electronics

MDPI

Article

Cross-Task Rumor Detection: Model Optimization Based on Model Transfer Learning and Graph Convolutional Neural Networks (GCNs)

Wen Jiang [1,2], Facheng Yan [1,2], Kelan Ren [1,2], Xiong Zhang [1,2], Bin Wei [1,2] and Mingshu Zhang [1,2,*]

[1] College of Cryptography Engineering, Engineering University of People's Armed Police, Xi'an 710086, China
[2] Key Laboratory of Network and Information Security, Engineering University of People's Armed Police, Xi'an 710086, China
* Correspondence: zms2099@163.com

Abstract: With the widespread adoption of social media, the rapid dissemination of rumors poses a severe threat to public perception and social stability, emerging as a major challenge confronting society. Hence, the development of efficient and accurate rumor detection models has become an urgent need. Given the challenges of rumor detection tasks, including data scarcity, feature complexity, and difficulties in cross-task knowledge transfer, this paper proposes a BERT–GCN–Transfer Learning model, an integrated rumor detection model that combines BERT (Bidirectional Encoder Representations from Transformers), Graph Convolutional Networks (GCNs), and transfer learning techniques. By harnessing BERT's robust text representation capabilities, the GCN's feature extraction prowess on graph-structured data, and the advantage of transfer learning in cross-task knowledge sharing, the model achieves effective rumor detection on social media platforms. Experimental results indicate that this model achieves accuracies of 0.878 and 0.892 on the Twitter15 and Twitter16 datasets, respectively, significantly enhancing the accuracy of rumor detection compared to baseline models. Moreover, it greatly improves the efficiency of model training.

Keywords: cross-task; BERT; GCN; transfer learning; rumor detection

Citation: Jiang, W.; Yan, F.; Ren, K.; Zhang, X.; Wei, B.; Zhang, M. Cross-Task Rumor Detection: Model Optimization Based on Model Transfer Learning and Graph Convolutional Neural Networks (GCNs). *Electronics* **2024**, *13*, 3757. https://doi.org/10.3390/electronics13183757

Academic Editor: Agnieszka Nowak-Brzezińska

Received: 9 August 2024
Revised: 16 September 2024
Accepted: 20 September 2024
Published: 21 September 2024

1. Introduction

In the information age, social media has emerged as one of the primary channels for information dissemination, significantly facilitating the exchange and sharing of information [1]. However, beneath this convenience lies a hidden risk, most notably the proliferation of false information. False information extends beyond directly misleading content to encompass various forms of fake reviews and rumors, posing severe threats to public perception, social stability, and even individual lives [2].

Fake comments, either as a marketing tactic by businesses or as malicious actions by competitors, often mislead consumers through exaggerations and fabrications, disrupting fair market competition [3]. Rumors, on the other hand, represent unverified or deliberately falsified information that spreads rapidly on the internet, potentially triggering social panic, misdirecting public opinion, or damaging the reputations of individuals and organizations. Although fake comment detection and rumor detection differ in their objectives, both essentially involve assessing the authenticity of information and require the application of natural language processing (NLP) techniques for analysis [4].

In recent years, significant progress has been made in fake comment detection and rumor detection, fueled by rapid advancements in deep learning technologies, particularly in the field of NLP [5]. Nevertheless, rumor detection currently confronts multiple challenges, including (1) difficulty in data annotation: rumor data are often difficult to obtain and annotation costs are high; (2) the complexity and dynamics of rumors: rumors are often accompanied by intricate emotions and socio-psychological factors and their propagation

processes exhibit high degrees of dynamics and uncertainty; (3) domain disparity: rumors in different domains vary significantly in content, form, and communication channels, limiting the generalization ability of models across domains; and (4) subjectivity influence: human subjective judgments can lead to misclassifications of rumors [6]. Most existing models are trained for specific tasks or domains, lacking cross-task transfer capabilities. In practical applications, due to differences in data distributions and feature representations across tasks, directly applying a model trained for one task to another often yields suboptimal results. Therefore, how to achieve cross-task rumor detection, transferring knowledge and experience from fake comment detection to rumor detection tasks, has emerged as an urgent problem to be solved [7].

This paper focuses on cross-task rumor detection, aiming to explore how to leverage transfer learning techniques to migrate useful information from fake comment detection models to rumor detection tasks, thereby enhancing the performance and generalization ability of rumor detection models. Specifically, we will integrate the powerful text representation capabilities of BERT (Bidirectional Encoder Representations from Transformers), the feature extraction capabilities of Graph Convolutional Networks (GCNs) on graph-structured data, and the advantages of transfer learning in cross-task knowledge sharing to construct a cross-task rumor detection model based on BERT–GCN–Transfer Learning. Through this model, we can effectively detect rumors on social media platforms, promptly identifying and containing their spread. The main contributions and innovations of this paper are as follows:

(1) Maximizing model advantages by leveraging the robust text representation capabilities of the BERT module to generate high-quality text feature representations, providing rich semantic information for rumor detection.

(2) Conducting effective structural feature extraction, where the GCN module captures global structural features in rumor propagation networks, offering vital supplementary information for rumor detection.

(3) Demonstrating efficient transfer learning capabilities, with the transfer learning module leveraging existing knowledge and experience to accelerate the model training process and improve model performance.

(4) Conducting experiments on the Twitter15 and Twitter16 datasets, comparing them with multiple baseline models. The results show that BERT–GCN–Transfer Learning achieves promising experimental outcomes.

Section 2 presents a literature review of related work in recent years, introducing the current state of rumor detection, Graph Convolutional Networks, and the application of transfer learning in rumor detection. Section 3 introduces the overall architecture and components of the proposed model. Section 4 details the datasets, experimental setup, and a thorough analysis of the experimental results, along with a comprehensive discussion of the ablation study results. Section 5 concludes the paper based on the research presented herein.

2. Related Work

2.1. Current Status of Rumor Detection Research

Initially, rumor detection research relied on traditional manual annotation methods, which were not only inefficient but also prone to subjectivity and bias. With the continuous development of computer science research in areas such as Machine Learning (ML), Data Mining (DM), and Natural Language Processing (NLP), researchers began to explore rumor detection techniques from diverse angles. Initially, decision trees (Castillo et al. [8]), random forests (Kwon et al. [9]), and Support Vector Machines (SVMs) (Yang et al. [10]) were utilized to classify rumor features for discriminating between true and false information. However, rumors are not strictly dichotomous, as they may contain both true and false elements, prompting researchers to create deep learning models based on message propagation processes for rumor detection. In 2016, Ma et al. [11–13] proposed using Long-Short Term Memory (LSTM) and Recurrent Neural Networks (RNNs) to extract temporal features

from rumors, enhancing detection efficiency to some extent. Nevertheless, these models required extensive training time and had complex internal parameters and computational processes. Furthermore, most studies focused solely on temporal feature learning, neglecting internal topological structures. In 2018, Chen et al. [14] improved upon this approach by integrating RNNs with attention mechanisms, vectorizing rumor-related text content, and training models to detect the textual features of rumors. Yu et al. [15] proposed a Convolutional Neural Network (CNN)-based method in 2017 for rumor detection, effectively extracting rumor features but failing to capture global structural relationships. In 2018, Liu et al. [16] combined an RNN and a CNN for rumor detection, enabling, to some extent, the analysis of user characteristics and the temporal sequence of rumor propagation. In 2020, T-Bian et al. [17], building upon Ma et al.'s propagation tree, employed bidirectional Graph Convolutional Networks (GCNs) for [13] rumor detection, capturing the global features of the propagation structure and achieving promising results. However, this approach was limited to representing rumor propagation as a static graph, ineffective for capturing dynamic rumor information. Khoo et al. [18] modeled the propagation structure as a tree to identify the structural features of message propagation, capturing more long-term dependencies. Wu et al. [19] proposed a rumor detection technique combining attention mechanisms with propagation graph neural networks in 2020, constructing propagation graphs by tracking tweet propagation on Twitter and repeatedly updating node representations through information exchange among neighbor nodes via relationship paths within limited time steps. In 2021, Jiho Choi et al. [20] introduced the Dynamic GCN, a novel graph convolutional network with attention mechanisms, effectively capturing both the structural and temporal information of rumor propagation. In 2022, Ma et al. [21] proposed Knowledge-aware Reasoning with Graph Convolutional Networks (KR-GCNs), integrating user–item interactions and knowledge graphs into heterogeneous graphs for enhanced recommendation performance and diverse explanations, though not directly applied to rumor detection. Looking ahead to 2024, Zhou et al. presented Clip-GCN, a multimodal fake news detection model, achieving an average accuracy of 88.7%, offering solutions to challenges posed by limited labels and multimodal breaking news [22]. Zhang et al. combined Graph Convolutional Networks with BERT and Co-Attention for fake news detection, introducing a novel model called GBCA [23].

2.2. Transfer Learning

Transfer learning is a method that transfers knowledge learned from one domain (source domain) to another different but related domain (target domain) [24]. In cross-domain rumor detection, transfer learning can utilize the abundant labeled data in the source domain to assist the rumor detection task in the target domain, where labeled data are scarce. Compared with traditional methods, transfer learning has obvious advantages. It can leverage existing knowledge to greatly save a large amount of the time and computational resources required for training a model from scratch. Through transfer learning, the model can improve its performance on new tasks by leveraging the feature representation and knowledge learned in the source domain, which enables the model to better solve practical problems and improve generalization ability. In addition, transfer learning can help understand how the model utilizes the knowledge and structure in the source domain during the learning process, which contributes to enhancing the interpretability and credibility of the model. For cross-domain transfer tasks, transfer learning can be applied not only between tasks in different domains but also between tasks within the same domain, allowing us to share knowledge in different scenarios and improve the efficiency and practicality of the model. According to whether the source and target tasks are the same and whether the source and target data are the same, transfer learning is mainly divided into four categories: multi-task learning, model transfer, domain adaptation, and cross-lingual learning. Among them, the goal of model transfer learning is to transfer knowledge learned from one task to another task; for example, training a model to recognize cats and dogs and then transferring this model to the task of recognizing lions and tigers [25].

