RumourGraphXplainer: Do Structures Really Matter in Rumour Detection

Chin Wai Kit Daniel, Kwan Hui Lim, *Member, IEEE* and Roy Ka-Wei Lee, *Senior Member, IEEE*

***Abstract*—The rise of social media has enabled individuals to rapidly share information, including rumours, which can have significant impacts on various domains. Traditional approaches to rumour control are impractical for social media platforms due to the volume and speed of information. Automated detection methods are needed that not only identify rumours early but also provide explanations for their decisions to protect free speech. Recent advancements in deep learning have shown promise in automating rumour detection. Graph-based models, such as Bi-directional Graph Convolution Network (Bi-GCN), capture propagation, and dispersion patterns to differentiate rumours from the truth. However, the interpretability of these deep learning models is a challenge. This paper focuses on Graph Convolution Networks (GCN), which lack attention maps for easy model attribution but excel at capturing global structural features. We investigate the importance of graph structure in rumour detection using two GCN models on a real-world dataset, analyzing the learned latent propagation and dispersion features. To the best of our knowledge, this is the first study to explore GCNs in rumour detection and investigate the significance of graph structure in this task. Our research addresses three primary questions: (1) the primary contributors to GCN-based rumour detection models and their differences across models, (2) the importance of graph structure for accurate predictions in GCN-based models, and (3) the latent propagation and dispersion features learned by GCN-based detection models during the rumour detection process.**

***Index Terms*—rumour detection, graph convolution networks, explainability**

I. Introduction

WITH the rise of social media, individuals have gained the power to rapidly create and disseminate information, reaching large audiences in a matter of seconds. While this facilitates the efficient and unrestricted sharing of ideas, it also opens the door to the dissemination of harmful content, including rumours [1]–[3]. The impact of rumours on various aspects, such as the economy, politics, and public health, has been evident during events like the COVID-19 pandemic in several countries [4].

In traditional media, the editing and vetting process before publication serves as a safeguard against rumour propagation. However, applying the same approach to social media platforms is neither practical nor reasonable due to the sheer volume and speed of information being shared. Although allowing users to report potentially rumour-related content for manual review could be a solution, it would still be labour-intensive and not entirely effective. Therefore, there is a need for automated detection methods that are both efficient and interpretable. To avoid accusations of unwarranted censorship, these detection methods must not only identify conversations or posts containing rumours as early as possible to prevent further spread but also provide explanations for their decisions, protecting the users' rights to free speech. Thus, the development of rumour detection techniques should prioritize both timely identification and interpretability.

Recent advancements in deep learning techniques have shown promising results in automated feature extraction from data, surpassing classical methods and offering a less labour-intensive approach for early rumour detection. These techniques encompass various models, including sequence models such as Long-Short Term Memory (LSTM) [5] and Gated Recurrent Unit (GRU) [5], as well as models utilizing tree or graph-structured data such as Propagation Tree Kernel with Support Vector Machine (PTK) [6], Recursive Neural Networks (RvNN) [7], [8], Bi-directional GCN (Bi-GCN) [9], Edge-enhanced Bayesian GCN (EBGCN) [10], and the recently proposed Claim-guided Hierarchical Graph Attention Network (ClaHi-GAT) [11] and Dual Attention GCN (DA-GCN) [12].

Earlier research by Vosoughi et al. [1] has demonstrated that rumours tend to spread farther, faster, and wider than truthful statements. In response, graph-based models were developed to capture the underlying propagation and dispersion patterns of rumours, enabling easier differentiation from the truth. For instance, Bi-GCN leverages tweet representations and the adjacency matrix of tweet relations to construct both a top-down and a bottom-up graph. This approach allows Bi-GCN to learn distinct node representations that capture diverse structural properties of the graph. Although these deep learning models exhibit superior performance compared to classical machine learning techniques, their decision-making process is more challenging to interpret.

This paper aims to investigate the effectiveness of Graph Convolution Networks (GCNs) in rumour detection, given their popularity as a model choice. Unlike Graph Attention Networks (GAT), GCNs lack attention maps that facilitate straightforward model attribution, making it more challenging to explain their decision-making process. Moreover, compared to models like Recursive Neural Networks (RvNN) and other sequence models, GCNs demonstrate superior capability in capturing global structural features [9], rendering them a worthwhile focus of study.

To assess the significance of graph structure in rumour detection, we conduct experiments using two models on the PHEME dataset [13] and the Twitter 15/16 dataset [6]. The former comprises tweets related to nine events, exhibiting a class imbalance favouring the non-rumour and true-rumour classes, while the latter comprises tweets from events in 2015 and 2016 with a balanced class distribution. In addition to evaluating the importance of graph structure in rumour detection, we analyse the latent propagation and dispersion features learned by GCN-based models to determine if they capture meaningful structural characteristics.

To the best of our knowledge, this study represents the first attempt to elucidate the functioning of GCNs in the context of rumour detection. Moreover, it is the first work to investigate the significance of graph structure in the rumour detection task. In summary, our research aims to address three primary research questions:

- **RQ1:** What are the primary contributors to GCN-based rumour detection models, and how do these contributors differ across models?
- **RQ2:** How crucial is the graph structure for GCN-based rumour detection models in making accurate predictions?
- **RQ3:** What latent propagation and dispersion features do GCN-based detection models capture and learn during the rumour detection process?

II. Related Work

A. *Graph-based Rumour Detection*

Rumour detection is a crucial task in combating misinformation in online media. Unlike fake news, which deliberately spreads misinformation, rumours can be both intentional and unintentional. Graph-based rumour detection models leverage graph-structured data to make predictions, enabling them to capture not only textual features but also topological and other graphical features.

There are several categories of graph-based rumour detection models, including kernel-based models, RvNN-based models, GCN-based models, and GAT-based models. Kernel-based models compute graph similarities by enumerating similar subgraphs, such as PTK and cPTK [6]. RvNN-based models learn node representations by aggregating information from neighbouring nodes through multiple recursions [7], [8]. However, RvNN-based models face challenges in capturing global structural features due to the use of sigmoid functions during the aggregation step, which can lead to vanishing gradients as the number of recursions increases [14]. On the other hand, GCN-based and GAT-based models do not employ sigmoid functions during aggregation, allowing them to effectively capture global structural features. Notable examples of GCN-based models are Bi-GCN [9] and EBGCN [10], while GAT-based models include ClaHi-GAT [11] and DA-GCN [12].

Although we have chosen Bi-GCN and EBGCN as objects of study, our work is not focused on model performance on the rumour detection task. Instead, we focus on explaining the importance of graph structure for GCN-based rumour detection models with Bi-GCN and EBGCN as specific examples.

B. *Graph Explainability*

There are two broad categories of explainability techniques for graph neural networks (GNNs): adapted methods and original methods. Adapted methods encompass Sensitivity Analysis (SA) [15], Layerwise Relevance Propagation (LRP) [16], [17], Gradient-weighted Class Activation Mapping (Grad-CAM) [18], Excitation Backpropagation (EB) [19], [20] and Local Interpretable Model-agnostic Explanations (LIME) [21]. On the other hand, original methods include GNNExplainer [22], PGExplainer [23], PGM-Explainer [24], RelEx [25], SubgraphX [26], CGAT [27].

Backpropagation-based methods such as SA, Grad-CAM, EB, and LRP are employed for model explanations. SA calculates the squared values of gradients from the model's output to input features, revealing the sensitivity of input features to the outputs. However, it is prone to saturation problems [15]. Grad-CAM backpropagates gradients from the target class to the desired convolution layer, resulting in coarse-grained model attribution achieved by multiplying these gradients with convolution feature maps [18]. LRP determines fine-grained model attribution by backpropagating relevance through all layers, and it provides more meaningful attribution in relative terms by normalizing relevance values after each layer's backpropagation [16], [17]. EB uses a top-down Winner-Takes-All approach to generate probabilities for each neuron's contribution to the final output [19], [20]. LIME obtains locally-linear approximations of model behaviour by observing the effect of perturbing input values on the model's predictions [21].

GNNExplainer trains a surrogate model to explain individual node predictions by learning a compact subgraph and relevant node features essential to a specific node's prediction. It can also extend its explanations to the graph level by aggregating individual node explanations [22]. PGExplainer, on the other hand, trains a surrogate model to explain graph-level predictions. It generates a probability graph of the input graph, identifying the subgraph that contributes the most to individual node predictions and graph-level predictions [23]. PGM-Explainer trains a probabilistic graphical model by perturbing node features randomly. The top dependent variables are selected using the Grow-Shrink algorithm, and a Bayesian network is trained on the reduced feature set to explain the original GNN [24]. RelEx trains a surrogate GCN model on perturbed data, generates explanations by applying soft masks on the surrogate GCN, and uses randomly sampled subgraphs to obtain predictions for the original GNN [25]. SubgraphX utilizes Monte Carlo Tree Search to generate subgraph-level explanations, identifying the most important subgraph as an explanation for the prediction [26]. Finally, CGAT utilises an adversarial approach via combining graphs to force the detector to learn more distinctive class features which we can then extract to perform explanations on [27].

In contrast to existing methods that primarily focus on node explanations, our proposed method aims to elucidate the latent propagation and dispersion features learned by GCN-based rumour detection models. Furthermore, our focus is on understanding the contribution of graph structure to the model's prediction, including the importance of edges, which is not directly explained by other methods.

III. Preliminaries

A. Graph Convolution Networks

1) Bi-directional Graph Convolution Network: The pioneering work of Bi-GCN introduced the application of GCNs to graph-based rumour detection [9]. Bi-GCN utilizes two separate GCNs to capture graph structure and text features from the top-down (TD) and bottom-up (BU) graphs, respectively. These features are then concatenated to form the input for a classifier. The model takes tweet vector representations and the corresponding TD/BU graph's edge index as input. Within each Graph Convolution Layer (GCL), a trainable feature extraction layer extracts textual features from the input and passes the learned lower-dimensional representations to the message-passing function. Notably, the model does not incorporate explicit edge weights as inputs. Instead, it employs a default edge weight generation function based on an inverse exponential function, where the number of hops from the root node determines the exponent. In the TD GCN, larger weights are assigned to the root node and nodes closer to it, while in the BU GCN, larger weights are assigned to leaf nodes and nodes closer to them.

2) Edge-enhanced Bayesian Graph Convolution Network: EBGCN builds upon the achievements of Bi-GCN by introducing an additional edge consistency loss to improve performance and infer edge weights [10]. In contrast to Bi-GCN, which relies on a simple approach to generate edge weights for unweighted edges in the top-down (TD) and bottom-up (BU) graphs, EBGCN incorporates a weight inference layer after the first Graph Convolution Layer (GCL). This dedicated layer is responsible for inferring edge weights for the second GCL. As the first GCL in EBGCN is identical to that of Bi-GCN, the graph contribution remains the same for this layer. In essence, EBGCN achieves the capability to dynamically weigh the importance of edges, enhancing its ability to capture informative graph structures.

B. Graph Centrality Measures

Graph centrality measures are fundamental tools in graph theory and network analysis for assessing the significance of nodes within a graph. These measures are valuable in various domains, such as identifying influential individuals in social networks, key nodes in network infrastructure, or analyzing disease outbreaks. Each centrality measure ranks nodes based on different notions of importance, offering diverse perspectives on what defines importance within a specific network. In our study, we have selected three centrality measures that are relevant to social network analysis, enabling us to evaluate the importance of nodes in the propagation graph and determine their significance based on their network positions.

- **Out-Degree:** The normalized out degree centrality is defined as follows: $C_{D_{out}}(v) = \frac{deg_{out}(v)}{max_{v\in V} deg_{out}(v)}$. Out-degree serves as an indicator of dispersion within the immediate neighbourhood of a node in TD graphs.
- **Betweenness:** The betweenness centrality is formulated as: $C_B(v) = \sum_{v\neq s\neq t\in V} \frac{\sigma_{st}(v)}{\sigma_{st}}$, where σ_{st} represents the number of shortest paths from node s to node t, and $\sigma_{st}(v)$ denotes the number of those paths that pass through node v. Betweenness can be employed as a measure of dispersion, as a higher betweenness score implies greater importance of a node as a conduit for information flow between the root and other nodes.
- **Closeness:** The closeness centrality is calculated using the formulation: $C_C(v) = \frac{N-1}{\sum_u d(u,v)}$, where N represents the number of nodes in the graph, and $d(u,v)$ denotes the shortest path distance between node u and node v. In the BU graph, a lower closeness score corresponds to a node that is further away from the root. Having more nodes with lower closeness values can be considered as an indication of message propagation in the graph, implying the spread of information to distant nodes.

By employing these centrality measures, we can effectively assess the importance of nodes within the network and gain insights into their roles in propagation dynamics.

IV. Methodology

In order to provide an explanation of the GCNs utilized in rumour detection, we focus on the constituents of the GCL and the intermediate outputs that are of particular interest to us. As depicted in **Fig. 1**, each GCN consists of two GCLs, denoted as GCNConv in the diagram, which we have highlighted with a red box. The *GCNConv* operation is a type of graph convolution introduced by Kipf et al. [28]. The detailed breakdown of this operation is illustrated in the enlarged view of the GCNConv module. Within each GCL, there exists a linear layer responsible for extracting textual features from the nodes. By examining the feature maps generated by these two layers, we can assess the relative importance of individual nodes within each layer, with each row in the feature map corresponding to a node in the graph.

To address **RQ1**, we aim to uncover the features learned by GCN models when applied to the task of rumour detection using graph-structured conversation threads. Each GCL is responsible for capturing both textual and graphical aspects. Initially, a linear layer extracts pertinent textual features, which serve as node representations for subsequent message-passing operations. Prior to message passing, self-directed edges are introduced for each node. These self-directed edges play a crucial role in the message-passing operation, facilitating the aggregation of extracted textual features based on their edge relationships. For instance, in a TD GCN, a node with n child nodes would have a total of $2n + 1$ edges, comprising $n + 1$ self-directed edges and n vectors. Each edge's vector, originating from the source node, is multiplied by its corresponding edge weight, resulting in $2n + 1$ vectors for the aggregation step. During aggregation, element-wise addition is performed for vectors associated with the same destination node. Ultimately, the GCL produces n vectors, representing the learned node representations for that specific layer. Consequently, the graph's structure can be implicitly captured in the node update step within the GCL. This is exemplified by Bi-GCN, where edge weights are naively generated, allowing the model to learn the graphical features indirectly. In contrast, EBGCN infers edge weights in the second GCL, enabling direct learning of graphical features by assigning appropriate weights to each edge.

