GRAIN: Gradient-Based Interaction Graph of Neurons for Explainable Misinformation Detection

Darshan Rathod
Fujitsu Research of India Pvt. Ltd.
Bengaluru, India
Rathod.Dineshkumar@fujitsu.com

Rahul Mishra
Fujitsu Research of India Pvt. Ltd.
Bengaluru, India
Rahul.N.Mishra@fujitsu.com

Abstract

Prior work on misinformation detection often applies large language models (LLMs) in a black-box manner, relying on knowledge implicitly encoded in model parameters or on pre-hoc and post-hoc reasoning techniques. However, the opaque nature of LLMs limits access to their internal decision-making processes, and existing approaches largely overlook the rich signals embedded within their hidden layers, instead focusing primarily on surface-level generative capabilities. To address these limitations, we propose GRAIN, an explainable misinformation detection framework that explicitly models neuron–neuron interaction graphs using dominancy scores. Experimental results across multiple benchmark datasets show that GRAIN consistently outperforms state-of-the-art baselines while providing interpretable insights into the neuronal patterns and cues that distinguish misinformation from factual content. Furthermore, we introduce NNI-IG, a GRAIN-guided variant of Integrated Gradients that leverages critical neurons to produce sharper and more focused token-level attributions, further enhancing model interpretability.

1 Introduction

Large Language Models (LLMs) have become the de facto choice for a wide range of downstream tasks due to their state-of-the-art performance, stemming from their ability to encode extensive world knowledge in parametric memory as well as their strong reasoning and planning capabilities, particularly in the era of LLM-based agents. Despite their success, the internal cues and mechanisms that drive the impressive performance of LLMs remain largely unexplored, owing to their black-box nature. Misinformation detection is one such task where the use of LLMs has been widely investigated ((Popat et al., 2018; Mishra and Setty, 2019; Liu et al., 2025; Bürger et al., 2024)). However, most prior work relies on post-hoc explanation methods or attention-based word- or sentence-level attributions to provide a notion of interpretability. Numerous studies have shown that such post-hoc explanations are often unreliable and insufficient for uncovering the true patterns and signals governing model behaviour within the internal representations of LLMs ((Jain and Wallace, 2019)). Moreover, existing approaches largely fail to leverage the rich signals embedded in the internal layers of LLMs, instead relying primarily on their generative language modelling capabilities to address misinformation detection (MD). Internal states of the LLMs are a rich resource to be further mined for important and conclusive patterns helpful in many downstream tasks ((Padakandla et al., 2025; Xie et al., 2024)).

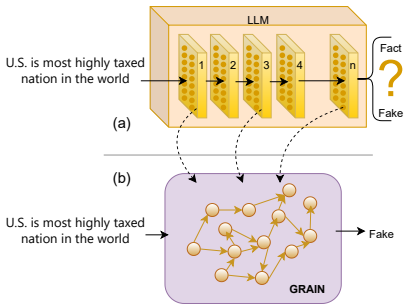


Figure 1: (a) Vanilla prompting LLM is inefficient. Whereas, (b) classifying the graph representation of model's internals is efficient.

To address these limitations, this paper explores two key directions: (1) whether the internal representations of LLMs capture latent patterns in the propagation of input information across layers that can be leveraged to improve misinformation detection performance; and (2) how to model and extract these internal signals in a reliable and comprehensive manner to reveal meaningful patterns that shed light on the model's decision-making process. To this end, we propose GRAIN, a neuron–neuron interaction–based explainable misinformation detection framework. GRAIN constructs a neuron

Findings of the Association for Computational Linguistics: IJCNLP-AACL 2026, pages 3655–3670
November 6-10, 2026 ©2026 Association for Computational Linguistics

interaction graph using dominancy scores, enabling the modeling of fine-grained relationships among neurons both within and across layers. This formulation not only enhances interpretability by exposing internal interaction patterns, but also consistently outperforms existing baseline methods. Our approach is motivated by the observation that neurons within a layer and even across layers, often exhibit coordinated behavior conditioned on the input prompt ((Gurnee et al., 2024; Yu and Ananiadou, 2024)). We hypothesise that misinformation and factual information induce discernible and structurally different interaction patterns among neurons. To capture these patterns, we model neuron relationships through a neuron–neuron interaction graph (NNI-graph) derived from dominancy scores and process the resulting interaction flow across layers using graph neural networks.

The key contributions of this work are as follows: 1) We introduce, to the best of our knowledge, the first approach for modelling neuron–neuron interactions using dominancy scores to uncover latent signals and cues relevant to misinformation detection. 2) We propose GRAIN, an explainable framework that leverages neuron–neuron interaction graphs to predict the veracity of information while providing interpretable insights into the model's decision-making process. 3) We conduct extensive evaluations of GRAIN on multiple publicly available datasets, including comprehensive ablation studies, to demonstrate the effectiveness of the proposed design choices and highlight the contribution of individual components. 4) We propose Neuron-neuron Interaction-based Integrated Gradients (NNI-IG), a neuron-aware variant of Integrated Gradients (IG) that leverages GRAIN-identified critical neurons to produce sharper and more informative token attributions than standard IG, improving interpretability for misinformation detection. 5) We release the neuron–neuron interaction graphs for all evaluated datasets, providing a rich resource to support future research.

2 Related work

2.1 MD works using DNN, Graphs, pre-LLM models

Early misinformation detection methods primarily relied on textual modelling with limited contextual awareness. Neural architectures such as CNNs and RNNs extracted semantic and stylistic features to classify fake versus genuine news ((Wang et al., 2018; Zhu et al., 2022)). To capture the social nature of misinformation, graph-based approaches modelled relational dependencies, improving robustness across domains and platforms ((Sun et al., 2023; Zhang et al., 2021; Shu et al., 2020)). Evolution of contextualized transformer models such as BERT enabled richer semantic representations and significantly improved performance on claim classification and credibility estimation tasks ((Devlin et al., 2019; Raffel et al., 2020; Popat et al., 2018)). Subsequent domain-aware methods further enhanced generalisation by incorporating multi-domain learning strategies (Nan et al., 2021). Despite these advances, pre-LLM approaches remain constrained by static architectures and a reliance on surface-level textual signals.

2.2 MD works using LLMs as agents and LLM internals

Recent studies increasingly leverage LLMs for misinformation detection due to their reasoning capabilities and broad knowledge grounding. Many approaches employ LLMs as black-box evaluators that analyse claims, retrieve evidence, or perform fact-checking through prompting ((Lin et al., 2025; Wan et al., 2024; Cui et al., 2025; Deng et al., 2025)). Others explore LLM-based agents capable of iterative verification and collaborative reasoning (Yi et al., 2025; Wang et al., 2025). While effective, these methods primarily operate at the output level and do not explicitly use model internals. In contrast, a growing line of work investigates LLM internals, including hidden activations and gradient-based signals, to understand how models encode factuality and misinformation ((Phan et al., 2023; Hu et al., 2024; Geva et al., 2022; Tong et al., 2025)). These studies reveal that deceptive content induces distinct internal patterns which could provide a better signal for classifying them.

Our work builds on this direction by accessing the internal LLM representations and gradients as structured signals for misinformation detection, moving beyond black-box inference toward neuron-level, interpretable modelling.

3 Problem Definition

Given an input claim, also referred to as a prompt (p), the objective is to determine whether the claim corresponds to misinformation. This is formulated as a binary classification task, where the model predicts whether p belongs to the positive class

(misinformation) or the negative class (fact).

4 Methodology

This section introduces GRAIN, a neuron-level graph construction framework for misinformation detection using large language models. GRAIN consists of three main modules: (a) obtaining model internals, (b) computing neuron influence scores, and (c) constructing a neuron-neuron interaction (NNI) graph. An overview of the architecture is shown in Figure 2. Given an input claim, GRAIN first extracts neuron activations and gradients from the LLM. It then identifies influential neurons through a sparsification-based neuron interaction mechanism and finally constructs a graph over selected neurons, which is used for downstream classification.

4.1 Obtaining Model Internals

Large language models consist of multiple hidden layers, each producing high-dimensional activation vectors. Let the LLM consist of n hidden layers, where each layer outputs a vector of dimension d_ℓ. We treat each scalar activation as a neuron, resulting in a total of $d = n \times d_\ell$ neurons.

