Journal of Theoretical and Applied Information Technology
30th November 2025. Vol.103. No.22
© Little Lion Scientific

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

BERT-GCN MULTI-TASK MODEL FOR DISASTER DETECTION, CLASSIFICATION, AND HELP IDENTIFICATION IN TWEETS

BASUDEV NATH[1], DEEPAK SAHOO[2], SATYENDR SINGH[3,*], SUNIL KUMAR SHARMA[4], SUDHANSU SHEKHAR PATRA[5,*]

[1] Research Scholar, Faculty of Engineering Technologies, Sri Sri University, Cuttack-754006, Odisha, India
[2]Associate Professor, Faculty of Engineering Technologies, Sri Sri University, Cuttack-754006, Odisha, India
[3]Assistant Professor, Department of Computer Science & Engineering, BML Munjal University, Gurugram, India
[4]Doctorate Level Scholar, Arizona State University, United States
[5]Associate Professor, School of Computer Applications, Kalinga Institute of Industrial Technology (KIIT) Deemed to be University, Bhubaneswar-751024, Odisha, India

E-mail: [1]nathbasudev@gmail.com, [2]deepak.s@srisriuniversity.edu.in, [3]satyendr@gmail.com, [4]sshar259@asu.edu, [5]sudhanshupatra@gmail.com

ABSTRACT

X, which used to be called Twitter, has become an important way to share information during natural disasters in this age of real-time digital contact. It is still very hard to get useful information from the huge amounts of social media data that are available. Our study suggests a multi-stage integrated deep learning models that uses both contextual and relational features to effectively sort and analyze tweets about disasters. In the first task, our model uses binary classification to figure out with 96.58% accuracy if a tweet is disaster related or not. A multi-class categorization of tweets into three categories—non-disaster, flood, and earthquake—is performed in the second task. The accuracy of this classification is 92.65%, 96.30%, and 95.48%, respectively. The third and most focused task is to identify specific support needs in earthquake-related tweets, such as calls for food and medical help, with an accuracy of 95.07%. Graph Convolutional Networks (GCNs) are used to learn the relationships between words, and BERT (Bidirectional Encoder Representations from Transformers) is used to get deep contextual meanings. The combined embeddings are passed into a fully connected neural network (FCNN) for classification. Experimental findings show that our hybrid approach not only works well across all tasks but also has practical relevance in improving real-time catastrophe response and focused humanitarian relief distribution.
Keywords: *Tweeter Analysis, BERT, GCN, NLP, FCNN, Tweeter Help Categorization*

1. INTRODUCTION

In recent years, the increasing frequency and intensity of natural disasters, such as floods, earthquakes, and wildfires, have raised considerable difficulties to emergency response and resource management. These issues have been brought about by several factors. During times of crisis, social media platforms such as Twitter have emerged as crucial channels for the dissemination of information in real time. People post about damage, needs, and ongoing relief efforts. On the other hand, authorities have a tough time extracting relevant insights in a timely manner due to the sheer number of these tweets and the fact that they are not organized in any way [1-3].

There has been a significant amount of investigation into the classification of tweets via the use of conventional machine learning techniques, such as Support Vector Machines (SVM) and Random Forests, in conjunction with statistical characteristics, such as TF-IDF [5 -15]. In spite of the fact that these techniques provide the groundwork, they typically fail to capture the contextual complexity and related patterns that are inherent in natural language. This is especially true in situations that include brief text, such as tweets. A number of deep learning models, including RNN [18], CNNs [19, 20], LSTMs, and transformers such as BERT [21], have shown improved performance since that time by using semantic understanding [16,17] [22]. However, even these very efficient algorithms often overlook word-structure links, which might be useful for comprehending and classifying tweets.

Even with these improvements, it is still very hard to accurately sort tweets about disasters. Tweets are hard to understand because they are short, loud, and don't give much context. Most current methods use either semantic context (like BERT) or structural relationships (like GCN), but not both. This gap leads to incomplete representations and makes current classification models less effective. The main issue this study looks at is how to make a unified framework that combines semantic and structural information in a way that improves the classification of disaster tweets and helps with quick crisis response.

This study investigates whether the integration of BERT and GCN enhances disaster tweet classification (relevance, type, and urgency) in comparison to models utilizing only BERT or only GCN. We believe that the hybrid BERT+GCN will surpass these benchmarks by utilizing both semantic and structural information. Section 4.1 presents an ablation analysis that supports this hypothesis.

We propose a hybrid deep learning strategy that merges the conceptual powers of BERT with the relational modelling skills of Graph Convolutional Networks (GCNs) in order to bridge this gap. Following the preprocessing of tweets and the transformation of those tweets into contextual embeddings using BERT, our procedure begins. Meanwhile, a word co-occurrence graph that is based on Normalized Pointwise Mutual Information (NPMI) is being constructed in order to capture important word connections that are present across the whole corpus. This graph is being developed in order to capture the importance of these interactions. This graph is one of the inputs that the GCN gets. The GCN is responsible for learning structural patterns and connections between words. In order to create a uniform feature representation, the embeddings from BERT and GCN are integrated and connected together. After that, the representation is sent to a fully connected neural network (FCNN) for the purpose of classification. There are three distinct tasks that our model is evaluated on.

Task1: To create and assess a system that automatically sorts tweets into two categories: related to disasters or not related to disasters.

Task 2. Categorize tweets into specific disaster types (non-disaster, flood, and earthquake)

Task 3. Identification of support requirements in tweets connected to earthquakes

By using this multi-stage pipeline, we are able to accomplish both broad classification and specific determination of support requirements. The test results show that our mixed BERT-GCN model is a strong and scalable way to look at disaster tweets in real time and respond to emergencies.

Organization.

The subsequent sections of our work are structured as follows: Section 2 delineates pertinent research in the domain of catastrophe tweet categorization. Section 3 delineates the proposed hybrid framework, including data preprocessing, graph generation using NPMI, and the BERT- GCN architecture. Section 4 presents the outcomes for each categorization task and including a performance evaluation. Section 5 finishes with a summary of the results and prospective research directions.

2. RELATED WORK

Our study intends to solve many issues in catastrophe tweet categorization and analysis. We first look at current studies on how Twitter data is processed and categorized during crisis events to provide a strong basis for our suggested method. We look at research on the categorization of tweets as disaster-related or not, the identification of certain catastrophe kinds, and the detection of help-related material within those tweets. Key techniques, models, and issues mentioned in prior studies are described in this part; they provide the foundation and inspiration for our multi-task framework.

Disaster-related tweets have been found in many research using traditional machine learning techniques like Support Vector Machines (SVM), Naive Bayes, and Conditional Random Fields (CRFs). Stowe et al. [4], for example, used an SVM with characteristics like unigrams, bigrams, part-of-speech tags, tweet timestamps, and retweet signals to categorize tweets during catastrophic situations. Focusing on situational awareness, Verma et al. [28] used maximum entropy classifiers and Naive

Bayes. Their feature set included lexical and stylistic elements like as tone, register, and subjectivity. Imran et al. [27] first filtered relevant tweets using a Naive Bayes model; then, they employed CRFs to derive practical insights. Although somewhat efficient, CRFs may be costly in terms of computation. Other conventional models have used manually created characteristics including tweet length, mentions, and hashtags [4], [26, 27], TF-IDF [30], and Bag-of-Words [29]. Though useful, these methods can experience performance decline on changing datasets because of their dependence on static training data distributions. Their sparse and high-dimensional character also creates scaling problems. Also, these old methods don't always work well with new or changing tweet patterns that happen in real time during a crisis.

To overcome these limitations, researchers have used deep learning architectures that can learn semantic features automatically to circumvent the constraints of human feature engineering. Disaster-related tweets have been classified using CNNs [31] and LSTMs [32]. Among the first to use CNNs for this goal, Caragea et al. [44] showed better results in finding useful tweets. Combining CNN's capacity to collect spatial information with LSTM's capability in modelling long-term dependencies, Bhoi et al. [45] put out a hybrid CNN-LSTM model. Their findings indicated that this mix surpassed conventional baselines. While deep learning models are great at capturing intricate language patterns, they need constant guidance in order to predict interdependencies between tweets or take use of the structural linkages between ideas.
Recently, transformer-based architectures, particularly BERT [21], have gained popularity because to their superior contextual comprehension. Liu et al. [46] created CrisisBERT for classifying tweets during crisis circumstances, using a compact form of BERT known as DistilBERT. BERT effectively represents word order and local context by integrating transformer encoders with multi-head self-attention mechanisms. Despite its efficacy, BERT may struggle to convey structural or relational information within the text, since it mostly emphasizes individual sequence modelling and lacks a way to capture higher-order co-occurrence patterns across the corpus.
In recent years, Graph Neural Networks (GNNs) provide novel methods for describing this type of related data [33–35]. Now, they are employed in NLP for tasks such as machine translation [37], naming sequences [36], and text classification [23]. One of the earliest significant works in this field was TextGCN by Yao et al. [24], which generated a complex graph with nodes representing texts and words and connections representing PMI and TF-IDF scores. The entire dataset (train and test) is required to construct the graph, as this method is transductive. SHINE, a model developed by Yaqing Wang et al. [25], is a hierarchical heterogeneous graph that learns to characterize documents by sharing graphs and assembling brief texts based on POS markers, entities, and words. In real-time environments, where new tweets are continuously received, the applicability of these models is restricted by their dependence on transductive learning.
For better generalization, many models have looked into inductive learning. TEXTING, by Zhang et al. [38], makes a graph for each short text by using co-occurrence relationships in a moving window and learned word representations to make sentence-level embeddings. Kunze Wang et al. [47] created InduT-GCN, which uses only training data statistics to build the graph and one-directional GCN transmission to guess the names of the classes. In their paper [42], Ye L et al. described STGCN, a transductive corpus-level model that combines document topics, word embeddings (from BiLSTM), and document structures into a single graph that can be used for classification. Despite the fact that these models have the potential to be useful, they often only handle a single goal, which might be either categorization or structural modelling. They do not extend to multifaceted, real-world difficulties, such as concurrently detecting different sorts of disasters and the requirements that are connected with them.