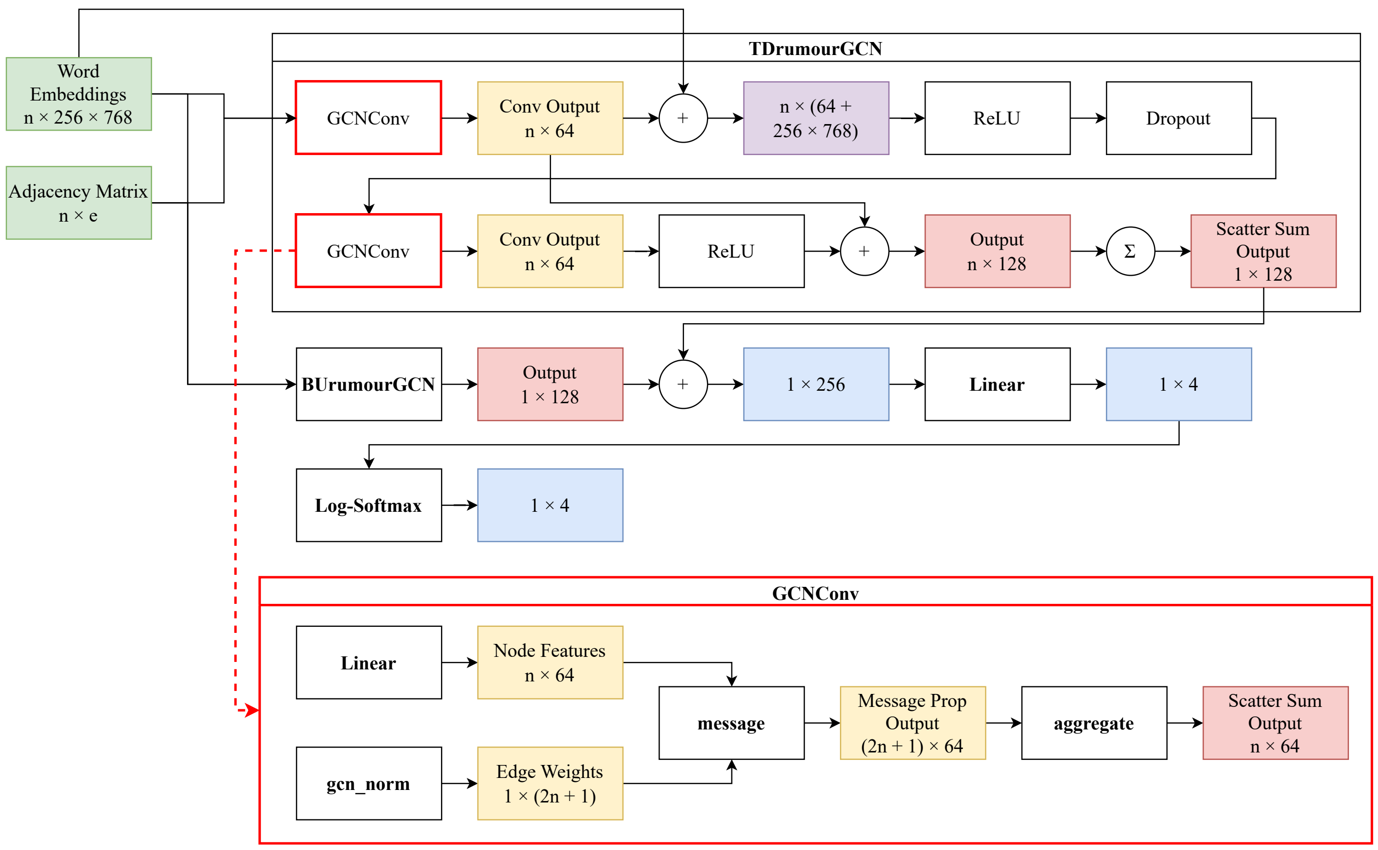

Fig. 1: Diagram of GCN layers

A. Identifying Model Component Contributions

To dissect how various components within our model contribute to its final output, a thorough understanding of these components is essential. Based on our in-depth knowledge of the model's architecture and the data it processes, we have identified two pivotal types of features learned by the model: (i) *textual features*, derived from content analysis, and (ii) *graphical features*, emanating from the propagation graph structure. Grounded on these distinct features, we categorize the contributions into three types: (i) *Textual Component Contribution*, assessing the impact of textual features; (ii) *Graphical Component Contribution*, evaluating the influence of graphical features; and (iii) *Joint Contribution*, which examines the synergistic effect of combining both types of features. In the following subsections, we delve into the specifics of how we compute these respective contributions. This detailed exploration will provide a clearer understanding of the individual and combined roles of textual and graphical components in shaping the model's performance and output.

1) Textual Component Contribution: To assess the influence of the textual component, we employ LRP to generate a relevance map for the activations emerging from the GCL layer. Subsequently, we introduce a slight modification to the approach presented in Montavon et al. [16] in order to distinguish the relevance flows through the two distinct edge types. Specifically, we focus solely on the rows in the relevance map that correspond to self-directed edges. Within these rows, we aggregate the relevance values to compute a relevance score for each node. The modified LRP procedure can be formulated as follows:

$$R_i^{(1)} = \sum_{j \in P_i} R_{ji}^{(1)} + R_{ii}^{(1)} \tag{1}$$

$$T_i^{(1)} = \frac{R_{ii}^{(1)} * W_{ii}^{(1)} * V_i^{(1)}}{\sum_{i \in n} R_{ii}^{(1)} * W_{ii}^{(1)} * V_i^{(1)}} \tag{2}$$

where $R^{(1)}i$ represents the relevance of the i^{th} node. The term $\sum j \in P_i R^{(1)} ji$ denotes the cumulative relevance flowing through all edges ji from parent nodes j to node i. Meanwhile, $R^{(1)} ii$ indicates the relevance flowing through the self-directed edge connected to node i. The relevance $T^{(1)} i$ corresponds to the relevance associated with the textual features extracted from node i. Additionally, $W^{(1)} ii$ represents the weight assigned to the self-directed edge ii, and $V_i^{(1)}$ signifies the feature vector of node i. The superscript (1) is utilized to indicate the values specific to the first GCL.

In the second GCL, as the input consistently includes additional information from the root node, we consider those parts as constants. We apply the modified LRP procedure, similar to the first step, to obtain a feature map. This feature map is utilized to calculate the contributions, following a similar approach as in the first GCL: $t_i^{(2)} = \sum_m v_{i,m}$. Next, we employ the following equation to determine the combined contribution of textual features:

$$t_i = t_i^{(1)} + t_i^{(2)} \qquad \forall i \in V \tag{3}$$

To obtain the relative contribution, we perform a normalization step:

$$c_{t_i} = \frac{t_i}{\sum_{i \in V} t_i} \tag{4}$$

where c_{t_i} is the relative textual contribution from each node. With this, we can obtain a ranked list of nodes by contribution for the TD and BU GCN from both models.

2) Graphical Component Contribution: To assess the contribution of the graphical component, we consider two aspects: the edges and the nodes. Firstly, we extract the edge weights generated by each GCL in both GCNs within the model. These weights indicate the importance attributed to the information flow along the corresponding edges and reflect their respective edge contributions. Additionally, each GCL generates a self-directed edge for every node, and the weight on this edge signifies the node's contribution.

Since the inputs to the second GCL include the node representations produced by the first GCL, the edges not only transmit information from the source node but also carry information from the preceding sources. Consequently, when determining the contribution of individual edges, this aspect must be taken into account. Given a triplet of nodes i, j, k, where node i is the parent of node j, which in turn is the parent of node k, the individual contribution of edge ij is computed as follows.

$$w_{ij} = w_{ij}^{(1)} + w_{ij}^{(2)} + \sum_{k \in A_j} w_{jk}^{(2)} \qquad \forall i,j,k \in V \tag{5}$$

Specifically, we denote $w_{ij}^{(1)}$ as the weight of the edge ij from the first Graph Convolution Layer (GCL), $w_{ij}^{(2)}$ as the weight of the same edge from the second GCL, and $w_{jk}^{(2)}$ as the weight of the edge jk from the second GCL. To normalize each weight and obtain the relative contribution, we normalize each weight by the sum of all weights:

$$c_{e_{ij}} = \frac{w_{ij}}{\sum_{i,j \in V} w_{ij}} \tag{6}$$

To calculate the overall contribution of each node, denoted by c_{n_i}, we consider the self-looping edge ii in each GCL. We sum the weights of the self-looping edge for each GCL and add the contributions from all edges where node i serves as the source. The equation for deriving the node contribution is as follows:

$$w_i = w_{ii}^{(1)} + w_{ii}^{(2)} + \sum_{j \in A_i} w_{ij} \qquad \forall i,j \in V \tag{7}$$

where $w_{ii}^{(1)}$ and $w_{ii}^{(2)}$ are the weights for edge ii from the first and second GCL respectively and A_i is the adjacency matrix of node i.

To obtain the relative contribution, we perform a similar normalization step as with the edge contributions:

$$c_{n_i} = \frac{w_i}{\sum_{i \in V} w_i} \tag{8}$$

where c_{n_i} is the relative contribution for each node. With the node and edge contribution maps obtained from both Bi-GCN and EBGCN for both the TD and BU graphs, we rank the nodes and edges by their contribution and compare the ranked lists with each other to look for differences.

3) Joint Contribution: To evaluate the combined contributions of the two components, we utilize LRP to generate a heatmap indicating their relative importance. LRP is chosen for its principled approach, as it normalizes intermediate inputs at each layer, enabling us to quantify their contributions accurately. The resulting heatmap highlights the nodes that the model deems most significant for the final output. To obtain the contributions, we follow a similar procedure to summing values across rows in the heatmap, as done for textual contributions:

$$c_{l_i}^{(n)} = \sum_m l_{i,m} \tag{9}$$

where $c_{l_i}^{(n)}$ is the contribution of the i^{th} node for the n^{th} output logit and $l_{i,m}$ is the relevance value for the m^{th} dimension of the i^{th} row. For each output class, we obtain a ranked list of nodes by contribution which will be used for comparison.

To obtain the relative contribution over all classes, we simply perform an element-wise addition weighted by the normalized value of the output logits $o^{(n)}$, for all the heatmaps before normalizing the values by the sum of all the heatmap values. The equation below shows how to compute this:

$$c_{l_i} = \frac{\sum_n o^{(n)} c_{l_i}^{(n)}}{\sum_n o^{(n)} \sum_i c_{l_i}^{(n)}} \qquad \forall i,j \in V \tag{10}$$

B. Importance of Graph Structure

To answer **RQ2** and determine the importance of the graph structure in model behaviour and performance, we perform a variety of tests:

- **Edge Deletion:** We perform edge deletion in the graph, which retains the complete textual content while reducing the influence of the graph structure.
- **Set Initial Edge Weights:** We initialize the edge weights uniformly to a value of 2. This configuration disregards any information propagated through self-loops.

For each of these configurations, we utilize LRP to extract the contributions associated with each class prediction. We then compare these contributions with those obtained from the original, unaltered data to identify and analyze any observed changes in behaviour.

C. Latent Propagation and Dispersion Features

To address **RQ3** and gain insights into the latent propagation and dispersion features learned by the model, we employ four centrality measures to generate ranked lists of node importance. These lists are then compared with the ranked

TABLE I: GCN-wise Jaccard Similarities for PHEME and Twitter15/16

Model/Output		Method/Dataset					
Anchor	**Comparison**	**LRP/PHME**	**Grad-CAM/PHEME**	**c-EB/PHEME**	**LRP/Twitter**	**Grad-CAM/Twitter**	**c-EB/Twitter**
Bi-GCN TD/Text	**EBGCN TD/Text**	0.207 ± 0.10	$\mathbf{0.177 \pm 0.15}$	0.211 ± 0.11	$\mathbf{0.173 \pm 0.09}$	0.206 ± 0.10	0.206 ± 0.10
	Bi-GCN BU/Text	0.131 ± 0.09	$\mathbf{0.098 \pm 0.09}$	0.135 ± 0.09	$\mathbf{0.138 \pm 0.07}$	0.144 ± 0.08	0.144 ± 0.08
	EBGCN BU/Text	0.167 ± 0.09	$\mathbf{0.078 \pm 0.09}$	0.160 ± 0.09	$\mathbf{0.145 \pm 0.07}$	0.146 ± 0.08	0.146 ± 0.08
EBGCN TD/Text	**Bi-GCN BU/Text**	0.177 ± 0.10	0.174 ± 0.10	$\mathbf{0.163 \pm 0.10}$	$\mathbf{0.143 \pm 0.08}$	0.180 ± 0.10	0.145 ± 0.08
	EBGCN BU/Text	0.321 ± 0.30	0.380 ± 0.31	$\mathbf{0.270 \pm 0.29}$	$\mathbf{0.154 \pm 0.08}$	0.160 ± 0.11	0.156 ± 0.09
Bi-GCN BU/Text	**EBGCN BU/Text**	$\mathbf{0.211 \pm 0.11}$	0.356 ± 0.14	0.213 ± 0.11	$\mathbf{0.162 \pm 0.08}$	0.297 ± 0.11	0.167 ± 0.09
Bi-GCN TD/Edge	**EBGCN TD/Edge**	0.192 ± 0.09	—	—	0.155 ± 0.08	—	—
	Bi-GCN BU/Edge	0.187 ± 0.10			0.149 ± 0.08		
	EBGCN BU/Edge	0.186 ± 0.10			0.149 ± 0.08		
EBGCN TD/Edge	**Bi-GCN BU/Edge**	0.363 ± 0.32	—	—	0.194 ± 0.10	—	—
	EBGCN BU/Edge	0.335 ± 0.29			0.201 ± 0.09		
Bi-GCN BU/Edge	**EBGCN BU/Edge**	0.369 ± 0.29	—	—	0.237 ± 0.12	—	—