2.3. Graph Convolutional Neural Network (GCN)

The Graph Convolutional Neural Network (GCN) is a scalable method for semi-supervised learning on graph-structured data, evolving from the Graph Neural Network (GNN) [26]. It enables the use of convolutional operations on graphs, effectively extracting node features and structural information, and provides multi-label classification. It can achieve an average acquisition of feature information from neighboring nodes on a single node and incorporate this mean into a neural network to obtain a more global result. GCNs can extract features from graph data, which can then be used for node classification (NC), graph classification (GC), link prediction (LP), and graph embedding (GE). In rumor detection, GCNs can be used to construct a graph model of rumor propagation and perform rumor detection based on this model.

In 2018, Scarselli et al. [27] first introduced the GCN as a special propagation model for undirected or directed graphs, which is one of the evolutionary forms of recurrent neural networks. Its framework requires repeatedly applying compressive mappings as propagation functions until the node representations reach a stable point. In existing research, GCNs can extract graph structural information and better describe the neighborhoods of nodes. They define multiple graph convolutional layers (GCLs) to iteratively aggregate the features of each node's neighbors and can be represented as a simple message-passing framework. Based on this development, research on rumor detection models has also been deeply inspired by GCNs. Similarly, for Twitter data processing tasks, graph embedding research methods are also essential. Nodes in the graph are represented as a low-dimensional vector space while preserving the network's topological structure and node information, enabling existing machine learning algorithms to be directly applied to subsequent graph analysis tasks and transforming the entire corpus into a heterogeneous graph through modeling [28].

3. Cross-Task Rumor Detection Model: BERT–GCN–Transfer Learning

3.1. Model Construction

The proposed BERT–GCN–Transfer Learning model in this paper primarily consists of five components:

(1) Preprocessing Module for Source Domain Dataset: This module selects the review data collected by Zhang et al. [29] in 2014, which comprises 3,605,300 reviews from 510,071 users on 209,132 businesses. This serves as the initial data input for the model.

(2) BERT Module: This module conducts a deep semantic analysis of the rumor text, extracting high-quality text feature representations. It leverages the powerful language understanding capabilities of BERT to enrich the contextual meaning of the rumors.

(3) GCN Module: This module captures user features, text features, and related propagation features constructed from the source tweets. The GCN is used to model the relationships and interactions between users, tweets, and their propagation patterns, which are crucial for rumor detection.

(4) Model Transfer Module: This module transfers the trained feature extraction layers from the source task to the target task of rumor detection. A fine-tuning strategy is employed to adjust the model parameters to better suit the specific requirements of rumor detection, leveraging the knowledge learned from the source domain.

(5) Prediction Layer Module: This module utilizes a Multi-Layer Perceptron (MLP) and a softmax function to classify the fused features. The cross-entropy loss function is employed to calculate the loss, enabling the model to optimize its predictions through backpropagation.

The overall architecture of the model is illustrated in Figure 1, showcasing the integration of these components and their interactions within the BERT–GCN–Transfer Learning framework.

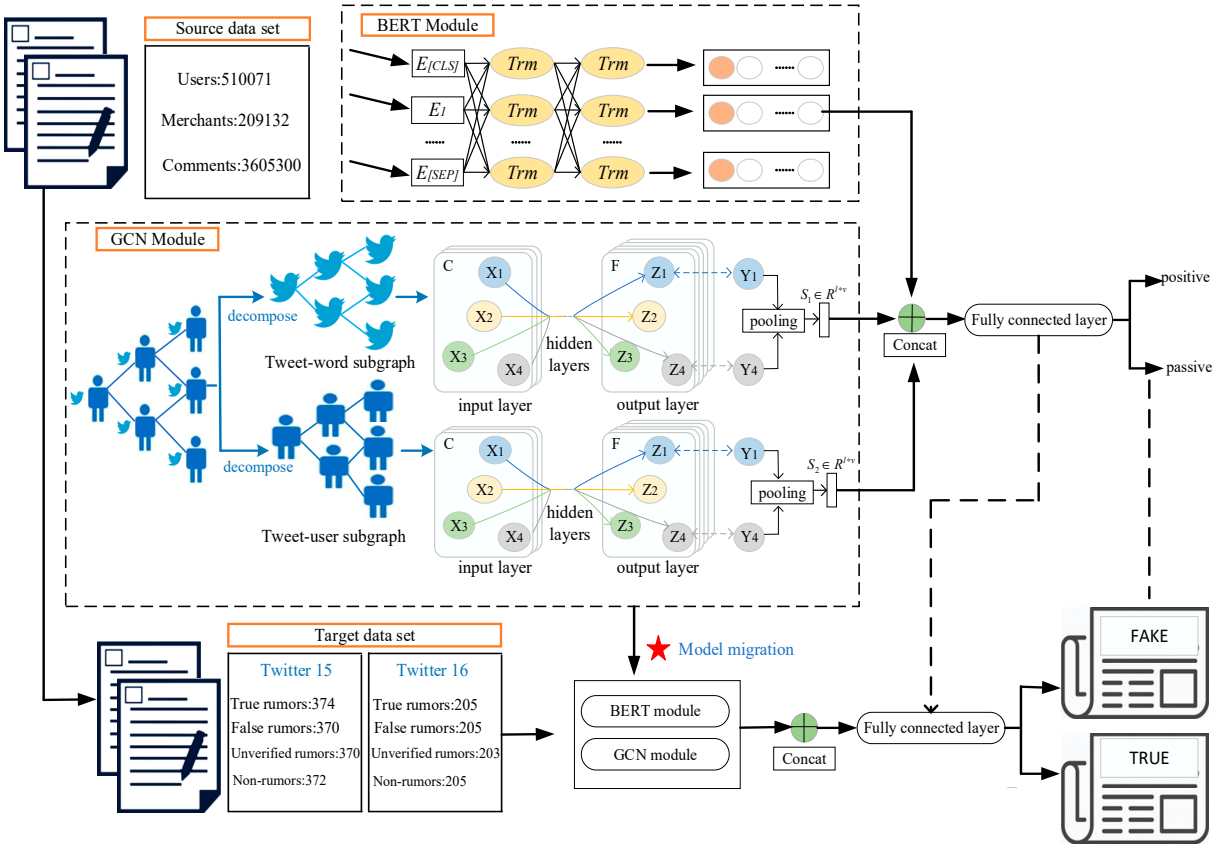

Figure 1. The overall architecture of the model.

As shown in Figure 1, the BERT–GCN–Transfer Learning model implements the complete process from data preprocessing to final rumor detection through these five modules. Firstly, the source domain dataset is prepared by the preprocessing module. Then, text features and relational features are extracted by the BERT module and GCN module, respectively. Then, the knowledge is transferred from the source task to the target task through the model transfer module. Finally, the fusion features are classified in the prediction layer module to realize rumor detection. This process makes full use of BERT's semantic understanding ability, GCN's relational modeling ability, and transfer learning's knowledge transfer ability, aiming to improve the accuracy and efficiency of rumor detection.

3.2. BERT Module

This paper leverages the powerful pre-trained language modeling capabilities of BERT (Bidirectional Encoder Representations from Transformers) to conduct a deep semantic analysis of rumor texts and extract high-quality text feature representations. The bidirectional training mechanism of BERT enables it to consider both the preceding and succeeding context information of the text simultaneously, thereby generating richer and more accurate text representations.

For a given rumor text, after being encoded by BERT, a set of word embeddings can be obtained.

$$W = \left[w_{[CLS]}, w_1, \cdots w_n, w_{[SEP]}\right] \tag{1}$$

where n represents the length of the tokenized sequence, [CLS] indicates the start of a sentence or document, and [SEP] is used to separate different sentences.

The BERT pre-trained model consists of two main parts, the coding layer and the Transformer encoder, as shown in Figure 2.

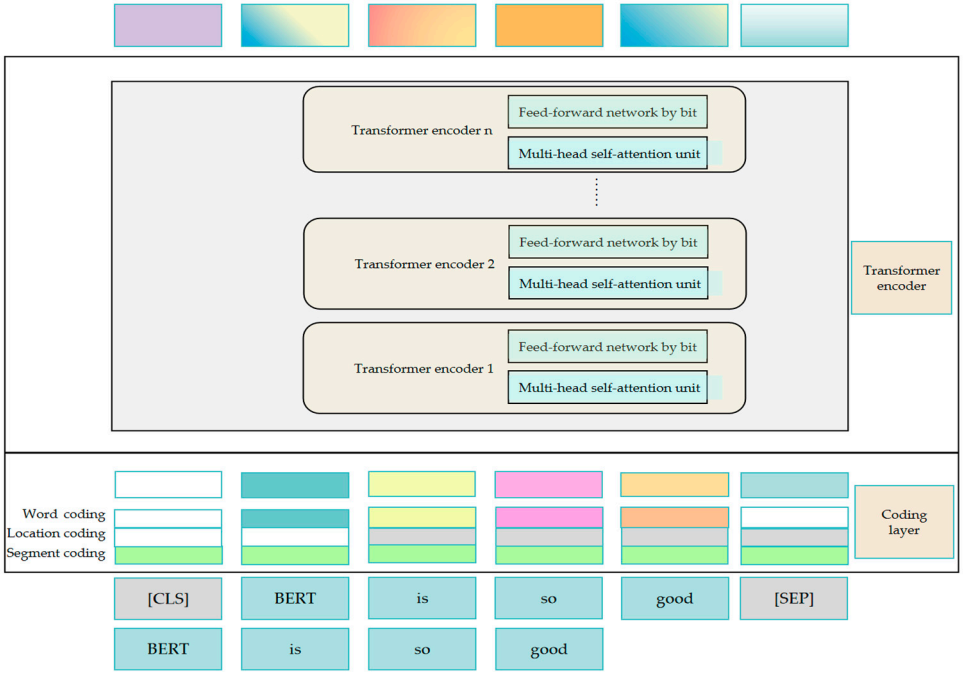

Figure 2. BERT model composition.