GATs are more advanced than GCNs because they have attention systems that learn how to adjust the edge weights between nodes. In their paper [39], Veličković et al. described the GAT model, which changes how nodes are represented by using self-attention between peers. Hu et al. [40] suggested HGAT, which works on a corpus-level heterogeneous graph with papers, topics, and entities and uses a dual-level attention method. Liu et al. [41] used attention spread to find secondary neighbour context and node-level attention to improve how documents are shown. Haitao et al. [43] made a model that predicts text groups using LSTM and GAT and builds text graphs based on dependency relations. Even though they are new, most GAT-based models only work on document-level tasks and don't consider the unique problems of tweet classification, especially when there is a lot of noise and the text is short. Our suggested model brings together contextual and structure learning

methods for classifying short texts, especially when it comes to analysing tweets about disasters. Transductive models like TextGCN and STGCN need access to the whole collection, including test data, in order to build a graph. Our model, on the other hand, uses an inductive method that lets it generalize to tweets that haven't been seen yet, which makes it better for real-time applications. Moreover, previous GNN-based models frequently use basic encoders like LSTM or don't include any deep semantic understanding at all. Our system, on the other hand, uses BERT to record rich contextual embeddings. Our model can use both lexical and structural links in text because it combines BERT with a GCN that is based on an NPMI-generated word co-occurrence graph. The suggested framework is also made to handle both binary and multi-class disaster classification, as well as finding specific support needs in tweets about earthquakes, which is an area that hasn't been looked into much in the past. This makes our model a unique, multitasking, and easily usable way to use social media data for real-time emergency reaction.
The Table 1 outlines key models used in catastrophe tweet categorization, emphasizing their advantages and drawbacks. It provides a comparative analysis of classical, deep learning, and graph-based methodologies, positioning our proposed BERT-GCN framework as a holistic approach that addresses both semantic and structural issues in this field.

This study is advantageous for both researchers and practitioners. For the academic community, it shows how important it is to use both semantic and structural learning to classify short texts. Our proposed model offers emergency response agencies a scalable method to pinpoint critical information in real time, facilitating swifter and more efficient crisis management.

3. PROPOSED MODEL DESCRIPTION

Our suggested hybrid deep learning architecture, shown in Figure 1, is meant to use a multi-task learning framework to sort tweets about disasters. The model uses structural embeddings made using a Graph Convolutional Network (GCN) based on word co-occurrence graphs constructed with NPMI and contextual representations taken from a pre-trained BERT model. The whole procedure starts with collecting and cleaning up tweets, and then it uses BERT and GCN to extract two sets of features. We combine these two representations to make a resulting vector, which we then pass through a Fully Connected Neural Network (FCNN) to get the final classification. The categorization goes through three steps in order: figuring out how relevant the catastrophe is, figuring out what kind of disaster it is, and finding tweets about earthquakes that are connected to assistance. The following subsections provide a detailed description of each part and function of the architecture.

3.1. Collection of tweets / Dataset Description

This study utilized tweet data sourced from two established platforms—CrisisNLP and CrisisLexT6—both of which offer labelled tweets pertaining to actual disaster events. A dataset comprising 33,370 tweets was prepared by integrating these sources. This dataset encompasses tweets pertaining to disasters as well as general tweets that are not associated with any crisis. The "Data Availability" section of this paper contains hyperlinks to the primary data sources.

In Task 1, binary classification was performed on the complete dataset. Tweets were categorized into two distinct classes: disaster-related (Target = 1) and non-disaster (Target = 0) as seen in Figure 2. The dataset exhibited a near balance, with 52.2% of tweets categorized as disaster and 47.8% classified as non-disaster. Figure 3 presents a visual summary of this distribution.

In Task 2, a refined dataset comprising 15,435 tweets was generated, derived from the original collection. The dataset was categorized into three distinct groups: non-disaster (4,783 tweets), flood-related (4,873 tweets), and earthquake-related (5,779 tweets). This procedure enabled the categorization of tweets according to distinct disaster types. Figures 4 and 5 illustrate the structure and balance of the dataset.

Task 3 concentrated exclusively on the 5,779 tweets related to earthquakes that were identified in Task 2. The tweets were systematically categorized according to the type of assistance they represent, including requests for food, medical aid, infrastructure support, emotional support, and other pertinent information. Figure 6 provides a summary of the final dataset, while Figure 7 depicts the distribution of these help requests across various categories.

All tweets were subjected to identical preprocessing procedures: text normalization, punctuation removal, word tokenization, and stop word filtering. The datasets utilized for Tasks 1 and 2 comprised three fields: Tweet ID, Tweet text, and Target label. Task 3 incorporated an additional column labeled "Help," which specifies the type of assistance referenced in the tweet.

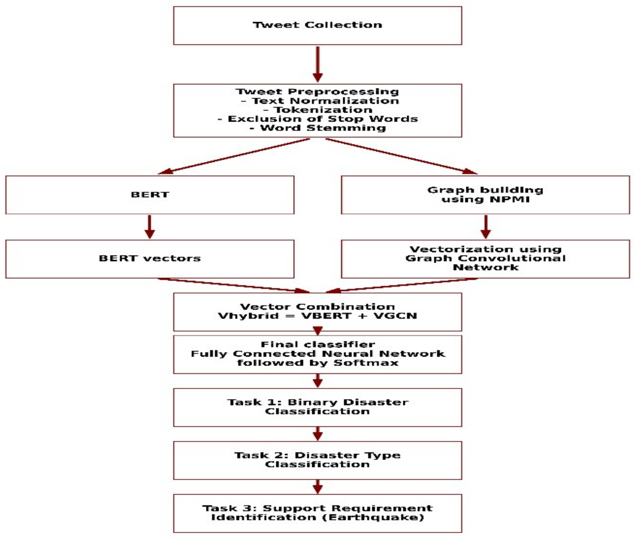
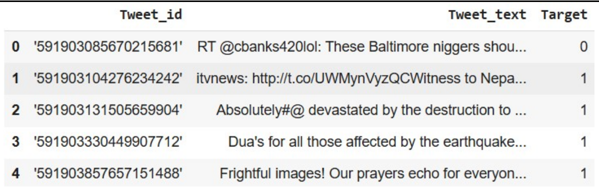

Tweet Collection
Tweet Preprocessing
- Text Normalization
- Tokenization
- Exclusion of Stop Words
- Word Stemming
BERT
Graph building using NPMI
BERT vectors
Vectorization using Graph Convolutional Network
Vector Combination
Vhybrid = VBERT + VGCN
Final classifier
Fully Connected Neural Network followed by Softmax
Task 1: Binary Disaster Classification
Task 2: Disaster Type Classification
Task 3: Support Requirement Identification (Earthquake)

Figure 1: Proposed BERT-GCN Hybrid Model Architecture

Table 1: Comparative Summary of Existing Approaches for Disaster Tweet Classification

Category	**Model**	**Strengths**	**Limitations**
Traditional ML	SVM, Naive Bayes, CRF [4], [27], [28], [29], [30]	Basic tweet categorization using n-grams, POS tags, retweet signals, and TF-IDF.	Sparse, high-dimensional features; poor adaptability to informal, real-time tweet streams.
Deep Learning	CNN, LSTM, CNN-LSTM [31], [32], [44], [45]	Automatically learns semantic features; no feature engineering required.	Unable to recreate inter-tweet or corpus-level structural connections.
Transformers	BERT, CrisisBERT [21], [46]	Self-attention and multi-head encoding capture rich contextual semantics	Neglects structural or co-occurrence correlations across tweets or the corpus.
Graph Neural Networks	TextGCN, SHINE, STGCN [24], [25], [42]	Learns corpus-wide structural word-document correlations	Transductive; needs whole dataset (train + test); not scalable for real-time classification.
Inductive GNN	TEXTING, InduT-GCN [38], [47]	Creates document graphs using word co-occurrence; enables unseen input during inference.	Not intended for multitasking like classification + support detection.
Graph Attention Networks	GAT, HGAT, STGAT [39], [40], [41], [43]	Gives neighbours learnable attention weights; captures more information during message passing	Best for document-level tasks, not tweets or other short, noisy material.
Our work	Bert + GCN (Inductive, NPMI-based Graph	Integrates BERT's semantic context with GCN's structural learning; facilitates binary, multi-class, and support detection tasks.	Needs two embedding processes and preprocessing, which adds a considerable amount of computational cost.



	Tweet_id	Tweet_text	Target
0	'591903085670215681'	RT @cbanks420lol: These Baltimore niggers shou...	0
1	'591903104276234242'	itvnews: http://t.co/UWMynVyzQCWitness to Nepa...	1
2	'591903131505659904'	Absolutely#@ devastated by the destruction to ...	1
3	'591903330449907712'	Dua's for all those affected by the earthquake...	1
4	'591903857657151488'	Frightful images! Our prayers echo for everyon...	1

Figure 2: The tweet dataset used in task 1

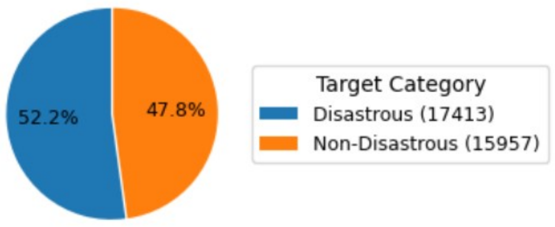
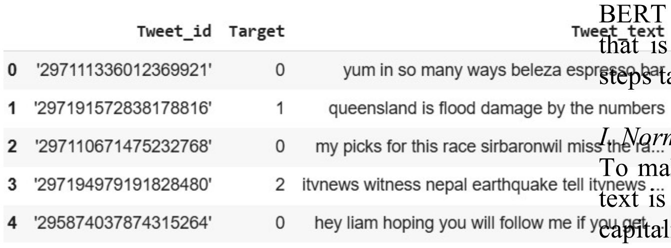


Figure 3: Target Distribution of classes



	Tweet_id	Target	Tweet_text
0	'297111336012369921'	0	yum in so many ways beleza espresso bar
1	'297191572838178816'	1	queensland is flood damage by the numbers
2	'297110671475232768'	0	my picks for this race sirbaronwil miss the ra...
3	'297194979191828480'	2	itvnews witness nepal earthquake tell itvnews ...
4	'295874037874315264'	0	hey liam hoping you will follow me if you get...