TABLE II: GCN-wise Szymkiewicz-Simpson Similarities for PHEME and Twitter15/16

Model/Output		Method/Dataset					
Anchor	**Comparison**	**LRP/PHME**	**Grad-CAM/PHEME**	**c-EB/PHEME**	**LRP/Twitter**	**Grad-CAM/Twitter**	**c-EB/Twitter**
Bi-GCN TD/Text	**EBGCN TD/Text**	0.462 ± 0.27	$\mathbf{0.380 \pm 0.27}$	0.438 ± 0.30	$\mathbf{0.286 \pm 0.12}$	0.492 ± 0.15	0.332 ± 0.13
	Bi-GCN BU/Text	0.222 ± 0.14	$\mathbf{0.167 \pm 0.14}$	0.227 ± 0.14	$\mathbf{0.236 \pm 0.11}$	0.393 ± 0.18	0.244 ± 0.12
	EBGCN BU/Text	0.367 ± 0.23	$\mathbf{0.196 \pm 0.21}$	0.348 ± 0.27	0.246 ± 0.11	$\mathbf{0.233 \pm 0.16}$	0.246 ± 0.12
EBGCN TD/Text	**Bi-GCN BU/Text**	0.429 ± 0.33	0.439 ± 0.30	$\mathbf{0.374 \pm 0.33}$	$\mathbf{0.243 \pm 0.12}$	0.295 ± 0.13	0.246 ± 0.12
	EBGCN BU/Text	0.466 ± 0.33	0.531 ± 0.37	$\mathbf{0.388 \pm 0.37}$	$\mathbf{0.258 \pm 0.12}$	0.263 ± 0.15	0.261 ± 0.13
Bi-GCN BU/Text	**EBGCN BU/Text**	0.456 ± 0.29	0.677 ± 0.21	$\mathbf{0.446 \pm 0.31}$	$\mathbf{0.271 \pm 0.12}$	0.432 ± 0.20	0.277 ± 0.12
Bi-GCN TD/Edge	**EBGCN TD/Edge**	0.457 ± 0.29	—	—	0.260 ± 0.12	—	—
	Bi-GCN BU/Edge	0.456 ± 0.30			0.254 ± 0.12		
	EBGCN BU/Edge	0.403 ± 0.27			0.252 ± 0.12		
EBGCN TD/Edge	**Bi-GCN BU/Edge**	0.505 ± 0.32	—	—	0.315 ± 0.14	—	—
	EBGCN BU/Edge	0.482 ± 0.30			0.325 ± 0.13		
Bi-GCN BU/Edge	**EBGCN BU/Edge**	0.536 ± 0.28	—	—	0.371 ± 0.15	—	—

contribution lists obtained in the previous steps to determine the following relationships:

- **Out Degree:** By assessing the similarity between the top-ranked nodes in the contribution list and the out degree list, we can infer the model's emphasis on the dispersion within a node's immediate neighbourhood. A higher similarity indicates that the model has effectively learned this dispersion feature.
- **Betweenness:** Evaluating the resemblance between the top-ranked nodes in the contribution list and the betweenness list helps us understand the model's prioritization of nodes that significantly influence a larger number of downstream nodes. This signifies dispersion over the entire graph, indicating the model's recognition of nodes that serve as conduits to the rest of the graph beyond the immediate neighbourhood.
- **Closeness/Farness:** Analyzing the overlap between the top-ranked nodes in the contribution list and the nodes in the BU graph with low closeness provides insights into the model's ability to distinguish graphs based on longer propagation. Conversely, if there is a significant overlap with nodes having high farness (reciprocal of closeness), it indicates the model's capability to recognize graphs with shorter propagation.

By examining these relationships, we can better understand how the model captures and learns various propagation and dispersion features within the graph.

V. Experiments

In this section, we outline the data pre-processing and training methodology employed for the two models under investigation. Subsequently, we provide details of the experiments conducted using the trained models. Finally, we present the obtained results and provide a comprehensive analysis of our findings.

A. Data Pre-processing and Training Procedure

We utilized the raw data from the PHEME dataset provided by Kochkina et al. [13] to generate tweet embeddings. To achieve this, we employed a multilingual BERT model pre-trained on Wikipedia articles [29]. The tweet embeddings were obtained using the pooled representation used by BERT in the next sentence prediction. Given that some events in the PHEME dataset involve languages other than English, we opted for a multilingual BERT. To accommodate the new input vector size, we made slight modifications to the original models by reducing the input dimension from 5000 to 768. For the Twitter 15/16 dataset, we could not obtain the raw data as most of the tweets were deleted by Twitter, so we used the preprocessed data generated by Ma et al. [6]

For training both models, we largely followed the default settings as per the source code obtained from their respective GitHub repositories. Both models featured a hidden layer of size 64, with a dropout rate of 0.2 for both TD and BU graphs. The learning rate was set at 0.0005, with a weight decay of 0.0001, and a mini-batch size of 32. We trained both models for a maximum of 200 epochs, employing early termination when the validation loss remained stagnant for 10 epochs. To ensure optimal model training under these settings, we implemented a learning rate scheduler that reduces the learning rate upon reaching a loss plateau. Additionally, due to the class imbalance present in the PHEME dataset, we incorporated class weighting into the loss function during

TABLE III: Bi-GCN Classwise Jaccard Similarities for PHEME and Twitter15/16

Class/Model/Output		**Method/Datatset**					
Anchor	**Comparison**	**LRP/PHEME**	**Grad-CAM/PHEME**	**c-EB/PHEME**	**LRP/Twitter**	**Grad-CAM/Twitter**	**c-EB/Twitter**
NR TD/Text	**FR TD/Text**	0.175 ± 0.10	0.473 ± 0.22	0.177 ± 0.11	0.161 ± 0.09	0.423 ± 0.14	0.174 ± 0.10
	TR TD/Text	0.170 ± 0.10	0.458 ± 0.22	0.172 ± 0.11	0.157 ± 0.08	0.452 ± 0.15	0.171 ± 0.10
	UR TD/Text	0.170 ± 0.10	0.470 ± 0.20	0.176 ± 0.11	0.158 ± 0.08	0.462 ± 0.15	0.172 ± 0.09
FR TD/Text	**TR TD/Text**	0.928 ± 0.12	0.807 ± 0.17	0.599 ± 0.22	0.754 ± 0.24	0.567 ± 0.21	0.699 ± 0.24
	UR TD/Text	0.926 ± 0.13	0.606 ± 0.21	0.646 ± 0.22	0.725 ± 0.25	0.575 ± 0.22	0.693 ± 0.24
TR TD/Text	**UR TD/Text**	0.910 ± 0.14	0.648 ± 0.21	0.566 ± 0.22	0.697 ± 0.28	0.628 ± 0.19	0.677 ± 0.28
NR BU/Text	**FR BU/Text**	0.189 ± 0.11	0.319 ± 0.17	0.186 ± 0.11	0.165 ± 0.08	0.364 ± 0.15	0.178 ± 0.09
	TR BU/Text	0.190 ± 0.11	0.321 ± 0.18	0.183 ± 0.11	0.166 ± 0.08	0.373 ± 0.15	0.179 ± 0.09
	UR BU/Text	0.189 ± 0.11	0.321 ± 0.18	0.186 ± 0.11	0.161 ± 0.08	0.377 ± 0.13	0.177 ± 0.09
FR BU/Text	**TR BU/Text**	0.982 ± 0.07	0.452 ± 0.20	0.778 ± 0.18	0.935 ± 0.17	0.489 ± 0.18	0.867 ± 0.19
	UR BU/Text	0.984 ± 0.06	0.465 ± 0.20	0.802 ± 0.17	0.927 ± 0.19	0.483 ± 0.16	0.883 ± 0.17
TR BU/Text	**UR BU/Text**	0.986 ± 0.06	0.466 ± 0.20	0.749 ± 0.19	0.912 ± 0.20	0.479 ± 0.18	0.849 ± 0.21
NR TD/Edge	**FR TD/Edge**	0.168 ± 0.11			0.166 ± 0.08		
	TR TD/Edge	0.167 ± 0.11	—	—	0.162 ± 0.08	—	—
	UR TD/Edge	0.168 ± 0.11			0.164 ± 0.08		
FR TD/Edge	**TR TD/Edge**	0.933 ± 0.13	—	—	0.766 ± 0.25	—	—
	UR TD/Edge	0.920 ± 0.14			0.750 ± 0.25		
TR TD/Edge	**UR TD/Edge**	0.902 ± 0.15	—	—	0.730 ± 0.27	—	—
NR BU/Edge	**FR BU/Edge**	0.405 ± 0.33			0.214 ± 0.19		
	TR BU/Edge	0.405 ± 0.33	—	—	0.226 ± 0.18	—	—
	UR BU/Edge	0.401 ± 0.33			0.212 ± 0.19		
FR BU/Edge	**TR BU/Edge**	0.914 ± 0.19	—	—	0.548 ± 0.31	—	—
	UR BU/Edge	0.918 ± 0.20			0.524 ± 0.32		
TR BU/Edge	**UR BU/Edge**	0.925 ± 0.19	—	—	0.492 ± 0.33	—	—

TABLE IV: Bi-GCN Classwise Szymkiewicz-Simpson Similarities for PHEME and Twitter15/16

Class/Model/Output		**Method/Datatset**					
Anchor	**Comparison**	**LRP/PHEME**	**Grad-CAM/PHEME**	**c-EB/PHEME**	**LRP/Twitter**	**Grad-CAM/Twitter**	**c-EB/Twitter**
NR TD/Text	**FR TD/Text**	0.284 ± 0.15	0.615 ± 0.20	0.288 ± 0.15	0.269 ± 0.12	0.582 ± 0.13	0.285 ± 0.13
	TR TD/Text	0.285 ± 0.15	0.600 ± 0.20	0.280 ± 0.15	0.263 ± 0.12	0.608 ± 0.14	0.281 ± 0.13
	UR TD/Text	0.285 ± 0.15	0.617 ± 0.18	0.285 ± 0.15	0.264 ± 0.12	0.618 ± 0.14	0.284 ± 0.13
FR TD/Text	**TR TD/Text**	0.958 ± 0.08	0.882 ± 0.11	0.724 ± 0.18	0.833 ± 0.20	0.700 ± 0.18	0.795 ± 0.20
	UR TD/Text	0.956 ± 0.08	0.733 ± 0.17	0.763 ± 0.17	0.810 ± 0.21	0.705 ± 0.19	0.793 ± 0.19
TR TD/Text	**UR TD/Text**	0.946 ± 0.09	0.765 ± 0.17	0.696 ± 0.19	0.785 ± 0.23	0.754 ± 0.16	0.770 ± 0.23
NR BU/Text	**FR BU/Text**	0.304 ± 0.15	0.459 ± 0.19	0.299 ± 0.15	0.275 ± 0.12	0.517 ± 0.16	0.293 ± 0.12
	TR BU/Text	0.305 ± 0.15	0.459 ± 0.20	0.297 ± 0.15	0.276 ± 0.12	0.526 ± 0.16	0.295 ± 0.12
	UR BU/Text	0.305 ± 0.15	0.460 ± 0.20	0.300 ± 0.15	0.268 ± 0.12	0.530 ± 0.16	0.292 ± 0.12
FR BU/Text	**TR BU/Text**	0.990 ± 0.04	0.596 ± 0.19	0.863 ± 0.12	0.956 ± 0.12	0.636 ± 0.17	0.914 ± 0.15
	UR BU/Text	0.991 ± 0.04	0.610 ± 0.19	0.880 ± 0.11	0.948 ± 0.15	0.635 ± 0.15	0.927 ± 0.13
TR BU/Text	**UR BU/Text**	0.992 ± 0.04	0.610 ± 0.19	0.843 ± 0.13	0.937 ± 0.16	0.628 ± 0.17	0.900 ± 0.16
NR TD/Edge	**FR TD/Edge**	0.275 ± 0.15			0.276 ± 0.12		
	TR TD/Edge	0.273 ± 0.15	—	—	0.270 ± 0.12	—	—
	UR TD/Edge	0.274 ± 0.15			0.274 ± 0.12		
FR TD/Edge	**TR TD/Edge**	0.960 ± 0.08	—	—	0.840 ± 0.20	—	—
	UR TD/Edge	0.952 ± 0.09			0.829 ± 0.20		
TR TD/Edge	**UR TD/Edge**	0.941 ± 0.10	—	—	0.810 ± 0.22	—	—
NR BU/Edge	**FR BU/Edge**	0.544 ± 0.32			0.321 ± 0.23		
	TR BU/Edge	0.543 ± 0.32	—	—	0.338 ± 0.23	—	—
	UR BU/Edge	0.539 ± 0.32			0.315 ± 0.23		
FR BU/Edge	**TR BU/Edge**	0.960 ± 0.11	—	—	0.652 ± 0.29	—	—
	UR BU/Edge	0.963 ± 0.11			0.628 ± 0.31		
TR BU/Edge	**UR BU/Edge**	0.966 ± 0.10	—	—	0.592 ± 0.32	—	—

training. However, equal weighting was used during validation to obtain a model with improved performance across all classes. To train our model, we adopted a random stratified 5-fold split when training on Twitter 15/16 data, consistent with previous rumour detection studies [5]–[10]. When training on PHEME data, we adopted an event-wise 9-fold split [10].

B. Contribution Analysis

To assess the contribution of textual and graphical features at the GCN level in both Bi-GCN and EBGCN models, we conducted a series of experiments. Additionally, we conducted a class-wise contribution analysis to evaluate if the models effectively learned distinct features for each class and to measure their uniqueness across classes. In our analysis, each model is subjected to our modified LRP technique. For a comprehensive comparative study, we also employ Grad-CAM [18] and Contrastive Excitation Backpropagation (c-EB) [19], [20] as baseline methodologies. The rationale behind selecting Grad-CAM and c-EB lies in their recognition as quintessential examples of gradient-based and decomposition-based explainability techniques, respectively. These techniques are pivotal for understanding how decisions are made within the model. In contrast, we consciously choose not to include perturbation-based and surrogate-based methods like GNNExplainer [22].