For a given neuron i, we denote its activation by a_i. Given an input claim p, the model produces an output distribution over the vocabulary: $\hat{t} = \mathcal{L}(p)$, where $\hat{t} \in \mathbb{R}^V$ and V is the vocabulary size. The misinformation detection task is framed as a binary classification problem with two target tokens: *fake* (positive class) and *fact* (negative class). Accordingly, we define one-hot target distributions t^+ and t^-, and compute the corresponding cross-entropy losses:

$$L^-_{\text{CE}}(\hat{t}, t^-) = -\sum_{i=1}^{V} t_i^- \log(\hat{t}_i), \tag{1}$$

$$L^+_{\text{CE}}(\hat{t}, t^+) = -\sum_{i=1}^{V} t_i^+ \log(\hat{t}_i). \tag{2}$$

By backpropagating these losses, we obtain gradients with respect to each neuron activation. Collecting these values yields three vectors: $\vec{a}, \vec{g}^-, \vec{g}^+ \in \mathbb{R}^d$, where $\vec{g}^-$ and $\vec{g}^+$ correspond to gradients induced by the negative and positive targets, respectively.

4.2 Computing the Dominancy Matrix

The gradient of a neuron reflects its contribution toward a target prediction. Since our objective is to identify neurons that support classification into the positive (misinformation) class, we focus exclusively on the positive-target gradient vector $\vec{g}^+$. To capture *inter-neuron influence*, we require a pairwise attribution measure that quantifies how strongly one neuron affects another. We define such influence using an inverse-gradient interaction. For neurons i and j, we define the dominancy score as:

$$\mathbf{D}_{i,j} = g_i^+ \cdot \frac{1}{(g_j^+ + \epsilon)}, \tag{3}$$

where $g_i^+ = \partial L^+ / \partial a_i$, and $0 < \epsilon \ll 1$ is a constant for numerical stability. We note that this ratio is a heuristic sensitivity proxy rather than an exact chain-rule derivative. Larger values of $\mathbf{D}_{i,j}$ indicate stronger directional influence from neuron i to neuron j. Collectively, these values form the **Dominancy Matrix** $\mathbf{D} \in \mathbb{R}^{d \times d}$. Where $\mathbf{D}_{i,j}$ represents the influence of neuron i on neuron j.

4.3 Block-wise Computation and Sparsification

Although $\mathbf{D}$ is defined over all neurons for notational clarity, it is never explicitly constructed due to its quadratic size. Instead, neurons are indexed by layer–neuron pairs (ℓ, h), where $\ell \in \{1, \ldots, n\}$ and $h \in \{1, \ldots, d_\ell\}$.

For each ordered pair of layers (ℓ_m, ℓ_n) with $\ell_m \leq \ell_n$, we compute a block wise dominancy matrix $\mathbf{D}^{\{m,n\}} \in \mathbb{R}^{d_\ell \times d_\ell}$. Within each block, we retain only the top-k entries by magnitude, enforcing sparsity and suppressing weak interactions. This produces an block-wise upper triangular binary matrix $\mathbf{B}$ defined as:

$$\mathbf{B}_{i,j}^{\{m,n\}} = \begin{cases} 1, & \text{if } \mathbf{D}_{i,j}^{\{m,n\}} \in \text{Top-}k \text{ of its block,} \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Matrix $\mathbf{B}$ is block-wise upper triangular because we do not binarise those blocks where $m > n$. This is because we only want to identify directional relationships from previous layers of the model to next layers of the model.

4.4 Neuron Influence Vector

From the binarized dominancy structure, we compute a neuron influence vector: $V \in \mathbb{Z}^d$, where the influence score of neuron i is defined as:

$$v_i = \sum_{j=1}^{d} \left(\mathbf{B}_{i,j} + \mathbf{B}_{j,i} \right). \tag{5}$$

This score captures how frequently a neuron participates in dominant interactions, either as a source

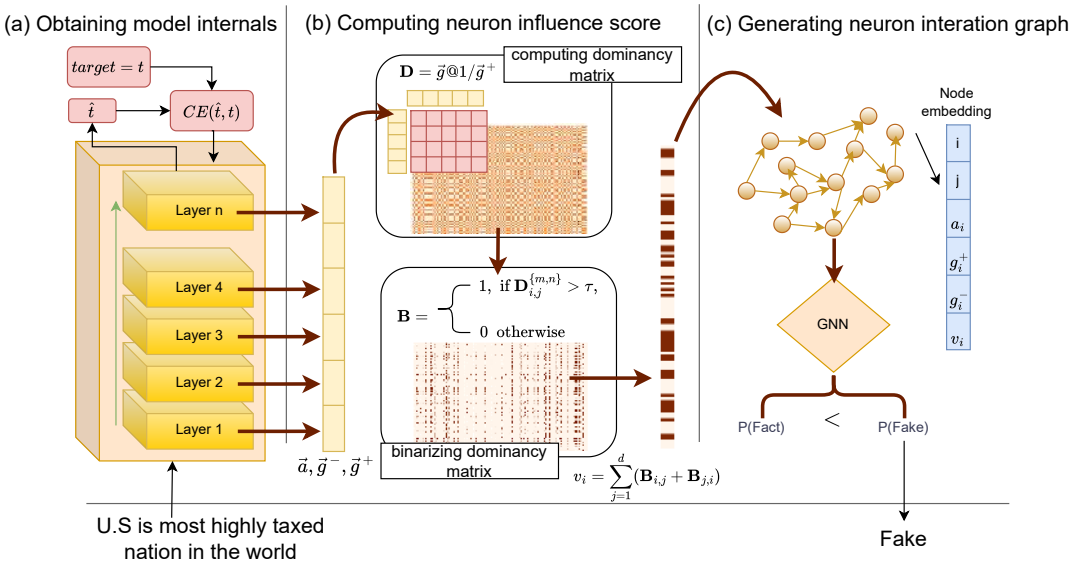


Figure 2: Overview of the GRAIN framework.

or a target. Neurons with higher v_i are considered more influential for the given input. The total number of neurons in a model is significantly high, and constructing a graph from all neurons is not practical. Through our method, we capture informative neurons whose total number is significantly lower than the total number of neurons in the model. The expected number of informative neurons is given by equation 6.

Lemma 1. *(Expected numbers of neurons selected)*

Given n number of layers, d_ℓ number of neurons per layer and top-k selection in each layer-layer block computation, the expected number of neurons per sample with non-zero influence is given by

$$\mathbb{E}[x] = nd_\ell \left(1 - \frac{\binom{(d_\ell - 1)^2}{k}}{\binom{d_\ell^2}{k}}\right) \tag{6}$$

The proof for the above equation is described in section A.1. We also empirically showcase that the formula holds for multiple models on various datasets with reasonable accuracy. The results in table 4 list the empirical expectation of informative neurons on three datasets and 4 different models. Furthermore, figure 6 shows the distribution of the number of informative nodes per sample for Llama-8B model. It shows that, on any dataset, the maximum number of neurons per sample do not go above 750 for $k = 10$, which is significantly lower compared to 131,072 neurons in a model.

4.5 Neuron-neuron interaction graph (NNI-graph) construction

We construct a directed graph $\mathcal{G} = (N, E, W)$ from sparsely selected informative neurons from $\mathcal{L}$, i.e. the non-zero elements of $\vec{v}$. Here, N is a set of nodes, E is the set of edges, and W is the graph-level attribute of the graph. The set of nodes is defined such that, $N = \{(n_k, X_{n_k}) | k \in \mathcal{I}\}$. Where, $\mathcal{I} = \{1, 2, \ldots, d | v_k > 0\}$ and $X_{n_k} = [l(k), \mathcal{N}(k), v_k, a_k, g_k^-, g_k^+] \in \mathbb{R}^6$. Here, X_{n_k} is the node embedding vector.

The node embedding X_{n_k} consists of 6 elements: 1) a layer index = $l(k) \in \{1, 2, \ldots, n_l\}$, 2) an intra-layer activation index $\mathcal{N}(k) \in \{1, 2, \ldots, d_l\}$, 3) its influence value, 4) its activation value a_k, 5) its negative-target gradient value g_k^- and 6) its positive-target gradient value g_k^+.

