Discover Artificial Intelligence

Research

Rumor detection using dual embeddings and text-based graph convolutional network

Barsha Pattanaik[1] · Sourav Mandal[1] · Rudra M. Tripathy[1] · Arif Ahmed Sekh[2]

Received: 6 June 2024 / Accepted: 5 November 2024
Published online: 19 November 2024

Abstract
Social media platforms like Twitter and Facebook have gradually become vital for communication and information exchange. However, this often leads to the spread of unreliable or false information, such as harmful rumors. Currently, graph convolutional networks (GCNs), particularly TextGCN, have shown promise in text classification tasks, including rumor detection. Their success is due to their ability to identify structural patterns in rumors and effectively use neighborhood information. We present a novel rumor detection model using TextGCN, which utilizes a word-document graph to represent rumor texts. This model uses dual embedding from two pre-trained transformer models: generative pre-trained transformers (GPT) and bidirectional encoder representations from transformers (BERT). These embeddings serve as node representations within the graph, enhancing rumor detection. Combining these deep neural networks effectively extracts significant contextual features from rumors. This graph undergoes convolution, and through graph-based learning, the model detects a rumor. We evaluated our model using publicly available rumor datasets, such as PHEME, Twitter15, and Twitter16. It achieved 88.64% accuracy on the PHEME dataset, surpassing similar models, and performed well on Twitter15 and Twitter16 with accuracies of 81.98% and 83.41%, respectively.

Keywords Rumor detection · Rumor classification · Dual word embedding · Graph convolution network (GCN) · TextGCN

1 Introduction

Social media is so widely used and accessible that rumors may spread quickly, substantially damaging society. There is uncertainty over depending exclusively on user comments to determine the legitimacy of rumors. Because social media users tend to disseminate rumors whose truth is still unknown. However, the rapid spread of rumors adversely affects society and individuals. Therefore, spotting those rumors on social media is crucial. Recently, most researchers have developed different models, mostly DL-based, to detect rumors early. Following our comprehensive survey [28], we observed that rumor detection is primarily approached as a binary or multi-class text classification task to discern whether the information is a rumor, a non-rumor, or otherwise. Presently, deep learning (DL) surpasses traditional

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s44163-024-00193-6.

✉ Barsha Pattanaik, barsha@xustudent.edu.in; ✉ Sourav Mandal, sourav@xim.edu.in; Rudra M. Tripathy, rudramohan@xim.edu.in; Arif Ahmed Sekh, skarifahmed@gmail.com | [1]School of Computer Science and Engineering, XIM University, Bhubaneswar 752050, Odisha, India. [2]Department of Computer Science, UiT The Arctic University, Tromsø 9017, Norway.

machine learning techniques in rumor detection. However, further enhancements are needed to achieve higher accuracy levels and increase confidence in Leading-edge techniques.

In prior studies, researchers employed sequence learning models like recurrent neural networks (RNNs) [8, 23, 37] and convolutional neural networks (CNNs) [13, 35], which focus on extracting text from consecutive word sequences but overlook global word co-occurrences and contextual nuances in the corpus. Consequently, some turned to graph neural networks (GNNs) [5, 26, 38], a graph-based learning model, to extract global features within broader contexts. GNNs constitute a subset of deep learning approaches tailored for analyzing data represented by graphs [7]. They offer a direct application to graph structures, enabling seamless execution of tasks involving node-level, edge-level, and graph-level predictions. GNNs fill a void where CNNs encountered limitations, extending capabilities to domains CNNs couldn't adequately address [27].

We use both BERT and GPT as embedding models for our approach because of their complementary strengths in managing contextual information. BERT, as a bidirectional transformer model, proficiently comprehends word context through surrounding words, rendering it ideal for tasks necessitating profound contextual knowledge. Conversely, GPT, a unidirectional model, produces coherent text by forecasting the subsequent word based on preceding words, which aids in comprehending sequential dependencies in text. We have been inspired by the analysis of BERT and GPT embedding reported in [16], where the authors show that these two different models have different embedding styles. GPT is more inclined to the contextual information, whereas BERT shows strong word similarity-based embedding. For example, the authors visualize the Position of the word "banks" in GPT and BERT. Hence, they are different and complementary.[1] By utilizing both models, we seek to obtain a more comprehensive representation of textual data, enabling us to harness the advantages of each model in various contextual dimensions. This dual embedding method improves the performance of our classification problem by offering varied contextual insights that a singular model may neglect.

1.1 GCN and TextGCN: a brief description

GCN represents a distinct category within GNNs, employing convolutional techniques to transmit data across nodes within a graph structure. A GCN, first proposed by [17], a multilayer graph neural network, generates node embeddings based on neighborhood properties within graphs. Formally, for a graph $G = (V, E)$ with nodes V ($| V | = n$) and edges E, each node v has a self-loop, $(v, v) \in E$.

The feature matrix $X \in \mathbb{R}^{n \times m}$ contains the features of all nodes (n) and where m represents the feature vectors' dimensions, with each row $X_v \in \mathbb{R}^m$ representing a node's feature vector. For a single layer GCN, the new k-dimensional node feature matrix $H^{(1)}$ is computed as follows:

$$H^{(1)} = \sigma\left(\hat{A}XW^{(0)}\right)$$

where the normalized adjacency matrix $\hat{A} = D^{-1/2}AD^{-1/2}$, weight matrix $W^{(0)}$, and activation function σ is like ReLU and the degree matrix is D, where $D_{ii} = \sum_j A_{ij}$. Higher-order information is incorporated, with $H^{(0)} = X$ and j the layer number, by stacking multiple layers:

$$H^{(j+1)} = \sigma\left(\hat{A}H^{(j)}W^{(j)}\right)$$

Thereafter, [40] first applied GCNs for text classification, introducing a model called TextGCN, which utilizes two GCN layers on word and document embedding (word-document graph). Their research [40] demonstrated that GCNs achieve higher accuracy in text classification compared to CNN and RNN models [19]. TextGCN performs text classification by emphasizing the extraction of spatial or global features from the text. In [40], word-document graphs are built using pointwise mutual information (PMI) for word-word correlations and term frequency-inverse document frequency (tf-idf) for correlations of word with documents, which serve as the adjacency matrix input for TextGCN. The nodes (all unique words plus documents) in the TextGCN are represented as one-hot vector embeddings ($X \in \mathbb{R}^{n \times m}$). The weighted adjacency matrix A is defined in [40] and given below.

[1] https://bert-vs-gpt2.dbvis.de/.