Figure 4: The tweet dataset used in task 2

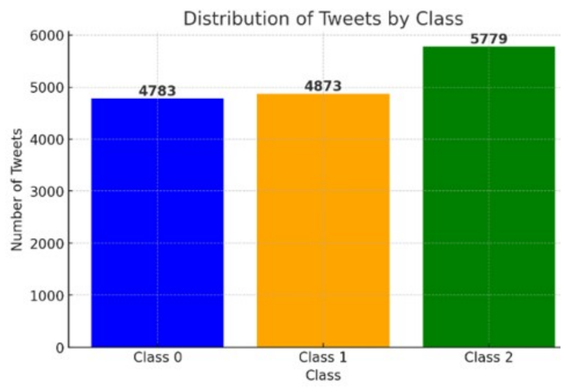
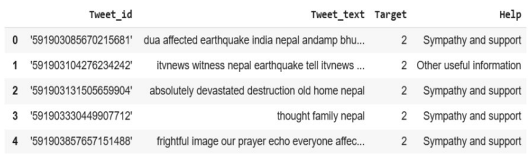


Figure 5: Distribution of Tweets by Class



	Tweet_id	Tweet_text	Target	Help
0	'591903085670215681'	dua affected earthquake india nepal andamp bhu...	2	Sympathy and support
1	'591903104276234242'	itvnews witness nepal earthquake tell itvnews ...	2	Other useful information
2	'591903131505659904'	absolutely devastated destruction old home nepal	2	Sympathy and support
3	'591903330449907712'	thought family nepal	2	Sympathy and support
4	'591903857657151488'	frightful image our prayer echo everyone affec...	2	Sympathy and support

Figure 6: Tweets about earthquake in Task 3

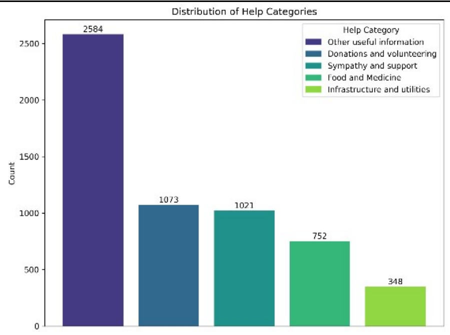


Figure 7: Histogram of help groups in earthquake-related tweets

3.2. Pre-processing of Tweets

All tweets go through a standard preparation workflow before they are used to train our model. This improves the quality of the data and makes sure that all tweets are represented the same way. This step is very important for both making the BERT encoding and the word co-occurrence graph that is used in the GCN. There are the following steps taken:

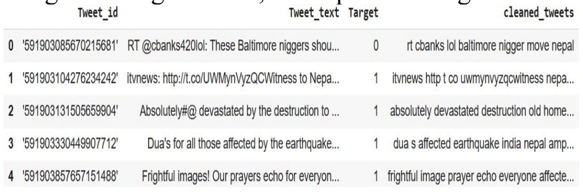
I. Normalizing the text
To make sure everything looks the same, all tweet text is changed to lowercase. This ensures that the capitalization of the same term does not cause any differences in its treatment. Special letters and punctuation signs (like #, @, etc.) are taken out because they don't usually add to the meaning of our tasks.

II. Tokenization
The text of the tweet is broken up into separate words, or "tokens." This makes it easier to use text as input patterns for both BERT and building graphs.

III. Stop Word Removal
Common words like "and," "the," and "is" that don't add much sense are taken out using a list of stop words that has already been made.

IV. Word Stemming
Stemming methods take words back to their base or root form, like "helping" becoming "help." This cuts down on repetition and groups things that are linked together. To ensure that the input data is clean, consistent, and acceptable for further learning, several preprocessing processes are carried out. This enhanced textual input is essential to the implementation of both the BERT-based contextual encoding and the GCN-based graph learning in order to produce useful features.

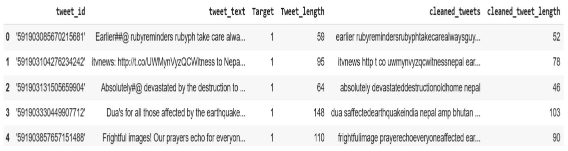
The tweets and their corresponding cleansed versions, which have been generated with designated target values, are depicted in Figure 8.



	Tweet_id	Tweet_text	Target	cleaned_tweets
0	'591903085670215681'	RT @cbanks420lol: These Baltimore niggers shou...	0	rt cbanks lol baltimore nigger move nepal
1	'591903104276234242'	itvnews: http://t.co/UWMynVyzQCWitness to Nepa...	1	itvnews http t co uwmynvyzqcwitness nepa...
2	'591903131505659904'	Absolutely#@ devastated by the destruction to ...	1	absolutely devastated destruction old home...
3	'591903330449907712'	Dua's for all those affected by the earthquake...	1	dua s affected earthquake india nepal amp...
4	'591903857657151488'	Frightful images! Our prayers echo for everyon...	1	frightful image prayer echo everyone affecte...

Figure 8: Tweets with cleaned tweets

Figure 9 illustrates a comparison of tweets before and after the cleaning procedure. The changes in text and length are shown.



	tweet_id	tweet_text	Target	Tweet_length	cleaned_tweets	cleaned_tweet_length
0	'591903085670215681'	Earlier##@ rubyreminders rubyph take care alwa...	1	59	earlier rubyremindersrubyphtakecarealwaysguy...	52
1	'591903104276234242'	itvnews: http://t.co/UWMynVyzQCWitness to Nepa...	1	95	itvnews http t co uwmynvyzqcwitnessnepal ear...	78
2	'591903131505659904'	Absolutely#@ devastated by the destruction to ...	1	64	absolutely devastateddestructionoldhome nepal	46
3	'591903330449907712'	Dua's for all those affected by the earthquake...	1	148	dua saffectedearthquakeindia nepal amp bhutan ...	103
4	'591903857657151488'	Frightful images! Our prayers echo for everyon...	1	110	frightfulimage prayerechoeveryoneaffected ear...	90

Figure 9: Length of tweets before and after cleaning

Figure 10 shows how cleaning affects tweet length. Most data points are below the y = x line, indicating tweet shortening after preprocessing. The continuously decreased cleaned_tweet_length values compared to Tweet_length explain the length reduction. Eliminating stop words, links, and special characters shortens tweets.

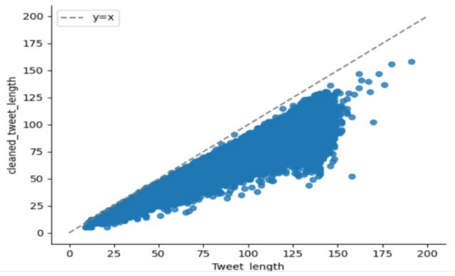


Figure 10: Influence of Pre-Processing on Tweet Lengths

3.3. Extraction of contextual features using BERT

We utilize BERT-base (Bidirectional Encoder Representations from Transformers) to acquire a comprehensive semantic representation of tweets, since it is a pre trained transformer-based language model that collects contextual information via bidirectional text processing. BERT is especially effective at processing short and informal messages, such as tweets, due to its ability to model word dependencies in both forward and backward directions simultaneously. Each tweet, after-preprocessing, is tokenized into a series of tokens and then enhanced with special tokens [CLS] and [SEP], which denote the beginning and end of the sequence, respectively. The complete tokenized input for a tweet is denoted as:

$$\mathbf{T = [[CLS]}, t_1, t_2, t_3 \cdots t_n \mathbf{[SEP]]} \qquad (1)$$

Where t_i represents the $i-th$ token in the tweet and [CLS] is used to identify the start of the sequence, and [SEP] is used to identify the end of the sequence. For each token t_i, an input embedding x_i is derived by aggregating three different types of embeddings:

1. **Token embeddings** $E_{token}(t_i)$
2. **Segment embeddings** $E_{segment}(t_i)$
3. **Position embeddings** $E_{position}(t_i)$.