The graph edges are created such that they encode the hierarchical influence from lower layers to higher layers. For any $i, j \in \mathcal{I}$, the set of edges is given by $E = \{(n_i, n_j) | i, j \in \mathcal{I}, l(i) < l(j)\}$ By default, this connects every pair of selected nodes across layers, regardless of the underlying dominancy score between them; the dominancy scores are instead used earlier, to decide *which* neurons are selected via the top-k sparsification. As shown in ablation (d) (Table 2), further restricting edges via the e_k threshold has little additional effect on performance, indicating that the identity of the selected neurons, rather than the precise edge topology among them, carries most of the discriminative signal.

We now pass this generated graph to a trained

graph neural network ($\mathcal{F}$) which classifies the graph into binary classification targets: $\hat{y} = \mathcal{F}(\mathcal{G}) \in \{0, 1\}$.

4.6 Graph Neural Network Architecture

To classify the neuron interaction graph constructed in the previous subsection, we employ a graph convolutional network (GCN) as the graph encoder. Given an input graph $\mathcal{G} = (N, E, W)$, the GCN aggregates local neighbourhood information to produce a graph-level representation suitable for binary misinformation classification.

Graph Convolution layers: We use a two-layer Graph Convolutional Network (GCN) (Kipf and Welling, 2017) to encode node representations. Let $X \in \mathbb{R}^{|N| \times F}$ denote the matrix of node features, where $F = 6$ in our case. The GCN updates node representations as:

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(l)}W^{(l)}\right) \quad (7)$$

where $\tilde{A} = A + I$ is the adjacency matrix with self-loops, $\tilde{D}$ is the corresponding degree matrix used for normalisation and $W^{(l)}$ is the trainable weight matrix, and $\sigma(\cdot)$ denotes the ReLU activation function. $H^0 = X$. Batch normalization is applied after each convolution to stabilize training.

Multi-level Feature Aggregation: To capture neuron interactions at different receptive field sizes, we concatenate the hidden representations obtained from the first and second GCN layers using $H = [H^{(1)} \| H^{(2)}]$. Where $\|$ denotes feature concatenation. This design allows the model to jointly leverage shallow and deeper relational patterns among neurons.

Graph-level Pooling: Since our task is to classify the graph, we aggregate node features into a single graph representation using global mean pooling as follows:

$$h_{\mathcal{G}} = \frac{1}{|N|}\sum_{n \in N} H_n \quad (8)$$

Global mean pooling provides a permutation-invariant summary of neuron interactions and is robust to varying graph sizes across samples.

Classification Head: The pooled graph representation $h_{\mathcal{G}}$ is passed through a two-layer multilayer perceptron (MLP) with ReLU activation and dropout, followed by a linear layer that outputs logits for binary classification: $\hat{y} = \text{MLP}(h_{\mathcal{G}})$.

5 Experiments

Datasets The experiments are conducted across three benchmark misinformation detection datasets [1] to evaluate the performance of GRAIN. *Gossipcop* is a misinformation dataset collected by (Shu et al., 2020) consisting of 11,800 pairs of claims and their veracity labels. *Politifact* dataset involves claims involving political intent and is a prevalent misinformation detection dataset. We use the dataset provided by (Popat et al., 2018). The original dataset consists of the following veracity labels: {*true, mostly true, half true, mostly false, false, pants-on-fire*}. Following (Rashkin et al., 2017) and (Wang et al., 2024), we divide the claims into two labels. Namely, *True* for {*true, mostly true, half true*} and *False* for {*mostly false, false, pants-on-fire*}. *Snopes* is a misinformation dataset collected from a fact-checking website with human-verified veracity labels. Dataset statistics, including train/validation/test counts, are reported in Appendix Table 5. All baselines and the GRAIN method are trained and evaluated on identical train/validation/test splits for each dataset, using only the claim text, without access to retrieved evidence or additional context.

Baselines Following (Nan et al., 2024; Hu et al., 2022), we categorise the baselines into three major categories. (1) **Content-based methods** often extract style features, semantic features, emotional features, etc. from the textual content using sentence transformers. In this category we include works like, **BERT** (Devlin et al., 2019), **EANN** (Wang et al., 2018), **ENDEF** (Zhu et al., 2022) and **BERT-Emo** (Zhang et al., 2021). (2) **Structure-aware methods** embed the claim either directly through sentence transformers or by accessing the model's internals. It further augments the representation of a claim using Graph or MoE. These augmented representations are used for classifying claims into fact or misinformation. This category includes works like, **MDFEND** (Nan et al., 2021), **MGIN-AG** (Sun et al., 2023) and **MSynFD** (Xiao et al., 2024). (3) **LLM-based methods** use latest text-generation models in their architecture for generating an explanation, profiled comment or an intent. These LLM-generated representations are then used for MD. These methods utilise the comprehension capabilities of LLM and exploit those capabilities in their architectures. It involves works

[1] The datasets can be downloaded from https://www.mpi-inf.mpg.de/dl-cred-analysis/

like, **GenFEND** (Nan et al., 2024) and **DM-Inter** (Wang et al., 2024).

6 Results and Discussions

6.1 Comparison to Baselines

We evaluate GRAIN on the *GossipCop* and *Politifact* datasets against a broad spectrum of misinformation detection approaches spanning content-based encoders, structure-aware graph models, and recent LLM-based systems. Table 1 summarises the results. Across both datasets, GRAIN consistently achieves the highest accuracy and macro-F1, demonstrating strong and stable superiority over all competing methods. Across repeated runs, GRAIN's macro-F1 varies with a standard deviation of approximately 0.8%, indicating that the reported gains over the strongest baselines are unlikely to be an artefact of training variability.

Compared to content-based methods, the GRAIN method achieves better macro-F1 by approximately +4-6% on *Gossipcop* and +3-5% on *Politifact*. The content-based methods are limited by their sentence-transformers' limited contextual representation, whereas the GRAIN method uses the rich graph representation from the dense context of the LLM models. This highlights that simply modeling textual semantics is insufficient for reliable misinformation detection, and deeper discriminative signal extraction is required.

Structure-aware methods such as MDFEND, MGIN-AG, and MSynFD introduce richer relational dependencies through modality fusion, mixture-of-experts routing, or syntax-graph integration. These models yield improvements over purely content-based methods, particularly on *GossipCop*. GRAIN clearly outperforms structure-aware methods, achieving +3–4% macro-F1 gains on *GossipCop* and *Politifact* datasets.

Furthermore, we compare GRAIN against LLM-based MD systems. We first compare to vanilla LLM methods, wherein we prompt the Llama-3.1-8B-Instruct model to classify the input claim into fact or misinformation. Here, we use zero-shot and a last-layer finetuned model for the task, and we find that the vanilla-LLM method significantly shows poor performance compared to GRAIN and other LLM-based methods. Remarkably, the higher results in GRAIN is obtained with substantially fewer trainable parameters compared to Llama-FT. The fine-tuned Llama model updates 425,984 parameters, while GRAIN optimises only 275,406, approximately 35% fewer. This shows that the GRAIN method captures the informative neurons and induced relational structures through the graph using comparatively few additional trainable parameters; we do not have reliable runtime or memory measurements to support a broader efficiency claim. Moreover, direct prompting LLM underutilizes the internal knowledge of LLM and showcases the requirement of richer representation of a claim for such MD task. GenFEND and DM-Inter incorporate more advanced LLM-guided or internal feature strategies and thus constitute the strongest baselines in MD task. GenFEND shows the second best performance on both datasets after the GRAIN method. The better results of the GRAIN method show that it captures and represents informative neurons from the LLM effectively for the MD task, while adding only a lightweight GCN component in terms of trainable parameters.

6.2 Ablation studies

We further ablate the key components of the GRAIN method to showcase the sensitivity to the selection of design parameters. We ablate four crucial design choices. Table 2 shows the results of four ablation parameters.

(a) Graph level array: Table 2 (a) shows the results of the GRAIN algorithm without a graph level array ($G = (N, E)$) and with a graph level array ($G = (N, E, W)$). Results show that the addition of graph level array improves the macro-F1 by +3.2%. Therefore, the GRAIN algorithm incorporates graph level array to the graph for better performance.

(b) Node embedding dimension: Table 2 (b) shows the results of the GRAIN algorithm with varying node embedding dimensions. The results show that adding more dimensions to the node embeddings improves the performance of the GRAIN algorithm. However, the improvement from $\mathbb{R}^5$ to $\mathbb{R}^6$ is very small (of 0.1%) compared to the improvement from $\mathbb{R}^4$ to $\mathbb{R}^5$ (of 1.8%).