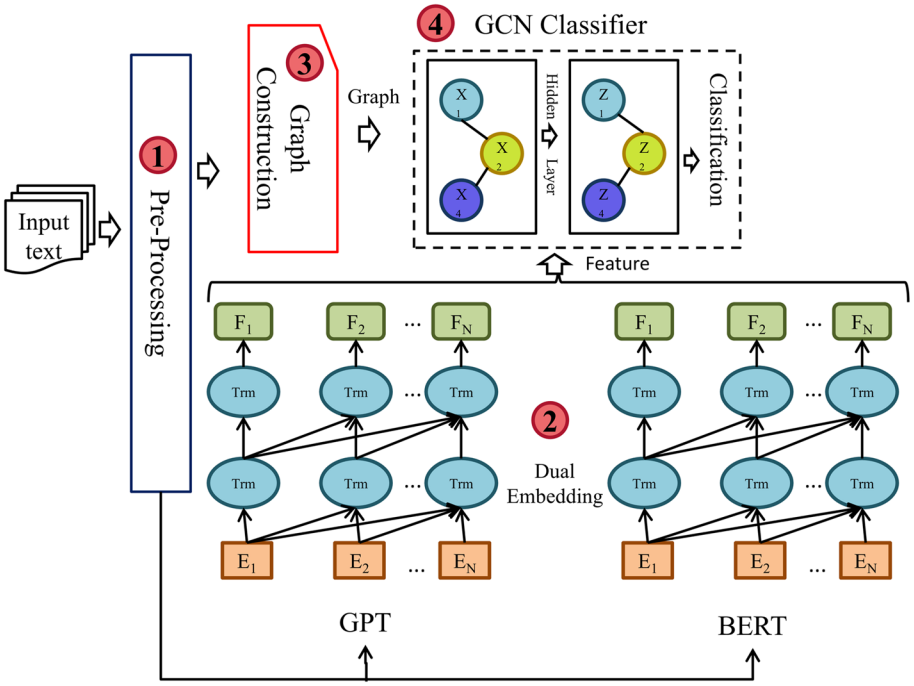

Fig. 1 The proposed method consists of 4 major components depicted by red circles. (1) is a pre-processing unit, (2) is a pre-trained feature extractor module and dual embedding, (3) is an algorithmic text-document graph construction unit, and (4) is a trainable GCN framework. The model utilizes BERT and GPT as feature extractors and a graph train a GCN model. x: Input features (e.g., combined embeddings from BERT and GPT) whereas, z: Output features after GCN processing (trans- formed node embeddings)

$$A_{rc} = \begin{cases} \text{PMI}(r,c) & \text{if } r \text{ and } c \text{ are words, PMI}(r,c) > 0 \\ \text{tf-idf}_{rc} & \text{if } r \text{ is a document, } c \text{ is a word} \\ 1 & \text{if } r = c \\ 0 & \text{otherwise} \end{cases}$$

After constructing the text graph, it is processed by a two-layer GCN similar to [17]. The second layer's node embeddings, matching the label set size, are fed into a softmax classifier for final classification:

$$H^{(2)} = \text{softmax}\big(\hat{A}\ \text{ReLU}\big(\hat{A}H^{(0)}W^{(0)}\big)W^{(1)}\big)$$

However, after [40], some researchers used TextGCN or GCN for text classification. Later, this model is also used for rumor detection or classification by many researchers. [15] used TextGCN and BERT-based word and document (sentence) embedding [11], to generate contextual embedding vectors. Documents are considered as a sequence of sentences to generate final document embedding vectors. [19] proposed a text classification model using transformer and textGCN and also applied part-of-speech (POS) features. Section 2 provides detailed research studies on rumor detection using DL-based approach, mostly with TextGCN.

1.2 Problem statement

Accurately determining whether a text includes a rumor is still a major difficulty. Conventional text classification techniques frequently produce poor classification results because they are unable to fully capture the intricate semantic links and contextual dependencies between words and texts. Inspired by the work of [15], we propose a TextGCN-based approach (Fig. 1) for rumor detection in this study. For our research work, we consider only the original texts/ news or tweets with their labels (rumor or not) as input to our model, and the output will be the prediction of whether the text is a rumor or not.

We applied GPT [29] and BERT-based [11] dual embedding for contextual feature extraction from rumor and used the combined feature vector as input to the TextGCN as node representation. Leveraging the strengths of both embedding techniques for feature vector generation for nodes in the word-document graph enhances accuracy up to 88.64% on the PHEME dataset [43]. Our GCN model used stochastic gradient descent (SGD) for training with early stopping to prevent overfitting. This is the first work to incorporate dual embeddings using both BERT and GPT for rumor detection, capturing complementary semantic information and improving classification accuracy and robustness. For our experiment, we use

publicly accessible datasets in this work, such as the PHEME [43], Twitter15, and Twitter16 [24] datasets discussed in 4.1. PHEME dataset contains a total of 5802 samples for five events, whereas Twitter15 contains 1490 and Twitter16 contains 818 samples. Evaluation metrics like accuracy and F1 score assess model performance.

Our motivation behind using dual embedding is to better represent the rumor text. GPT and BERT are most used in recent times and have complementary strengths in natural language processing. BERT is proficient in comprehending the context of words in a phrase and captures relationships in both directions. By contrast, unidirectional training of GPT makes it efficient at producing text and comprehending sequential relationships. Integrating rich contextual information from BERT and sequential patterns from GPT, achieved by merging these embeddings with a Graph Convolutional Network (GCN), enhances the model's performance in rumor identification. The dual embeddings increase the model's capacity to identify minute details in rumor-related information by offering a more thorough representation of the text data.

1.3 Our significant contributions in this study

The following are the primary contributions:

- We employ dual word embeddings with BERT and GPT in conjunction with TextGCN, a novel methodology (Fig. 1) that enhances rumor detection accuracy to 88.64%, surpassing previous GCN-based models on the PHEME dataset.
- GPT and BERT have complementary strengths. GPT excels in generating contextualized language models with a focus on generating coherent sequences, while BERT is designed for deep bidirectional understanding of text. By employing both models, we leverage GPT's capability in capturing context and sequence dependencies and BERT's strength in understanding deep contextual relationships.
- We provide an ablation study (Table 2) using different kinds of embeddings with TextGCN models as a variant of the proposed model components.

The article continues as follows: we discuss relevant research (Sect. 2), detail our model (Sect. 3), analyze results (Sect. 4), and conclude (Sect. 5).

2 Related work

Early research on rumor identification often employed conventional machine-learning (ML) techniques based on human-annotated features. Researchers used different handcrafted features from a labeled dataset, such as sentiments or temporal [39] or semantic features [12]. So, they focused on DL-based models to detect rumors on social media to strengthen the shortcomings of hand-crafted features and improve the model's performance. [23] used RNNs to build hidden representations, which can capture changes in contextual information of pertinent postings over time. [21] developed an LSTM-based model to detect rumors by analyzing continuous changes in content, spreaders, and diffusion structures throughout the spreading process, both early on and later. Other researchers [8, 18, 21, 25] also developed a few RNN-based models to detect rumor in social media, whereas researchers like [35] developed a CNN-based model that combined text and proposition structure representation learning. Other researchers like [1, 13] have also used CNN-based rumor detection models. The details of various models for rumor detection are available in some recent survey papers [6, 28, 33, 36]. These rumor detection techniques mostly used RNN or CNN, which extract text features from word sequences locally [23]. However, they ignored global and long-distance word relationships with non-consecutive meanings. Graph-based learning prioritized global feature exploitation, which deals directly with complicated structured data [15]. So, researchers gave attention to GCN-based models [5, 26, 38] for detecting rumors by analyzing different features of the input texts. This is mostly due to their ability to retain the global structure in graph embeddings and assess complex relational structures.