The final embedding for token t_i is:

$$x_i = E_{token}(t_i) + E_{segment}(t_i) + E_{position}(t_i) \qquad (2)$$

These embeddings are stacked into the input matrix X:

$$X = [x_0, x_1, x_2 \cdots x_{n+1}] \in R^{(n+2)\times d} \qquad (3)$$

where d is the model's hidden dimension, which is usually 768 in BERT-base and n is the number of tokens in the tweet. This matrix X is then passed through a total of 12 transformer encoder layers, each of which is comprised of multi-head self-attention mechanism and a feed-forward network. In each layer, the input is first transformed into Query (Q), Key (K), and Value (V) representations by the application of learnt weight matrices:

$$Q = XW^Q,\ K = XW^K,\ V = XW^V \qquad (4)$$

where W^Q, W^K, $W^V \in R^{d\times d_k}$ are learned projection matrices and d_k represents dimension of each attention head size. The attention weights are calculated with scaled dot-product attention:

$$\textbf{Attention (Q, K, V) = SoftMax} \left(\frac{QK^T}{\sqrt{d_k}}\right)V \qquad (5)$$

This method allows the model to assess the significance of each word in relation to every other word in the sequence. BERT utilizes multiple attention heads concurrently to capture diverse contextual features. The outputs from all attention heads are concatenated and then processed by an output projection matrix $W^O \in R^{hd_k \times d}$ where h

denotes the number of attention heads. This yields the outcome of multi-head attention:

$$MultiHead(X)=Concat(head_1\ldots,\ \ head_h)W^O \tag{6}$$

The multi-head attention output is then passed through a feed-forward neural network, with a residual connection and layer normalization included to enhance training stability. The hidden state of the $i\text{-}th$ token in the $l\text{-}\ th$ layer is modified utilizing:

$$h_i^{(l)} = LayerNorm\ (h_i^{(l-1)}) + \text{FFN}\ (MultiHead\ (h_i^{(l-1)}))) \tag{7}$$

with the initial hidden state set as $h_i^{(0)} = x_i$. After going through all 12 levels, we have the final contextualized token representations:

$$\mathbf{H} = [\ h_0^{(L)}\ ,\ h_1^{(L)}\ \ldots\ldots\ldots\ h_{n+1}^{(L)}\ ] \tag{8}$$

We take the last hidden state that goes with the [CLS] token to make a fixed-length vector that shows the whole tweet is represented as:

$$V_{BERT} = h_0^{(L)} \in R^d \tag{9}$$

This vector encapsulates the tweet's contextual significance and semantic attributes, serving as the contextual representation in the subsequent phase of our model, where it integrates with graph-based structural embeddings generated by the GCN.

3.4. Construction of graphs employing NPMI

Our suggested methodology analyses twitter content using two concurrent modules—BERT and a graph-based system—to concurrently collect semantic and structural attributes. To create the input for the Graph Convolutional Network (GCN), we develop a word co-occurrence graph, whereby nodes signify important words and edges reflect statistically significant co-occurrence associations extracted from the twitter corpus.

The procedure starts with text preprocessing, including normalization, stop word elimination, stemming, and tokenization. To ensure the graph consists only significant words, we use TF-IDF filtering and exclude tokens with excessively high or low document frequencies, which often introduce noise. Figure 11 illustrates the top 20 phrases ranked by both TF-IDF score and raw frequency, offering insight into essential disaster-related keywords derived from the twitter corpus.

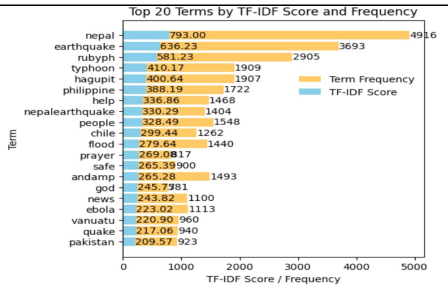


Figure 11: Distribution of TF-IDF Terms in Disaster Tweets

After the completion of the TF-IDF filtering process, the next step involves analyzing the strength of the connection between each pair of words. We use NPMI, or Normalized Pointwise Mutual Information, for this. NPMI is a statistical measure that shows us how probable it is that two words would come together in the same context, in comparison to what would be predicted by chance. When compared to basic co-occurrence counts, NPMI takes word frequency into account and prioritizes true word relationships. The calculation for NPMI between two terms, ωi and ωj is as follows:

$$\textbf{NPMI}\ (\omega_{i,}\omega_j) = \frac{log\left(\frac{P(\omega_{i,}\omega_j)}{p(\omega_i)P(\omega_j)}\right)}{-log\ p(\omega_{i,}\omega_j)} \tag{10}$$

In this case, $p(\omega_i)$ and $p(\omega_j)$ are the marginal probabilities of each word, while $p(\omega_i, \omega_j)$ is the joint probability of the pair happening inside a sliding window of size 5 throughout the tweet corpus. To figure out these probabilities, we divide the counts of co-occurrences or individual terms by the total number of windows that were scanned.
The range of possible NPMI scores is from -1 to +1, where: +1 indicates that the two words consistently co-occur. A value of 0 indicates that the occurrences are random and do not exhibit any correlation. -1 indicates that they do not co-occur. The specified range enhances the use of NPMI for constructing a word-level graph. This process enables the retention of only the most strong and significant connections between words while eliminating weak or extraneous links. In order to accomplish this, a threshold T is established, and an edge is created between two words solely if their NPMI score exceeds T. The adjacency matrix is characterized as follows:

$$A_{ij} = \begin{cases} 1, & if\ NPMI\ (\omega_{i,}\omega_j) > T \\ 0, & \text{Otherwise} \end{cases} \tag{11}$$

Our Experimental results indicate that a threshold value of T = 0.2 provides the optimal balance.

Lower values, such as 0.1, lead to graphs that exhibit excessive density and incorporate numerous irrelevant connections. Conversely, higher values, such as 0.3 or above, eliminate an excessive number of significant links, resulting in a graph that is overly sparse. The selected value maintains a compact and informative graph structure.

The final word co-occurrence graph is characterized as undirected, featuring nodes that represent significant words, while the edges illustrate strong connections among them. Figure12. illustrates the resulting graph, with red nodes denoting words that are prevalent in disaster-related tweets, while purple nodes signify words identified in non-disaster tweets.

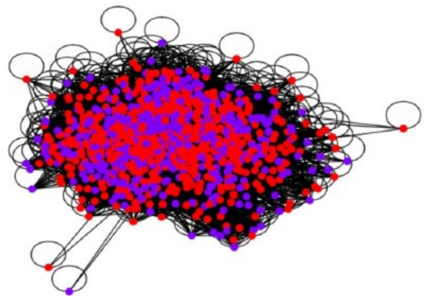


Figure 12: Sparse Word Graph After NPMI Thresholding

We use a contextual word embedding from BERT to set up each node in the network. We find the average of the contextual embeddings for each unique word node ω_i over all tweets that include it. These embeddings are created by BERT's last encoder layer. This form is the first node feature vector, or xi. A two-layer GCN then processes these node embeddings. It learns relational properties depending on the topology of the graph. The graph module adds to the sentence-level semantic representation from BERT by capturing word dependencies at the corpus level. By modelling both lexical semantics and word relationships together, this hybrid architecture makes the model better at recognizing information connected to disasters.

3.5. Vectorization using Graph Convolutional Networks (GCN)

We build a word co-occurrence graph $G = (V, E)$ as shown in Section 3.3. In this graph, nodes $v_i \in$ V represent important words that have been filtered using TF-IDF, while edges $e_{ij} \in$ E show statistically significant co-occurrences based on NPMI. We start each word node with a 768-dimensional contextual embedding from BERT. This is done by averaging the final-layer token representations of that word over all tweets where it occurs.

The input feature matrix $X \in R^{n\times d}$ is made up of these BERT-initialized node embeddings. Here, n is the number of nodes and d=768 represents the dimensionality of the embedding. Prior to inputting the graph into the GCN, we initially compute the symmetrically normalized adjacency matrix $\hat{A}$ to help with neighbourhood aggregation during graph convolution. The symmetric normalization formula is given as follows:

$$\hat{A} = \widetilde{D}^{-1/2}(\mathrm{A+I})\, \widetilde{D}^{-1/2} \quad (12)$$

In this context, A represents the binary adjacency matrix derived from the co-occurrence graph, I denotes the identity matrix that incorporates self-loops (thereby preserving a node's own features throughout aggregation), and $\widetilde{D}$ signifies the degree matrix, where $\widetilde{D}_i = \sum_j (A_{i,j} + I_{i,j})$

To get structural embeddings, we use a Graph Convolutional Network (GCN) with two layers. The following is the propagation rule for each layer:

$$H^{l+1} = \sigma\left(\hat{A}H^{(l)}W^{(l)}\right) \quad (13)$$

Where:

$H^{(l)}$ is the node feature matrix at the l-th layer

$W^{(l)} \in R^{d\times d'}$ is a trainable weight matrix for layer l

σ is a non-linear activation function such as ReLU

In the input layer, we establish $H^{(0)} = X$, indicating that the initial node features correspond to the BERT embeddings. Following two GCN layers, the ultimate representation of each node is denoted as $H^{(2)} \in R^{n\times d'}$. where we set d′=128.

The message-passing mechanism is shown in Figure 13, whereby each node aggregates information from its adjacent neighbors to update its representation. For instance, node A aggregates feature from adjacent nodes B, C, and D, facilitating the model's acquisition of more nuanced structural embeddings. The ultimate output from the GCN functions as a structure-enhanced embedding space. These embeddings are then integrated with sentence-level BERT representations in the following phase of our hybrid model. This integration allows the model to use both the global graph structure and the local semantic meaning, hence improving classification efficacy across all tasks.

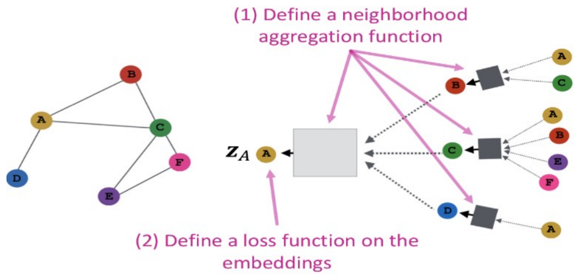

(1) Define a neighborhood aggregation function
(2) Define a loss function on the embeddings
z_A
A
B
C
D
E
F

Figure 13: Graph Convolutional Network (GCN) for Word Embedding [48]

3.6. Vector Combination (Feature Fusion)

We combine the BERT-based and GCN-based tweet representations into a single hybrid vector to fully use both local semantic meaning and global structural context. From Section 3.3, we derive the tweet-level semantic embedding $V_{BERT} \in R^{768}$ from the [CLS] token of BERT. According to Section 3.4, the GCN delivers revised word-level embeddings $hi \in \mathbb{R}^{128}$. To calculate the structural embedding of a tweet including m words $\{\omega_1, \omega_2, \omega_3 \cdots \omega_m\}$, we average the GCN outputs of these words:

$$V_{GCN} = \frac{1}{m}\sum_{i=1}^{m} h_i \tag{14}$$

In our implementation, we experimentally specified the GCN output dimension as $d2 = 128$, yielding a fixed-size tweet representation $V_{GCN} \in R^{128}$. This vector encapsulates the graph-based structural links among the words in the tweet and is then integrated with the BERT-based contextual representation for final categorization. The two embeddings V_{BERT} and V_{GCN} are concatenated to create a hybrid vector:

$$V_{hybrid} = [\, V_{BERT} \| V_{GCN}] \tag{15}$$

We use a linear projection and a ReLU activation to minimize dimensionality and match this concatenated vector with the final classifier's input requirements:

$$V_{proj} = \text{ReLU}\,(W_p \cdot V_{hybrid} + b_p) \tag{16}$$

where $W_p \in R^{512\times896}$ is a trainable weight matrix, and $b_p \in R^{512}$ is a bias vector. This yields a fixed-size vector $V_{proj} \in R^{512}$, which effectively encapsulates both contextual and structural information. The projected vector is then input into a fully connected neural network for classification in the model's final step.