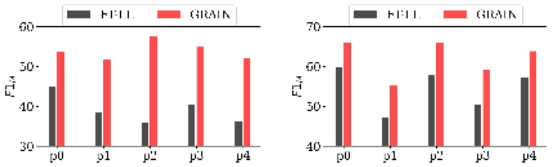
(c) Effect of top-k: Table 2 (c) shows the results of the GRAIN algorithm with varying k. To reiterate from section 4, the parameter k represents the top-k^{th} value for the generation of the neuron influence vector v_i from the dominancy matrix $\mathbf{D}$. The higher value of k will generate a larger number of nodes in the graph. We study three values of $k \in \{4, 10, 16\}$. Increasing k from 4 to 10 improves the macro-F1 by +1.5%. This is attributed to the increased number of nodes in the graph and

Category	Method	GossipCop				Politifact			
		Acc	$F1_M$	$F1_R$	$F1_F$	Acc	$F1_M$	$F1_R$	$F1_F$
Content-based	BERT	84.4	78.7	89.7	67.8	60.8	60.4	62.0	60.9
	BERT-Emo	84.6	79.1	89.7	68.5	62.7	62.5	61.5	63.4
	EANN	85.2	79.3	90.4	68.2	61.5	61.1	61.6	62.1
	ENDEF	85.2	80.1	90.2	69.9	62.8	61.9	63.6	61.3
Struct-aware	MDFEND	85.6	80.1	90.2	69.6	60.3	61.9	70.9	52.9
	MGIN-AG	85.9	80.4	90.7	70.7	62.5	62.3	63.0	64.9
	MSynFD	87.0	80.5	91.5	70.3	62.5	62.6	60.0	65.3
LLM based	Llama-zeroshot	59.7	52.6	79.1	34.4	52.5	52.5	52.3	53.7
	Llama-FT	65.7	62.4	73.6	51.2	59.7	59.7	60.0	59.5
	GenFEND	86.0	83.7	90.0	73.0	63.8	63.8	64.1	63.7
	DM-Inter	85.5	79.8	95.6	69.0	61.4	61.3	63.1	59.4
GRAIN	Llama-8B	87.1	84.5	90.9	78.1	66.6	66.0	70.4	61.6

Table 1: Experimental results of GRAIN method with baselines and State-of-the-art MD methods.

	Method	Acc	$F1_M$	$F1_R$	$F1_F$
(a)	$G = (N, E)$	63.1	62.8	66.7	58.8
	$G = (N, E, W)$	66.6	66.0	70.4	61.6
(b)	$X_{n_k} \in \mathbb{R}^4$	64.9	64.1	69.4	58.8
	$X_{n_k} \in \mathbb{R}^5$	66.4	65.9	69.9	61.8
	$X_{n_k} \in \mathbb{R}^6$	66.6	66.0	70.4	61.6
(c)	$k = 4$	65.1	64.5	69.2	59.7
	$k = 10$	66.6	66.0	70.4	61.6
	$k = 16$	65.7	65.1	69.9	60.2
(d)	$e_k = 5$	66.4	65.8	70.2	61.4
	$e_k = 10$	66.2	65.6	70.1	61.1
	$e_k = 25$	65.7	65.1	69.9	60.2
	$e_k = 50$	66.4	65.8	70.2	61.4
	$e_k = 100$	65.7	65.1	69.8	60.4
	$e_k = -1$	66.6	66.0	70.4	61.6

Table 2: Ablation experiments results



(a) Llama-1B (b) Llama-8B

Figure 3: Perturbation experiment results. p0 refers to no-perturbation. p1, p2, p3 and p4 are 4 types of perturbation applied to the sample.

a richer representation of graph-level information. Whereas, further increasing k to 16 reduces signal to noise ratio, which leads to a reduction in performance by -0.9%.

(d) Effect of number of edges: As mentioned in section 4, a directed edge is created between the node of a lower layer and the node of a higher layer. This approach is identified as default and is marked as $k = -1$ in table 2 (d). Moreover, a threshold can be provided for edge creation such that, if $\mathbf{D}_{i,j} > e_k$ then the edge is created. The threshold parameter e_k is chosen such that e_k is the top-k^{th} value of $\mathbf{D}_i$. Thus, $e_k = 25$ means that an edge is created between the node i and j if $\mathbf{D}_{i,j}$ lies in the top-25 values of $\mathbf{D}_i$. Table 2 (d) shows the effect of e_k in the GRAIN method. Interestingly, we do not observe a significant effect of e_k in performance of the GRAIN method. This is because even though the number of edges is reduced overall in the graph, the informative edge for the MD task is preserved amongst the top edges from any given node.

(e) Graph structure vs. node selection: To assess how much of GRAIN's performance is attributable to the graph structure and GCN classifier, as opposed to simply selecting informative neurons, we replace the GCN with a non-graph classifier (logistic regression / SVM) applied directly to the same selected-neuron node features. This ablation reduces accuracy and macro-F1 by approximately 2% relative to the full GRAIN method, indicating that both the neuron-selection step and the graph-based relational modelling contribute to performance, with the majority of the gain coming from neuron selection itself.

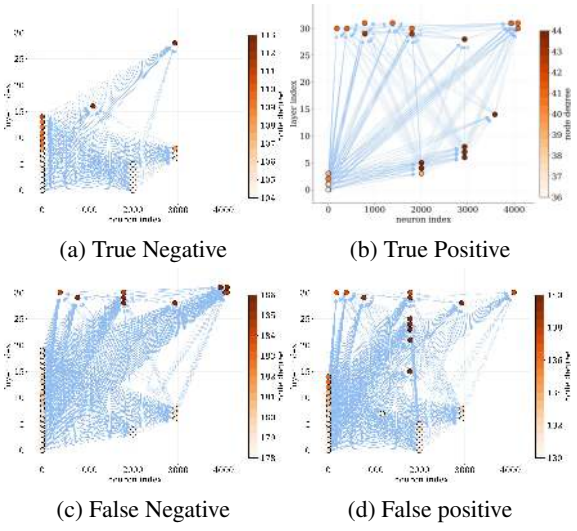
6.3 GRAIN with different LLM

We apply the GRAIN method on four different finetuned models across two sizes and two families of models each: 1) Llama-3.2-1B-Instruct (Llama-1B), 2) Llama-3.1-8B-Instruct (Llama-8B), 3) Gemma-1.1-2B-it (Gemma-2B) and 4) Gemma-1.1-7B-it (Gemma-7B). We additionally compare models on the third dataset, *Snopes*. Table 3 shows the macro-F1 comparison of the experiment. The GRAIN method significantly outperforms the underlying finetuned-LLM despite having fewer trainable parameters.

Perturbation experiment: To analyse the stability of the algorithm, we perturb the input statement

Model	Method	Params	D1	D2	D3
L-1B	FT-LL	212,992	41.3	31.0	44.9
	GRAIN	78,798	75.0	48.6	53.6
L-8B	FT-LL	425,984	60.6	41.8	59.7
	GRAIN	275,406	84.5	61.1	66.0
G-2B	FT-LL	204,800	44.6	46.5	56.9
	GRAIN	86,900	68.7	56.8	60.4
G-7B	FT-LL	458,752	40.8	43.2	52.9
	GRAIN	185,294	71.9	59.2	64.9

Table 3: F1-Macro for finetuned LLM and GRAIN method across D1, D2 and D3 misinformation datasets; D1 - Gossipcop; D2 - Snopes and D3 - Politifact. L-1B, L-8B, G-2B and G-7B are Llama-1B, Llama-8B, Gemma-2B and Gemma-7B model respectively.



(a) True Negative (b) True Positive

(c) False Negative (d) False positive

Figure 4: Intersecting graph over multiple samples for Llama-8B.

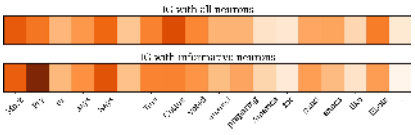
and pass that perturbed statement to the finetuned-LLM and the GRAIN method. In perturbation, we only alter the statement without changing the underlying meaning or any information from the statement. We use Llama-8B model to generate the altered statements. We perform 4 types of perturbations, namely p1) Active passive voice changeover, p2) Add remove neutral words, p3) Clause reordering and p4) Synonym replacement. The details of each perturbation with their examples are provided in Appendix A. Figure 3 shows the results of the perturbation experiments on Llama-1B and Llama-8B model. The GRAIN method outperforms the finetuned model on perturbed data. Moreover, the variation in performance with perturbation is lower in the GRAIN method compared to the finetuned model, which showcases the robustness of the GRAIN method.