TABLE V: EBGCN Classwise Jaccard Similarities for PHEME and Twitter15/16

Class/Model/Output		**Method/Datatset**					
Anchor	**Comparison**	**LRP/PHEME**	**Grad-CAM/PHEME**	**c-EB/PHEME**	**LRP/Twitter**	**Grad-CAM/Twitter**	**c-EB/Twitter**
NR TD/Text	**FR TD/Text**	0.415 ± 0.37	0.548 ± 0.36	0.374 ± 0.32	0.219 ± 0.20	0.316 ± 0.17	0.231 ± 0.19
	TR TD/Text	0.411 ± 0.37	0.657 ± 0.33	0.368 ± 0.32	0.198 ± 0.18	0.339 ± 0.16	0.222 ± 0.17
	UR TD/Text	0.408 ± 0.36	0.653 ± 0.33	0.372 ± 0.33	0.205 ± 0.18	0.311 ± 0.17	0.224 ± 0.18
FR TD/Text	**TR TD/Text**	0.958 ± 0.12	0.626 ± 0.34	0.803 ± 0.26	0.799 ± 0.26	0.400 ± 0.18	0.665 ± 0.23
	UR TD/Text	0.957 ± 0.12	0.580 ± 0.34	0.816 ± 0.24	0.794 ± 0.26	0.412 ± 0.18	0.658 ± 0.23
TR TD/Text	**UR TD/Text**	0.948 ± 0.13	0.687 ± 0.32	0.799 ± 0.26	0.779 ± 0.28	0.425 ± 0.17	0.652 ± 0.25
NR BU/Text	**FR BU/Text**	0.370 ± 0.36	0.671 ± 0.31	0.336 ± 0.31	0.208 ± 0.19	0.307 ± 0.18	0.202 ± 0.13
	TR BU/Text	0.370 ± 0.36	0.677 ± 0.30	0.330 ± 0.31	0.207 ± 0.18	0.308 ± 0.19	0.202 ± 0.13
	UR BU/Text	0.369 ± 0.36	0.672 ± 0.30	0.328 ± 0.31	0.207 ± 0.18	0.317 ± 0.18	0.203 ± 0.13
FR BU/Text	**TR BU/Text**	0.990 ± 0.05	0.683 ± 0.31	0.847 ± 0.24	0.925 ± 0.20	0.320 ± 0.17	0.945 ± 0.13
	UR BU/Text	0.988 ± 0.06	0.674 ± 0.31	0.845 ± 0.24	0.915 ± 0.23	0.361 ± 0.19	0.954 ± 0.11
TR BU/Text	**UR BU/Text**	0.987 ± 0.06	0.719 ± 0.28	0.825 ± 0.27	0.912 ± 0.22	0.366 ± 0.18	0.936 ± 0.15
NR TD/Edge	**FR TD/Edge**	0.428 ± 0.38	—	—	0.216 ± 0.20	—	—
	TR TD/Edge	0.424 ± 0.37			0.200 ± 0.18		
	UR TD/Edge	0.422 ± 0.37			0.200 ± 0.18		
FR TD/Edge	**TR TD/Edge**	0.962 ± 0.12	—	—	0.796 ± 0.26	—	—
	UR TD/Edge	0.963 ± 0.12			0.779 ± 0.27		
TR TD/Edge	**UR TD/Edge**	0.956 ± 0.13	—	—	0.774 ± 0.28	—	—
NR BU/Edge	**FR BU/Edge**	0.383 ± 0.35	—	—	0.221 ± 0.24	—	—
	TR BU/Edge	0.375 ± 0.34			0.221 ± 0.23		
	UR BU/Edge	0.367 ± 0.33			0.210 ± 0.23		
FR BU/Edge	**TR BU/Edge**	0.896 ± 0.21	—	—	0.867 ± 0.24	—	—
	UR BU/Edge	0.882 ± 0.22			0.849 ± 0.26		
TR BU/Edge	**UR BU/Edge**	0.882 ± 0.21	—	—	0.832 ± 0.28	—	—

TABLE VI: EBGCN Classwise Szymkiewicz-Simpson Similarities for PHEME and Twitter15/16

Class/Model/Output		**Method/Datatset**					
Anchor	**Comparison**	**LRP/PHEME**	**Grad-CAM/PHEME**	**c-EB/PHEME**	**LRP/Twitter**	**Grad-CAM/Twitter**	**c-EB/Twitter**
NR TD/Text	**FR TD/Text**	0.521 ± 0.38	0.689 ± 0.31	0.554 ± 0.36	0.317 ± 0.25	0.458 ± 0.18	0.354 ± 0.25
	TR TD/Text	0.518 ± 0.38	0.787 ± 0.27	0.533 ± 0.36	0.298 ± 0.22	0.484 ± 0.18	0.344 ± 0.23
	UR TD/Text	0.517 ± 0.38	0.786 ± 0.26	0.534 ± 0.36	0.306 ± 0.23	0.451 ± 0.19	0.348 ± 0.23
FR TD/Text	**TR TD/Text**	0.982 ± 0.05	0.763 ± 0.27	0.904 ± 0.17	0.857 ± 0.22	0.550 ± 0.18	0.783 ± 0.19
	UR TD/Text	0.980 ± 0.06	0.727 ± 0.28	0.916 ± 0.15	0.856 ± 0.21	0.562 ± 0.18	0.775 ± 0.19
TR TD/Text	**UR TD/Text**	0.976 ± 0.07	0.810 ± 0.24	0.888 ± 0.19	0.840 ± 0.24	0.578 ± 0.17	0.762 ± 0.21
NR BU/Text	**FR BU/Text**	0.463 ± 0.39	0.787 ± 0.27	0.495 ± 0.36	0.310 ± 0.23	0.443 ± 0.20	0.320 ± 0.17
	TR BU/Text	0.462 ± 0.39	0.801 ± 0.26	0.479 ± 0.35	0.309 ± 0.23	0.442 ± 0.21	0.318 ± 0.17
	UR BU/Text	0.462 ± 0.39	0.794 ± 0.26	0.479 ± 0.35	0.309 ± 0.23	0.454 ± 0.20	0.320 ± 0.17
FR BU/Text	**TR BU/Text**	0.995 ± 0.03	0.801 ± 0.26	0.926 ± 0.16	0.944 ± 0.16	0.461 ± 0.19	0.966 ± 0.10
	UR BU/Text	0.994 ± 0.03	0.794 ± 0.26	0.926 ± 0.16	0.933 ± 0.19	0.502 ± 0.21	0.973 ± 0.08
TR BU/Text	**UR BU/Text**	0.993 ± 0.03	0.837 ± 0.22	0.894 ± 0.21	0.934 ± 0.18	0.511 ± 0.20	0.959 ± 0.11
NR TD/Edge	**FR TD/Edge**	0.530 ± 0.39	—	—	0.314 ± 0.25	—	—
	TR TD/Edge	0.527 ± 0.38			0.301 ± 0.22		
	UR TD/Edge	0.525 ± 0.38			0.301 ± 0.23		
FR TD/Edge	**TR TD/Edge**	0.983 ± 0.06	—	—	0.857 ± 0.21	—	—
	UR TD/Edge	0.984 ± 0.06			0.842 ± 0.23		
TR TD/Edge	**UR TD/Edge**	0.979 ± 0.07	—	—	0.836 ± 0.24	—	—
NR BU/Edge	**FR BU/Edge**	0.490 ± 0.36	—	—	0.308 ± 0.28	—	—
	TR BU/Edge	0.490 ± 0.35			0.314 ± 0.27		
	UR BU/Edge	0.484 ± 0.35			0.297 ± 0.27		
FR BU/Edge	**TR BU/Edge**	0.954 ± 0.10	—	—	0.905 ± 0.19	—	—
	UR BU/Edge	0.946 ± 0.11			0.887 ± 0.22		
TR BU/Edge	**UR BU/Edge**	0.947 ± 0.11	—	—	0.873 ± 0.24	—	—

This exclusion is based on the premise that such techniques do not effectively trace the information flow within the model, which is essential for our objective of comprehending the internal mechanics of these Graph Convolutional Network (GCN)-based models. Our approach is thus tailored to provide insight into the actual inner workings of the models, aligning with the specific requirements of our study.

Using the obtained contribution maps, we identified nodes with values higher than the upper quartile threshold. To compare the sets of selected nodes, we computed Jaccard Similarity and Szymkiewicz-Simpson Similarity. Each experiment involved analyzing the respective models for each split on the test set. To ensure a meaningful comparison, we excluded data samples with fewer than 20 nodes in their graph. After filtering out graphs with less than 20 nodes, we are left with 1799 remaining graphs for PHEME and 1294 remaining graphs for Twitter 15/16, providing sufficient data for our analysis. To facilitate the examination of similarity distributions, we represented the Jaccard and Szymkiewicz-Simpson Similarities using split violin plots. These plots depict the quartiles (upper, lower, and median) using dashed lines in green, blue, and black, respectively, while the mean is represented by a solid red line.

1) GCN-wise Contribution: First, we examine the GCN-wise text contributions to determine if there are notable differences in the types of text features learned by the GCNs for each of the four classes. Our findings are summarized in Tables I and II. In these tables, we primarily showcase the results obtained from our modified LRP, as the Grad-CAM and c-EB baselines do not effectively identify graphical contributions. To enhance our understanding of these findings, we present visual distributions derived from the modified LRP applied to the PHEME dataset. The choice of PHEME for in-depth analysis is driven by two factors. Firstly, the preprocessing procedure for the Twitter15/16 dataset is not transparent to us, rendering it less suitable for a detailed study. Secondly, Twitter15/16 relies on vector representations of a 5000-word vocabulary, leading to potential data leakage between training and test sets. This leakage stems from the specific nature of the vector representations employed. Despite these considerations, it is noteworthy that the overall results obtained from our method align closely with those from the baseline techniques.

Surprisingly, we observe that the similarity distributions across the four classes exhibit remarkable similarities across the four GCNs. For brevity, we present the consolidated results for all classes in Fig. 2. Next, we investigate the GCN-wise edge contributions to assess whether there are distinct disparities in the types of graph features learned by the GCNs for each of the four classes. Similar to the text contributions, we observe consistent trends in the similarity distributions and provide the compiled results in Fig. 3. Upon analyzing the figures, we observe that, in general, the TD and BU GCNs in EBGCN exhibit higher similarity in terms of learned text and graph features compared to those in Bi-GCN. However, it is important to note that even within the comparisons between GCNs, the similarities remain relatively low. This indicates that each GCN indeed learns distinct textual and graphical features.

2) Class-wise Text Contribution: To gain deeper insights into the class differences and identify unique text features learned by the GCNs for each class, we compare the top contributors within the same GCN for each class. The results are presented in Table III, IV, V, VI with deeper analysis shown in Fig. 4, 5, 6, and 7. From the figures, we observe that each GCN demonstrates the ability to learn distinct features for the Non-Rumour (NR) class, as indicated by the low similarities when compared to the False-Rumour (FR), True-Rumour (TR), and Unverified-Rumour (UR) classes. However, it is noteworthy that while the TD GCN in Bi-GCN successfully captures more distinct features between the FR, TR, and UR classes, the BU GCN in Bi-GCN and both GCNs in EBGCN struggle to learn distinct text features. Examining Fig. 5, 6, and 7, we observe remarkably high similarities, almost reaching 1, for the comparisons between the three classes. However, for the BU GCN in Bi-GCN, we note that the median and lower quartile values in the similarity distribution for comparisons between NR and FR/TR/UR are higher compared to both the TD and BU GCNs in EBGCN. This suggests that EBGCN exhibits greater differences in textual features learned between the NR class and the FR/TR/UR classes. This finding potentially contributes to the superior performance of EBGCN compared to Bi-GCN.