3.7. Classification by Fully Connected Neural Network

We combine the contextual features from BERT and the structural features from GCN into one hybrid vector $V_{proj} \in R^{512}$. Our Fully Connected Neural Network (FCNN), which acts as the last classifier for tweet classification, then receives this merged representation. The FCNN is composed of one or more dense layers, concluding with an output layer that features a number of neurons equivalent to the number of target classes associated with the specific task. In our model, to separate "non-disaster" tweets from "disaster" tweets for Task 1 we use an output layer with two neurons. In Task 2 comprises three neurons in the output layer, which correspond to "non-disaster," "flood," and "earthquake." In Task 3 the output layer consists of 5 neurons, with each neuron corresponding to a distinct support category: food, medical, infrastructure, emotional support, and other. The FCNN performs a linear modification of the input vector, then using a SoftMax activation to derive class probabilities:

$$z = W_c \,.V_{proj} + b_c \tag{17}$$

$$\hat{y} = \text{softmax}(z) \tag{18}$$

Where:
$W_c \in R^{C\times512}$ is the learned weight matrix of the classifier,
b_c is the bias vector,
C represents the number of output classes relevant to the specific activity,
$\hat{y} \in R^{C}$ denotes the projected probability distribution over the classes

In order to make the output understandable as a probability distribution, the SoftMax function ensures that the total of the predicted probabilities across all classes is equal to one. For the purpose of making the ultimate forecast, the category that has the greatest probability is chosen.
In order to train this classifier, we make use of the categorical cross-entropy loss function, which measures the degree to which the predicted probabilities differ from the actual class label:

$$L = \sum_{i=1}^{C} y_i \, log(\hat{y}) \tag{19}$$

$y_i \in \{0, 1\}$ signifies the actual label for class i, represented by one-hot encoding. This indicates that only the actual class has a value of 1, while all other classes are assigned a value of 0. y_i represents the model's estimated probability for class i.

All trainable parameters of the model, encompassing the FCNN weights, GCN layers, and BERT layers, are optimized using the Adam optimizer and mini-batch backpropagation. To enhance training efficiency and reduce overfitting, regularization methods like as dropout and batch normalization are used between layers.

3.8. Multi task classification

In the last part of our model, we set up a sequential multi-task classification process. This means that each task is done one at a time, depending on how the previous one went. We don't look at tweet classification as a single problem; instead, we break it up into three subtasks that depend on each other rationally. This organized layout not only imitates how humans think, but it also makes the model more accurate and efficient by focusing more on one thing at a time.

Task 1: Binary Disaster Classification

The first task is for the model to use binary classification to figure out whether a tweet is about a disaster or not. Tweets that are not about disasters are discarded, while tweets that are about disasters go on to the next step for further examination.

Task 2: Disaster Type Classification

After then, tweets that are tagged as being about a catastrophe are sent to a multi-class classifier that sorts them by kind of disaster. We look at three types of catastrophes in this paper: earthquakes, floods, and others. This level of detail helps us better grasp the situation, which lets later processing change dependent on the kind of disaster.

Task 3: Support Category Identification (Specific to Earthquake Tweets)
The third task only looks at tweets that are in the "earthquake" category. In this case, the model checks to see whether the tweet has any information about help, including requests or offers for shelter, food, medication, infrastructural aid, or sympathy words.

This top-down sequential architecture makes sure that irrelevant information is removed early, disaster-type insights are found when they are needed, and actionable support-related tweets are found quickly. Figure 1 shows this approach in a way that is easy to see. The detailed sequential procedure of this multi-task classification framework is presented in Algorithm 1, demonstrating the organized decision-making logic utilized across the model pipeline.

```
Input: Unprocessed tweets dataset D = {Tweet_1, Tweet_2, ..., Tweet_N}
Results: Multi-level categorization results for Task 1, Task 2, and Task 3
// ----------- Collection and Preprocessing of Tweets
1: For each tweet t in D
    • normalize the text by converting it to lowercase and removing punctuation.
    • Tokenize the tweet into individual words.
    • Eliminate stop words.
    • Implement stemming
2: End for
3: // ----------- Contextual Feature Extraction using BERT
4: For each tweet t in D
    • Add the special token [CLS] at the beginning and append [SEP] at the end.
    • Subsequently, generate token, position, and segment embeddings.
    • Input embeddings into the pre-trained BERT model.
    • Extract the final hidden state of the [CLS] token, denoted as $V_{BERT} \in R^{768}$.
5. End for
// ----------- Graph Construction using TF-IDF and NPMI
6: Construct vocabulary V from pre-processed tweets.
7: Compute TF-IDF scores for all terms in V
8: For each word pair ($W_i, W_j$) inside a window size of 5, do
    • Calculate the co-occurrence frequency and NPMI ($W_i, W_j$)
9: If NPMI ($W_i, W_j$) exceeds T (empirically T = 0.2), then
    • Establish edge E (I, j) in graph G
10: End if
11: End For
// ----------- Extraction of Structural Features Utilizing GCN
12: Initialize each node in G with its BERT embedding vector.
13: For I = 1 to L (the number of GCN layers) do
        - Perform Graph convolution as:
          $H^{l+1} = \sigma(\hat{A}H^{(l)}W^{(l)})$
14. End For
```

```
15: Aggregate node embeddings for each tweet to create V_GCN_t ∈ $R^{d_2}$
// ----------- Vector Fusion
16: For each tweet t, perform the following:
    • Concatenate:V_proj_t = concat(V_BERT_t, V_GCN_t) → $R^{512}$
17: End for
// ----------- Classification using FCNN
18: For each tweet t do the following:
29:     Predict $y_1$ = FCNN₁(V_proj_t) // Task 1: Disaster versus
        Non-disaster
20:   if $y_1$ = = 'Disaster' then
    • Predict $y_2$ = FCNN₂(v_proj_t) // Task 2: Disaster type
21:       if $y_2$ = = 'Earthquake' then
    •     Predict $y_3$ = FCNN₃(v_proj_t) // Task 3: Support
          requirement
22:       End if
23:   End if
2: End for
25: Define loss functions L (categorical cross-entropy)
26: Optimize all parameters via the Adam optimizer and mini-batch backpropagation.
27: Implement dropout and batch normalization to reduce overfitting.
```

4. RESULTS AND DISCUSSION

We performed a number of tests to evaluate the efficacy of our hybrid model, which combines contextual embeddings from BERT with structural representations derived from Graph Convolutional Networks (GCN) for the categorization of disaster-related tweets. All tests were conducted with Python 3.8 on the Google Colab platform. To comprehensively assess the model, we generated three specialized datasets for three distinct classification tasks, each partitioned using an 80:20 train-test ratio. The GCN component was constructed with two layers containing 128 and 64 neurons, respectively. The training used the Adam optimizer, including a learning rate of 0.01 and a weight decay of 5e-4. Dropout at a rate of 0.5 and L2 regularization were used to prevent overfitting.
The BERT model functioned as the contextual encoder, delivering tweet-level embeddings that contain semantic significance. The embeddings were merged with GCN-derived structural characteristics to produce the final input vector. The vector was then input into a Fully Connected Neural Network (FCNN), with an architecture meticulously designed for each classification assignment.
In Task 1 (disaster vs non-disaster classification), we constructed the FCNN with two hidden layers of 512 and 256 neurons, respectively, culminating in a softmax output layer with two neurons. In Task 2 (disaster type classification: earthquake, flood, or other), we used hidden layers consisting of 256 and 128 neurons, culminating in a 3-neuron output layer. In Task 3, which aims to detect support categories in earthquake-related tweets, the FCNN had hidden layers of 512 and 128 neurons, culminating in a 5-neuron output layer for classification into donation, shelter, food and medication, infrastructure damage, and sympathy.
All hidden layers used ReLU activation, however the output layers applied softmax for multi-class classification. The FCNNs were trained with a dropout rate of 0.3 and batch normalization to improve generalization and training stability. The Adam optimizer (learning rate: 0.001) and early stopping were used to enhance convergence and minimize overfitting. The model's performance was evaluated using conventional metrics: accuracy, precision, recall, F1-score, and confusion matrices. This section's findings demonstrate the efficacy of the BERT + GCN model in all three classification tests.

4.1. Results of Disaster Identification (Task 1)

We compare our suggested model against various well-established baseline techniques to assess its efficacy. The experimental findings for every model are shown in Table 2.
Bi-LSTM [46]: This model classifies disaster-related tweets using a Bidirectional Long Short-Term Memory network. It captures contextual dependencies within the tweet content by analyzing sequences in both forward and backward directions.
STGCN [42]: This design combines a bidirectional LSTM with network Convolutional Networks (GCN), interpreting the dataset as a heterogeneous network where words, concepts, and documents operate as linked nodes, hence boosting contextual awareness.
HGAT [40]: Intended for short text categorization, this model connects words, themes, and papers to create a corpus-level heterogeneous graph. It uses a dual-attention approach to improve classification accuracy.
TextGCN [24]: This approach builds a global network with nodes representing both documents

and words. Graph-based learning enhances categorization by modelling long-range relationships throughout the corpus.