[26] proposed a recursive neural network, RVNN, for rumor detection in microblogs, integrating propagation structure and content semantics with top-down and bottom-up information flow. To check the model's performance, the authors used two publicly available datasets, such as Twitter15 and Twitter16, developed by [23] and obtained an accuracy of 72.3% on Twitter15 and 73.7% on Twitter16 datasets. They focused only on sequential rumor propagation, neglecting rumor dispersion. Furthermore, [41] developed a CNN-based model to identify misinformation by capturing local correlations but couldn't handle global word relationships. So, looking at these points, [5] analyzed

both the propagation and dispersion structures of rumors to identify essential features. The decision to utilize GCN stemmed from the aim of acquiring sophisticated representations from structured data. The authors used publicly available datasets such as Weibo [23], Twitter15, and Twitter16 and obtained an accuracy of 96% on the Weibo and 88.6% on Twitter15 and 88.0% on Twitter16 datasets.

[38] showed that [5] aggregates neighbor's feature based on reply or retweet. To address uncertainty, they proposed an edge-enhanced Bayesian GCN model, EBGCN, that analyzed text bi-directionally, focusing on rumor propagation and dispersion. They evaluated its performance on Twitter15, Twitter16, and PHEME datasets [43]. They showed that their model had an accuracy of 89.2% on Twitter15, 91.5% on Twitter16, and 71.5% on PHEME datasets. By considering the connections among every tweet about a certain topic, [4] suggested a model, EGCN, based on GCN. The model is built on interactions between source tweets and retweets, considering both local and global message structures. The authors employed an ensemble structure of a textCNN layer a GCN with a SortPooling layer, and a 1-dimension convolution layer to construct the model. The geographical features-based GCN network discovered the association between the source and reply tweets. Their model obtained 70% accuracy on the PHEME dataset. An adversary-aware model was also proposed by [31] to produce adversarial responses, considering the propagation structure's response location.

[34] developed a BERT-based GCN model where initially BERT embedding was applied to the texts to convert all the texts to numerical vector. They used BBC News and the IMDB movie reviews dataset for this experiment. Then, GCN was applied to the embedded vectors for classification. [32] developed another GCN-based model, DDGCN, to study the dynamic characteristics of the knowledge information for the rumor detection task. One GCN works on the evolving propagation graph for the spatial and temporal structure of the input message as a dynamic propagation representation, and the other operates on the evolving knowledge graphs associated with the message to learn a dynamic knowledge representation. The authors compared the work of [5] those who neglected temporal information in diffusion. Their model produced 94.8% accuracy on Weibo datasets and 85.5% on PHEME datasets. Further, [10] used attention mechanisms to hierarchically integrate external knowledge into the text and employed GCN to reveal semantic connections, modeling both local and global information. Their model, KAGN, produced an accuracy of 89.2% on Twitter15, 90.1% on Twitter16, and 86.5% on PHEME datasets. [2] developed a BERT-based rumor detection model by using sentence embedding to extract the contextual meaning of Twitter sentences and reveal the specific linguistic structures of a tweet. They used the PHEME dataset and showed an accuracy of 85.5%, which was a better result than existing state-of-the-art techniques.

Recently, [3] developed a concise and efficient BERT-based model, CE-BERT, for rumor detection. According to the researchers, their model performed very well for the PHEME dataset compared to the other two datasets, Twitter15 and Twitter16. Their model obtained an accuracy of 85.5% on the PHEME, 89% on Twitter15, and 86% on Twitter16 dataset. [20] proposed a model to understand both representations of user correlation and information transmission. In particular, they utilized graph neural networks to study the information representation and propagation using a tree structure and user correlation with a bipartite graph that highlights the correlations between users and source tweets. Rumors are then classified by merging the learnt representations from these two modules. They also devised a greedy attack technique for evaluating the cost of three adversarial attacks: graph attack, comment attack, and joint attack. They demonstrated that their model gave 86.0% accuracy on Twitter15 and 85.3% on Twitter16 datasets.

Recently, some researchers have used dual embedding in the field of text or multimodal data classification. [22] developed a dual word embedding model for cross-domain sentiment classification. They used both BERT and Word2vec to extract syntactic as well as semantic information from the document. [9] developed a Dual-view distilled BERT based model. The goal is to preserve the efficiency of Siamese BERT networks while integrating word-level interaction features into sentence embeddings. The method, which is based on multi-view learning, employs two views: (1) the Siamese View, in which the fundamental framework is provided by Siamese BERT networks, which are efficient in producing sentence embeddings and capturing semantic similarity; and (2) Interaction View, in which multiple teachers are provided by standard pre-trained models with cross-sentence interactions, assisting the Siamese networks. Using two different word embedding methods, which combine both GloVe and Word2vec models to represent the text, [42] proposed a text sentiment classification model that forms a combinatory input of dual channels of a CNN. To identify the CNN sentiment classification model with superior combination input compared to a single vector representation, the initial word vector was continually learned and updated based on the word vector fine-tuning technique. [30] developed a contrastive learning-based model named CLIP to analyze both text and image data. In this paper, a model for training vision systems using massive amounts of text data was provided. Aligning both image and word embeddings in a common space enables the model to interpret and predict visual notions from written descriptions. This method sought to enhance the model's comprehension and generalization of visual ideas by making use of the rich and varied data found in the text.

The above papers on dual embedding based on either text or image text, inspired us to use both GPT and BERT stems from the necessity to capture the full spectrum of features from the original tweets for rumor detection.

After doing a lot of surveys, we found that researchers used different complex models based on TextGCN to analyze the propagation path of the rumor spread. However, few explained how to construct a graph to understand the whole corpus. Furthermore, most recent word embedding models were neglected to reduce the computation time. Here, we introduce dual word embedding using GPT and BERT to include the larger context of a word along with TextGCN. Our model demonstrates superior performance compared to those utilizing single embedding on the PHEME dataset.

3 Proposed method

In this section, we elaborate on our proposed rumor detection model. The method comprises four distinct modules: (1) pre-processing, (2) dual embedding for feature extraction, (3) constructing a text-document graph, and (4) employing a trainable GCN classifier with fused features. Each module has unique characteristics and roles (see Fig. 1).