3.3. GCN Module

3.3.1. Heterogeneous Graph

A heterogeneous graph is a graph structure composed of different types of nodes and edges, where nodes and edges can be divided into different types according to their properties and relationships. By constructing a tweet–word–user heterogeneous graph, which covers user information, message content, and message comments, a more comprehensive analysis of the related factors of rumor generation can be conducted. Among them, tweet, word, and user, respectively, represent three different nodes, and the attributes of each node are also very different, as shown in Figures 3–5.

Figures 3–5 are related attributes based on tweet, word, and user, respectively. As can be seen from Figure 3, related attributes of a tweet include release time, release location, release content, release user, number of likes, number of retweets, number of favorites, and so on. As can be seen from Figure 4, the relevant attributes of the words include user name, IP address, number of fans, number of followers, number of messages, and so on. As can be seen from Figure 5, the relevant attributes of the user include text, pictures, symbols, emojis, and so on. It is precisely because each node has different attributes that the interaction between the nodes is diversified, thus increasing the complexity of the graph data. The heterogeneous diagram representing user, tweet, and word information is shown in Figure 6.

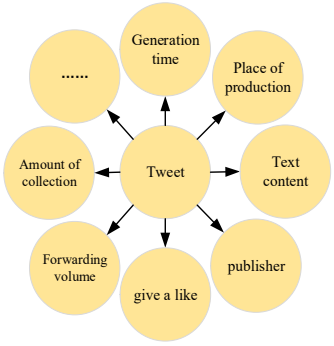

Figure 3. Tweet-related attributes.

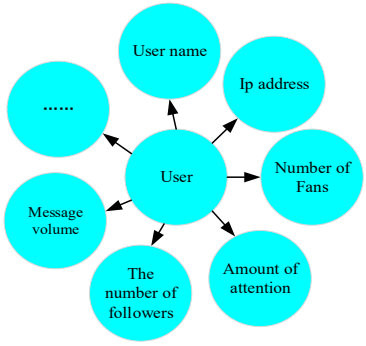

Figure 4. User-related attributes.

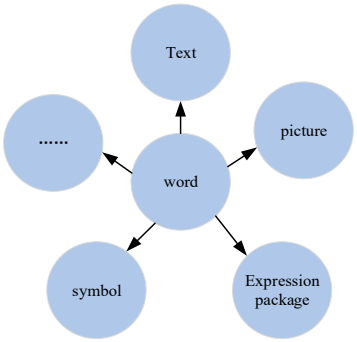

Figure 5. Word-related attributes.

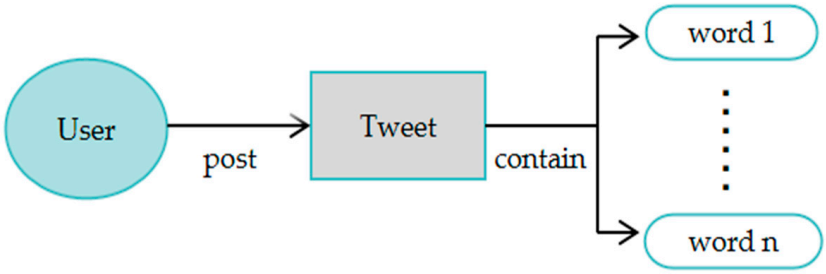

Figure 6. The heterogeneous diagram representing the user, tweet, and words.

3.3.2. Theoretical Basis of Graph Convolutional Neural Networks

Rumor detection is fundamentally approached as a multi-classification task, where the objective is to train classifiers capable of distinguishing the veracity of a source tweet based on the information generated through its continuous forwarding. This section delves into the theoretical underpinnings of leveraging Graph Convolutional Networks (GCNs) for this purpose.

Given a rumor detection dataset consisting of events, each event is characterized by its source tweet and a set of related forwarded posts. The propagation structure of each event is represented as a graph, where nodes represent tweets (both source and forwarded), and directed edges from response tweets to their corresponding forwards depict the propagation path. The adjacency matrix of this graph, initially defined in Equation (2), captures these relationships.

In form, if $C = \{c_1, c_2, c_3, \ldots, c_m\}$ is a rumor detection dataset, which has a total of m events, c_i represents the i-th event. Let $c_i = \left\{r_i, x_1^i, x_2^i, \ldots, x_{n_i-1}^i, G_i\right\}$ and r_i represent the source tweets, and x_j^i represents the j-th related forwarded post. The propagation structure is $G_i = \langle V_i, E_i \rangle$, where nodes $V_i = \left\{r_i, x_1^i, x_2^i, \ldots, x_{n_i-1}^i\right\}$ and $E_i = \{e_{st}^i | s, t = 0, \ldots, n_i - 1\}$

represent a set of directed edges from the response tweet to the corresponding forward. Let $A_i \in \{0,1\}^{n_i \times n_i}$ be the adjacency matrix of the graph, and its initial value is Formula (2):

$$\sigma_{ts}^{i} = \begin{cases} 1, \ if \ e_{st}^{i} \in E_i \\ 0, \ otherwise \end{cases} \tag{2}$$

Furthermore, each event c_i is associated with a ground truth label $y_i \in Y$, where Y represents the fine-grained class, encompassing four labels $\{N, F, T, U\}$: N for non-rumor, F for false rumor, T for true rumor, and U for unverified rumor. When a dataset is given, $f : C \to Y$ is used to predict the label of rumor events based on user features, text features, and relevant propagation features constructed from the source tweet.

For a graph with nodes, each endowed with distinct features, these features are organized into a feature matrix. The relationships between nodes are captured by the adjacency matrix. GCNs employ a layer-wise propagation rule, outlined in Equation (3), that iteratively updates node representations by aggregating information from neighboring nodes. Assuming a set of graph data that contains N nodes with distinct node features, the features of N nodes are organized into an $N \times D$-dimensional matrix, and the relationships between nodes are represented as an $N \times N$-dimensional matrix A. In this context, X represents the input feature matrix, and A is the adjacency matrix. Given X and A as inputs, the GCN (Graph Convolutional Network) employs a layer-wise propagation rule as described in Equation (3):

$$H^{(l+1)} = \sigma(\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}) \tag{3}$$

The meanings of each symbol are shown in Table 1:

Table 1. The meaning of the symbols.

Symbol	Implication
l	Identity matrix
A	Adjacency matrix
$H^{(l)}$	The eigenvector of layer l
$H^{(l+1)}$	The eigenvector of layer $l+1$
$W^{(l)}$	Parameters of the level l convolution
$\widetilde{A}$	Self-contiguous adjacency matrix
$\widetilde{D}$	Metric matrix
σ	Activation function

Node representations are initialized using text features, as specified in Equation (4), which represents the hidden feature matrix of the input layer. For undirected graphs (or directed graphs with reciprocated edges considered), we augment the adjacency matrix with self-connections as shown in Equation (5), effectively adding a self-loop to each node.

$$H(0) = x \, ; H(l) \in R^{N \times D} \tag{4}$$

$$\widetilde{A} = A + l \tag{5}$$

This is equivalent to adding a self-loop to each node. In order to avoid a large value of multi-layer convolutional backvector, it is necessary to normalize the matrix A:

$$A = D^{-\frac{1}{2}}AD^{-\frac{1}{2}} \tag{6}$$

$$A_{ij} = \frac{A_{ij}}{\sqrt{d_i}\sqrt{d_j}} \tag{7}$$

The degree of node v_i is expressed as:

$$\widetilde{D}_{ii} = \sum j\widetilde{A}_{ij} \tag{8}$$

According to the two-layer GCN model set up in this experiment, $softmax$ and $ReLu$ activation functions are adopted, respectively, and the prediction formula is as follows:

$$\hat{A} = \widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}} \tag{9}$$

$$Z = softmax\Big(\hat{A}ReLu\Big(\hat{A}XW^{(0)}W^{(1)}\Big)\Big) \tag{10}$$

The loss function is:

$$L = -\sum_{l\in Y_L}\sum_{f=1}^{F} Y_{lf}\ln Z_{lf} \tag{11}$$

where Y_L is a set of nodes with class objects, and F is the number of classes. This loss function is optimized during training to improve the model's ability to accurately classify rumors.