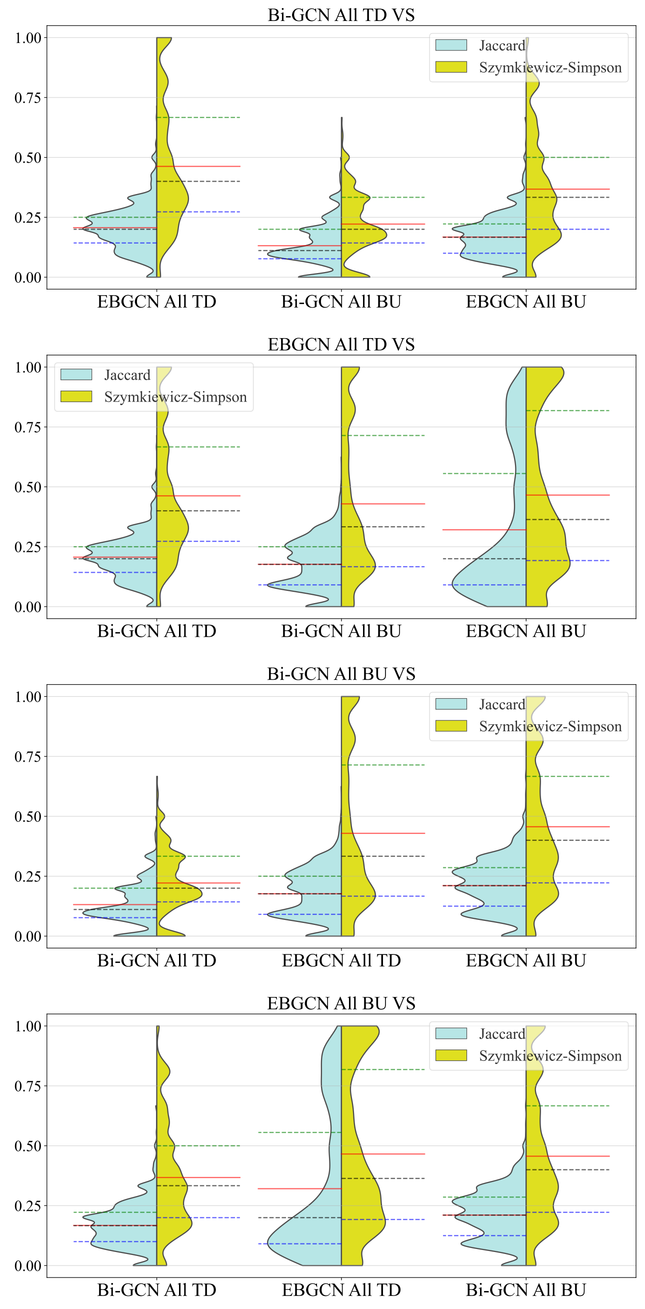

Fig. 2: Similarity between Upper Quartile Text Contributors for all Classes

3) Class-wise Edge Contribution: To examine the differences in class-specific graphical features learned by each GCN, we analyze the top contributors among the edges for each class within the same GCN. The corresponding results are illustrated in Fig. 8, 9, 10, and 11. From the figures, we observe a similar trend in the distributions as observed in the class-specific text contributions. Similar to the text contributions, each GCN demonstrates the ability to learn distinct graphical features for the NR class while exhibiting less distinction between the FR, TR, and UR classes. Notably, the TD GCN in Bi-GCN shows greater capability in learning distinct graphical features across all classes, whereas the other GCNs exhibit

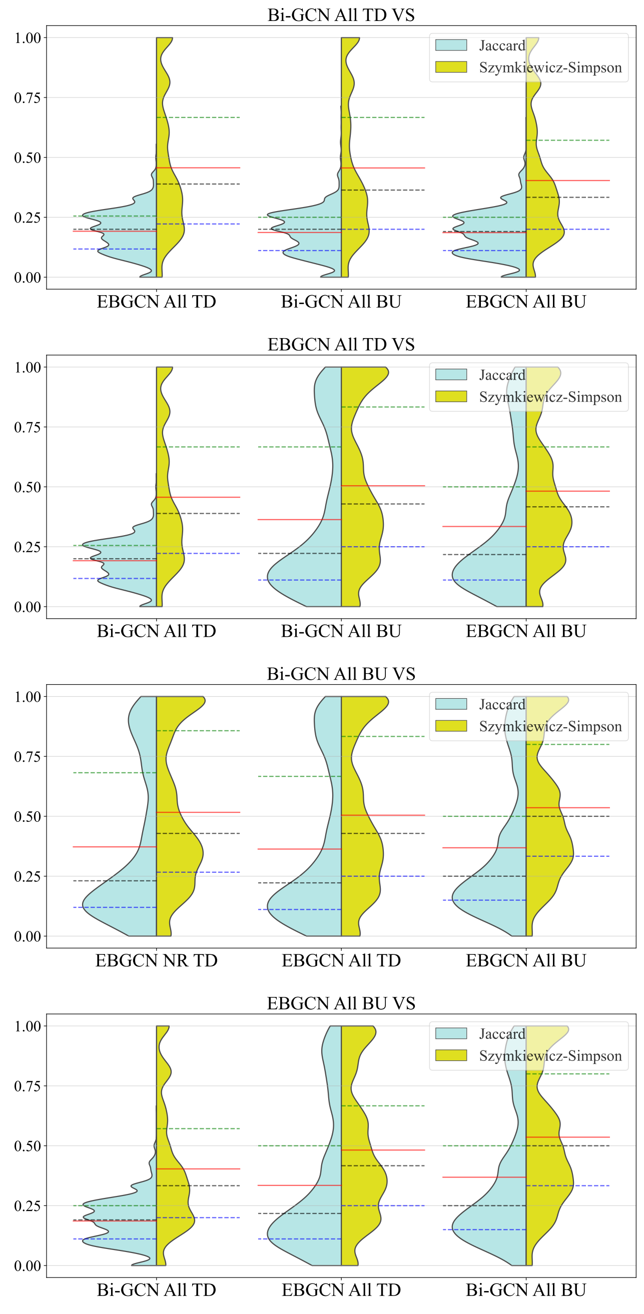

Fig. 3: Similarity between Upper Quartile Edge Contributors for all Classes

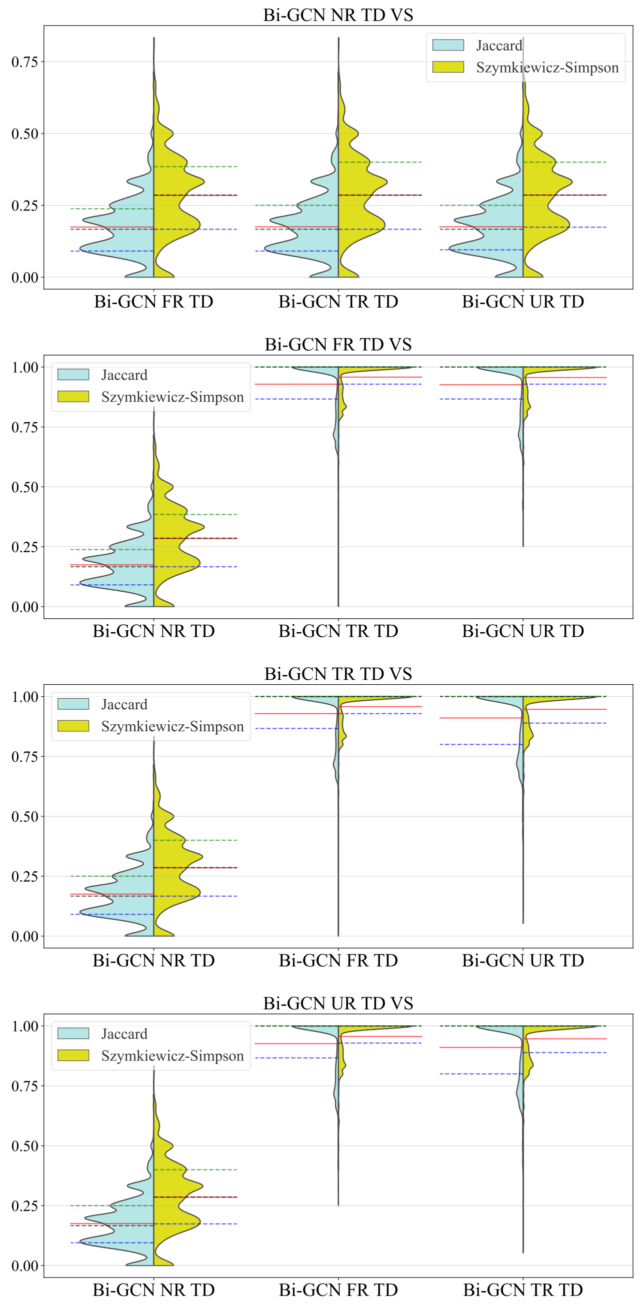

Fig. 4: Similarity between Upper Quartile Text Contributors for Bi-GCN TD GCN

relatively lower distinction in their learned graphical features for the FR, TR, and UR classes. Furthermore, in Fig. 10, we observe even less dissimilarity between the NR class and the other three classes compared to the dissimilarity observed in the text contributions. Conversely, both the TD and BU GCNs in EBGCN demonstrate greater dissimilarity between the NR class and the FR/TR/UR learned graphical features.

C. Importance of Graph Structure

To assess the significance of graph structure, we analyze the class-specific similarities between the upper quartile contributors for each GCN under different modifications. Notably, we observe interesting findings when examining the modifications to graph structure.

In the case of the TD GCN in Bi-GCN, as depicted in Fig. 12, the distributions appear highly similar to each other, except for the upper quartile values. From the figure, it is evident that the top contributors for the modified graph structures bear considerable resemblance to those for the unmodified graph. This similarity is observed across all four classes, suggesting that, for the Bi-GCN TD GCN, graph structure has limited importance as it does not significantly alter the top contributors for all four classes. Conversely, for the TD GCN in EBGCN, Fig. 13 reveals a contrasting result. The similarity distributions

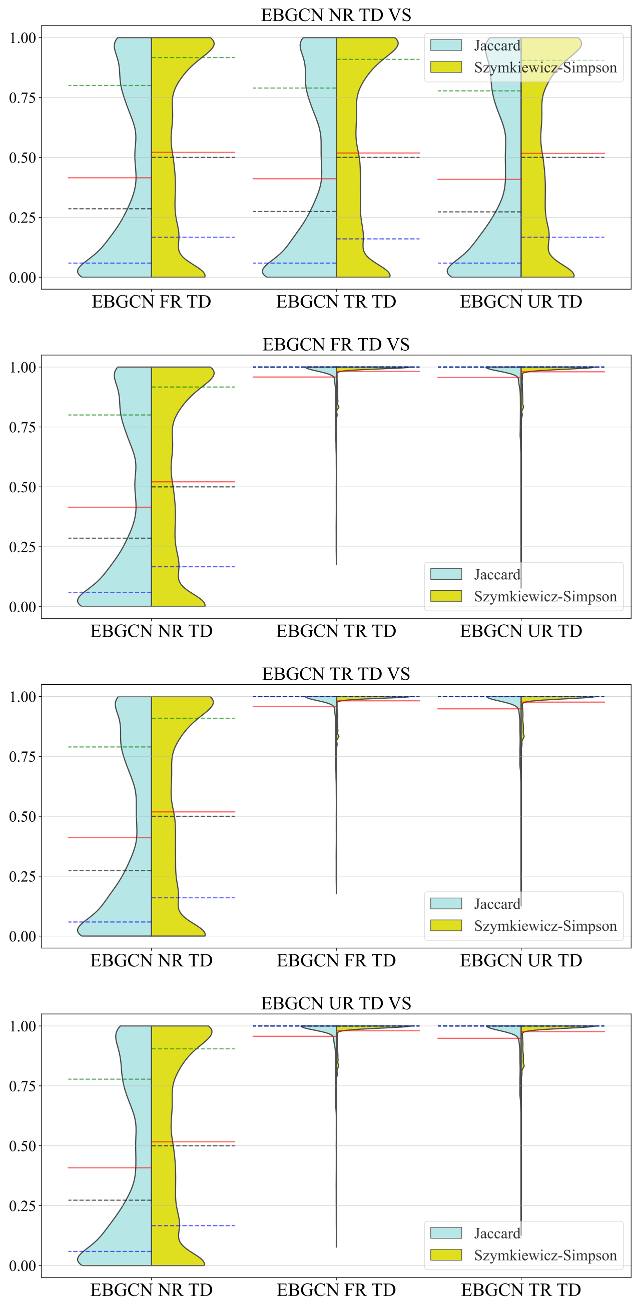

Fig. 5: Similarity between Upper Quartile Text Contributors for EBGCN TD GCN

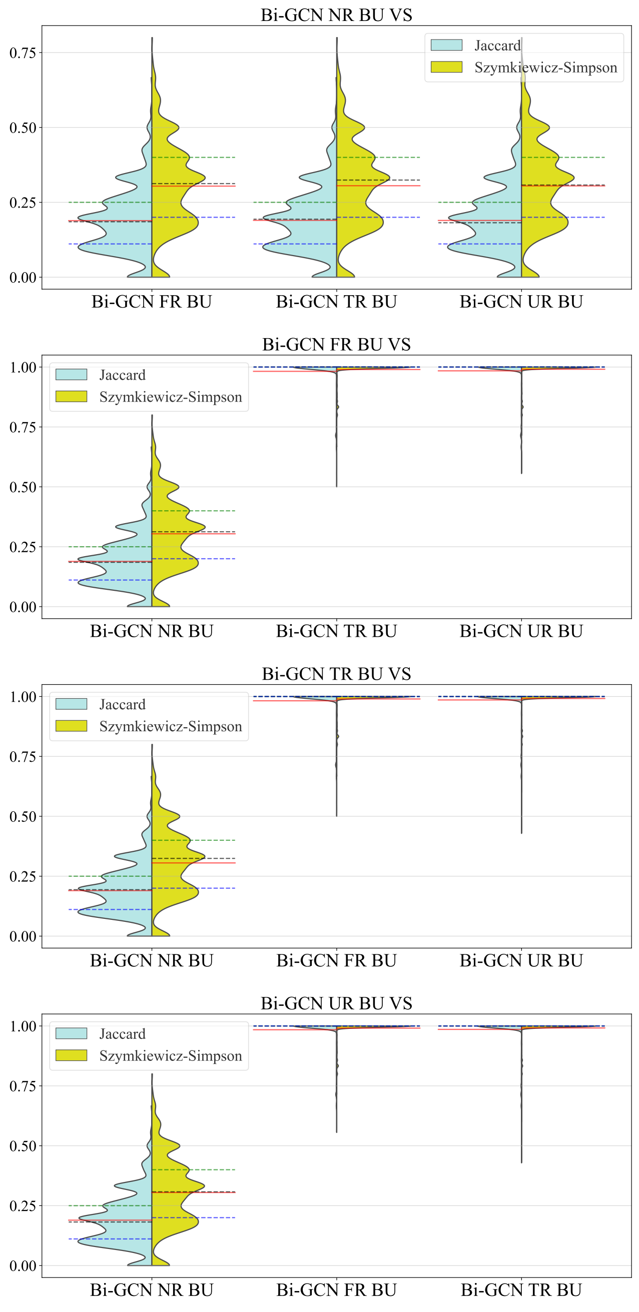

Fig. 6: Similarity between Upper Quartile Text Contributors for Bi-GCN BU GCN

demonstrate that the top contributors for the modified graphs differ considerably from those of the unmodified graph across all four classes. This indicates that graph structure in EBGCN plays a significant role, as it notably influences the results.