3.1 Pre-processing

This step involves cleaning the data. The module cleans each sentence by removing non-alphanumeric characters, extra spaces, and other unwanted characters (eg., special characters such as punctuation marks (e.g., !, @, #),). It also converts the text to lowercase. After cleaning, the sentences are tokenized into words. Stop words (common words like 'and,''the,' etc.) are removed, and a minimum word frequency threshold threshold of word frequency=2) is applied to filter out less frequent words.

3.2 Feature extraction and dual embedding

The proposed model uses a dual embedding for both word and sentence embedding from two different modalities of transformers. We utilize pre-trained BERT to produce contextual word embeddings for every word in a sentence. We employ mean-pooling to generate sentence embeddings by averaging the word embeddings to create a fixed-length representation of the phrase. This method captures the comprehensive semantics of the statement and was selected for its simplicity and efficacy in our experiments. It incorporates BERT and GPT embeddings to capture rich semantic information from text data. The feature matrix X can be represented as $X = X_{\text{BERT}} \oplus X_{\text{GPT}}$, where, X_{BERT} is the feature matrix obtained from BERT embeddings and X_{GPT} is the feature matrix obtained from GPT embeddings. As two different types of transformers are designed and trained in different datasets, they carry different contextual information for a given text.

Our model features a graph with nodes that symbolize both words and texts. The initialization characteristics for these nodes are derived via a synthesis of GPT-2 and BERT embeddings, as detailed below:

Document representations We tokenize the text of each document utilizing both GPT-2 and BERT tokenizers. Subsequently, we input the tokenized text into pre-trained GPT-2 and BERT models to acquire their corresponding sentence embeddings. GPT-2 and BERT encode the complete document (up to a designated token limit, generally 512 tokens). The embeddings derived from these models encapsulate semantic information at the document level. We compute the average of the hidden states from the last layer of GPT-2 and BERT to represent each document. The embeddings are subsequently concatenated to provide a unified document representation.

Word Representations Token-level representations for each word in the papers are taken from both GPT-2 and BERT. The token embeddings are either padded or shortened to a uniform length for uniformity. When a word is present in both GPT-2 and BERT tokenizations, we utilize the lesser of the two embeddings (element-wise) to achieve a cohesive representation. This aids in capturing both contextual and grammatical subtleties of the term. For any term in the vocabulary absent from the document, we initialize its embedding as a null vector. Consequently, BERT and GPT-2 encode representations of both words and documents. The embeddings are amalgamated to initialize the node features in the graph, with document nodes represented by the aggregated document embeddings and word nodes represented by the word embeddings obtained from the token representations of BERT and GPT-2.

This method offers extensive contextual information for both word and document nodes, rendering the network appropriate for further tasks such as categorization or rumor detection.

Aggregation The architectural differences between BERT (encoder-only) and GPT (decoder-only) models and their embedding compatibility are also different. Both methods produce high-dimensional embeddings (e.g., 768 dimensions

for basic versions) via distinct mechanisms-BERT employs bidirectional context, whereas GPT operates unidirectionally. To guarantee compatibility, we initially adjust the dimensions of the embeddings through a linear projection layer. Following alignment, we integrate the embeddings for each token, utilizing the advantages of both models. Since BERT and GPT are pre-trained on similar language modeling objectives and possess overlapping vocabulary, their outputs encapsulate complimentary semantic information, facilitating effective integration in our model. This methodology ensures the harmonious integration of both embeddings within our model. We have taken the final fully connected layer output of the pre-trained models (BERT and GPT). Here, the final embedding is the combination of the two vectors.

3.3 Graph construction

In this module, the input rumor text is converted to a word document graph $G = (V, E)$, where V represents all nodes which are both words and documents (words plus documents) and E represents all edges which are relationships between nodes. (A) is a weighted adjacency square matrix. The graph created for our model is undirected, which illustrates the edges connecting words and documents, signifying mutual relationships and facilitating symmetrical information flow in both directions. The method is described in Algorithm 1.

Algorithm 1 Graph Construction

Input: $D : \{d_1, d_2, ..., d_N\}$ ▷ List of tokenized documents.
Output: A ▷ Adjacency matrix
procedure MAKEGRAPH(D)
  $A = [][], WF = []$ ▷ WF = *Word Frequency*
  for d_i in D **do**
    for t_j in d_i **do**
      $WF[t_j] += 1$
    end for
  end for
  $node\_size = len(WF) + len(D)$
  $A = [node\_size][node\_size] = 0$ ▷ Initialize to zero
  for i in $node\_size$ **do**
    for j in $node\_size$ **do**
      if $i \leq len(WF)\ AND\ j \leq len(WF)$ **then** ▷ i, j represents Word
        $A[i][j] = CALCULATE - PMI(i, j)$
      else if $i > len(WF)\ OR\ j > len(WF)$ **then** ▷ i, j is Document-Word
        $A[i][j] = TF - IDF_{ij}$
      else if $i = j$ **then**
        $A[i][j] = 1$
      end if
    end for
  end for
  return A
end procedure
procedure CALCULATE-PMI(D)
  $p(i) = \frac{F_i}{Total\ Window}$ ▷ Freq. of $Word_i$ in Sliding Window
  $p(i, j) = \frac{F_i j}{Total\ Window}$ ▷ Freq. of $Word_i$ and $Word_j$ in Sliding Window
  $PMI = \log \frac{p(i,j)}{p(i)p(j)}$ ▷ Freq. of $Word_i$ and $Word_j$ in Sliding Window
  return PMI
end procedure

This technique constructs a graph from tokenized documents, with nodes representing words and documents, while edges are weighted according to statistical metrics. The graph is organized with an adjacency matrix A, which encapsulates the associations between words and documents via Pointwise Mutual Information (PMI) and Term Frequency-Inverse Document Frequency (TF-IDF). Tokenized documents are inputs to the model for graph construction. The technique generates a network from a collection of tokenized documents, where nodes signify both words and documents, and edges delineate links via Pointwise Mutual Information (PMI) and Term Frequency-Inverse Document Frequency (TF-IDF). The process commences with the computation of word frequencies throughout the corpus, followed by the establishment of an adjacency matrix predicated on the aggregate number of words and documents. Edges connecting word nodes are weighted using Pointwise Mutual Information (PMI) to assess co-occurrence, whereas edges linking the document and word nodes are weighted by Term Frequency-Inverse Document Frequency (TF-IDF) to indicate word significance within the document. Self-loops are incorporated for all nodes, and the finalized adjacency matrix is provided, depicting the graph topology.

3.4 Trainable GCN network

A GCN network is trained to fine-tune the prediction using domain-specific data (rumor detection). The model takes feature vectors (X) and A to predict the text labels. The input data (A, X) are passed through the GCN-based trainable classifier, which consists of GCN layers.