3.3.3. GCN Experiments

The GCN experiment is mainly divided into three steps:

(1) The first step

According to the factors involved in the process of rumor propagation, the rumor dataset is constructed as a tweet–word–user heterogeneous graph, so that the user node information in the graph corresponds to the node information of published posts. It is decomposed into a propagation subgraph about posts (i.e., tweet–word subgraph) and a propagation subgraph about users (i.e., tweet–user subgraph). In the tweet–word subgraph, the nodes are related posts of Twitter, which contain the text content and comment content involved in the spread of the rumor source's tweets; in the tweet–user subgraph, where the nodes are Twitter users, it contains the characteristics of users, such as the identity characteristics of users, the time that users post, and the mutual attention between users.

(2) The second step

Sets up a two-layer GCN model, which recovers a rich class of convolutional filter functions by stacking convolutional layers, achieving the filtering effect of high-order polynomial frequency response functions to some extent. The number of GCN layers can be appropriately reduced or increased according to the experimental requirements, but considering that deep GCNs have certain residual connections, this paper chooses to use a two-layer GCN for stacking. Its expression is:

$$H_1 = \sigma\big(\hat{A}_i XW_0\big) \tag{12}$$

$$H_2 = \sigma\big(\hat{A}_i XW_1\big) \tag{13}$$

where $H_1 \in R^{n\times v_1}$ and $H_2 \in R^{n\times v_2}$ represent the hidden features of the two layers of the GCN, while $W_0 \in R^{d\times v_1}, W_1 \in R^{v_1\times v_2}$ represent the filter parameter matrices of the two layers of the GCN.

(3) The third step

The node features of tweet–word subgraph and tweet–user subgraph are aggregated by means of mean pooling, and the expression is:

$$S_1 = MEAN\Big(\widetilde{H}_1\Big) \tag{14}$$

$$S_2 = MEAN\Big(\widetilde{H}_2\Big) \tag{15}$$

Then, the user information and post information are combined. Since the figure tweet–word subgraph and the figure tweet–user subgraph have the same dimensional matrix after GCN, the combined expression is as follows:

$$S = concat(s_1, s_2) \tag{16}$$

Finally, with a full connection and a *softmax* layer, the label $\hat{y}$ expression is given as:

$$\hat{y} = softmax(FC(S)) \tag{17}$$

Here $\hat{y} \in R^{l \times c}$ represents the probability vector for all classes that predict the event label. The softmax function is an activation function commonly used in multi-classification problems that can compress a K-dimensional vector with any real number into another K-dimensional real vector, such that every element is in the range (0, 1) and the sum of all elements is 1. Concat function: a Concat function, also known as a concatenation operation, is an operation that concatenates two or more tensors (multidimensional arrays) along a specified dimension.

3.4. Model Transfer Learning Module

3.4.1. Transferability Analysis

Neural networks are composed of layers of hidden layers. The more hidden layers, the more complex the model structure and the more knowledge it can learn [30]. Due to the structural layers of deep neural tissue, it provides flexibility, customization, and transferability. The source dataset serves as the evaluation set, which shares similarities with the rumor set of the target dataset in that both express opinions and carry a certain degree of emotional color. The pre-trained language model BERT can eliminate the need for a large amount of initial training costs and reduce the requirement for the number of labeled datasets in the target task.

3.4.2. Model Fine-Tuning

After fully pre-training the BERT model and GCN model by using rich comment data on source tasks, various rich knowledge such as user information, message content, and message comments of rumor text is obtained, and some parameters of the GCN layer are migrated. Since different layers capture different types of information, differentiated fine-tuning strategies should be adopted to set appropriate fine-tuning learning rates to ensure that the model can effectively utilize the knowledge of the source task and adapt to the characteristics of the target task [31].

The update of parameter θ of the model at layer L at time t is shown in Formula (18):

$$\theta_t^l = \theta_{t-1}^l - \eta^l \cdot \nabla_{\theta^l} J(\theta) \tag{18}$$

where η is the learning rate and $\nabla_{\theta^l} J(\theta)$ is the gradient of the model objective function.

According to the different contributions of different layers to the target rumor detection task, parameter θ is decomposed into $\{\theta^1, \ldots, \theta^L\}$, where θ^L is the parameter of the layer L model and L is the number of layers of the model.

η is decomposed into $\{\eta^1, \ldots, \eta^L\}$, where η^L is the learning rate of layer L.

Since there are significant differences in structure and function between the deep features and the shallow features of neural networks, the shallow features are mainly used to capture low-level information in the data and are suitable for processing simple problems. The deep features can extract more complex and abstract high-level information, which is suitable for the processing of complex problems. Therefore, if the learning rate of the shallowest layer is set to η^l, the learning rate of the deepest layer is:

$$\eta^{l-1} = \eta^l * 1.3 \tag{19}$$

3.5. Prediction Classification Layer

The features extracted by the fine-tuned feature extraction layer are input into the L classification layer, and the cross-entropy loss function is used to calculate the loss to determine whether a rumor text is true or false.

$$\hat{y}_2 = softmax(d) \tag{20}$$

$$L_1 = -\sum_{i}^{N}\left(y_2^i \log \hat{y}_2^i + \left(1 - y_2^i\right) \log(1 - \hat{y}_2^i)\right) \tag{21}$$

where y_2^i is the true value, $\hat{y}_2^i$ is the predicted value, and d is the fine-tuned feature extraction vector.

4. Experimental Design and Result Analysis

4.1. Experimental Settings

4.1.1. Experimental Environment

The experimental environment settings are shown in Table 2:

Table 2. The experimental environment settings.

Operating System	Windows 10
CPU	Intel(R) Core (TM) i5-10750H
GPU	NVIDIA GeForce MX150
Environment configuration	python3.5.2 and python3.6.2

As can be seen from Table 2, the operating system selected for this experiment is Windows 10; the computer's inherent CPU is Intel(R) Core(TM) i5-10750H; the GPU is NVIDIA GeForce MX150; and during data preprocessing, the environment was set to python 3.6.2. In the process of model building, python3.5.2 was selected in the environment configuration to ensure the smooth operation of the experiment, based on the different versions of third-party packages supported in the code and certain dependencies between different libraries.

4.1.2. Parameter Setting

In this paper, a two-layer graph convolutional neural network (GCN) is used for the experiment. In order to prevent the model from overfitting, Dropout is introduced to ensure the randomness in the training process. The parameter of Dropout is set to 0.5, and the learning rate parameter is set to 0.001 to train the model. Stop the code running early when the window size is 10; in other words, stop the training of the model if the verified loss rate after 10 consecutive training iterations does not decrease.

4.2. Experimental Data Set

The source dataset required for this experiment is the review dataset from DianPing.com, and the target datasets are the two publicly available datasets published in 2016 and 2017; namely, the Twitter15 dataset and the Twitter16 dataset. For details, see Tables 3 and 4:

Table 3. Source dataset.

Classification	Statistical Result
user	510,071
merchant	209,132
comment	3,605,300
category	Positive/Negative comment
label	1/0

And the target dataset is shown in Table 4:

Table 4. Target dataset.

Statistical Data	Twitter15	Twitter16
user	2,764,818	173,487
post	331,612	204,820
incident	1490	818
True rumor	374	205
False rumor	370	205
unverified rumor	374	203
non-rumor	372	205

4.3. Data Preprocessing

In real life, the data are usually incomplete, some lack attribute values, some have errors, outliers, and so on, and if the database is too large it will also lead to low data mining efficiency. The use of data preprocessing is to eliminate the above possible problems one by one, which can extract more suitable information from a large number of data and provide technical support for its own research direction. The data preprocessing process usually changes slightly for different tasks and different dataset attributes. Figure 7 shows the following flowchart:

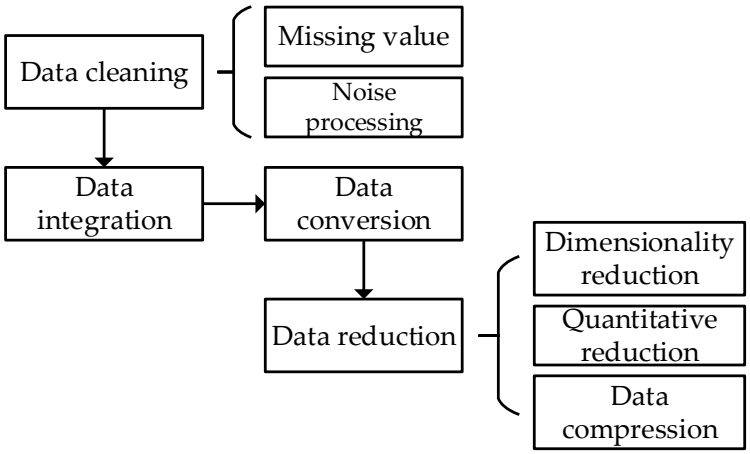

Figure 7. Common flow charts for data preprocessing.

4.3.1. The Process of Data Preprocessing

In this experiment, two public datasets Twitter15 and Twitter16 were processed, respectively, mainly based on message content and comment content in the data of Twitter15 and Twitter16. The specific processing process is as follows:

Step 1: View data.

First, the original dataset is decompressed, the file is decompressed to the data directory, the absolute path of the dataset file is read, the rumor and non-rumor labels are set, and the total number of rumor data and non-rumor data are counted, respectively. By traversing related data of rumors and non-rumors and analyzing rumor data and non-rumor data, the total amount of rumor data and non-rumor data can be obtained.

Step 2: Build the dictionary and list.

When all the data are read out, the data are generated into a tuple and then converted into a dictionary form, so that Chinese characters and numbers correspond one by one. Then add an unknown character, save these dictionaries to the local disk, and set the saving path of the data dictionary. At this time, the data dictionary is generated, and the length of the dictionary can be obtained. Before generating the data, the files of the training set and the test set can be cleared first, the serialized representation data can be created, the

training dataset and the verification dataset can be divided in a certain proportion, the saving path of the data list can be set, and then the data list can be generated.