Similar trends emerge when analyzing the BU GCNs in both Bi-GCN and EBGCN. Fig. 14 and 15 illustrate that edge deletion leads to distinct top contributors in all four classes. However, an intriguing observation is that setting edge weights to 0.5 and 2.0 produces comparable results. In both cases, the similarities between these modified graphs and the unmodified graph approach 1 across all four classes. This finding suggests that the edge weights used within the GCNs are likely less than 0.5. Notably, in the BU GCN, a node's representation incorporates more information from its neighbours compared to the TD GCN, which only receives information from a single neighbour. The results of the edge deletion experiment indicate that the BU GCNs emphasize the impact of textual contributions on model predictions, as the edges facilitate a richer node representation. Simultaneously, the minimal changes in top contributors for the modified edge weights experiments demonstrate that although edges are important, the BU GCN assigns greater significance to the textual information originating from the node itself.

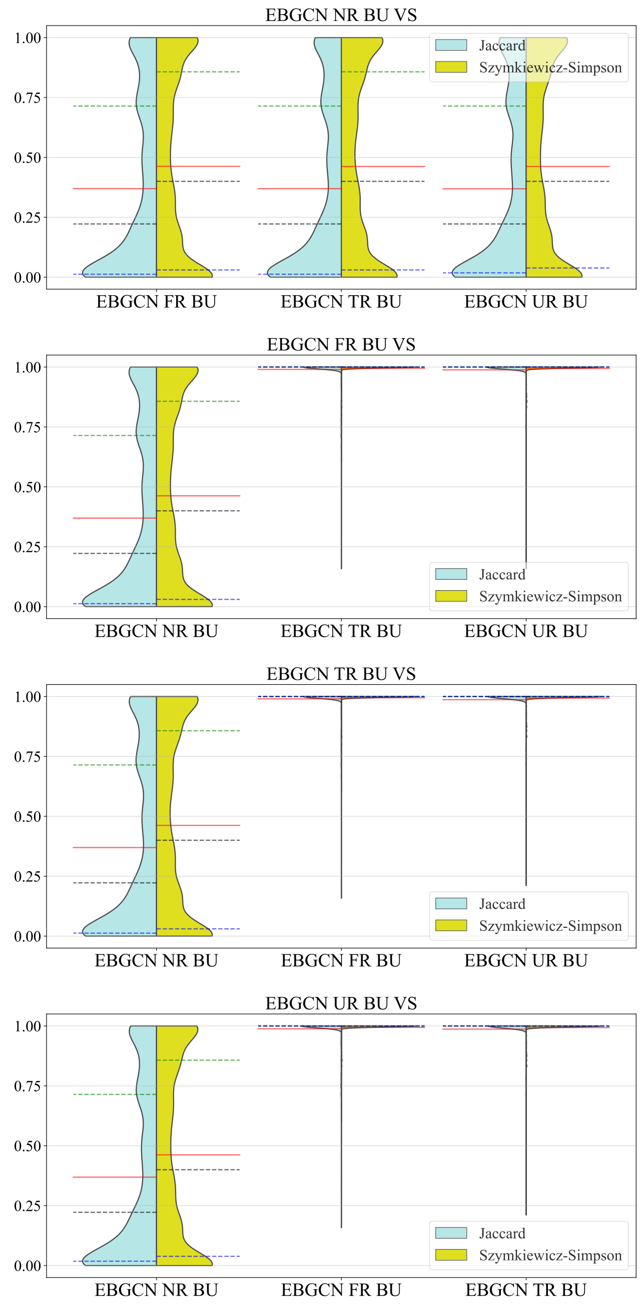

Fig. 7: Similarity between Upper Quartile Text Contributors for EBGCN BU GCN

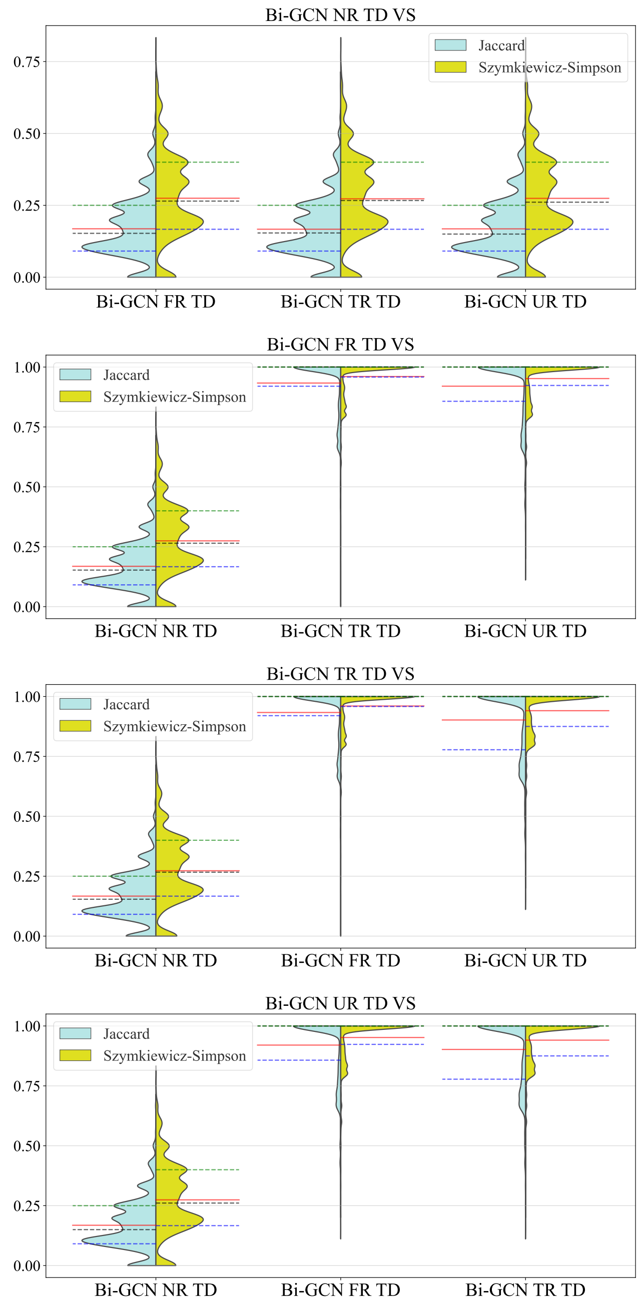

Fig. 8: Similarity between Upper Quartile Edge Contributors for Bi-GCN TD GCN

D. Latent Propagation and Dispersion Features

Finally, to investigate the latent propagation and dispersion features learned by the GCNs, we compare the top contributors for each GCN with the top-ranked nodes based on centrality measures for all four classes. The summarized results are presented in Tables VIII and VII, including the mean and standard deviation. Notably, the similarities across the four classes are quite similar. This finding challenges the assumption that GCNs would be better equipped to capture the unique latent propagation and dispersion features of rumours and non-rumours, consequently leading to improved performance in rumour detection.

In addition to this observation, we find that both the TD and BU GCNs in Bi-GCN exhibit a stronger focus on nodes with high out-degree. This is evident from the higher similarities between the top-ranked nodes for out-degree and the top contributors for the TD and BU GCNs in Bi-GCN, as compared to other centrality measures. Furthermore, we observe that the TD GCN in Bi-GCN pays particular attention to leaf nodes in the graph, as indicated by the second-highest similarities for farness, following out-degree.

In the case of EBGCN, both the TD and BU GCNs demonstrate lower similarities, suggesting a reduced emphasis on propagation and dispersion features. This observation is

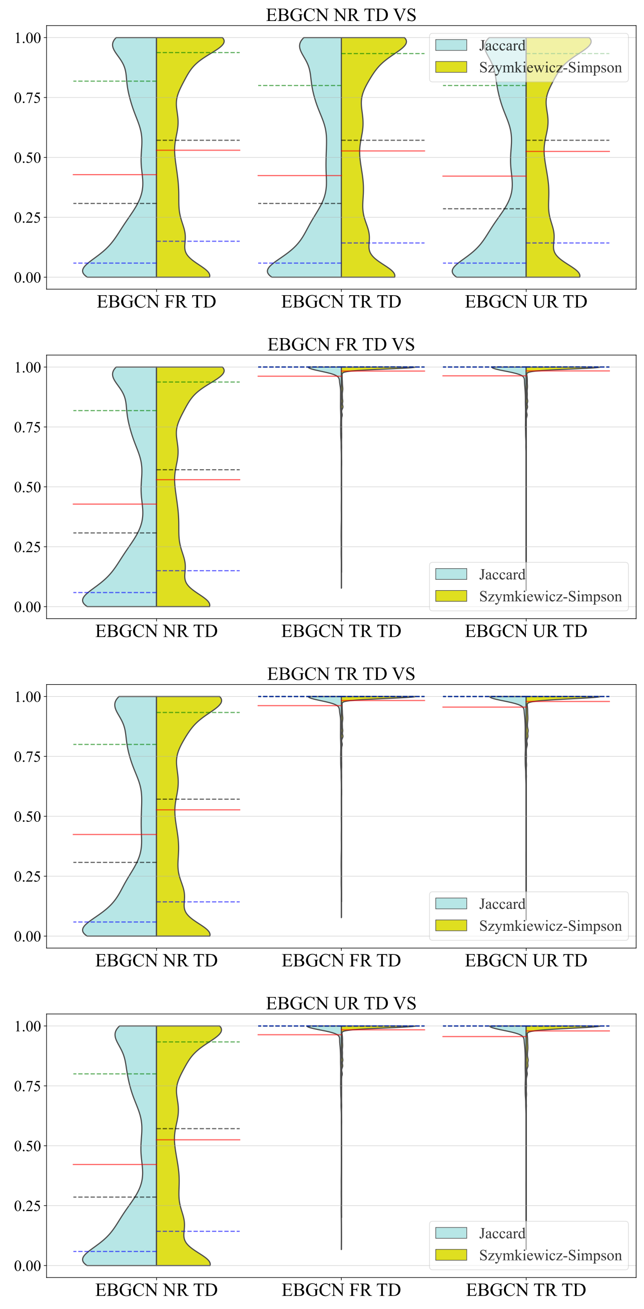

Fig. 9: Similarity between Upper Quartile Edge Contributors for EBGCN TD GCN

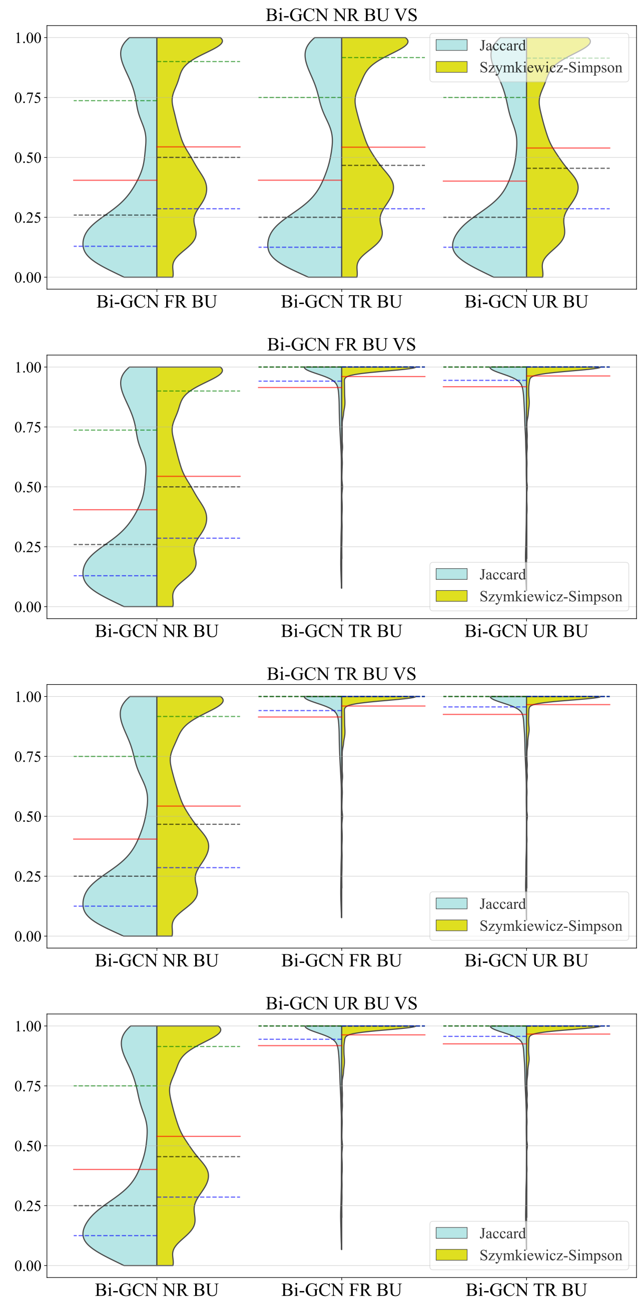

Fig. 10: Similarity between Upper Quartile Edge Contributors for Bi-GCN BU GCN

especially pronounced for the TD GCN in EBGCN, which exhibits notably low similarities across all four centrality measures. These results indicate that EBGCN does not assign as much importance to nodes considered significant by classical centrality measures, implying that its improved performance stems from its ability to better capture textual features. However, an interesting finding is that the farness measure exhibits the highest similarity for the BU GCN in EBGCN, which contrasts with the pattern observed for the BU GCN in Bi-GCN.