GCN Layer: This layer performs the essential operation of a GCN: aggregating and transforming node features using the graph structure. It consists of state-of-the-art weight and bias parameters and dropout. During the forward pass, the layer uses the weight matrix directly as the node features. Otherwise, it multiplies the input features by the weight matrix to obtain transformed features. These features are then aggregated according to the graph's adjacency matrix through sparse matrix multiplication. This process effectively blends each node's features with those of its neighbors, capturing the local graph structure. An optional bias term can be added, and if an activation function is provided, it is applied to the output. This results in a new feature set that combines the original node features and the graph topology, ready to be passed to the next layer or used for prediction.

The model aims to learn node representations H by aggregating information from neighboring nodes in the graph. This can be expressed using the following equation for a single layer of the GCN [15].

$$H^{(l+1)} = \sigma\left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right)$$

where:

$H^{(l)}$ is nodes, $W^{(l)}$ is weight at layer l. $\hat{A} = A + I$ is the adjacency matrix. $\hat{D}$ is the degree matrix of $\hat{A}$, and σ represents the activation function, typically ReLU.

Here, $H^{(l)}$ is defined by the dual embedding feature as:

$$H^{(l)} = X \tag{1}$$

The layer forwarded as:

$$H^{(l+1)} = ReLu\left(AH^{(l)}W^{(l)}\right) \tag{2}$$

The GCN Model: The method uses a two-layer GCN using graph convolution. The first graph convolution layer maps the input features to the hidden dimension with ReLU activation, and dropout propagates to the next layer. We have used 2 such layers in our experiment. The final graph convolutional layer maps the hidden dimension to the number of classes without any activation and finally, a softmax is used for a final classification as follows:

$$Z = softmax\left(AH^{L}W^{L}\right), Listhelastlayer \tag{3}$$

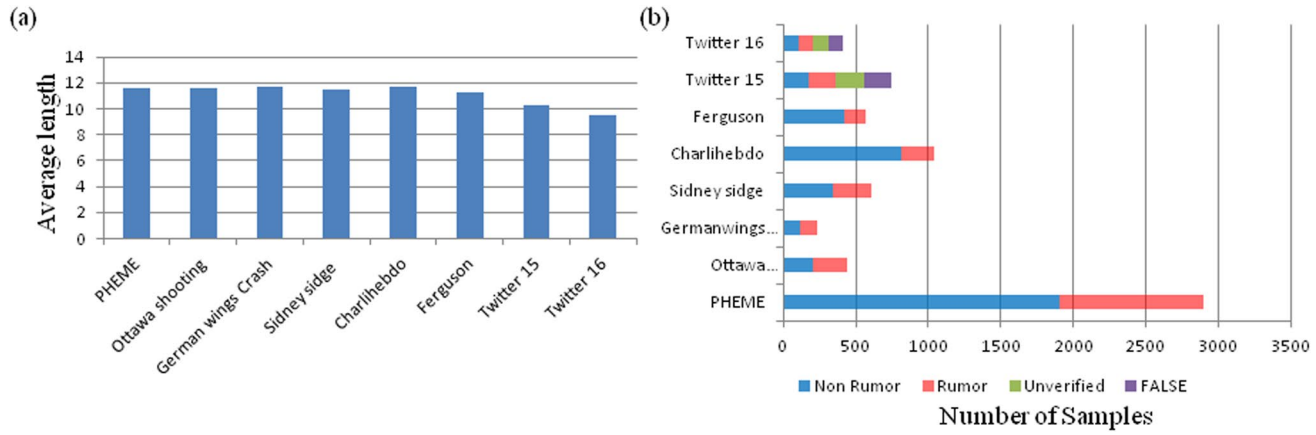

Fig. 2 Statistics of the datasets. **a** is the comparison of the average input length (after pre-processing). **b** Distributions of the classes

4 Experiment and result analysis

4.1 Datasets

For our experiment, we use three publicly available datasets, such as PHEME [43], Twitter15, and Twitter16 [24], that are widely used as reference datasets in rumor detection. PHEME dataset contains five events (Ottawa shooting, German wings Crash, Sidney sidge, Charlihebdo, and Ferguson). We evaluated our model using these datasets containing tweets labeled with predefined categories such as rumor and non-rumor. In contrast, the two other datasets, such as Twitter15 and Twitter16, are annotated with four labels. The number of data points (rumor and non-rumor) is "PHEME" (1910, 989), "Ottawa Shooting" (209, 236), "Germanwings Crash" (118, 116), "Sidney sidge" (343, 267),"Charlihebdo" (816, 223), "Ferguson" (424, 147), "Twitter 15" (178, 185), and "Twitter 16" (106, 100). The dataset is evenly split into training and test sets (50:50), with each document represented as a sequence of words and sentences. We use a random normal distribution for each class (rumor and non-rumor) to split the data to ensure 50% of the data for each class are distributed for training and testing. Figure 2(a) shows the average length of text, Fig. 2(b) the number of samples after the pre-processing stage, and the distribution of classes for both PHEME and Twitter datasets. Further dataset details are provided below.

4.2 Training and evaluation

During the training, the graph is passed as an adjacency matrix (A) combined with the dual embedding feature (X). The model is trained using Adam Optimizer and with an initial learning rate of 0.02. An early stop criteria over 10 epochs (maximum 200 epochs) with a weight decay of 0.001 is employed to reduce the overfitting. In this study, we set EDGE to 2, to capture both local and global links, we used a mix of document-to-word, word-to-word, and document-to-document edges. With a hidden dimension of 200 and a two-layer architecture (NUM_LAYERS = 2), we were able to capture complicated patterns with enough capacity. We set a dropout rate of 0.5. These hyperparameters were carefully adjusted during the model's 200-epoch training process to achieve a compromise between efficiency and performance.

The training objective aims to minimize the cross-entropy loss function. We minimize the cross-entropy loss, which quantifies the divergence between the predicted class distribution H(L) and the actual distribution Y (i.e., ground truth labels). The cross-entropy loss is mathematically expressed as

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} Y_{i,c} log\left(H(L)_{i,c}\right)$$

where:

- N is the number of samples,
- C is the number of classes,
- Y is the ground truth label matrix.

Table 1 Model performance comparison with our proposed model with similar GCN-based models

Models	PHEME (%)	Twitter15 (%)	Twitter16 (%)
Bi-GCN[5]	65.24	81.5	82.26
EBGCN [38]	71.50	85.21	87.53
DDGCN [32]	85.51	81.70	82.53
Bert+GCN [15]	86.13	74.72	78.78
Bert+GPT+GCN (Proposed model)	88.64	81.98	83.41

- Cross-Entropy computes the cross-entropy loss between predicted and ground truth labels.