6.4 Graph insights

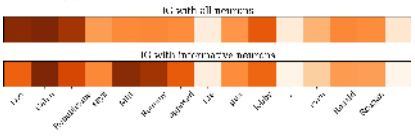
The GRAIN method creates a graph for a sample from the model's informative neurons. It is difficult to analyse the single-sample graph due to the high number of edges in a graph, as seen in figure 11. Hence, we analyse the graph patterns from intersecting graphs to gain a deeper understanding on GRAIN method. Appendix A.5 describes the method to get the intersecting graph. Figure 4 shows the intersecting graph on the Llama-8B model. Interestingly, we find a significant number of edges in the intersecting graph, indicating that the *fact* and *fake* samples consist of some common underlying graph structure. Moreover, the graph pattern and node degrees in *True* classes (figure 4 (a) and (b)) differ significantly. For *True* negatives, the nodes are concentrated in initial layers, whereas for *True* positives, the nodes are distributed in initial and later layers. The degree of nodes also ranges from 100 - 115 and 35 - 45 for *True* negatives and *True* positives, respectively. To further probe the significance of the identified neurons, we perform a *neuron intervention experiment*: we switch off the neurons identified as informative for the *fake* class and measure the resulting change in the model's output probability for *fake*-class tokens. Switching off the 45 neurons identified for the *fake* class (against 95 identified for the *fact* class) reduces the *fake*-token probability by 62%, averaged over the test dataset, showing that GRAIN identifies a small but functionally consequential subset of neurons out of 131,072 in the model. We note that this intervention does not include a random-neuron or magnitude-matched control, so it demonstrates that the selected neurons are causally relevant, but not that they are uniquely or maximally so relative to other neuron subsets of the same size; we leave such controlled comparisons to future work. Interestingly, the intersecting graph for *False* negative and *False* positive do not show a significant pattern difference, and we can see some overlapping structures of *True* negative and *True* positive in False predicted classes. Such non-discernible graph patterns make the claim hard to correctly classify.

6.5 Neuron-neuron Interaction-based Integrated Gradients (NNI-IG)

We introduce a variant of the Integrated Gradient(IG) method ((Sundararajan et al., 2017)) to gain further insights into how the GRAIN method



(a) LLM - Fact, GRAIN - Fake



(b) LLM - Fake, GRAIN - Fact

Figure 5: Integrated gradients on verdicts with contradicting predictions of LLM and GRAIN. True prediction is of the GRAIN method. Underlying model = Llama-8B

attributes token importance. In the NNI-IG method, we only consider important neurons identified from GRAIN, whereas IG without GRAIN considers all the neurons. Further details on method implementation are in section A.7. Figure 5 shows the token attribution heatmap plot for both methods. Compared to only the IG method, the NNI-IG method attributes very sharp attribution to critical tokens, which helps it to classify the claim. For e.g., in figure 5(a) NNI-IG method attributes sharp attention to the underlying person's name token, which is crucial for classifying the claim. Whereas the IG method's attention is diffused throughout the claim. Similarly, in figure 5(b), we observe that the NNI-IG method attributes sharp attention to both the person's name in the claim. This shows that the GRAIN method optimally identifies critical neurons responsible for the misinformation detection task.

7 Conclusions

The growing demand for explainable misinformation detection highlights the limitations of existing methods in exposing the internal reasoning processes behind model predictions. In this paper, we presented GRAIN, a novel explainable framework that constructs neuron–neuron interaction graphs from a model's internal representations. Extensive experiments on multiple benchmark datasets show that GRAIN consistently outperforms strong baseline approaches while providing meaningful insights into the decision-making signals underlying misinformation detection. Our results demonstrate that modeling neuron-level interactions is an effective step toward more transparent and reliable misinformation detection systems.

8 Limitations

In this study, we apply the GRAIN framework to models with up to 8B parameters, primarily due to hardware and computational resource constraints. Nevertheless, the proposed approach is model-agnostic with respect to scale and can, in principle, be extended to larger LLMs given sufficient resources. GRAIN requires access to a model's internal representations and gradients, which restricts its applicability to white-box models and prevents direct use with closed-source or API-only systems. We evaluate our approach on widely used misinformation detection benchmarks, which predominantly focus on political claims. As a result, the effectiveness of GRAIN on orthogonal domains, such as scientific or medical misinformation, as well as on multilingual claims, remains to be systematically explored. We leave these extensions to future work.

References

Lennart Bürger, Fred A. Hamprecht, and Boaz Nadler. 2024. Truth is Universal: Robust Detection of Lies in LLMs. In *NeurIPS 2024*.

Zikun Cui, Tianyi Huang, Chia-En Chiang, and Cuiqianhe Du. 2025. Toward Verifiable Misinformation Detection: A Multi-Tool LLM Agent Framework. In *Proceedings of the 2025 International Conference on Generative Artificial Intelligence for Business*, GAIB '25, pages 179–185, New York, NY, USA. Association for Computing Machinery.

Boyi Deng, Wenjie Wang, Fengbin Zhu, Qifan Wang, and Fuli Feng. 2025. CrAM: credibility-aware attention modification in LLMs for combating misinformation in RAG. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'25/IAAI'25/EAAI'25. AAAI Press.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Mor Geva, Avi Caciularu, Kevin Ro Wang, and Yoav Goldberg. 2022. Transformer Feed-Forward Layers Build Predictions by Promoting Concepts in the Vocabulary Space. *arXiv preprint*. ArXiv:2203.14680 [cs].

Wes Gurnee, Theo Horsley, Zifan Carl Guo, Tara Rezaei Kheirkhah, Qinyi Sun, Will Hathaway, Neel Nanda, and Dimitris Bertsimas. 2024. Universal Neurons in GPT2 Language Models. *Transactions on Machine Learning Research*.

Beizhe Hu, Qiang Sheng, Juan Cao, Yuhui Shi, Yang Li, Danding Wang, and Peng Qi. 2024. Bad Actor, Good Advisor: Exploring the Role of Large Language Models in Fake News Detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(20):22105–22113. ArXiv:2309.12247 [cs].

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-Rank Adaptation of Large Language Models. *arXiv preprint*. ArXiv:2106.09685 [cs].

Linmei Hu, Siqi Wei, Ziwang Zhao, and Bin Wu. 2022. Deep learning for fake news detection: A comprehensive survey. *AI Open*, 3:133–155.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. 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)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv preprint*. ArXiv:1609.02907 [cs].

Hongzhan Lin, Yang Deng, Yuxuan Gu, Wenxuan Zhang, Jing Ma, See-Kiong Ng, and Tat-Seng Chua. 2025. FACT-AUDIT: An Adaptive Multi-Agent Framework for Dynamic Fact-Checking Evaluation of Large Language Models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 360–381, Vienna, Austria. Association for Computational Linguistics.

Shuliang Liu, Hongyi Liu, Aiwei Liu, Duan Bingchen, Zheng Qi, Yibo Yan, He Geng, Peijie Jiang, Jia Liu, and Xuming Hu. 2025. A Survey on Proactive Defense Strategies Against Misinformation in Large Language Models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 18144–18155, Vienna, Austria. Association for Computational Linguistics.

Rahul Mishra and Vinay Setty. 2019. SADHAN: Hierarchical Attention Networks to Learn Latent Aspect Embeddings for Fake News Detection. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval*, ICTIR '19, pages 197–204, New York, NY, USA. Association for Computing Machinery.

Qiong Nan, Juan Cao, Yongchun Zhu, Yanyan Wang, and Jintao Li. 2021. MDFEND: Multi-domain Fake News Detection. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, pages 3343–3347, New York, NY, USA. Association for Computing Machinery. Event-place: Virtual Event, Queensland, Australia.

Qiong Nan, Qiang Sheng, Juan Cao, Beizhe Hu, Danding Wang, and Jintao Li. 2024. Let Silence Speak: Enhancing Fake News Detection with Generated Comments from Large Language Models. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, CIKM '24, pages 1732–1742, New York, NY, USA. Association for Computing Machinery.