Step 3: Text processing.

View the data in the rumor dataset, use the jieba third-party package to segment the text data, and define an empty list to store the data after word segmentation. Use the string third-party package to de-punctuate text, remove stops in text, and so on. In the original dataset, the redundant parts of the data are removed to better extract the key features of the information.

Step 4: Make a word cloud map.

Use the matplotlib drawing tool and word cloud third-party package to draw the processed data and more directly show the effect of data preprocessing. In this experiment, the basic parameter settings are shown in Figure 8:

```
from wordcloud import WordCloud
import matplotlib
import matplotlib.pyplot as plt
matplotlib.rcParams['figure.figsize'] = (10.0, 10.0)
#%%
word_counts_top100 = words_count.most_common(100)
word_counts_top100 = dict(word_counts_top100)
wordcloud=WordCloud(font_path="./data/simhei.ttf",background_color="white",
                    max_font_size=70,prefer_horizontal=0.5,max_words=100,
                   height=200,width=200,margin=1).generate_from_frequencies(word_counts_top100)
wordcloud.to_image()
plt.imshow(wordcloud)
plt.axis('off')
plt.show()
```

Figure 8. The basic parameter setting of word cloud map.

4.3.2. Results of Data Preprocessing

The visualization of the Twitter dataset is shown in Figures 9 and 10.

Figures 9 and 10 are visualizations of a Twitter dataset preprocessed based on message content. Figure 9 is the bar graph generated by the data preprocessing based on message content in Twitter data, and Figure 10 is the word cloud graph generated by the data preprocessing based on message content in Twitter data. Figure 9 shows the number of message rumor features extracted as the data of message content in the dataset continues to increase, showing the length of the rumor viewed. The histogram is a visual expression used to observe the frequency distribution pattern of data, the horizontal coordinate represents the isometric segment with a value of 10, the vertical coordinate represents the frequency statistics in each isometric segment, and the continuous graph represents the density distribution of data about rumor words. The word cloud map generated in Figure 10 is based on the results of message content processing in the Twitter data, and it shows some high-frequency words that appear after message text processing. In addition, the size of the font in the figure indicates different frequency of occurrence, and the larger the font, the higher the frequency of relative words. It can be seen that words such as soldiers, reports, found, Afghan, and missing appear frequently in the data of the message content.

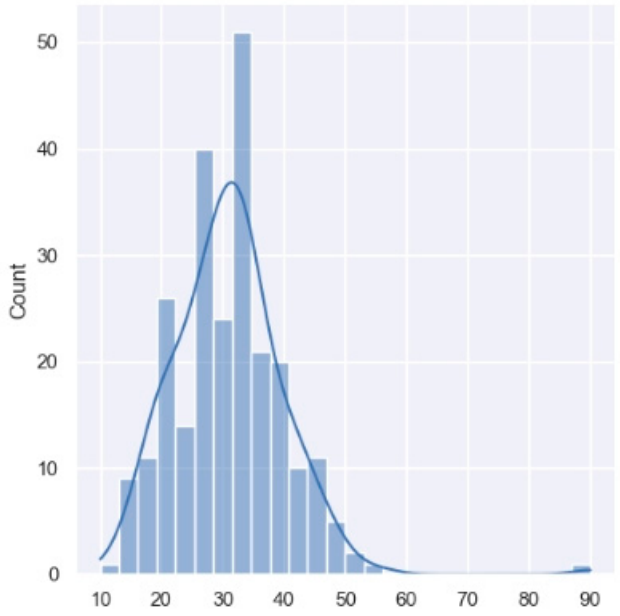

Figure 9. Bar chart.

Figure 10. Word cloud map.

Figures 11 and 12 are visual images of the preprocessed Twitter dataset based on message comments. Figure 11 is a histogram generated by preprocessed Twitter data based on message comments. Figure 12 shows the word cloud map generated by the data preprocessing based on message comments in the Twitter data. In Figure 11, the horizontal coordinate of the histogram represents the isometric segment with a value of 200, and the vertical coordinate represents the frequency statistics of each isometric segment. The continuous graph shows the density distribution of data about rumor words. The word cloud map generated in Figure 12 is the result of processing comments in Twitter data, and it shows some high-frequency words that appear in the process of comment text processing. It can be seen that Breaking, 7News, News, WCVB, Reports, and other words appear very frequently in the data of message comments.

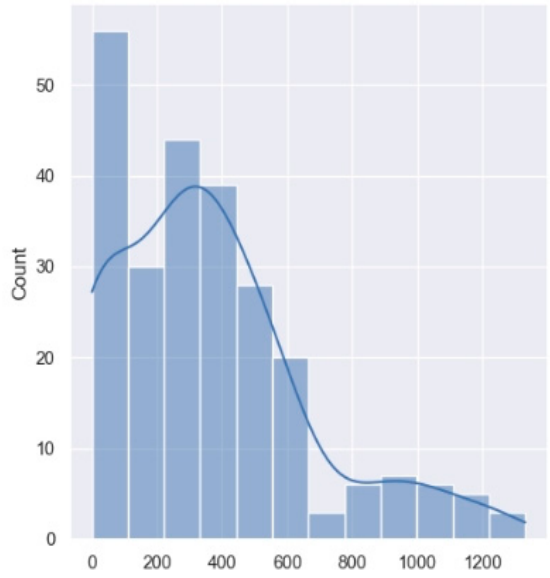

Figure 11. Bar chart 2.

Figure 12. Word cloud map 2.

4.4. Experimental Results and Analysis

4.4.1. Evaluation Indicators

The evaluation index [32] is an important reference for verifying the performance of a model. In this paper, the evaluation index commonly used in classification algorithms, namely accuracy and F1 score, is used as an evaluation index to test the effect of the model in this paper. Firstly, a confusion matrix is introduced, which is a standard format for accuracy evaluation. Its basic meaning is as follows: (1) True Positive (TP): the positive example is predicted to be positive; False Positive (FP): The negative example is predicted to be positive; True Negative (TN): predicts the negative example as a negative example; False Negative (FN): indicates that the positive example is predicted as a negative example. The details are as follows:

(1) Accuracy represents the proportion of the number of samples with correct classification in all samples. The calculation formula is:

$$Accuracy = \frac{TP + FN}{TP + FP + TN + FN} \tag{22}$$

(2) Precision represents the proportion of the number of correctly classified positive samples to the number of all positive samples in the classifier. The calculation formula is as follows:

$$Precision = \frac{TP}{TP + FP} \tag{23}$$

(3) Recall represents the proportion of the number of correctly classified positive samples to the number of positive samples. The formula is as follows:

$$Recall = \frac{TP}{TP + FN} \tag{24}$$

(4) The F1 score represents the weighted harmonic average of the accuracy rate and recall rate, where P represents the accuracy rate and R represents the recall rate. The calculation formula is:

$$F_1 = \frac{2 * P * R}{P + R} \tag{25}$$

4.4.2. Baseline Model

(1) PPC_RNN+CNN [33]: a rumor detection model combining an RNN and a CNN, which learns the representation of rumors by the characteristics of users in the rumor propagation path.

(2) Bi-GCN: a model based on graph convolutional networks for capturing propagation patterns, with message-passing architecture.

(3) DYNGCN: a model based on graph convolutional networks with an attention mechanism to capture graph snapshot time dynamics.

(4) GraMuFeN [34]: a model utilizing the function of graph convolutional neural networks, with GCN and LSTM networks as the core of the text encoder and Resnet-152 as the image encoder.

(5) MFAN [35]: a model that integrates text, visual, and social graph features into a single framework and considers the complementarity and alignment between the different modes. The model achieves better multimodal fusion by introducing self-supervised losses to align representations in different views.

(6) TDEDA [36]: a multimodal fusion neural network that performs a high-level information interaction at the text–image object level and captures visual features associated with keywords using an attention mechanism.

(7) CAFE [37]: CAFE analyzes cross-modal ambiguity learning from the perspective of information theory, adaptively aggregates single-modal and cross-modal correlation features, and performs rumor detection.

4.4.3. Experimental Results

(1) Intuitive representation of the model

From Figure 13 and Table 5, it can be observed that the BERT–GCN–Transfer Learning model exhibits outstanding performance on both the Twitter15 and Twitter16 datasets, with an even more remarkable performance on Twitter16. In terms of accuracy, the proposed model achieves exceptionally high accuracy rates of 0.878 (Twitter15) and 0.892 (Twitter16), respectively, indicating its exceptional ability to classify inputs into the correct categories. The improvement in accuracy by 0.014 from Twitter15 to Twitter16 might suggest that the model has learned more effective feature representations on Twitter16.