The model is trained using 50% of the data and the rest is used for testing. Choosing to split up 50% of the data for training was made in accordance with standard practices in this region, striving to achieve equilibrium between training and validation datasets while guaranteeing adequate data for model training. Further, this split allows for robust evaluation while preventing over-fitting. In this work, we evaluate model performance using accuracy [14], Macro F1, and Weighted F1. Accuracy measures the percentage of correctly classified samples in the test set. The F1 score, derived from precision and recall, is the harmonic mean of these metrics. Precision measures the ratio of correctly classified rumor messages to all classified rumors. At the same time, recall gauges the ratio of correctly classified rumors to all messages that should be classified as such. Weighted F1 score averages F1 scores for each class, with weights proportional to the true instances in each class. Further, as we know, both the BERT and GPT models include a lot of parameters, so we made sure the dataset size was big enough to allow for adequate fine-tuning. Our experimental methodology, which pays close attention to the training dynamics to prevent overfitting, validates that the dataset was sufficient for this objective.

4.3 Baseline and state-of-the-art (GCN-based)

In this section, some GCN-based remarkable systems are listed below with their baseline accuracy given in Table 1.

- Bi-GCN [5]: A GCN-based model which used both the propagation and dispersion structures for node representations with root node attributes.
- EBGCN [38]: Enhancing uncertainty propagation for rumor detection via an edge-enhanced Bayesian GCN.
- DDGCN [32]: A GCN-based model which considered the dynamic characteristics of the knowledge information for the rumor detection task.
- Bert+GCN [15]: Examined node and edge embeddings in graphs along with GCN learning techniques for text classification. We re-implemented their model and tested it on rumor datasets to yield results.

4.4 Results and analysis

Here, we compare and discuss the experimental results of our proposed model using various embeddings and juxtapose them with other cutting-edge models.

4.4.1 Comparison with other models

Table 1 shows the performance comparisons on the datasets. Our model produced the best accuracy on the PHEME dataset and outperformed all the other models in all the metrics, such as accuracy, macro F1, and weighted F1. Our model, which uses dual embedding with GCN, is the first-ever rumor detection model using dual embedding. Our BERT+GPT+GCN model performed well on all events of the PHEME dataset, as shown in Table 2. Somehow, for Twitter15 and Twitter16, the results are lower than some state-of-the-art models [38]. The small size of the Twitter15 and Twitter16 datasets and the imbalanced distribution of their four categories (unverified, non-rumor, true rumor, and false rumor) might significantly affect the performance of our proposed method.

The deep learning models often require bigger datasets to perform well in generalization since they employ complex feature extraction techniques like GPT, BERT, and GCN. We can also add other features to improve the performance. We have also done an ablation study where we used different types of embeddings, which is discussed in the below section.

Table 2 Ablation study on rumor datasets

Dataset	Ottawa shooting			Germanwings Crash			Sydney siege			Charlie Hebdo			Ferguson			Twitter15			Twitter16		
Methods	Accuracy (%)	Macro F1 (%)	Weighted F1 (%)	Accuracy (%)	Macro F1 (%)	Weighted F1 (%)	Accuracy (%)	Macro F1 (%)	Weighted F1 (%)	Accuracy (%)	Macro F1 (%)	Weighted F1 (%)	Accuracy (%)	Macro F1 (%)	Weighted F1 (%)	Accuracy (%)	Macro F1 (%)	Weighted F1 (%)	Accuracy (%)	Macro F1 (%)	Weighted F1 (%)
BERT+ Logistic Regression	87	86	86	83	83	83	80	79	79	86	85	85	81	80	80	63	62	63	67	67	67
GPT + Logistic Regression	80	79	80	82	82	82	80	80	80	85	84	8 5	81	81	81	54	54	54	59	59	59
BERT+Support Vector Machine	84	84	84	81	81	81	75	75	75	87	81	87	80	71	79	63	63	63	77	77	77
BERT+ Decision tree	74	74	74	67	67	67	64	63	64	76	68	77	73	66	74	38	38	38	44	44	44
GPT+Support Vector Machine	84	84	84	78	78	78	79	79	79	88	81	87	82	75	82	62	62	62	67	67	67
GPT+ decision tree	68	68	68	66	66	66	65	64	65	77	67	77	75	67	75	40	40	40	50	50	50
Bert+ GCN	85.57	85.43	85.58	80.70	79.78	79.87	73.55	73.35	73.66	85.59	80.07	85.85	80.94	76.39	81.75	74.72	73.45	74.72	78.78	78.64	78.87
Roberta+ GCN	84.04	83.96	84.02	80.51	80.49	80.48	76.97	76.46	77.15	86.23	81.26	86.55	83.25	79.40	83.47	73.32	73.34	73.32	76.21	76.23	76.21
GPT +GCN	87.57	87.55	87.57	82.88	82.49	82.88	82.49	82.75	82.97	86.45	80.57	86.42	82.59	73.17	80.58	77.68	77.28	77.26	77.41	77.54	77.46
ROBERTA +GPT +GCN	87.39	87.35	87.37	85.83	85.82	80.83	82.65	81.12	82.85	86.28	86.98	85.14	85.14	78.09	84.65	78.31	78.01	78.12	82.21	81.92	82.14
BERT +GPT +GCN	88.64	88.63	88.64	88.09	87.99	88.02	83.74	82.57	83.94	88.12	87.23	87.89	87.78	82.64	86.64	81.98	80.23	80.54	83.41	83.12	83.41

Table 3 Prediction for the proposed model on PHEME dataset

Tweet	True Label	Predicted Label	Possible reasons
Texas police officer receives death threats after video shows him fatally shooting a puppy http://t.co/h7nQ4ZG5eS http://t.co/nRWe0jqKYy	Non-rumor	Rumor	Few words such as "death threats" must be with sensationalized topic and connected with rumors in the trained data.
Full White House statement on Obama's call to #Ottawa http://t.co/7UAGgehJms	Rumor	Non-rumor	Trust in Official Sources like words "Obama" and "White House"

4.4.2 Ablation study

We conduct ablation studies to scrutinize the impact of our proposed components; these involve defining variations on embeddings, including Roberta and GPT individually, as well as in combination with each other, alongside GCN integration. Through this systematic examination, we aim to discern the relative contributions of each component to the overall performance of the model, elucidating how their interplay influences the effectiveness of the proposed approach; such a comprehensive analysis facilitates a deeper understanding of the intricate dynamics at play within the model architecture and sheds light on the optimal configurations for achieving optimal results in rumor detection tasks, thereby providing valuable insights for future research in this domain. Table 2 shows the comparison of our developed models using different options of embedding models. Initially, we used Roberta or GPT embedding instead of Bert and found there was no major change in the performance of these models as compared to the BERT-GCN model [15]. But while we use both ROBERTA and GPT embedding, accuracy increased. We observe that the BERT+GPT+GCN model performed better in both PHEME (performances for separate events are given) and Twitter datasets, as in Table 2.

Although ROBERTA optimizes BERT's architecture, it retains similar bidirectional properties, which limits its diversity in contextual representation. Our ablation studies indicate that the pairing of ROBERTA and GPT do not outperform BERT and GPT primarily because ROBERTA's advantages are more pronounced with larger datasets. Further, ROBERTA may perform better on large datasets, the BERT+GPT combination proves more effective for our specific tasks as per our result.