Sindhu Padakandla, Sadbhavana Babar, Rathod Darshan D, and Manohar Kaul. 2025. SafeQuant: LLM Safety Analysis via Quantized Gradient Inspection. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2522–2536, Albuquerque, New Mexico. Association for Computational Linguistics.

Huyen Trang Phan, Ngoc Thanh Nguyen, and Dosam Hwang. 2023. Fake news detection: A survey of graph neural network methods. *Applied Soft Computing*, 139:110235.

Kashyap Popat, Subhabrata Mukherjee, Andrew Yates, and Gerhard Weikum. 2018. DeClarE: Debunking Fake News and False Claims using Evidence-Aware Deep Learning. *arXiv preprint*. ArXiv:1809.06416 [cs].

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, and Yejin Choi. 2017. Truth of Varying Shades: Analyzing Language in Fake News and Political Fact-Checking. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2931–2937, Copenhagen, Denmark. Association for Computational Linguistics.

Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, and Huan Liu. 2020. FakeNewsNet: A Data Repository with News Content, Social Context, and Spatiotemporal Information for Studying Fake News on Social Media. *Big Data*, 8(3):171–188.

Tiening Sun, Zhong Qian, Peifeng Li, and Qiaoming Zhu. 2023. Graph Interactive Network with Adaptive Gradient for Multi-Modal Rumor Detection. In *Proceedings of the 2023 ACM International Conference on Multimedia Retrieval*, ICMR '23, pages 316–324, New York, NY, USA. Association for Computing Machinery.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR.

Zhao Tong, Yimeng Gu, Huidong Liu, Qiang Liu, Shu Wu, Haichao Shi, and Xiao-Yu Zhang. 2025. Generate First, Then Sample: Enhancing Fake News Detection with LLM-Augmented Reinforced Sampling. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 24276–24290, Vienna, Austria. Association for Computational Linguistics.

Herun Wan, Shangbin Feng, Zhaoxuan Tan, Heng Wang, Yulia Tsvetkov, and Minnan Luo. 2024. DELL: Generating Reactions and Explanations for LLM-Based Misinformation Detection. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 2637–2667, Bangkok, Thailand. Association for Computational Linguistics.

Bing Wang, Ximing Li, Changchun Li, Bo Fu, Songwen Pei, and Shengsheng Wang. 2024. Why Misinformation is Created? Detecting them by Integrating Intent Features. *arXiv preprint*. ArXiv:2407.19196 [cs].

Yaqing Wang, Fenglong Ma, Zhiwei Jin, Ye Yuan, Guangxu Xun, Kishlay Jha, Lu Su, and Jing Gao. 2018. EANN: Event Adversarial Neural Networks for Multi-Modal Fake News Detection. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, pages 849–857, New York, NY, USA. Association for Computing Machinery.

Zhengjia Wang, Qiang Sheng, Danding Wang, Beizhe Hu, and Juan Cao. 2025. Bridging Thoughts and Words: Graph-Based Intent-Semantic Joint Learning for Fake News Detection. *arXiv preprint*. ArXiv:2509.01660 [cs] version: 1.

Liang Xiao, Qi Zhang, Chongyang Shi, Shoujin Wang, Usman Naseem, and Liang Hu. 2024. MSynFD: Multi-hop Syntax aware Fake News Detection. *arXiv preprint*. ArXiv:2402.14834 [cs] version: 2.

Yueqi Xie, Minghong Fang, Renjie Pi, and Neil Gong. 2024. GradSafe: Detecting Jailbreak Prompts for LLMs via Safety-Critical Gradient Analysis. *arXiv preprint*. ArXiv:2402.13494 [cs].

Jingyuan Yi, Zeqiu Xu, Tianyi Huang, and Peiyang Yu. 2025. Challenges and Innovations in LLM-Powered Fake News Detection: A Synthesis of Approaches and Future Directions. In *Proceedings of the 2025 2nd International Conference on Generative Artificial Intelligence and Information Security*, pages 87–93. Association for Computing Machinery, New York, NY, USA.

Zeping Yu and Sophia Ananiadou. 2024. Neuron-Level Knowledge Attribution in Large Language Models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3267–3280, Miami, Florida, USA. Association for Computational Linguistics.

Xueyao Zhang, Juan Cao, Xirong Li, Qiang Sheng, Lei Zhong, and Kai Shu. 2021. Mining Dual Emotion for Fake News Detection. In *Proceedings of the Web Conference 2021*, WWW '21, pages 3465–3476, New York, NY, USA. Association for Computing Machinery.

Yongchun Zhu, Qiang Sheng, Juan Cao, Shuokai Li, Danding Wang, and Fuzhen Zhuang. 2022. Generalizing to the Future: Mitigating Entity Bias in Fake News Detection. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2120–2125. ArXiv:2204.09484 [cs].

A Appendix

A.1 Informative nodes distribution

Lemma. *(Expected numbers of neurons selected) Given n number of layers, d_ℓ number of neurons per layer and top-k selection in each layer-layer block computation, the expected numbers of neurons per sample with non-zero influence is given by*

$$\mathbb{E}[x] = nd_\ell \left(1 - \frac{\binom{(d_\ell-1)^2}{k}}{\binom{d_\ell^2}{k}}\right)$$

Proof. We want to compute expected number of neurons with non-zero influence score, x. A neuron's influence score is computed by summing the i^{th} row and column of $\mathbf{B}$ matrix, equation 5.

Lets denote this by I:

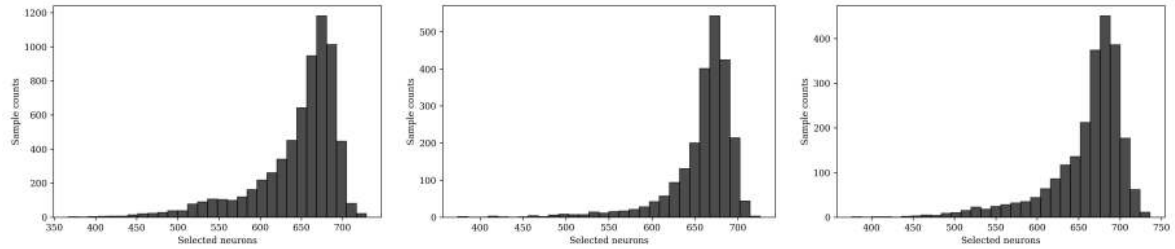
$$I_i = \begin{cases} 1, & \text{if row/column } i \text{ has} \geq 1 \text{ top-}k \text{ value,} \\ 0, & \text{otherwise.} \end{cases}$$

To get the expected value, we first need to find the probability of at least one top-k value being in i^{th} row or column, which can also be computed by complementing the probability of not finding any top-k value in i^{th} row or column. Thus,

$$P(I_i = 1) = 1 - P(I_i = 0)$$

To find $P(I_i = 0)$, we compute total number of ways the top-k values can be arranged in d_ℓ^2 cells except row i and column i. Total number of remaining cells are $d_\ell^2 - 2d_\ell + 1 = (d_\ell - 1)^2$. Adding back 1 to account for the overlapping cell where row and column i intersect (which is subtracted twice when removing row i and column i separately). Thus,

$$P(I_i = 0) = \frac{\binom{(d_\ell-1)^2}{k}}{\binom{d_\ell^2}{k}}$$



(a) Gossipcop (b) Snopes (c) Politifact

Figure 6: Distribution of number of informative neurons per sample on Llama-8B model on three datasets, for $k = 10$.

which gives,

$$P(I = 1) = 1 - \frac{\binom{(d_\ell - 1)^2}{k}}{\binom{d_\ell^2}{k}}$$

A neuron gets accounted if at least one top-k lies in either its column or row in $\mathbf{B}$ matrix. Thus, multiplying the above probability $P(I = 1)$ with total number of rows d_ℓ, gives us the expected value of number of neurons selected in a block, denoted by $\mathbb{E}[y]$.

$$\mathbb{E}[y] = d_\ell \left(1 - \frac{\binom{(d_\ell - 1)^2}{k}}{\binom{d_\ell^2}{k}}\right)$$

Total numbers of blocks computations in $\mathbf{B}$ are $n(n+1)/2$. But, if a neuron is selected in a single block then it gets accounted in the selected-neurons set, irrespective of later computations. Hence, the number of blocks which are responsible for computing influence score for unique neurons are n. Thus, over the whole binarized dominancy matrix $\mathbf{B}$, the expected numbers of influential neurons per sample becomes

$$\mathbb{E}[x] = n d_\ell \left(1 - \frac{\binom{(d_\ell - 1)^2}{k}}{\binom{d_\ell^2}{k}}\right)$$

Hence proved. □

Model	Eq:6	D1	D2	D3
Llama-8B	638.5	645.6	659.3	659.5
Llama-1B	318.5	276.1	285	286.6
Gemma-2B	358.3	310.2	312	316
Gemma-8B	559.3	556.9	536.4	537.3

Table 4: Number of neurons with non-zero information score for $k = 10$. Actual values of all 4 models for three datasets are shown. D1, D2 and D3 corresponds to Gossipcop, Snopes and Politifact datasets respectively.

