H.S., X.L. and X.Z.; methodology, H.S., X.L. and X.Z.; data preprocessing, H.S.; implementation, H.S.; result analysis, H.S., X.L. and X.Z.; writing—original draft preparation, H.S.; writing—review and editing, H.S., X.L. and X.Z.; visualization, H.S.; supervision, X.Z.; project administration, X.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Natural Science Foundation of China (61272374, 61300190), the Key Project of Chinese Ministry of Education (313011), and the Foundation of Department of Education of Liaoning Province (L2015001).

Acknowledgments: We would like to thank all reviewers and editors.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

LSTM	long short-term memory network
TextCNN	text convolutional neural network
Bi-LSTM	bi-directional long short-term memory network
GCN	graph convolutional network
GAT	graph attention network
fGAT	frequency-based graph attention network
SVM	support vector machines
SMFSR	supervised matrix factorization method with social regularization
SSDM	social spammer detection in microblogging
MVSD	multi-view learning for social spammer detection

References

1. Statista. Number of Monetizable Daily Active Twitter Users (mDAU) Worldwide from 1st Quarter 2017 to 4th Quarter 2021. 2022. Available online: https://www.statista.com/statistics/970920/monetizable-daily-active-twitter-users-worldwide/ (accessed on 24 March 2022).
2. Zhu, Y.; Wang, X.; Zhong, E.; Liu, N.; Li, H.; Yang, Q. Discovering spammers in social networks. In Proceedings of the AAAI Conference on Artificial Intelligence, Toronto, ON, Canada, 22–26 July 2012; Volume 26, pp. 171–177.
3. Hu, X.; Tang, J.; Zhang, Y.; Liu, H. Social spammer detection in microblogging. In Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence, Beijing, China, 3–9 August 2013; pp. 2633–2639.
4. Shen, H.; Ma, F.; Zhang, X.; Zong, L.; Liu, X.; Liang, W. Discovering social spammers from multiple views. *Neurocomputing* **2017**, *225*, 49–57. [CrossRef]
5. Wu, T.; Liu, S.; Zhang, J.; Xiang, Y. Twitter spam detection based on deep learning. In Proceedings of the Australasian Computer Science Week Multiconference, Geelong, Australia, 31 January–3 February 2017; pp. 1–8.
6. Ban, X.; Chen, C.; Liu, S.; Wang, Y.; Zhang, J. Deep-learnt features for Twitter spam detection. In Proceedings of the 2018 International Symposium on Security and Privacy in Social Networks and Big Data (SocialSec), Santa Clara, CA, USA, 10–11 December 2018; pp. 208–212.
7. Alom, Z.; Carminati, B.; Ferrari, E. A deep learning model for Twitter spam detection. *Online Soc. Netw. Media* **2020**, *18*, 100079. [CrossRef]

8. Elakkiya, E.; Selvakumar, S.; Leela Velusamy, R. TextSpamDetector: Textual content based deep learning framework for social spam detection using conjoint attention mechanism. *J. Ambient. Intell. Humaniz. Comput.* **2021**, *12*, 9287–9302. [CrossRef]
9. Fazil, M.; Sah, A.K.; Abulaish, M. DeepSBD: A Deep Neural Network Model With Attention Mechanism for SocialBot Detection. *IEEE Trans. Inf. Forensics Secur.* **2021**, *16*, 4211–4223. [CrossRef]
10. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
11. Nguyen, D.Q.; Vu, T.; Nguyen, A.T. BERTweet: A pretrained language model for English Tweets. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Online, 16–20 November 2020; pp. 9–14.
12. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; Bengio, Y. Graph Attention Networks. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
13. Rao, S.; Verma, A.K.; Bhatia, T. A review on social spam detection: Challenges, open issues, and future directions. *Expert Syst. Appl.* **2021**, *186*, 115742. [CrossRef]
14. Zheng, X.; Zeng, Z.; Chen, Z.; Yu, Y.; Rong, C. Detecting spammers on social networks. *Neurocomputing* **2015**, *159*, 27–34. [CrossRef]
15. Sohrabi, M.K.; Karimi, F. A feature selection approach to detect spam in the Facebook social network. *Arab. J. Sci. Eng.* **2018**, *43*, 949–958. [CrossRef]
16. Le, Q.; Mikolov, T. Distributed representations of sentences and documents. In Proceedings of the International Conference on Machine Learning, PMLR, Bejing, China, 22–24 June 2014; pp. 1188–1196.
17. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
18. Kim, Y. Convolutional Neural Networks for Sentence Classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; Association for Computational Linguistics: Doha, Qatar, 2014; pp. 1746–1751. doi: 10.3115/v1/D14-1181. [CrossRef]
19. Lee, K.; Eoff, B.; Caverlee, J. Seven months with the devils: A long-term study of content polluters on twitter. In Proceedings of the International AAAI Conference on Web and Social Media, Barcelona, Spain, 17–21 July 2011; Volume 5, pp. 185–192.
20. Kwak, H.; Lee, C.; Park, H.; Moon, S. What is Twitter, a social network or a news media? In Proceedings of the 19th International Conference on World Wide Web, Raleigh, NC, USA, 26–30 April 2010; pp. 591–600.
21. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, 2–7 June 2019; Volume 1, pp. 4171–4186.
22. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv* **2019**, arXiv:1907.11692.
23. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781.
24. Scarselli, F.; Gori, M.; Tsoi, A.C.; Hagenbuchner, M.; Monfardini, G. The graph neural network model. *IEEE Trans. Neural Netw.* **2008**, *20*, 61–80. [CrossRef] [PubMed]
25. Asif, N.A.; Sarker, Y.; Chakrabortty, R.K.; Ryan, M.J.; Ahamed, M.H.; Saha, D.K.; Badal, F.R.; Das, S.K.; Ali, M.F.; Moyeen, S.I.; et al. Graph neural network: A comprehensive review on non-euclidean space. *IEEE Access.* **2021**, *9*, 60588–60606 . [CrossRef]
26. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.
27. Weng, J.; Lim, E.P.; Jiang, J.; He, Q. Twitterrank: Finding topic-sensitive influential twitterers. In Proceedings of the Third ACM International Conference on Web Search and Data Mining, New York, NY, USA, 4–6 February 2010; pp. 261–270.
28. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 1–11 .
29. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. Pytorch: An imperative style, high-performance deep learning library. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 1–12 .
30. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
31. Loshchilov, I.; Hutter, F. Fixing Weight Decay Regularization in Adam. 2018. Available online: https://openreview.net/forum?id=rk6qdGgCZ (accessed on 4 May 2018).