VI. Conclusion

The experiments demonstrate that the TD GCN in Bi-GCN learns more distinct textual features among the four GCNs, while the BU GCN in Bi-GCN and both GCNs in EBGCN learn more similar textual features. Similarly, in terms of edge contributions, the BU GCN in Bi-GCN and the two GCNs in EBGCN exhibit greater similarity, whereas the TD GCN in Bi-GCN displays more distinct differences between the classes. This highlights the significance of the edge inference module in EBGCN, as it alters the behaviour of the TD GCN. Furthermore, the results of the experiments evaluating graph importance reveal that both GCNs in EBGCN rely more on

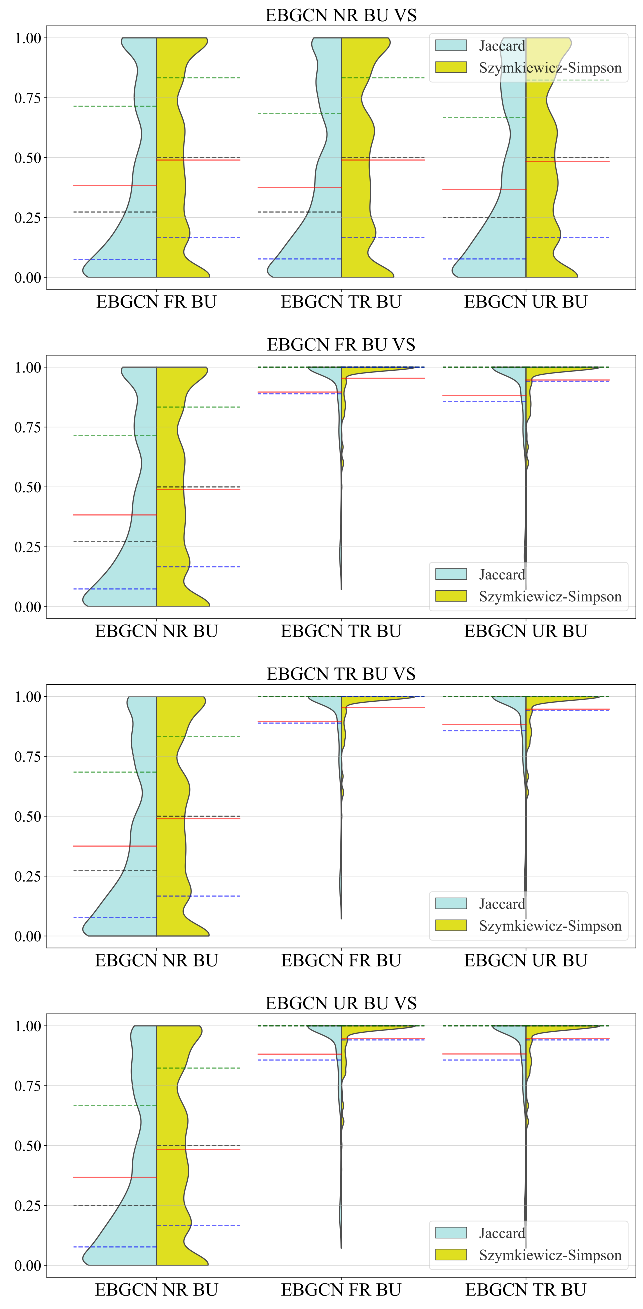

Fig. 11: Similarity between Upper Quartile Edge Contributors for EBGCN BU GCN

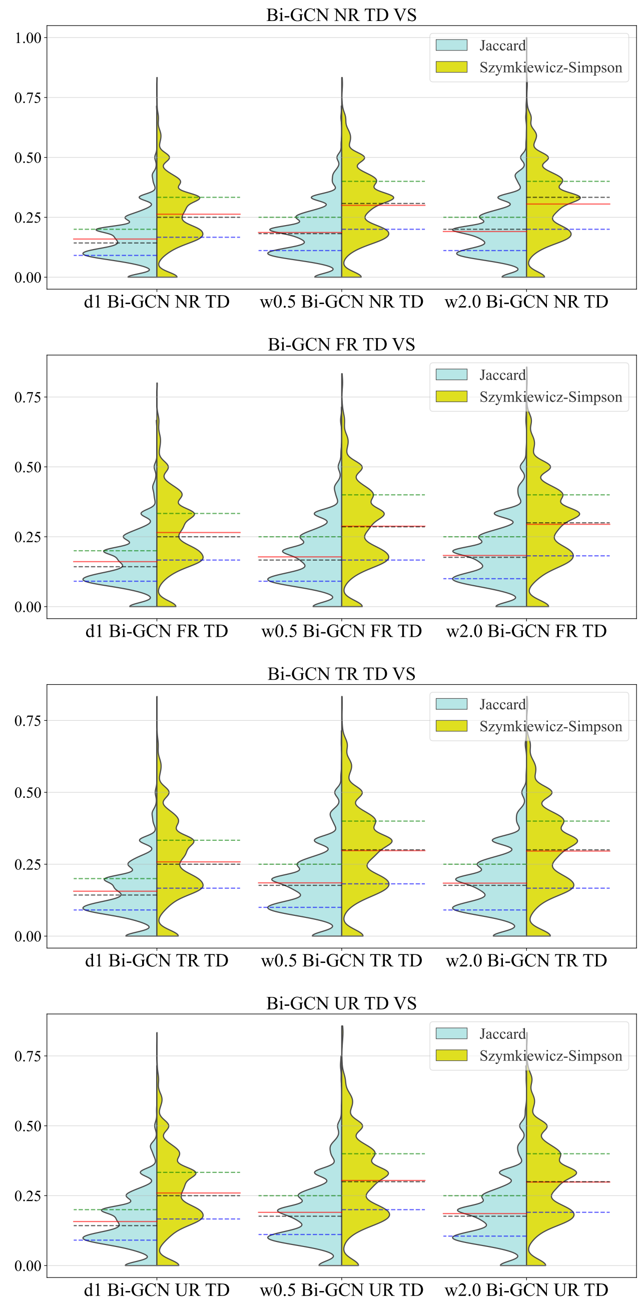

Fig. 12: Similarity between Upper Quartile Contributors for Bi-GCN TD GCN with Modified Graph Structure

neighbouring nodes for their node representations. This, along with the findings for the BU GCN, underscores the importance of neighbouring nodes in learning node representations while demonstrating how edge inference can address issues related to poorer node representations, such as those learned by the TD GCN in Bi-GCN.

Lastly, the experiments on latent propagation and dispersion feature learning demonstrate that EBGCN assigns less importance to nodes considered significant by classical centrality measures compared to Bi-GCN. Both GCNs in Bi-GCN capture dispersion features more effectively than EBGCN, and the TD GCN in Bi-GCN also captures propagation features well. These results indicate that superior performance arises from the learning of better textual features, rather than from the implicit learning of graphical features, as previously believed. Particularly, the edge inference module in EBGCN enables the model to select relevant textual features more effectively compared to Bi-GCN. Furthermore, the results indicate that the improved performance of GCNs, especially EBGCN, stems from better learning of textual features, rather than from learning unique latent propagation and dispersion features, as initially assumed. In the future, we plan to apply the same techniques to analyse GCNs in Fake News Detection.

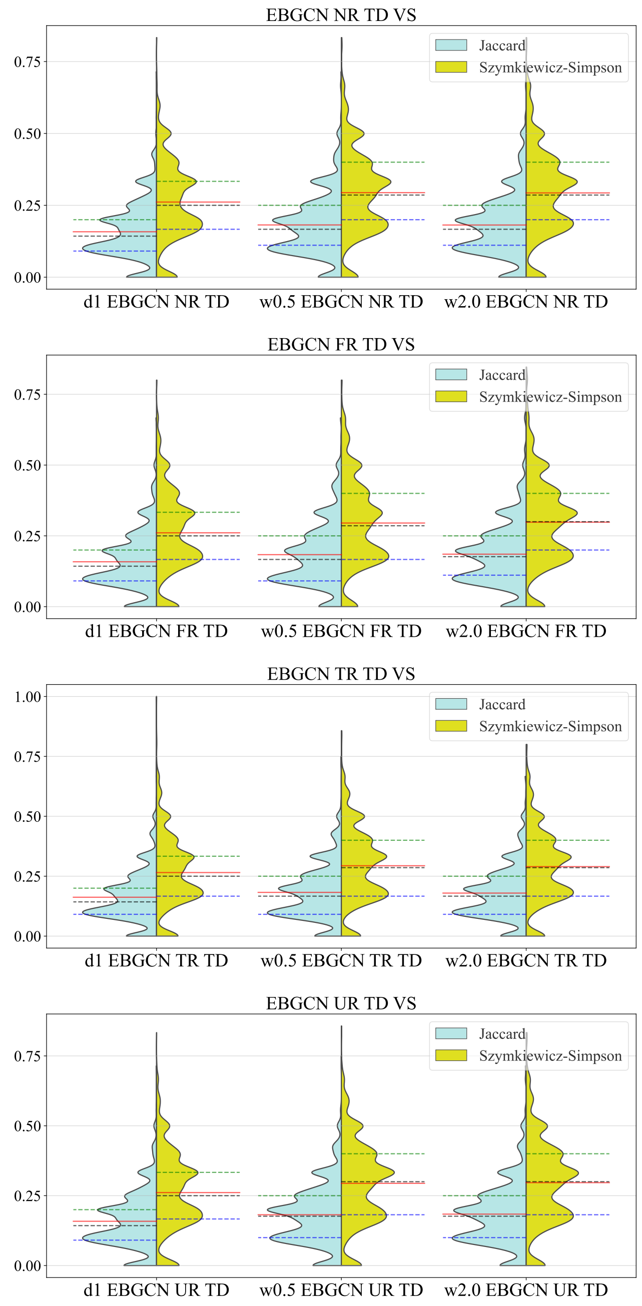

Fig. 13: Similarity between Upper Quartile Contributors for EBGCN TD GCN with Modified Graph Structure

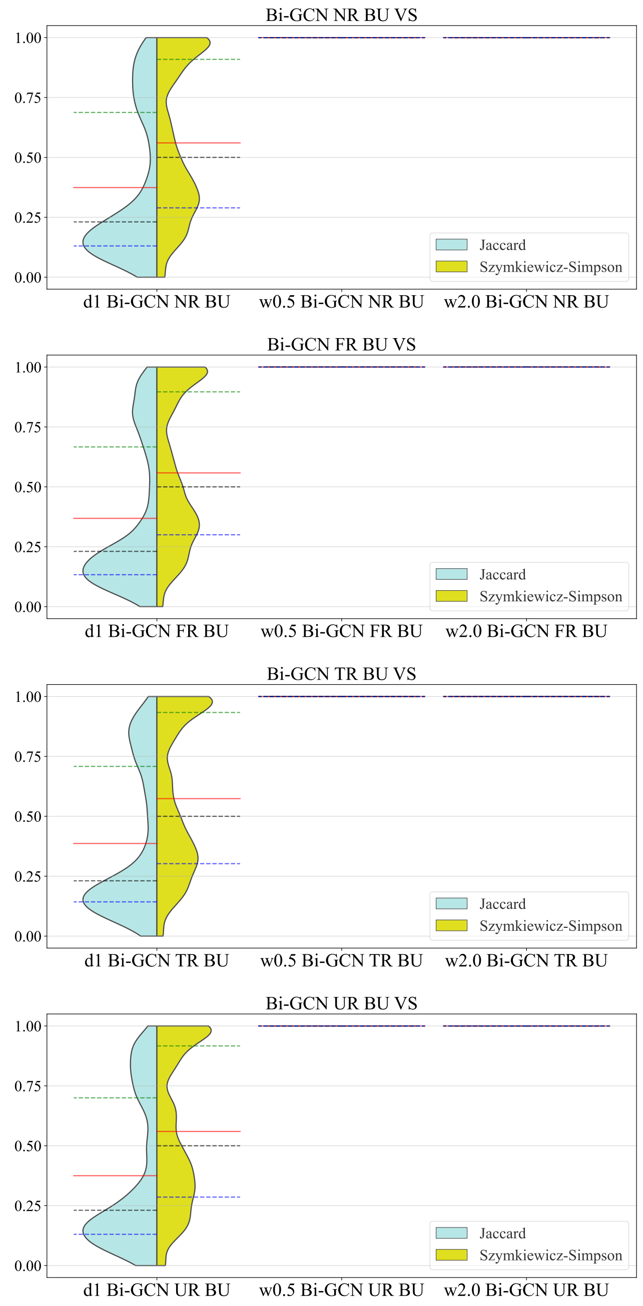

Fig. 14: Similarity between Upper Quartile Contributors for Bi-GCN BU GCN with Modified Graph Structure

References

[1] S. Vosoughi, D. Roy, and S. Aral, "The spread of true and false news online," *Science*, vol. 359, no. 6380, pp. 1146–1151, 2018. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.aap9559

[2] M. R. Awal, M. D. Nguyen, R. K.-W. Lee, and K. T. W. Choo, "Muscat: Multilingual rumor detection in social media conversations," in *2022 IEEE International Conference on Big Data (Big Data)*. IEEE, 2022, pp. 455–464.

[3] B. Liu, X. Sun, Q. Meng, X. Yang, Y. Lee, J. Cao, J. Luo, and R. K.-W. Lee, "Nowhere to hide: Online rumor detection based on retweeting graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[4] S. Tasnim, M. M. Hossain, and H. Mazumder, "Impact of rumors and misinformation on covid-19 in social media," *Journal of Preventive Medicine and Public Health*, vol. 53, no. 3, p. 171–174, 2020. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/32498140/

[5] J. Ma, W. Gao, P. Mitra, S. Kwon, B. J. Jansen, K.-F. Wong, and M. Cha, "Detecting rumors from microblogs with recurrent neural networks," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, ser. IJCAI'16. AAAI Press, 2016, p. 3818–3824.