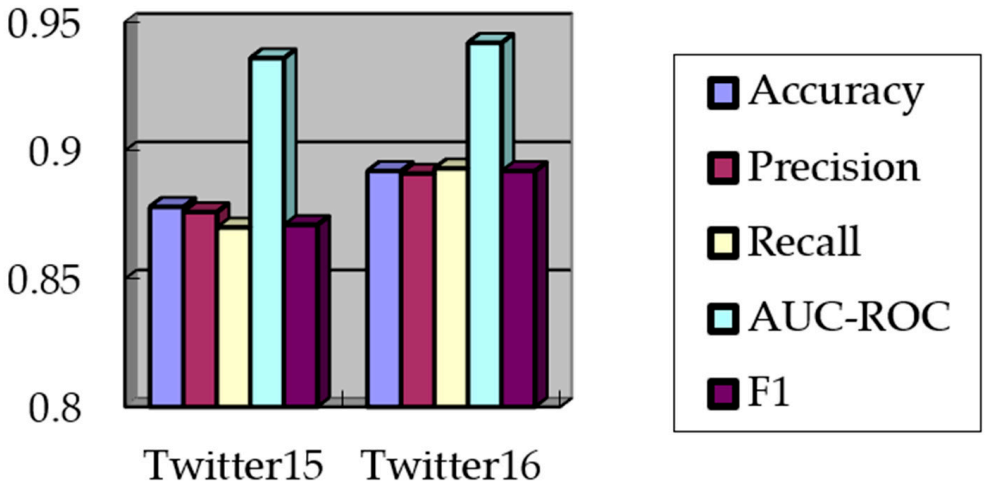

Figure 13. Performance diagram of the model in this paper.

Table 5. Data comparison.

Index	Twitter15	Twitter16
Accuracy	0.878	0.892
Precision	0.876	0.891
Recall	0.870	0.893
AUC-ROC	0.936	0.942
F1	0.871	0.892

Regarding precision and recall, both metrics remain close and at a high level across the two datasets. Slightly higher precision and recall on Twitter16 compared to Twitter15 suggest that the model is better at identifying true positives and reducing false judgments on Twitter16. The F1 score, which is the harmonic mean of precision and recall, serves as a comprehensive evaluation of model performance. With F1 scores close to 0.89 on both datasets, the model achieves a good balance between precision and recall. The slightly higher F1 score on Twitter16 further validates its superior performance on this dataset.

Analyzing the AUC-ROC (Area Under the ROC Curve), an essential metric for assessing classification models that is independent of classification thresholds reveals AUC-ROC values exceeding 0.93 on both datasets, demonstrating the model's high discriminative power. The marginally higher AUC-ROC on Twitter16 suggests that the model exhibits stronger capabilities in distinguishing positive and negative samples on this dataset.

Differences in text length, vocabulary distribution, topic types, or sentiment orientations between Twitter15 and Twitter16 may influence the model's learning and understanding of features.

Overall, this underscores that the BERT–GCN–Transfer Learning model, integrating BERT's robust text representation capabilities with GCN's (Graph Convolutional Network) advantages in graph-structured data, achieves enhanced performance through transfer learning and fine-tuning on the target dataset.

(2) Comparison with baseline models

Comparing the results from Tables 6 and 7, the proposed BERT–GCN–Transfer Learning model excels in the rumor detection task, not only leading in accuracy but also achieving high scores in multiple key metrics including NR, FR, TR, and UR. This could be attributed to the model's superior capability in feature extraction and context understanding, particularly through the combination of BERT's pre-training abilities and the structural information processing capabilities of Graph Convolutional Networks (GCNs).

Table 6. Comparison between the proposed model and the baseline model on the Twitter15 dataset (non-rumor (NR), false rumor (FR), true rumor (TR), unsubstantiated rumor (UR)).

Method	Acc	NR	FR	TR	UR
		F1	F1	F1	F1
PPC_RNN+CNN	0.842	0.818	0.875	0.811	0.790
Bi-GCN	0.814	0.768	0.811	0.793	**0.872**
DYNGCN	0.827	0.746	0.769	0.837	0.820
GraMuFeN	0.890	0.823	0.819	0.835	0.858
MFAN	0.874	0.839	0.824	0.809	0.866
TDEDA	0.872	0.842	0.856	0.838	0.870
CAFE	0.806	0.803	0.799	0.807	0.813
BERT–GCN–Transfer Learning	**0.892**	**0.857**	**0.880**	**0.842**	0.871

Table 7. Comparison between the proposed model and the baseline model on the Twitter16 dataset (non-rumor (NR), false rumor (FR), true rumor (TR), unsubstantiated rumor (UR)).

Method	Acc	NR	FR	TR	UR
		F1	F1	F1	F1
PPC_RNN+CNN	0.863	0.824	0.883	0.826	0.810
Bi-GCN	0.804	**0.895**	0.787	0.718	0.799
DYNGCN	0.836	0.741	0.804	0.880	0.853
GraMuFeN	0.857	0.798	0.872	0.832	**0.856**
MFAN	0.838	0.856	0.861	0.882	0.849
TDEDA	0.824	0.846	0.813	0.834	0.807
CAFE	0.840	0.855	0.830	0.842	0.837
BERT–GCN–Transfer Learning	**0.878**	0.869	**0.899**	**0.884**	0.852

Accuracy, which measures the proportion of correctly classified samples among all samples, reached 0.892 for BERT–GCN–Transfer Learning on the Twitter 15 dataset. Its F1 scores for NR, FR, and TR were 0.857, 0.880, and 0.842, respectively, outperforming other baseline models. However, for UR, Bi-GCN surpassed the proposed model with an F1 score of 0.872 compared to the model's 0.852. On the Twitter16 dataset, BERT–GCN–Transfer Learning exhibited outstanding performance across all metrics, achieving an accuracy of 0.878, which is 1.7% higher than the second-best accuracy of 0.863. Notably, it achieved an F1 score of 0.899 for FR, surpassing the second-best F1 score of 0.883 by 1.8%. FR refers to the ability to identify content falsely labeled as rumors, and BERT–GCN–Transfer Learning's best-in-class F1 score of 0.899 demonstrates its prowess in recognizing and avoiding the misclassification of non-rumor content as rumors.

Overall, the BERT–GCN–Transfer Learning model has achieved remarkable results in the rumor detection task, offering new insights and methodologies for research in this field. Figure 14 illustrates the performance comparison of the BERT–GCN–Transfer Learning model with other baseline models on the Twitter15 dataset, while Figure 15 shows the same comparison on the Twitter16 dataset.

4.5. Ablation Study

To investigate the contributions of each component in our model, this section conducts an ablation study to evaluate the impact of different components on the cross-task rumor detection algorithm.

The specific setup is as follows: the BERT module, GCN module, and model transfer module are removed individually to observe the changes in model performance after removing these modules.

Experiment ①: Remove the BERT module and only retain the GCN module and model transfer module for rumor detection in the experiment.

Experiment ②: Remove the GCN module and only retain the BERT module and model transfer module for rumor detection in the experiment.

Experiment ③: This description contains an error. It should be "Remove the model transfer module, and only retain the BERT module and GCN module for rumor detection in the experiment".

Experiment ④: The proposed BERT–GCN–Transfer Learning model, which includes the BERT module, GCN module, and model transfer module.

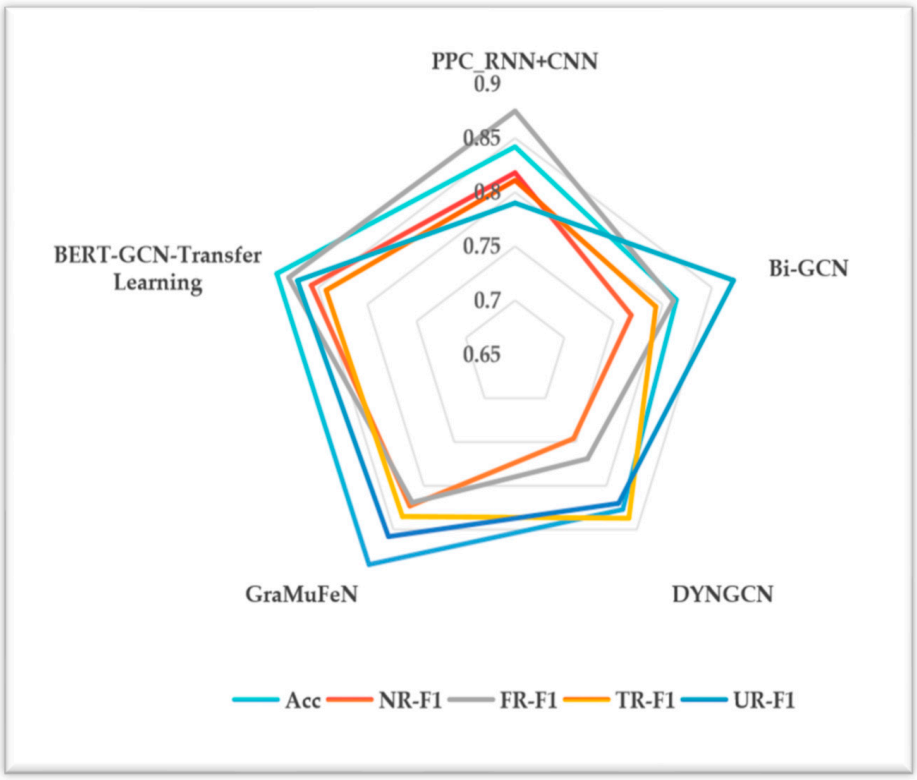

Figure 14. Performance comparison with other baseline models on dataset Twitter15.