A.2 Implementation Details

GRAIN method could be applied on any transformer based model with open access to its internals. We apply GRAIN on Llama and Gemma family of models across two model sizes. We use four models in our study, 1) Llama-3.2-1B-Instruct, 2) Llama-3.1-8B-Instruct, 3) Gemma-1.1-2B-it and 4) Gemma-1.1-7B-it. We obtain the activations ($\vec{a}$) and corresponding gradient vector ($\vec{g}$) for a given input prompt. During training phase of GNN, we utilise Adam as the optimiser for GNN with a batch size of 32. The GNN architecture involves two graph convolutional layers (GCN) followed with two linear layers. The output of GCN is batch normalized before passing it forward to next layer. The hidden dimension of both the GCN layer is 64. The output of both the GCN layer is concatenated and passed to global mean pooling which generates the graph level information of the input graph. Simultaneously, the graph level attribute (W) is passed to MLP which is concatenated with the GCN-generated graph level information. The concatenated information is passed to the linear heads to generate the classification probabilities. In our experiments, we compare our method with zero-shot and finetuned LLMs. Here, we only finetune the model's last layer Q, K and V projection matrices. We use the LoRA ((Hu et al., 2021)) method with a rank of 12 in finetuning of the model. We use early stopping on the validation dataset and report the result for the test set of that checkpoint.

A.3 Evaluation metrics

We report the following four metrics for evaluation of a method, namely Accuracy, macro-F1, F1-Real and F1-Fake. The *Gossipcop* and *Snopes* datasets have highly unbalanced positive and negative samples in their test dataset, and thus, we compare the macro-F1 score to evaluate various methods and

Method	#Train		#Validation		#Test	
	Real	Fake	Real	Fake	Real	Fake
Gossipcop	1950	1950	150	150	1746	600
Snopes	850	850	100	100	293	107
Politifact	816	815	117	116	234	233

Table 5: Statistics of MD datasets used in the study.

models. However, all four metrics are reported for all experiments for better insights to the reader.

A.4 Perturbation experiment examples

We use Llama-3.1-8B-Instruct model with few-shot examples for generating perturbed statement from a given input statement.

Active passive voice changeover examples

Input: The cat chased the mouse.
Output: The mouse was chased by the cat.
Input: The senate bill was approved by the president of the US.
Output: The president of the US approved the senate bill.

Add remove neutrals examples

Input: The cat sat on the mat.
Output: The cat comfortably sat on the mat.
Input: Unfortunately, the project was delayed due to unforeseen circumstances.
Output: The project was delayed due to unforeseen circumstances.

Clause reordering examples

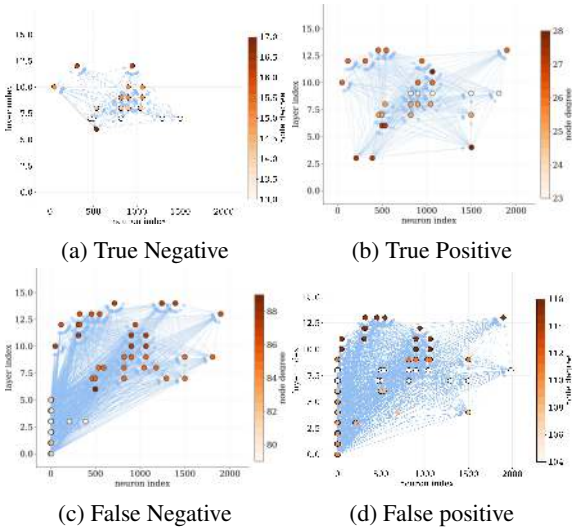
Input: Although it was raining, we went for a walk.
Output: We went for a walk, although it was raining.
Input: She completed her homework before she went out to play.
Output: Before she went out to play, she completed her homework.

Synonym replacement examples

Input: She enjoys reading books in her free time.
Output: She loves perusing novels in her leisure time.
Input: The weather today is sunny and warm.
Output: The climate today is bright and hot.

A.5 Intersecting graph analysis

Method to obtain intersecting graph We take multiple samples from a common class, i.e., True negatives, True positives, False negatives and False positives. We then list the edges from all the samples and make a union of them, keeping the count of occurrence of all the edges. We then threshold the edges by their occurrence counts and thus



(a) True Negative (b) True Positive

(c) False Negative (d) False positive

Figure 7: Intersecting graph over multiple samples for Llama-1B.

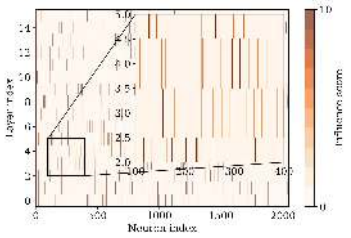
obtain the intersecting graph of m samples out of n samples. In this study we take 40 samples of True positives and True negatives and threshold the edges at 40 occurrences. This gives us the intersecting graph of 40 samples. Similarly, we get the intersecting graphs of 20 samples from false negatives and false positives each, with a threshold as 20 occurrences. Figure 4 and figure 7 shows intersecting graphs for Llama-8B and Llama-1B model respectively.

A.6 Plot of influence vector

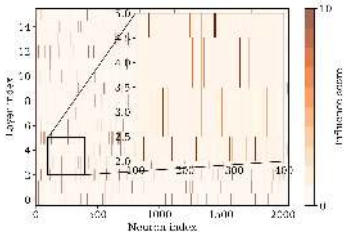
We take the influence vector generated for a sample and convert it into shape $n \times d_\ell$. i.e. number of layers times neuron index in a layer. As the influence matrix for a sample is very sparse, we sum the influence vector across the samples of the same class. Figure 8 and figure 9 show the sum of influence vote vector over multiple samples for the Llama-1B model and the Llama-8B model, respectively. Furthermore, we showcase the zoomed-view of a selected region in the plot to showcase that there are different patterns in influence scores of neurons for *True* positive and *True* negative samples.

A.7 NNI-IG method

We use a variant of the IG method, wherein only the selected influential neurons are activated in LLM. In IG method described by (Sundararajan et al., 2017), the whole LLM is used without any neuron de-activations. This tells us how the base LLM attributes importance to input tokens for a misinformation classification task. To compute the gra-

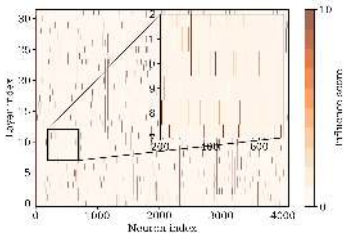


(a) True negative

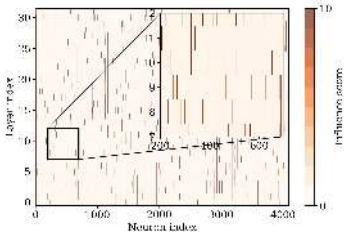


(b) True positive

Figure 8: Sum of influence vector over multiple samples for Llama-1B model.



(a) True negative



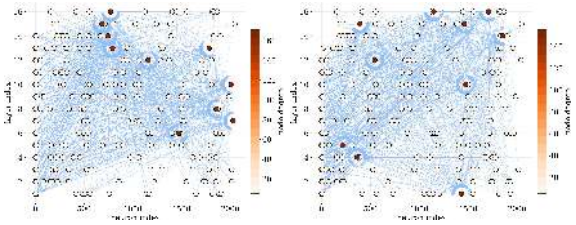
(b) True positive

Figure 9: Sum of influence vector over multiple samples for Llama-8B model.