4.4.3 Failure analysis

Our goal in this study was to improve the accuracy of rumor detection inside a TextGCN framework by utilizing dual embeddings such as BERT and GPT embeddings. After checking the performance, we found our system gave very good results on PHEME dataset as compared to other baseline models but subpar performance for Twitter15 from several problems despite thorough testing and deployment. This section explores the consequences of the failure's contributing components and analyses them. The table 3 explains the prediction results of our proposed model on PHEME dataset. This explains that, somehow, our model fails to detect the output correctly. From the table, we can see a few tweets are there where the actual label and the predicted label do not match. This generates the failure of our approach, which may be due to several factors such as data quality, improper tuning of hyperparameters, or inadequate graph structure. Similarly, Table 4 shows the prediction of the Twitter15 dataset where our models also somehow failed to predict the actual label. For example, in Twitter15, one text is "minor leaguer hits grand slam, later finds out that he broke his own windshield...," showing as non-rumor, but its true label is true rumor. So, we can say when an occurrence appears unusual and realistic, the models could find it difficult to distinguish it from a rumor. However, the "Possible Reasons" as given in these tables are completely anticipated by us without any strong justification or experimental support.

Relying exclusively on BERT and GPT embeddings may not be adequate for successful rumor detection, as evidenced by the failure to attain the requisite classification accuracy and highlight the limits of our current technique. This failure highlights the necessity for more reliable integration techniques and close attention to model convergence. This analysis shows the room for further work in this area. A few can be as given below.

- Better results might be obtained by experimenting with other embeddings and stronger features.

Table 4 Prediction for the proposed model on Twitter15 dataset

Tweet/dataset	True Label	Predicted Label	Possible reasons
what's the no. True killer of americans? these rankings have changed little over the years URL URL (Twitter15)	Non-rumor	TRUE	"Rankings have changed little" might suggest a concealed truth and the model could misunderstand "True killer as evidence of misinformation or a conspiracy
paula deen moves to have discrimination suit dismissed. just in time for her new book "how to cook a jew"	FALSE	TRUE	The approach may place a lot of emphasis on the text's controversial or offensive passages (e.g., "How to Cook a Jew").
minor leaguer hits grand slam, later finds out that he broke his own windshield URL URL	TRUE	Non-rumor	When an occurrence appears unusual and realistic, models could have trouble telling it from a rumor.
confirmed: #mikebrown had no criminal record. URL #ferguson URL	Unverified	Non-rumor	Even if the claim is still unverified, the model might interpret "confirmed" as a significant signal of true content.

- Enhancement of the performance of TextGCN model with fine-tuning of the hyper-parameters and attention network.

5 Conclusion

We aimed to understand the underlying factors influencing rumor detection based on TextGCN. In light of this, we investigated graph-based learning and graph construction techniques. By incorporating both BERT and GPT embeddings, we tried to leverage the strengths of both models: BERT for capturing bidirectional contextual information, and GPT for generating contextually rich embeddings. This combined approach potentially improved the performance of our novel TextGCN-based classification model by providing a more comprehensive representation of the input texts. Our model excelled on PHEME datasets and showed promise on Twitter15 and Twitter16 datasets. As an immediate extension of this model, we will incorporate dual embeddings with a Graph Attention Network (GAT), which differs slightly from Graph Convolutional Networks (GCNs). We can also try with more powerful GPT 3.5 or GPT 4. Additionally, an ensemble model that combines TextGCN and long short-term memory (LSTM) can be implemented for the same purpose.

It is also observed that many datasets such as PHEME, specifically Charlihebdo and Ferguson event suffer from a class imbalance problem. Using synthetic data or data augmentation can help improve accuracy.

Unfortunately, large datasets for rumor detection in English are limited, with the primary option being the Weibo dataset in Chinese, which is inappropriate for English. A large-scale dataset creation may attract researchers.

The accuracy and loss graphs for all models discussed in Table 2 and Subsection 4.4.2 are provided in a separate document accompanying this article (Supplementary material 1).

Author contributions Barsha Pattanaik has contributed to conceptualization, model design and coding, validation, original draft preparation, and formal analysis. Sourav Mandal has contributed to methodology, model design and development, result analysis and comparison, writing, review and editing, supervision, and resource allocation. Rudra M. Tripathy has contributed to conceptualization and review. Arif Ahmed Sekh has done model design and development, formal analysis, and validation.

Data availability This article does not involve data sharing as no datasets were generated or analyzed during the study. All referenced datasets are appropriately cited.

Declarations

Competing interests The authors assert that there are no Competing interests associated with the research presented in this paper.

References

1. Alsaeedi A, Al-Sarem M. Detecting rumors on social media based on a cnn deep learning technique. Ar J Sci Eng. 2020;45(12):10,813-10,844.
2. Anggrainingsih R, Hassan GM, Datta A BERT based classification system for detecting rumours on twitter. CoRR (2021) arXiv:2109.02975
3. Anggrainingsih R, Hassan GM, Datta A CE-BERT: concise and efficient bert-based model for detecting rumors on twitter. IEEE Access 2023:11:80,207–80,217. https://doi.org/10.1109/ACCESS.2023.3299858,
4. Bai N, Meng F, Rui X, et al Rumour detection based on graph convolutional neural net. IEEE Access 2021:9:21,686–21,693. https://doi.org/10.1109/ACCESS.2021.3050563
5. Bian T, Xiao X, Xu T, et al Rumor detection on social media with bi-directional graph convolutional networks. In: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020. AAAI Press, 2020;549–556, https://ojs.aaai.org/index.php/AAAI/article/view/5393
6. Bondielli A, Marcelloni F. A survey on fake news and rumour detection techniques. Inf Sci. 2019;497:38–55. https://doi.org/10.1016/J.INS.2019.05.035.
7. Cai H, Zheng VW, Chang KC. A comprehensive survey of graph embedding: Problems, techniques, and applications. IEEE Trans Knowl Data Eng. 2018;30(9):1616–37. https://doi.org/10.1109/TKDE.2018.2807452.
8. Chen T, Li X, Yin H, et al Call attention to rumors: Deep attention based recurrent neural networks for early rumor detection. In: Trends and Applications in Knowledge Discovery and Data Mining - PAKDD 2018 Workshops, BDASC, BDM, ML4Cyber, PAISI, DaMEMO, Melbourne, VIC, Australia, June 3, 2018, Revised Selected Papers, Lecture Notes in Computer Science. Springer, 2018;11154,40–52, https://doi.org/10.1007/978-3-030-04503-6_4
9. Cheng X Dual-view distilled BERT for sentence embedding. 2021 arXiv:2104.08675,
10. Cui W, Shang M. Kagn:knowledge-powered attention and graph convolutional networks for social media rumor detection. J Big Data. 2023;10(1):45. https://doi.org/10.1186/S40537-023-00725-4.
11. Devlin J, Chang M, Lee K, et al 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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, Volume 1 (Long and Short Papers). Association for Computational Linguistics, 2019;4171–4186, https://doi.org/10.18653/v1/n19-1423
12. Enayet O, El-Beltagy SR Niletmrg at semeval-2017 task 8: Determining rumour and veracity support for rumours on twitter. In: Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, August 3-4, 2017. Association for Computational Linguistics, 2017;470–474, https://doi.org/10.18653/v1/S17-2082
13. Guo M, Xu Z, Liu L, et al An adaptive deep transfer learning model for rumor detection without sufficient identified rumors. Mathematical Problems in Engineering 2020
14. Han J, Kamber M Data Mining: Concepts and Techniques, Second Edition. The Morgan Kaufmann series in data management systems, Elsevier 2006
15. Han SC, Yuan Z, Wang K, et al Understanding graph convolutional networks for text classification. 2022;2203.16060
16. Kehlbeck R, Sevastjanova R, Spinner T, et al Demystifying the embedding space of language models. kopsuni-konstanzde 2021
17. Kipf TN, Welling M Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net, 2017 https://openreview.net/forum?id=SJU4ayYgl
18. Li Q, Zhang Q, Si L Rumor detection by exploiting user credibility information, attention and multi-task learning. In: Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers. Association for Computational Linguistics, 2019;1173–1179, https://doi.org/10.18653/v1/p19-1113
19. Liu B, Guan W, Yang C, et al. Transformer and graph convolutional network for text classification. Int J Comput Intell Syst. 2023;16(1):161. https://doi.org/10.1007/S44196-023-00337-Z.
20. Liu T, Cai Q, Xu C, et al Rumor detection with a novel graph neural network approach. 2024 https://doi.org/10.48550/ARXIV.2403.16206,
21. Liu Y, Jin X, Shen H. Towards early identification of online rumors based on long short-term memory networks. Inf Process Manag. 2019;56(4):1457–67. https://doi.org/10.1016/j.ipm.2018.11.003.