LSTM-GAT [43]: For categorization, this method combines LSTM with a Graph Attention Network (GAT) and creates a dependency-based graph for every tweet. We re-implemented the model since the previous implementation was unavailable; this could cause some differences from the stated outcomes.

The BERT+GCN hybrid model obtained the maximum accuracy of 96.58% among all models, substantially surpassing single-models such as GCN (80.98%) and BERT (83.02%).

Table 2: Accuracy for each model

Model	Accuracy (%)	Precision	Recall	F1-score
BERT+ GCN (our model)	96.58	0.9780	0.9734	0.9757
GCN- Only	80.98	0.8094	0.8098	0.8091
BERT - Only	83.02	0.8373	0.8561	0.8443
Bi-LSTM [46]	87.00	*	*	*
TEXTGCN [24]	80.31	*	*	*
STGCN [42]	82.31	*	*	*
HGAT [40]	80.45	*	*	*
LSTM-GAT [43]	88.40	*	*	*

* Not Available

In Figure 14, we can see the confusion matrices that show how well GCN, BERT, and Hybrid model (BERT+ GCN) performed in terms of categorization. While GCN and BERT show somewhat higher levels of false positives and false negatives, Hybrid model has the lowest misclassification rate. Figure 15 shows the growth of accuracy over epochs for GCN, BERT, and Hybrid model.

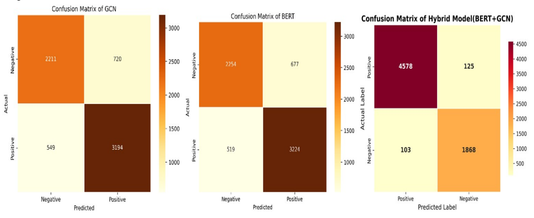


Figure 14: Confusion matrices of GCN, BERT and Hybrid model

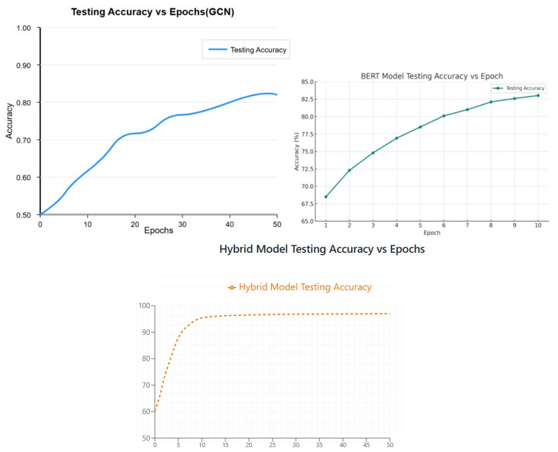


Figure 15: Testing accuracy curves of GCN, BERT and Hybrid model

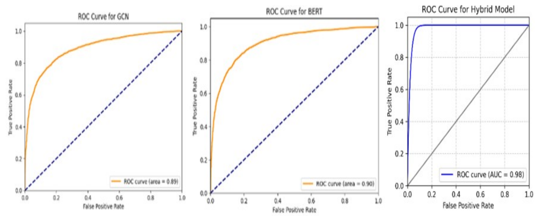


Figure 16: ROC curves of GCN, BERT and Hybrid Model

Figure 16 shows the ROC curves for the three models. Hybrid model has the best AUC value of 0.98, followed by BERT at 0.90 and GCN at 0.89, hence validating its higher classification efficacy.

4.2. Results of Disaster Type Classification (Task 2)

In Table 3, the model's learning trajectory across 50 epochs is shown by disaster categorization performance metrics. The model's generalization is shown by a continuous drop in training and validation loss. Successful learning and convergence increase accuracy, precision, recall, and F1-score across epochs. Figure 17 shows the confusion matrix heatmap illustrating how effectively the model identifies three catastrophes. High diagonal numbers (855, 934, 1084) indicate that the model produces many valid predictions.

Table 3: Epoch-wise Performance Metrics for Disaster Type Classification

Epoch	Training Loss	Validation Loss	Class	Accuracy (%)	Precision	Recall	F1-Score
1	0.163214	0.252341	No Disaster	89.50	0.8600	0.9100	0.8836
1	0.163214	0.252341	Flood	91.01	0.9230	0.8350	0.8756
1	0.163214	0.252341	Earthquake	90.20	0.8540	0.9030	0.8778
25	0.119123	0.181234	No Disaster	91.70	0.8830	0.9280	0.9051
25	0.119123	0.181234	Flood	94.20	0.9490	0.8590	0.9013
25	0.119123	0.181234	Earthquake	93.20	0.8760	0.9220	0.8977
50	0.082987	0.095642	No Disaster	**92.65**	0.8933	0.9300	0.9112
50	0.082987	0.095642	Flood	**96.30**	0.9582	0.9631	0.9606
50	0.082987	0.095642	Earthquake	**95.48**	0.9381	0.9556	0.9480

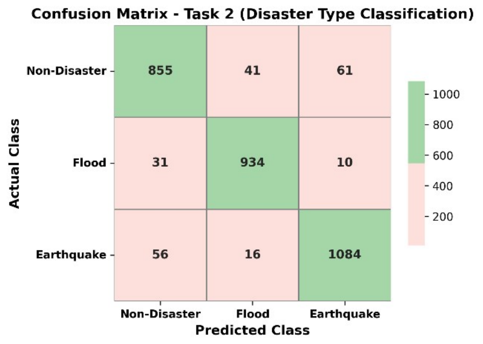


Figure 17: Confusion matrix for Disaster Type Classification

The disaster category classification accuracy for BERT-only, GCN-only, and the suggested hybrid model is displayed in Table 4. With an overall accuracy of 93.04%, our hybrid model performs best, scoring 92.65% for No Disaster, 96.30% for Flood, and 95.48% for earthquake. BERT-only and GCN-only, on the other hand, record 89.12% and 87.34%, respectively, demonstrating the efficacy of feature combination.

Figure 18 shows the 50-epochs Training, Validation Loss, and Accuracy Curve. It shows how loss affects model training accuracy. Right y-axis shows accuracy (%), left y-axis shows loss values, and x-axis shows epochs. As epochs rise, training and validation loss decrease, improving model learning and generalization.

Table 4: Disaster Type Classification Accuracies across models

Model Variant	No Disaster (%)	Flood (%)	Earthquake (%)	Overall Accuracy (%)
BERT only	87.50	91.20	88.65	88.12
GCN only	85.40	89.01	86.70	87.34
BERT + GCN	92.65	96.30	95.48	93.04

Improvement in classification occurs as accuracy rises. This image shows the model's learning trajectory, stressing the balance between loss reduction and accuracy improvement.

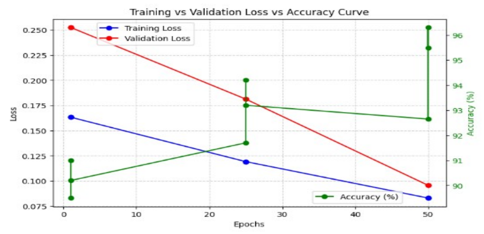


Figure 18: Training vs. Validation Loss and Accuracy Curve Over Epochs

4.3. Results of Help-Seeking Detection (Task 3)

The accuracy and loss metrics for our hybrid model across 50 epochs are shown in Table 5. The test accuracy rises from 90.00% in epoch 1 to a peak of 95% in epoch 50, accompanied by a reduction in test loss from 0.1157 to 0.022168. Figure 19. a) and 19. b) illustrate successful model learning and convergence via a continuous reduction in loss and a steady improvement in accuracy.

Table 5: Training and Testing Performance of hybrid Model Across Epochs

Epoch	Training Accuracy	Testing Accuracy	Training Loss	Testing Loss
1	0.850314	0.900482	0.120376	0.115729
6	0.927621	0.910158	0.050194	0.090413
11	0.945682	0.920721	0.030987	0.080145
16	0.950219	0.930352	0.020875	0.070593
21	0.955174	0.937603	0.015623	0.060482
26	0.958764	0.942847	0.012598	0.052731
31	0.960348	0.944527	0.010674	0.045289
36	0.963914	0.946328	0.007543	0.040713
41	0.966289	0.948728	0.006534	0.032127
50	0.967415	0.9507632	0.005523	0.022168

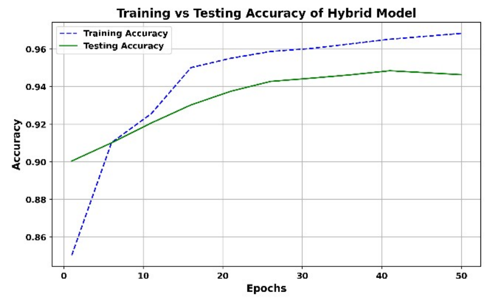


Figure 19: a) Training vs Testing Accuracy Over Epochs of Hybrid model

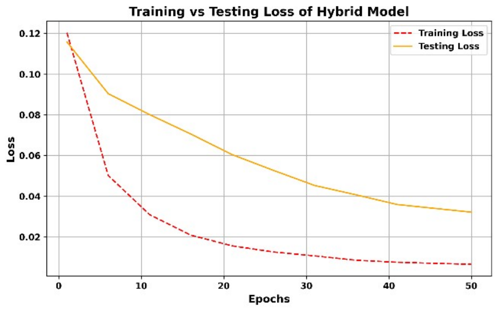


Figure 19: b) Training vs Testing Loss Over Epochs of Hybrid model

Table 6 shows the accuracy, recall, and F1-score for each kind of help. Our hybrid model (BERT + GCN) has the best accuracy at 95.07%. The category-wise performance of numerous models, including LSTM, Bi-LSTM, GCN, and BERT, is shown in Table 7. The models are evaluated based on precision (P), recall (R), and F1-score (F1), and our model shows the highest performance in almost all of the categories.