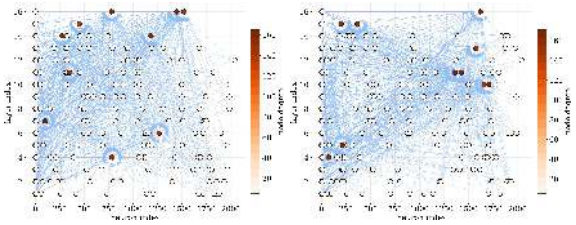
dients for the IG method, we use *Fake* token as a target token. Now, in the NNI-IG method, only allow influential neurons identified by the GRAIN method ($\vec{v}$) to be activated and the rest we deactivate. Then we perform the same computations as in the IG method to get the token level attribution.

A.8 Plot of sample level graph

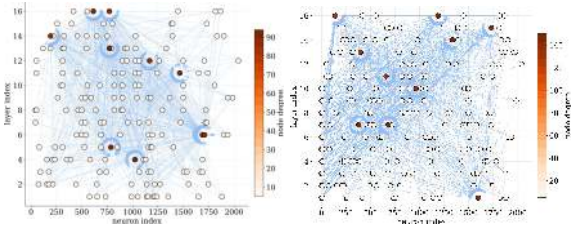
To visualise how the generated graph looks in the GRAIN method, we plot the graph for a few samples. However, due to a high number of edges in the graph for a single sample, we threshold the number of edges using the e_k parameter. Only for sample level graphs, we keep $e_k = 5$. The plots for *True* negatives and *True* positives for Llama-1B and Llama-8B models are shown in figure 10 and figure 11, respectively. Moreover, we tried different node arrangements, and we found that layer-wise and neuron-wise node arrangements give better intuition to the graph structure. Some of the arrangements are shown in figure 12 and figure 13, where we tried different circular arrangements of nodes. In this arrangement as well, we arrange the node layer-wise in a circular manner.



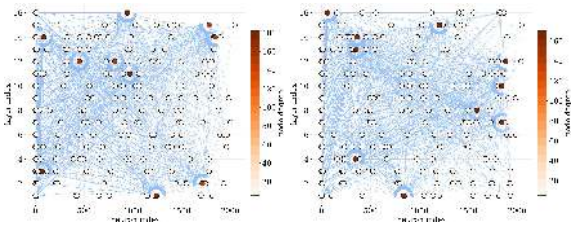
(a) True negative (b) True positive



(c) True negative (d) True positive

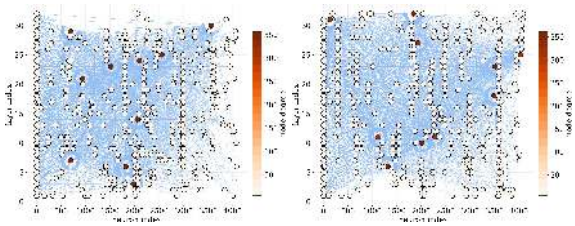


(e) True negative (f) True positive

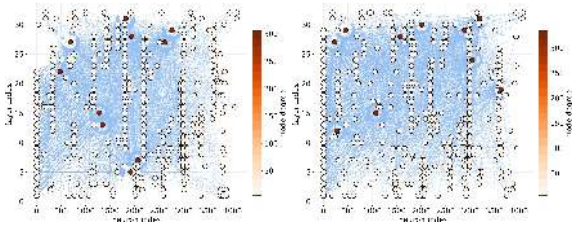


(g) True negative (h) True positive

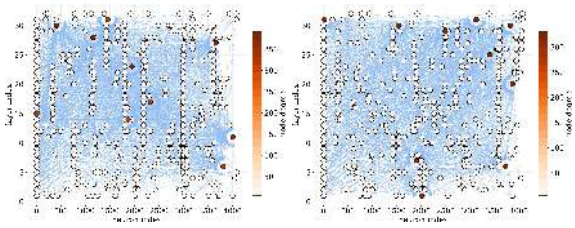
Figure 10: Per-sample graphs for Llama-1B model.



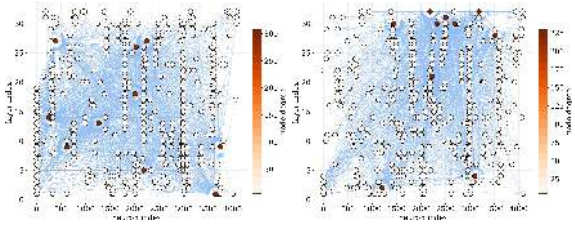
(a) True negative (b) True positive



(c) True negative (d) True positive

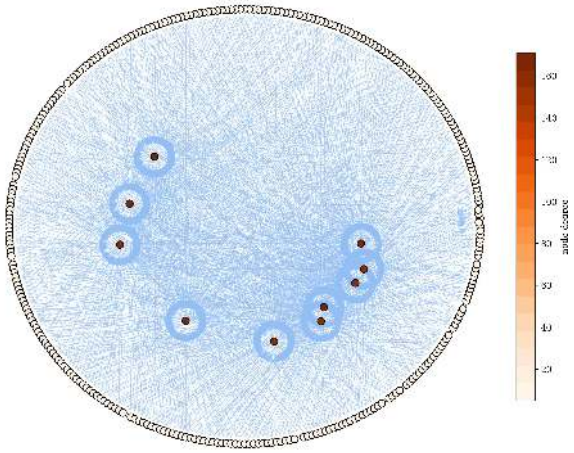


(e) True negative (f) True positive

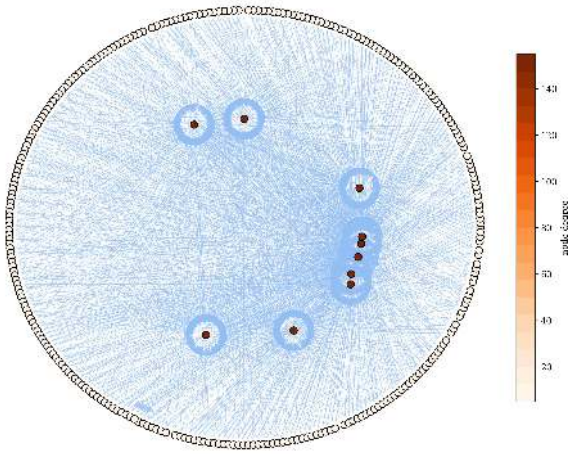


(g) True negative (h) True positive

Figure 11: Per-sample graphs for Llama-8B model.

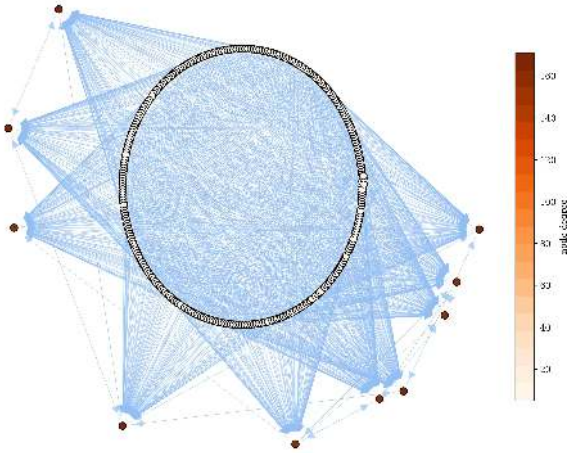


(a) True negative

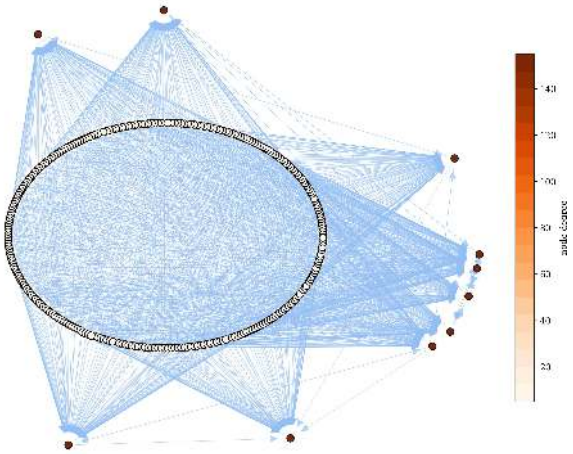


(b) True positive

Figure 12: Per-sample graphs for Llama-1B model.



(a) True negative



(b) True positive

Figure 13: Per-sample graphs for Llama-1B model.