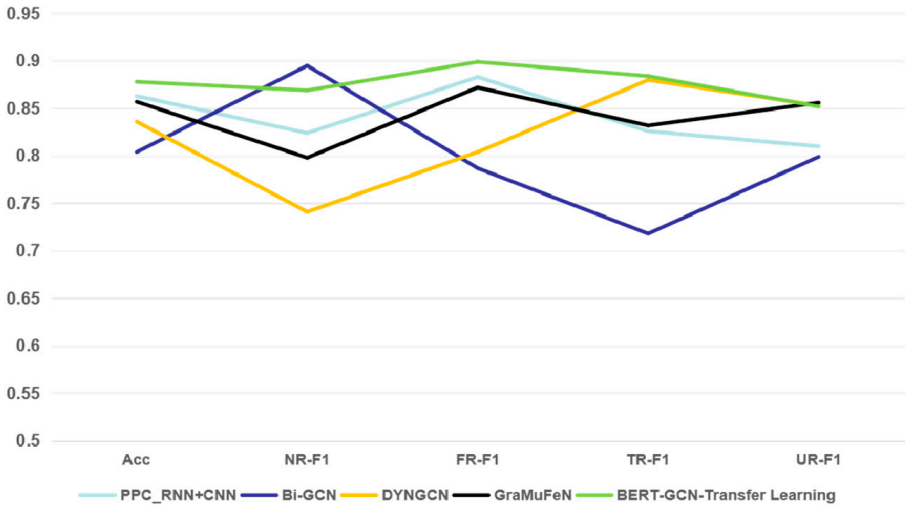

Figure 15. Performance comparison with other baseline models on dataset Twitter16.

These three experiments are validated on the target datasets Twitter15 and Twitter16, respectively, to measure the performance and rationality of different modules, and they are compared with the proposed BERT–GCN–Transfer Learning model. The experimental results are shown in Table 8.

Table 8. Comparison of ablation results.

	Method	Accuracy	TR			FR		
			Precision	Recall	F1-Score	Precision	Recall	F1-Score
Twitter15	①	0.869	0.839	0.833	0.836	0.801	0.701	0.794
	②	0.838	0.836	0.834	0.835	0.799	0.797	0.798
	③	0.880	**0.842**	0.836	0.839	0.874	0.831	0.852
	④	**0.892**	0.839	**0.845**	**0.842**	**0.882**	**0.878**	**0.880**
Twitter16	①	0.871	0.853	0.875	0.864	0.872	**0.884**	0.880
	②	0.849	0.879	0.877	0.878	0.859	0.865	0.862
	③	0.874	**0.887**	0.885	**0.886**	0.792	0.794	0.794
	④	**0.878**	0.874	**0.894**	0.884	**0.901**	0.808	**0.899**

Through analysis, it can be observed that the proposed BERT–GCN–Transfer Learning model performs outstandingly overall, surpassing other methods in classification performance. On the Twitter15 dataset, the model achieves an accuracy of 0.892, which is 1.4% higher than Experiment ③ and 6.4% higher than Experiment ②. When determining whether a rumor is true, the BERT–GCN–Transfer Learning model achieves a recall rate of 0.845 and an F1 score of 0.842, both higher than other methods. In identifying false rumors, the model achieves an F1 score of 0.880, indicating that the BERT module, GCN module, and model transfer module play active roles in cross-task rumor detection.

Experiment ③, which removes the model transfer module, achieves a precision rate of 0.842 when determining whether a rumor is true, but falls short in other metrics compared to the proposed model, suggesting that the model transfer method is crucial for tackling cross-task rumor detection.

Experiment ② excludes the GCN module, retaining only the BERT module and model transfer module for rumor detection. From the results, on the Twitter15 and Twitter16 datasets, Experiment ② achieves accuracies of 0.838 and 0.849, respectively, lower than the 0.880 accuracy of Experiment ③. Since the GCN module possesses feature extraction capabilities for graph-structured data, and the ability to extract features is directly related to the effectiveness of detection in rumor detection, this data comparison also demonstrates the importance of the GCN module.

Experiment ① removes the BERT module, retaining only the GCN module and model transfer module for rumor detection. Since BERT possesses robust text representation capabilities and the benefits of a pre-trained model, removing the BERT module negatively impacts the text representation process. Compared to the proposed BERT–GCN–Transfer Learning model, the performance of Experiment ① declines in all metrics.

Thus, the BERT–GCN–Transfer Learning model, by combining BERT's text representation capabilities, the GCN's feature extraction abilities for graph-structured data, and the cross-task generalization of transfer learning, exhibits exceptional performance in rumor detection tasks. Specifically, the high accuracy on the Twitter15 dataset and the high recall and F1 scores in true and false rumor classification fully prove the effectiveness and rationality of the model design. Among them, as the core text representation component, BERT learns rich linguistic knowledge and contextual information through pre-training, enabling it to generate high-quality text vectors. In rumor detection, the accurate representation of text content is crucial for subsequent classification and judgment. The significant performance drop in Experiment ① after removing the BERT module validates its importance in text representation.

The GCN (Graph Convolutional Network) excels at processing graph-structured data, capturing complex relationships between nodes. In rumor detection, rumor propagation paths, user interactions, etc., can be represented as graph-structured data. By extracting these graph-structured features, the GCN module provides additional information sources for the model, enhancing detection performance. The decrease in accuracy on both the

Twitter15 and Twitter16 datasets in Experiment ② after removing the GCN module proves its pivotal role in feature extraction.

Transfer learning enables models to apply knowledge learned from one task to another related but different task. From the source dataset of reviews to the target datasets of Twitter15 and Twitter16, from fake review detection to rumor detection, this demonstrates that transfer learning can significantly improve the model's generalization ability. The three modules each have their strengths and contribute to the overall performance improvement through their collaborative work. Intuitive ablation experiment comparisons are shown in Figures 16 and 17, where the yellow bars represent the performance of the proposed BERT–GCN–Transfer Learning model.

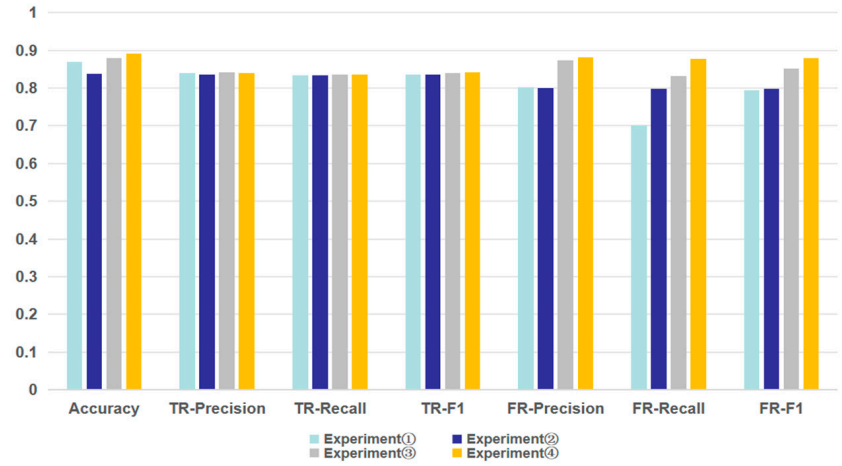

Figure 16. Comparison of ablation results on dataset Twitter15.

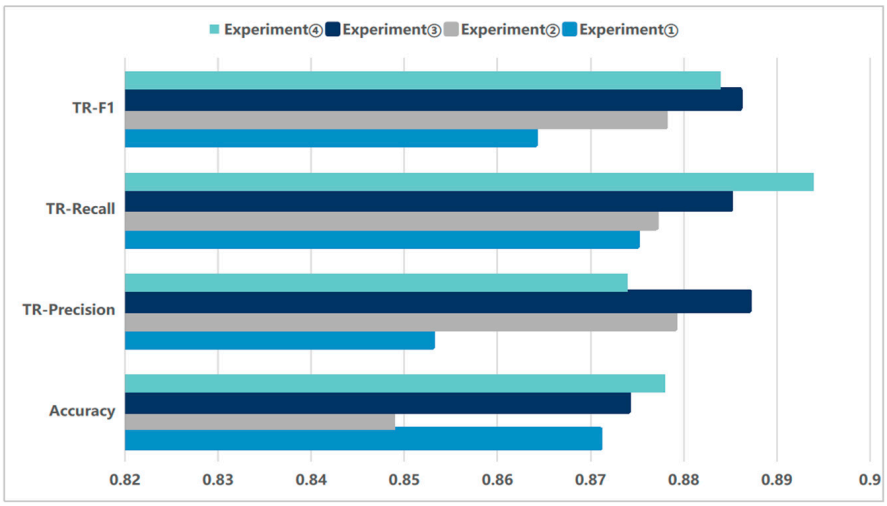

Figure 17. Comparison of ablation results on dataset Twitter16.

5. Conclusions and Future Work

This paper delves into the key technical challenges in cross-task rumor detection and innovatively proposes a rumor detection model, the BERT–GCN–Transfer Learning model, which integrates BERT, Graph Convolutional Network (GCN), and transfer learning techniques. This model leverages the advantages of multiple modules in knowledge sharing and transfer across tasks, effectively addressing issues such as data scarcity, complex features, and difficulties in cross-task knowledge transfer in rumor detection tasks. The experimental results demonstrate that our model not only achieves significant improvement in training efficiency but also attains remarkable detection accuracy on multiple benchmark datasets, significantly outperforming existing baseline models and showcasing its immense

potential in practical applications. In our next steps, we will attempt to incorporate more types of graph-structured data, such as user relationship networks and topic correlation graphs, to more comprehensively capture the complex relationships in the rumor-spreading process, further enhancing the accuracy and robustness of the detection model.