Table 6: Model Performance by Help category using hybrid model

Category	Precision	Recall	F1-Score	Support
Other Useful Information	0.9593	0.9593	0.9687	517
Donation & Volunteering	0.9447	0.9534	0.9490	215
Sympathy & Support	0.9417	0.9509	0.9463	204
Food & Medicine	0.9220	0.9466	0.9342	150
Infrastructure & Utilities	0.8611	0.8857	0.8732	70
Accuracy	**95.07**			
Macro Average	**0.9296**	**0.9392**	**0.9343**	
Weighted Average	**0.9512**	**0.907**	**0.9509**	

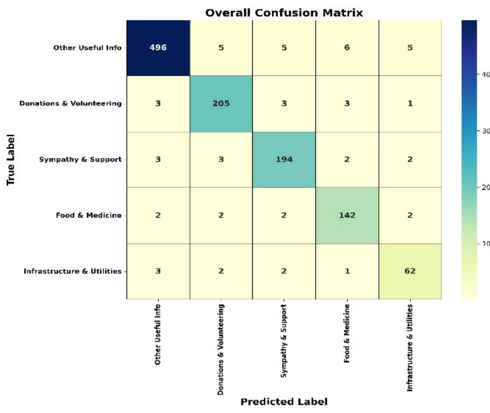
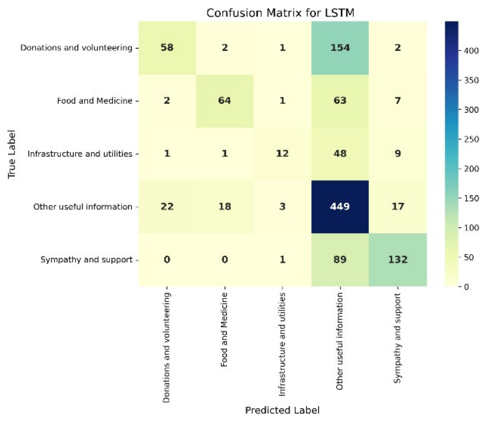
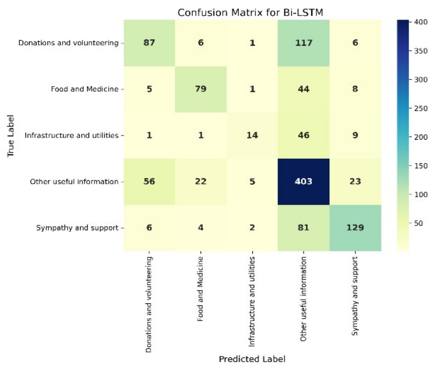


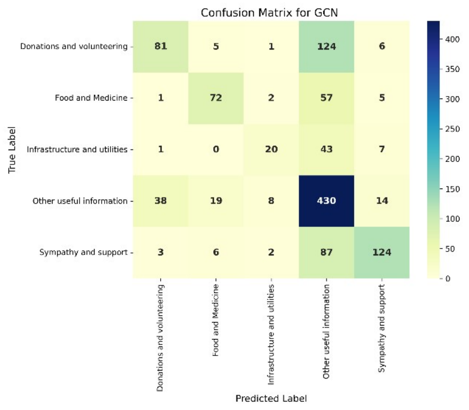
Figure 20: Confusion matrix for help category

Figure 20 provides a visual representation of the proposed model, which appropriately categorizes tweets into five different sorts of help. The bulk of predictions lie along the diagonal, which indicates that the model performs very well within each category. It has been noted that there are certain minor misclassifications that occur between categories that are closely related, such as Other Useful Information and Sympathy.

The confusion matrices of all deep learning models can be seen in Figure 21, and the ROC curves for each model and category can be seen in Figure 22. This lets us get a better idea of how well the classification works. Finally, Table 8 shows the classification accuracy of each model.







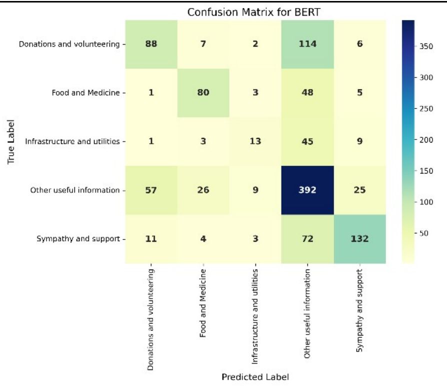


Figure 21: Multiclass Confusion Matrices for different Deep learning models

4.4. Ablation Analysis

We did an ablation analysis to learn more about how well our suggested hybrid framework works. In this research, we methodically eliminated or altered essential elements of the model to assess their distinct contributions. Table 9 shows a summary of the results. The hybrid model did much better than BERT-only (83.02%) and GCN-only (80.98%) for Task 1 (disaster vs. non-disaster), with a score of 96.58%. This shows that combining contextual and structural features makes binary classification a lot better. Table 4 already showed the per-class accuracies for Task 2 (disaster type classification). In this case, we look at the overall accuracy. The hybrid got 93.04%, while BERT-only got 89.12% and GCN-only got 87.34%. These findings validate that the integration of semantic and relational data enhances multi-class disaster recognition. In Task 3 (help/support category detection), the hybrid got 95.07%, but BERT-only and GCN-only did poorly, getting only 60.99% and 62.89%, respectively. The big difference in performance shows that both models are needed for difficult classification tasks. In general, the ablation analysis shows that neither BERT nor GCN can do as well as the hybrid on its own. For strong tweet classification in crisis situations, it is important to combine BERT's contextual embeddings with GCN's structural representations.

Table 7. Model performance metric by Help category using Deep learning models

Help Category	**LSTM (P, R, F1)**	**Bi-LSTM (P, R, F1)**	**GCN (P, R, F1)**	**BERT (P, R, F1)**
Donations & Volunteering	(0.61, 0.32, 0.42)	(0.55, 0.43, 0.48)	(0.56, 0.40, 0.47)	(0.50, 0.40, 0.44)
Food & Medicine	(0.77, 0.42, 0.54)	(0.74, 0.49, 0.59)	(0.77, 0.49, 0.66)	(0.73, 0.48, 0.58)
Infrastructure & utilities	(0.92, 0.13, 0.22)	(0.91, 0.23, 0.37)	(0.88, 0.26, 0.41)	(0.79, 0.13, 0.22)
Other Useful Information	(0.55, 0.85, 0.67)	(0.57, 0.79, 0.68)	(0.57, 0.82, 0.68)	(0.56, 0.79, 0.65)
Sympathy & Support	(0.78, 0.58, 0.67)	(0.76, 0.60, 0.67)	(0.82, 0.58, 0.68)	(0.76, 0.56, 0.64)

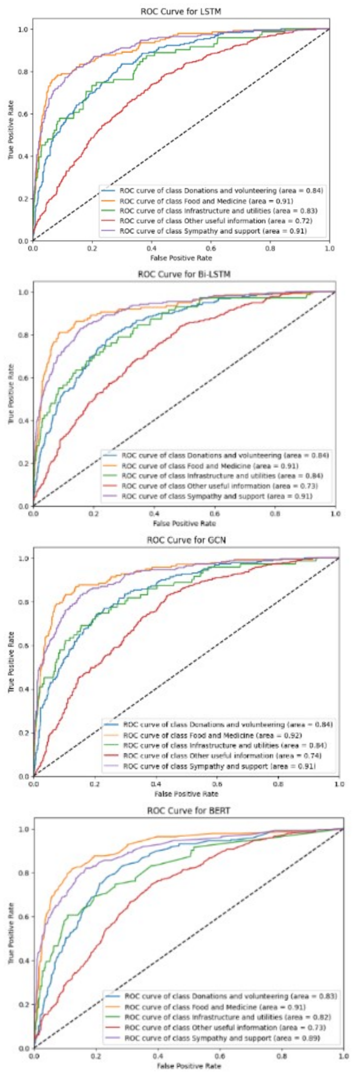


Figure 22: Multi class ROC Curves for different Deep learning models

Table 8: Summary of Models Accuracy by Help Category

Model	Accuracy (%)
BERT+GCN (Our model)	95.07
LSTM	61.85
Bi-LSTM	61.59
GCN	62.89
BERT	60.99

Table 9: Ablation Study Results

Model Variant	**Task 1 Accuracy (%)**	**Task 2 Accuracy (%)**	**Task 3 Accuracy (%)**
BERT only	83.02	89.12	60.99
GCN only	80.98	87.34	62.89
BERT + GCN (Proposed)	96.58	93.04	95.07

6. CONCLUSION & FUTURE WORK

We suggested a hybrid deep learning architecture in this paper combining BERT and Graph Convolutional Networks (GCN) to improve the categorization of disaster-related tweets. The model combines structural links obtained from word co-occurrence graphs built using TF-IDF filtering and NPMI with BERT's contextual word embeddings. By means of this dual-representation strategy, we addressed the drawbacks of both sequential and graph-based models used separately. Our model's experimental findings showed its efficacy by outperforming all three tasks: binary disaster detection, disaster type categorization, and support need identification. Especially suitable for real-time social media analysis during emergencies, the model's ability to handle short, informal text and extract both semantic and structural elements. While our model performs well, it requires substantial memory for BERT and graph processing. Also, it is trained on English tweets and may need further tuning for other languages or domains.

We want to expand the approach for future work to handle multilingual tweets, hence allowing more relevance during worldwide catastrophes. Including temporal and geographic information as well might help to improve categorization accuracy by contextualizing the tweets within certain timeframes or locales. To better capture subtle interactions, we also want to investigate more sophisticated graph-based encoders such Heterogeneous GCNs or Graph Attention Networks (GATs). Deploying the model in a real-time monitoring system might help humanitarian groups by giving timely and organized insights from streaming social media material during crisis occurrences.