[6] J. Ma, W. Gao, and K.-F. Wong, "Detect rumors in microblog posts using propagation structure via kernel learning," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 708–717. [Online]. Available: https://aclanthology.org/P17-1066

[7] ——, "Rumor detection on Twitter with tree-structured recursive neural networks," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 1980–1989. [Online]. Available: https://aclanthology.org/P18-1184

[8] J. Ma, W. Gao, S. Joty, and K.-F. Wong, "An attention-based rumor

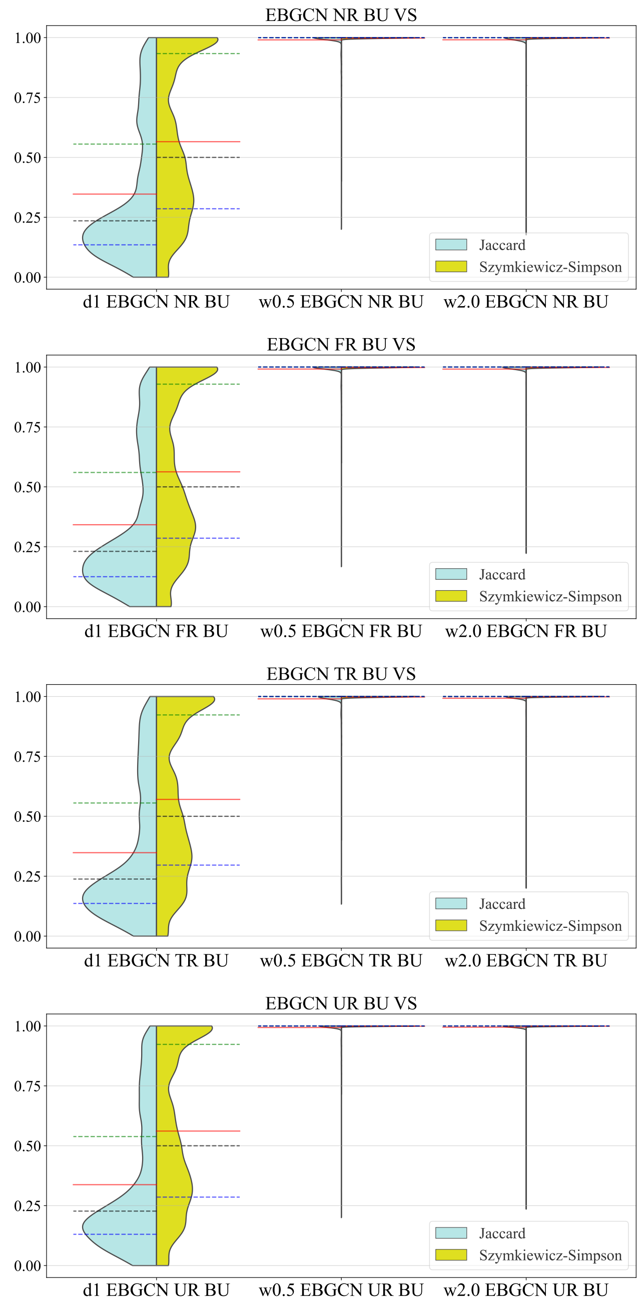

Fig. 15: Similarity between Upper Quartile Contributors for EBGCN BU GCN with Modified Graph Structure

TABLE VII: Classwise GCN Szymkiewicz-Simpson Similarities between Centrality Measures

GCN	**Class**	**Centrality**			
		Out Degree	**Betweenness**	**Closeness**	**Farness**
Bi-GCN TD	**NR**	0.851 ± 0.22	0.826 ± 0.27	0.789 ± 0.30	$0.827 \pm 0.25^{*}$
	FR	0.851 ± 0.21	0.831 ± 0.26	$0.796 \pm 0.29^{*}$	0.832 ± 0.24
	TR	0.851 ± 0.21	0.828 ± 0.27	$0.787 \pm 0.30^{*}$	0.832 ± 0.24
	UR	0.851 ± 0.22	0.827 ± 0.27	$0.787 \pm 0.30^{*}$	$0.835 \pm 0.24^{*}$
Bi-GCN BU	**NR**	$0.724 \pm 0.25^{*}$	$0.618 \pm 0.32^{*}$	$0.586 \pm 0.30^{*}$	$0.531 \pm 0.32^{*}$
	FR	$0.715 \pm 0.26^{*}$	0.616 ± 0.32	$0.578 \pm 0.31^{*}$	$0.538 \pm 0.32^{*\dagger}$
	TR	$0.709 \pm 0.26^{*}$	$0.610 \pm 0.32^{*}$	$0.578 \pm 0.30^{*}$	$0.547 \pm 0.32^{*\dagger}$
	UR	$0.715 \pm 0.26^{*}$	0.613 ± 0.33	$0.578 \pm 0.30^{*}$	$0.538 \pm 0.32^{*\dagger}$
EBGCN TD	**NR**	$0.492 \pm 0.36^{*}$	$0.485 \pm 0.40^{*}$	0.521 ± 0.32	$0.543 \pm 0.29^{*\dagger}$
	FR	$0.511 \pm 0.36^{*}$	$0.496 \pm 0.40^{*}$	$0.518 \pm 0.31^{*}$	$0.538 \pm 0.28^{*\dagger}$
	TR	$0.481 \pm 0.37^{*}$	$0.479 \pm 0.41^{*}$	$0.526 \pm 0.32^{*}$	$0.544 \pm 0.29^{*\dagger}$
	UR	$0.484 \pm 0.37^{*}$	$0.482 \pm 0.41^{*}$	$0.514 \pm 0.32^{*}$	$0.545 \pm 0.28^{*\dagger}$
EBGCN BU	**NR**	$0.575 \pm 0.35^{*}$	$0.533 \pm 0.39^{*}$	0.465 ± 0.39	0.646 ± 0.29
	FR	$0.574 \pm 0.35^{*}$	$0.531 \pm 0.39^{*}$	0.469 ± 0.39	0.648 ± 0.29
	TR	$0.586 \pm 0.35^{*}$	$0.546 \pm 0.38^{*}$	0.472 ± 0.38	0.643 ± 0.29
	UR	$0.582 \pm 0.35^{*}$	$0.543 \pm 0.38^{*}$	0.472 ± 0.38	0.648 ± 0.29

* Denotes at least one p-value < 0.05 when compared with other class distributions within the same GCN module for the same centrality measure
† Denotes at least one p-value ≥ 0.05 when compared with other class distributions from other GCN modules for the same centrality measure

TABLE VIII: Classwise GCN Jaccard Similarities between Centrality Measures

GCN	**Class**	**Centrality**			
		Out Degree	**Betweenness**	**Closeness**	**Farness**
Bi-GCN TD	**NR**	0.465 ± 0.26	$0.413 \pm 0.30^{*}$	$0.373 \pm 0.26^{*}$	$0.450 \pm 0.25^{*}$
	FR	0.469 ± 0.26	$0.422 \pm 0.30^{*}$	$0.382 \pm 0.26^{\dagger}$	$0.460 \pm 0.24^{*}$
	TR	0.464 ± 0.26	$0.414 \pm 0.30^{*}$	$0.372 \pm 0.26^{*}$	$0.453 \pm 0.24^{*}$
	UR	0.467 ± 0.26	0.415 ± 0.30	$0.373 \pm 0.27^{*}$	$0.452 \pm 0.25^{*}$
Bi-GCN BU	**NR**	0.465 ± 0.26	0.402 ± 0.28	$0.387 \pm 0.28^{*\dagger}$	$0.355 \pm 0.33^{*}$
	FR	0.469 ± 0.26	0.399 ± 0.28	$0.382 \pm 0.28^{\dagger}$	$0.361 \pm 0.33^{*}$
	TR	0.464 ± 0.26	0.397 ± 0.28	$0.381 \pm 0.28^{*\dagger}$	$0.370 \pm 0.33^{*}$
	UR	0.467 ± 0.26	0.399 ± 0.28	$0.381 \pm 0.28^{\dagger}$	$0.361 \pm 0.33^{*}$
EBGCN TD	**NR**	$0.164 \pm 0.10^{*}$	$0.168 \pm 0.11^{*}$	$0.220 \pm 0.11^{*}$	0.199 ± 0.09
	FR	$0.176 \pm 0.11^{*}$	$0.176 \pm 0.11^{*}$	$0.219 \pm 0.11^{*}$	$0.198 \pm 0.09^{*}$
	TR	$0.158 \pm 0.10^{*}$	$0.165 \pm 0.11^{*}$	$0.222 \pm 0.10^{*}$	$0.200 \pm 0.09^{*}$
	UR	$0.161 \pm 0.10^{*}$	$0.166 \pm 0.11^{*}$	$0.215 \pm 0.10^{*}$	$0.201 \pm 0.09^{*}$
EBGCN BU	**NR**	0.323 ± 0.28	$0.315 \pm 0.28^{*}$	0.293 ± 0.30	0.392 ± 0.29
	FR	0.322 ± 0.28	$0.314 \pm 0.29^{*}$	0.294 ± 0.29	0.392 ± 0.29
	TR	0.328 ± 0.28	$0.327 \pm 0.28^{*}$	0.298 ± 0.29	0.389 ± 0.29
	UR	0.327 ± 0.28	$0.328 \pm 0.28^{*}$	0.298 ± 0.29	0.395 ± 0.29

* Denotes at least one p-value < 0.05 when compared with other class distributions within the same GCN module for the same centrality measure
† Denotes at least one p-value ≥ 0.05 when compared with other class distributions from other GCN modules for the same centrality measure

detection model with tree-structured recursive neural networks," *ACM Trans. Intell. Syst. Technol.*, vol. 11, no. 4, jun 2020. [Online]. Available: https://doi.org/10.1145/3391250
[9] T. Bian, X. Xiao, T. Xu, P. Zhao, W. Huang, Y. Rong, and J. Huang, "Rumor detection on social media with bi-directional graph convolutional networks," in *AAAI*, 2020.
[10] L. Wei, D. Hu, W. Zhou, Z. Yue, and S. Hu, "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 (Volume 1: Long Papers)*. Online: Association for Computational Linguistics, Aug. 2021, pp. 3845–3854. [Online]. Available: https://aclanthology.org/2021.acl-long.297
[11] H. Lin, J. Ma, M. Cheng, Z. Yang, L. Chen, and G. Chen, "Rumor detection on twitter with claim-guided hierarchical graph attention networks," *arXiv preprint arXiv:2110.04522*, 2021.
[12] X. Liu, Z. Zhao, Y. Zhang, C. Liu, and F. Yang, "Social network rumor detection method combining dual-attention mechanism with graph convolutional network," *IEEE Transactions on Computational Social Systems*, vol. 10, no. 5, pp. 2350–2361, 2023.
[13] E. Kochkina, M. Liakata, and A. Zubiaga, "All-in-one: Multi-task learning for rumour verification," in *Proceedings of the 27th International Conference on Computational Linguistics*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, pp. 3402–3413. [Online]. Available: https://aclanthology.org/C18-1288
[14] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*. PMLR, 2013, pp. 1310–1318.
[15] F. Baldassarre and H. Azizpour, "Explainability techniques for graph convolutional networks," *ArXiv*, vol. abs/1905.13686, 2019.
[16] G. Montavon, A. Binder, S. Lapuschkin, W. Samek, and K.-R. Müller, *Layer-Wise Relevance Propagation: An Overview*. Cham: Springer International Publishing, 2019. [Online]. Available: https://doi.org/10.1007/978-3-030-28954-6_10
[17] T. Schnake, O. Eberle, J. Lederer, S. Nakajima, K. T. Schutt, K.-R. Muller, and G. Montavon, "Higher-order explanations of graph neural networks via relevant walks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, pp. 7581–7596, 2022.
[18] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," *International Journal of Computer Vision*, vol. 128, pp. 336–359, 2017.
[19] J. Zhang, S. A. Bargal, Z. Lin, J. Brandt, X. Shen, and S. Sclaroff, "Top-down neural attention by excitation backprop," *International Journal of*

Computer Vision, vol. 126, no. 10, pp. 1084–1102, 2018.

[20] P. E. Pope, S. Kolouri, M. Rostami, C. E. Martin, and H. Hoffmann, “Explainability methods for graph convolutional neural networks,” in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 10 764–10 773.

[21] M. T. Ribeiro, S. Singh, and C. Guestrin, “” why should i trust you?” explaining the predictions of any classifier,” in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.

[22] Z. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec, “Gnnexplainer: Generating explanations for graph neural networks,” *Advances in neural information processing systems*, vol. 32, 2019.

[23] D. Luo, W. Cheng, D. Xu, W. Yu, B. Zong, H. Chen, and X. Zhang, “Parameterized explainer for graph neural network,” *Advances in neural information processing systems*, vol. 33, pp. 19 620–19 631, 2020.

[24] M. N. Vu and M. T. Thai, “Pgm-explainer: Probabilistic graphical model explanations for graph neural networks,” *ArXiv*, vol. abs/2010.05788, 2020.

[25] Y. Zhang, D. DeFazio, and A. Ramesh, “Relex: A model-agnostic relational model explainer,” *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, 2020.

[26] H. Yuan, H. Yu, J. Wang, K. Li, and S. Ji, “On explainability of graph neural networks via subgraph explorations,” *ArXiv*, vol. abs/2102.05152, 2021.

[27] X. Yang, Y. Lyu, T. Tian, Y. Liu, Y. Liu, and X. Zhang, “Rumor detection on social media with graph structured adversarial learning,” in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, C. Bessiere, Ed. International Joint Conferences on Artificial Intelligence Organization, 7 2020, pp. 1417–1423, main track. [Online]. Available: https://doi.org/10.24963/ijcai.2020/197

[28] T. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” *ArXiv*, vol. abs/1609.02907, 2017.

[29] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.

Chin Wai Kit Daniel completed his Bachelor’s degree in Engineering from Singapore University of Technology and Design in 2019. He is currently a PhD student in the Information Systems Technology and Design Pillar, Singapore University of Technology and Design. His research interests are in Explainable Artificial Intelligence, Graph Neural Networks and Social Computing.

Roy Ka-Wei Lee is an Assistant Professor at the Information Systems Technology and Design Pillar, Singapore University of Technology and Design. He is a faculty of the transformative Design and Artificial Intelligence programme. His research lies in the intersection of data mining, machine learning, social computing, and natural language processing. He is leading the Social AI Studio, a research group that focuses on designing next-generation social artificial intelligence systems. He has published in top-tier venues in data mining and computation linguistics domains. He serves on the program committees of multiple top data mining and natural language processing conferences.

Kwan Hui Lim is an Assistant Professor at the Information Systems Technology and Design Pillar, Singapore University of Technology and Design, and leads the Social and Urban Analytics Lab. Previously, he was a Research Fellow at the School of Computing and Information Systems, University of Melbourne, Research Engineer at the Living Analytics Research Centre, Singapore Management University, Research Intern at IBM Research - Australia, and various visiting appointments. He received his PhD from the University of Melbourne, and MSc (Research) and BCompSci (1st Class Honours) from the University of Western Australia. He is a recipient of the 2016 Google PhD Fellowship in Machine Learning. His research interests are in Data Mining, Machine Learning, Artificial Intelligence, Social Network Analysis, and Social Computing.