22. Lu Z, Hu X, Xue Y Dual-word embedding model considering syntactic information for cross-domain sentiment classification. Mathematics 2022;10(24). https://doi.org/10.3390/math10244704, https://www.mdpi.com/2227-7390/10/24/4704
23. Ma J, Gao W, Mitra P, et al Detecting rumors from microblogs with recurrent neural networks. In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016. IJCAI/AAAI Press, 2016;3818–3824, http://www.ijcai.org/Abstract/16/537
24. Ma J, Gao W, Wong K Detect rumors in microblog posts using propagation structure via kernel learning. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers. Association for Computational Linguistics, 2017;708–717, https://doi.org/10.18653/v1/P17-1066
25. Ma J, Gao W, Wong K Detect rumor and stance jointly by neural multi-task learning. In: Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018. ACM, 2018a;585–593, https://doi.org/10.1145/3184558.3188729
26. Ma J, Gao W, Wong K Rumor detection on twitter with tree-structured recursive neural networks. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers. Association for Computational Linguistics, 2018b;1980–1989, https://aclanthology.org/P18-1184/
27. Menzil A Graph neural network and some of gnn applications: Everything you need to know. 2023 https://neptune.ai/blog/graph-neural-network-and-some-of-gnn-applications[Accessed: Jan,2024)]
28. Pattanaik B, Mandal S, Tripathy RM. A survey on rumor detection and prevention in social media using deep learning. Knowl Inf Syst. 2023;65(10):3839–80. https://doi.org/10.1007/S10115-023-01902-W.
29. Radford A, Wu J, Child R, et al. Language models are unsupervised multitask learners. OpenAI blog. 2019;1(8):9.
30. Radford A, Kim JW, Hallacy C, et al Learning transferable visual models from natural language supervision. In: Meila M, Zhang T (eds) Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event, Proceedings of Machine Learning Research. PMLR, 2021;139,8748–8763, http://proceedings.mlr.press/v139/radford21a.html
31. Song Y, Chen Y, Chang Y, et al Adversary-aware rumor detection. In: Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021, Findings of ACL, vol ACL/IJCNLP 2021. Association for Computational Linguistics, 2021;1371–1382, https://doi.org/10.18653/v1/2021.findings-acl.118
32. Sun M, Zhang X, Zheng J, et al DDGCN: dual dynamic graph convolutional networks for rumor detection on social media. In: Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022. AAAI Press, 2022;4611–4619, https://doi.org/10.1609/AAAI.V36I4.20385,
33. Tan L, Wang G, Jia F, et al. Research status of deep learning methods for rumor detection. Multim Tools Appl. 2023;82(2):2941–82. https://doi.org/10.1007/S11042-022-12800-8.
34. Tran LH, Tran T, Mai A Text classification problems via BERT embedding method and graph convolutional neural network. 2021 arXiv:2111.15379
35. Tu K, Chen C, Hou C, et al. Rumor2vec: a rumor detection framework with joint text and propagation structure representation learning. Inf Sci. 2021;560:137–51. https://doi.org/10.1016/j.ins.2020.12.080.
36. Varshney D, Vishwakarma DK. A review on rumour prediction and veracity assessment in online social network. Expert Syst Appl. 2021;168(114):208. https://doi.org/10.1016/j.eswa.2020.114208.
37. Wang W, Qiu Y, Xuan S, et al Early rumor detection based on deep recurrent q-learning. Secur Commun Networks 2021:5569,064:1–5569,064:13. https://doi.org/10.1155/2021/5569064
38. Wei L, Hu D, Zhou W, et al Towards propagation uncertainty: Edge-enhanced bayesian graph convolutional networks for rumor detection. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021. Association for Computational Linguistics, 2021;3845–3854, https://doi.org/10.18653/v1/2021.acl-long.297
39. Yang F, Liu Y, Yu X, et al Automatic detection of rumor on sina weibo. In: MDS '12 2012
40. Yao L, Mao C, Luo Y Graph convolutional networks for text classification. In: The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019. AAAI Press, 2019;7370–7377, https://doi.org/10.1609/AAAI.V33I01.33017370,
41. Yu F, Liu Q, Wu S, et al A convolutional approach for misinformation identification. In: Sierra C (ed) Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017. ijcai.org, 2017;3901–3907, https://doi.org/10.24963/IJCAI.2017/545,
42. Zhou M, Liu D, Zheng Y, et al A text sentiment classification model using double word embedding methods. Multim Tools Appl 2022;81(14):18,993–19,012. https://doi.org/10.1007/S11042-020-09846-X,
43. Zubiaga A, Liakata M, Procter R Exploiting context for rumour detection in social media. In: Social Informatics - 9th International Conference, SocInfo 2017, Oxford, UK, September 13-15, 2017, Proceedings, Part I, Lecture Notes in Computer Science, vol 10539. Springer, 2017;109–123, https://doi.org/10.1007/978-3-319-67217-5_8