REFERENCES:

[1]. N. Noor, R. Okhai, T. B. Jamal, N. Kapucu, Y. G. Ge, & S. Hasan, "Social-media-based crisis communication: Assessing the engagement of local agencies in Twitter during Hurricane Irma" , *International Journal of Information Management Data Insights*, 4(2), 2024, 100236.

[2]. A. Mukkamala, B. Gupta, & R. R. Mukkamala, "The Role of Social Media in Disaster Management: Opportunities, Challenges, and Future Directions", Book: *Technology Innovation for Sustainable Development of Healthcare and Disaster Management*, Springer Nature, 2024, pp. 13-29.

[3]. N. R. Paul, & R. C. Balabantaray, "Disaster related tweet classification method based on BERT and GAT", *International Journal of Information Technology*, 2025, pp. 1-13.

[4]. P. Khare, G. Burel, & H. Alani, "Classifying crises-information relevancy with semantics", *in European semantic web conference, Springer International Publishing,* 2018, pp. 367-383.

[5]. N. R. Paul, R. C. Balabantaray, & D. Sahoo, " Fine-tuning transformer-based representations in active learning for labelling crisis dataset of tweets", *SN Computer Science*, 4(5), 2023, 553.

[6]. A. Kumar, J. P. Singh, N. P. Rana, & Y. K. Dwivedi, "Multi-channel convolutional neural network for the identification of eyewitness tweets of disaster", *Information Systems Frontiers*, 25(4), 2023, pp. 1589-1604.

[7]. N. Algiriyage, R. Sampath, R. Prasanna, K. Stock, E. Hudson-Doyle, & D. Johnston, "Identifying Disaster-related Tweets: A Large-Scale Detection Model Comparison", In ISCRAM, 2021, pp. 731-743.

[8]. M. Kejriwal, P. Zhou, "On detecting urgency in short cri- sis messages using minimal supervision and transfer learning", *Social Network Analysis and Mining*, 10(1), 2020, 58.

[9]. N. R. Paul, D. Sahoo, R. C. Balabantaray, " Classification of crisis-related data on Twitter using a deep learning-based framework", Multimedia Tools and Applications, 82(6), 2022, pp. 8921-8941.

[10]. N. R. Paul, R. C. Balabantaray, "Detecting Crisis Event on Twitter Using Combination of LSTM, CNN Model", *In Annual Convention of the Computer Society of India*, Singapore: Springer Singapore, 2020, pp. 71-80.

[11]. J. Yin , A. Lampert, M. Cameron, B. Robinson, R. Power, "Using social media to enhance emergency situation awareness", *IEEE Intelligent Systems*, 27(6), 2012, pp.52–59

[12]. T. Sakaki, M. Okazaki, Y. Matsuo, "Earthquake shakes twit- ter users: real-time event detection by social sensors", *In Proceedings of the 19th international conference on World wide web*, 2010, pp.851–860

[13]. C. Li, A. Sun, A. Datta, "Twevent: Segment-based event detection from tweets", *In Proc of CIKM ACM*, 2012, pp. 155–164

[14]. C. Caragea, N. McNeese, A. Jaiswal A, G. Traylor, H. Kim , P. Mitra , D. Wu, A. Tapia, L. Giles, B. J. Jansen et al., "Classifying text messages for the Haiti earthquake", *In Proc. of ISCRAM*, 2011.

[15]. M. Imran, C. Castillo, J. Lucas, P. Meier, S. Vieweg, "AIDR: artificial intelligence for disaster response," *In Proc. of WWW (Companion). IW3C2*, 2014, pp.159-162

[16]. T. Mikolov, K. Chen, G. Corrado, J. Dean, "Efficient estimation of word representations

in vector space", arXiv preprint arXiv:1301.3781, 2013.

[17]. J. Pennington, R. Socher, C. D. Manning, "GloVe: global vectors for word representation", *In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543

[18]. T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, S. Khudanpur, "Recurrent neural network-based language model", *Interspeech*, 2(3), 2010, pp.1045–1048.

[19]. M. Jaderberg, K. Simonyan, A. Vedaldi, A. Zisserman, " Reading text in the wild with convolutional neural networks", *International Journal of Computer Vision*, 116(1), 2016, pp. 1–20.

[20]. P. Wang , B. Xu, J. Xu, G. Tian, C. L. Liu, H. Hao, "Semantic expansion using word embedding clustering and convolutional neural network for improving short text classification. *Neurocomputing,* 174, 2016, pp. 806–814.

[21]. J. Devlin, M.W. Chang, K. Lee, K. Toutanova, "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*, volume 1 (long and short papers), 2019, pp. 4171-4186.

[22]. F. Scarselli, M. Gori, A.C. Tsoi, M. Hagenbuchner, G. Monfardini, "The graph neural network model", *IEEE Transcation on Neural Networks*, 20(1), 2008, pp. 61–80

[23]. M. Defferrard, X. Bresson, P. Vandergheynst , "Convolu tional neural networks on graphs with fast localized spectral filtering," *Advances in neural information processing systems*, 2016, 29.

[24]. L. Yao, C. Mao, Y. Luo, "Graph convolutional networks for text classification", *In Proceedings of the AAAI conference on artificial intelligence*, Vol. 33, No. 01, 2019, pp. 7370-7377.

[25]. Y. Wang, S. Wang, Q. Yao, & D. Dou, "Hierarchical heterogeneous graph representation learning for short text classification", arXiv preprint arXiv: 2111.00180, 2021.

[26]. L. Huang, D. Ma, S. Li, X. Zhang, H. Wang, "Text level graph neural network for text classification", *In: EMNLP-IJCNLP*, 2019, pp. 3442-3448

[27]. M. Imran , S. Elbassuoni, C. Castillo, F. Diaz, P. Meier, " Practical extraction of disaster-relevant information from social media" , *International Conference on World Wide Web*, 2013, pp. 1021-1024.

[28]. S. Verma et al. "Natural language processing to the rescue? extracting situational awareness tweets during mass emergency", *Proceedings of the International AAAI Conference on Web and Social Media*. Vol. 5. No. 1, 2011, pp. 385-392.

[29]. D. M. Blei , A. Y. Ng , M. I. Jordan, "Latent dirichlet allocation", *Journal of Machine Learning Research*, 3, 2003, pp. 993–1022

[30]. C. C. Aggarwal, C. Zhai "A survey of text classification algorithms. Mining text data", Springer, 2012, pp 163–222

[31]. Y. Kim "Convolutional neural networks for sentence clas- sification", arXiv:1408.5882, 2014.

[32]. S. Hochreiter , J. Schmidhuber, "Long short-term memory", *Neural Computation*, 9(8), 1997, pp. 1735–1780

[33]. Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang , Yu PS, "A comprehensive survey on graph neural networks", *IEEE Transactions on Neural Networks and Learning Systems*, 32(1), 2021, pp. 4–24

[34]. Z. Zhang, P. Cui, W. Zhu, "Deep learning on graphs: a survey", *IEEE Transactions on Knowledge and Data Engineering*, 34(1), 2022, pp. 249–270

[35]. J. Zhou "Graph neural networks: a review of methods and applications" , *AI Open*, 1, pp. 57–81, 2020.

[36]. Y. Zhang, Q. Liu, L. Song, "Sentence-state LSTM for text representation, arXiv preprint arXiv:1805.02474, 2018.

[37]. J. Bastings, I. Titov, W. Aziz, D. Marcheggiani, K. Sima'an, " Graph convolutional encoders for syntax-aware neural machine translation, arXiv preprint arXiv:1704.04675, 2017.

[38]. Y. Zhang, X. Yu, Z. Cui, S. Wu, Z. Wen, L. Wang, "Every document owns its structure: Inductive text classification via graph neural networks", arXiv preprint arXiv:2004.13826, 2020.

[39]. P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò ,Y. Bengio, "Graph attention networks. arXiv preprint arXiv:1710, 10903, 2018.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[40]. H. Linmei, Y. Tianchi, S. Chuan, J. Houye, L. Xiaoli, " Heterogeneous graph attention networks for semi-supervised short text classification", *In Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, 2019, pp. 4821-4830.

[41]. Y. Liu , R. Guan, F. Giunchiglia, Y. Liang, X. Feng, "Deep attention diffusion graph neural networks for text classification", *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021, pp. 8142-8152.

[42]. Y. Zhihao, J. Gongyao, L. Ye, L. Zhiyong , Y. Jin, "Document and word representations generated by graph convolutional net work and BERT for short text classification", *In European Conference on Artificial Intelligence*, 2021, pp. 2275–2281

[43]. W. Haitao, L. Fangbing "A text classification method based on LSTM and graph attention network", *Connection Science*, 34(1), 2022, pp.2466– 2480.

[44]. C. Caragea, A. Silvescu, A.H. Tapia "Identifying informative messages in disaster events using convolutional neural networks", *In International conference on information systems for crisis response and management*, 2016, pp 137–147

[45]. A. Bhoi, S. P. Pujari, R. C. Balabantaray, "A deep learning-based social media text analysis framework for disaster resource management", *Social Network Analysis and Mining*, 10(1), 2020, pp.1–14.

[46]. J. Liu et al. "Crisisbert: a robust transformer for crisis clas- sification and contextual crisis embedding", *In proceedings of the 32nd ACM Conference on hypertext and social media*. 2021, pp.133-141.

[47]. K. Wang, C. H. Soyeon, P. Josiah, "InducT-GCN: Inductive graph convolutional networks for text classification", *26th International Conference on Pattern Recognition (ICPR),* 2022, pp.1243-1249.

[48]. https://neptune.ai/blog/graph-neural-network-and-some-of-gnn-applications.