Author Contributions: Conceptualization, W.J. and X.Z.; methodology, W.J.; software, F.Y.; validation, W.J., F.Y. and K.R.; formal analysis, B.W.; investigation, W.J.; resources, M.Z.; data curation, X.Z.; writing—original draft preparation, W.J.; writing—review and editing, X.Z.; visualization, W.J. and X.Z.; supervision, M.Z.; project administration, B.W.; funding acquisition, M.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Social Science Foundation of China (20BXW101, 18XXW015). Innovation Research Project for the Cultivation of High-level Scientific and Technological Talents (Top-notch Talents of the Discipline) (ZZKY2022303). National Natural Science Foundation of China (No. 62102451, No. 62202496). Basic frontier innovation project of Engineering University of People's Armed Police (WJX202317). This work is also supported by the National Natural Science Foundation of China (No. 62172436) and Engineering University of PAP's Funding for Scientific Research Innovation Team, Engineering University of PAP's Funding for Basic Scientific Research, and Engineering University of PAP's Funding for Education and Teaching. Natural Science Foundation of Shaanxi Province (No. 2023-JCYB-584).

Data Availability Statement: Dataset available on request from the authors.

Acknowledgments: We would like to give our heartfelt thanks to all the people who have ever helped us and we thank the reviewer for the positive and constructive comments regarding our paper.

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

References

1. Berrondo-Otermin, M.; Sarasa-Cabezuelo, A. Application of artificial intelligence techniques to detect fake news: A review. *Electronics* **2023**, *12*, 5041. [CrossRef]
2. Aïmeur, E.; Amri, S.; Brassard, G. Fake news, disinformation and misinformation in social media: A review. *Soc. Netw. Anal. Min.* **2023**, *13*, 30. [CrossRef]
3. Fang, Y.; Wang, H.; Zhao, L.; Yu, F.; Wang, C. Dynamic knowledge graph based fake-review detection. *Appl. Intell.* **2020**, *50*, 4281–4295. [CrossRef]
4. Yu, Z.; Lu, S.; Wang, D.; Li, Z. Modeling and analysis of rumor propagation in social networks. *Inf. Sci.* **2021**, *580*, 857–873. [CrossRef]
5. Lauriola, I.; Lavelli, A.; Aiolli, F. An introduction to deep learning in natural language processing: Models, techniques, and tools. *Neurocomputing* **2022**, *470*, 443–456. [CrossRef]
6. Choudhry, A.; Khatri, I.; Jain, M.; Vishwakarma, D.K. An emotion-aware multitask approach to fake news and rumor detection using transfer learning. *IEEE Trans. Comput. Soc. Syst.* **2022**, *11*, 588–599. [CrossRef]
7. Meel, P.; Vishwakarma, D.K. Fake news, rumor, information pollution in social media and web: A contemporary survey of state-of-the-arts, challenges and opportunities. *Expert Syst. Appl.* **2020**, *153*, 112986. [CrossRef]
8. Castillo, C.; Marcelo, M.; Poblete, B. Information Credibility on Twitter. In Proceedings of the 20th International Conference on World Wide Web, Hyderabad India, 28 March–1 April 2011; pp. 675–684.
9. Kwon, S.; Cha, M.; Jung, K. Prominent Features of Rumor Propagation in Online Social Media. In Proceedings of the 2013 IEEE 13th International Conference on Data Mining, Dallas, TX, USA, 7–10 December 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 1103–1108.
10. Yang, F.; Yu, X.; Liu, Y. Automatic Detection of Rumor on Sina Weibo. In Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics, Beijing China, 12–16 August 2015; pp. 1–7.
11. Ma, J.; Gao, W.; Mitra, P. Detecting Rumors from Microblogs with Recurrent Neural Networks. In Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence 2016, New York, NY, USA, 9–15 July 2016; pp. 3818–3824.
12. Ma, J.; Gao, W.; Wong, K.F. Detect Rumors in Microblog Posts Using Propagation Structure via Kernel Learning. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics 2017, Vancouver, Canada, 30 July–4 August 2017; pp. 708–717.
13. Ma, J.; Gao, W.; Wong, K.F. Rumor Detection on Twitter with Tree-structured Recursive Neural Networks. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, 15–20 July 2018; pp. 1980–1989.

14. Chen, T.; Wu, L.; Li, X. Call Attention to Rumors: Deep Attention Based Recurrent Neural Networks for Early Rumor Detection. In *Trends and Applications in Knowledge Discovery and Data Mining, proceedings of the PAKDD 2018 Workshops, BDASC, BDM, ML4Cyber, PAISI, DaMEMO, Melbourne, VIC, Australia, 3 June 2018*; Springer International Publishing: Berlin/Heidelberg, Germany, 2018; pp. 40–52.
15. Yu, F.; Liu, Q.; Wu, S. Attention-based Convolutional Approach for Misinformation Identification from Massive and Noisy Microblog Posts. *Comput. Secur.* **2019**, *83*, 106–121. [CrossRef]
16. Liu, Y.; Wu, Y.F. Early Detection of Fake News on Social Media Through Propagation Path Classification with Recurrent and Convolutional Networks. *Proc. AAAI Conf. Artif. Intell.* **2018**, *32*, 354–361. [CrossRef]
17. Bian, T.; Xiao, X.; Xu, T. Rumor Detection on Social Media with Bi-Directional Graph Convolutional Networks. *Proc. AAAI Conf. Artif. Intell.* **2020**, *34*, 549–556. [CrossRef]
18. Khoo, L.; Chieu, H.L.; Qian, Z.; Jiang, J. Interpretable Rumor Detection in Microblogs by Attending to User Interactions. *Proc. AAAI Conf. Artif. Intell.* **2020**, *34*, 8783–8790. [CrossRef]
19. Wu, Z.; Pi, D.; Chen, J.; Xie, M.; Cao, J. Rumor Detection Based on Propagation Graph Neural Network with Attention Mechanism. *Expert Syst. Appl.* **2020**, *158*, 0957–4174. [CrossRef] [PubMed]
20. Choi, J.; Ko, T.; Choi, Y. Dynamic Graph Convolutional Networks with Attention Mechanism for Rumor Detection on Social Media. *PLoS ONE* **2021**, *16*, e0256039. [CrossRef] [PubMed]
21. Ma, T.; Huang, L.; Lu, Q.; Hu, S. Kr-gcn: Knowledge-aware reasoning with graph convolution network for explainable recommendation. *ACM Trans. Inf. Syst.* **2023**, *41*, 1–27. [CrossRef]
22. Zhou, Y.; Pang, A.; Yu, G. Clip-GCN: An adaptive detection model for multimodal emergent fake news domains. *Complex Intell. Syst.* **2024**, *10*, 1–18. [CrossRef]
23. Zhang, Z.; Lv, Q.; Jia, X.; Yun, W.; Miao, G.; Mao, Z.; Wu, G. GBCA: Graph Convolution Network and BERT combined with Co-Attention for fake news detection. *Pattern Recognit. Lett.* **2024**, *180*, 26–32. [CrossRef]
24. Neyshabur, B.; Sedghi, H.; Zhang, C. What is being transferred in transfer learning? *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 512–523.
25. Niu, S.; Liu, Y.; Wang, J.; Song, H. A decade survey of transfer learning (2010–2020). *IEEE Trans. Artif. Intell.* **2020**, *1*, 151–166. [CrossRef]
26. Qian, S.; Hu, J.; Fang, Q.; Xu, C. Knowledge-aware multi-modal adaptive graph convolutional networks for fake news detection. *ACM Trans. Multimed. Comput. Commun. Appl. (TOMM)* **2021**, *17*, 1–23. [CrossRef]
27. Scarselli, F.; Tsoi, A.C.; Hagenbuchner, M. The Vapnik–chervonenkis Dimension of Graph and Recursive Neural Networks. *Neural Netw.* **2018**, *108*, 248–259.
28. Zhang, C.; Song, D.; Huang, C.; Swami, A.; Chawla, N.V. Heterogeneous graph neural network. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 793–803.
29. Zhang, Y.; Lai, G.; Zhang, M.; Zhang, Y.; Liu, Y.; Ma, S. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval, Gold Coast, QLD, Australia, 6–11 July 2014; pp. 83–92.
30. Samek, W.; Montavon, G.; Lapuschkin, S.; Anders, C.J.; Müller, K.R. Explaining deep neural networks and beyond: A review of methods and applications. *Proc. IEEE* **2021**, *109*, 247–278. [CrossRef]
31. Vrbančič, G.; Podgorelec, V. Transfer learning with adaptive fine-tuning. *IEEE Access* **2020**, *8*, 196197–196211. [CrossRef]
32. Yacouby, R.; Axman, D. Probabilistic extension of precision, recall, and f1 score for more thorough evaluation of classification models. *Proc. First Workshop Eval. Comp. NLP Syst.* **2020**, 79–91.
33. Mishra, R. Fake news detection using higher-order user to user mutual-attention progression in propagation paths. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020; pp. 652–653.
34. Kananian, M.; Badiei, F.G.; Ghahramani, S.A. GraMuFeN: Graph-based multi-modal fake news detection in social media. *Soc. Netw. Anal. Min.* **2024**, *14*, 104. [CrossRef]
35. Zheng, J.; Zhang, X.; Guo, S.; Wang, Q.; Zang, W.; Zhang, Y. MFAN: Multi-modal Feature-enhanced Attention Networks for Rumor Detection. *IJCAI* **2022**, *2022*, 2413–2419.
36. Han, H.; Ke, Z.; Nie, X.; Dai, L.; Slamu, W. Multimodal fusion with dual-attention based on textual double-embedding networks for rumor detection. *Appl. Sci.* **2023**, *13*, 4886. [CrossRef]
37. Chen, Y.; Li, D.; Zhang, P.; Sui, J.; Lv, Q.; Tun, L.; Shang, L. Cross-modal ambiguity learning for multimodal fake news detection. *Proc. ACM Web Conf.* **2022**, 2897–2905. [CrossRef]