EDN: A Novel Edge-Dependent Noise Model for Graph Data

Pintu Kumar[1] ✉[0000−0001−6111−1458] and Nandyala Hemachandra[1][0000−0003−2917−1551]

Indian Institute of Technology Bombay, India {pintuk,nh}@iitb.ac.in

Abstract. An important structural feature of a graph is its set of edges, as it captures the relationships among the nodes (the graph's topology). Existing node label noise models like Symmetric Label Noise (SLN) and Class Conditional Noise (CCN) disregard this important node relationship in graph data; and the Edge-Dependent Noise (EDN) model addresses this limitation. EDN posits that in real-world scenarios, label noise may be influenced by the connections between nodes. We explore three variants of EDN. A crucial notion that relates nodes and edges in a graph is the degree of a node; we show that in all three variants, the probability of a node's label corruption is dependent on its degree. Additionally, we compare the dependence of these probabilities on node degree across different variants. We performed experiments on popular graph datasets using 5 different GNN architectures and 8 noise robust algorithms for graph data. The results demonstrate that 2 variants of EDN lead to greater performance degradation in both Graph Neural Networks (GNNs) and existing noise-robust algorithms, as compared to traditional node label noise models. We statistically verify this by posing a suitable hypothesis-testing problem. This emphasizes the importance of incorporating EDN when evaluating noise robust algorithms for graphs, to enhance the reliability of graph-based learning in noisy environments. Link to code: https://github.com/pintu-dot/edn

Keywords: Graph Learning · New Label Noise Model for Graphs · Noise Robust Node Classification · Structure Aware Noise Model.

1 Introduction

Graph Neural Networks (GNNs) have shown good performance on the graph node classification task [1, 2]. GNNs assume that the available labels for training data are clean and noise-free, which may not be the case when working with real-world data [3]. Labels of real-world data are prone to noise, and noise can creep into data for many reasons, like expensive labelling, lack of expertise, human weariness, erroneous devices, adversaries changing labels, insufficient information to provide labels, etc [4, 3]. Hence, effectively learning for graph data in the presence of label noise has gained attention from the community [5–11].

One of the main reasons behind GNN's superior performance compared to traditional multilayer perceptron is that GNNs incorporate structural information during learning. Structural information is an integral part of graph data.

However, all current work on noise-robust graph learning uses one of the following: 1. Symmetric Label noise, 2. Pairwise noise / Class-Conditional noise, 3. Instance-Dependent noise. These noise models were originally proposed for i.i.d. data and not for graph data, and hence, they assume that label noise is independent of the structure of the node.

Consider a graph where nodes represent users in an online discussion forum. An edge between two nodes captures the interaction between users on a platform, such as a reply, comment, or question-answer exchange. Every user is assigned one of two labels *{helpful, not helpful}*. In such a graph, label noise can creep in two scenarios: **1.** Two helpful users who interacted with each other disagreed with each other or, due to some misunderstanding, couldn't convey their point of view, and hence, labelled each other as *not helpful*; **2.** Similarly, two unhelpful users might get incorrectly labelled as *helpful* because of a rare good discussion or due to colluding. In such graphs, noise dependent on just one node is not useful; rather, noise should be passed through edges. The labels of nodes on both sides of the edge should be changed together. We refer to this approach of adding noise to node labels as an edge-dependent noise model (EDN). In this work, we study three variants of edge-dependent noise and their impact on learnability.

The main contributions of the paper are as follows: **1.** Propose three variants of EDN, in which noise is passed through edges. **2.** For these noise models, we derive closed-form expressions for the probabilities of label change in terms of node degree. **3.** We analytically compare these probabilities as a function of node degree. **4.** We perform detailed experiments to check the behaviour of existing GNN architectures and noise-robust graph learning algorithms in the presence of EDN, using 5 different GNN architectures and 8 noise-robust graph learning algorithms. **5.** Based on confidence intervals of test accuracies for various noise models on many datasets, we observe that two variants of EDN at many noise levels substantially degrade the performance as compared to the existing noise label models. **6.** We pose this as a suitable hypothesis test problem to statistically verify our observations, and we conclude the same.

2 Related Work - Existing Node Label Noise Models

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph, where $\mathcal{V}$ denotes the set of vertices and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denotes the set of edges. Each node $v_i \in \mathcal{V}$, have an associated label $y_i \in \{1, 2, \cdots, K\}$. In models for label flipping, we can associate a discrete-time Markov chain [12] on the state space of labels $\{1, 2, \cdots, K\}$ in which each flip in the label corresponds to a state transition of the Markov Chain. We give details of the existing methods for adding noise to node labels; in each case, we identify the transition probability matrix of the associated Markov chain.

2.1 Symmetric Label Noise

Symmetric Label Noise (SLN) [13, 4] assumes that the label of a node is changed with some fixed probability ρ (and hence retained with probability $1 - \rho$). Also, the probability of a label being reassigned to each of the other classes is the same, which is $\rho/(K-1)$. Mathematically, if y and y' denote true and noisy label respectively, then $P(y'=n|y=m)=\frac{\rho}{K-1}$, where $n,m \in \{1,2,\ldots,K\}$ and $m \neq n$. Transition probability matrix for SLN (Q_{sln}) is given by

$$Q_{sln} = \begin{bmatrix} 1-\rho & \frac{\rho}{K-1} & \frac{\rho}{K-1} & \cdots & \frac{\rho}{K-1} \\ \frac{\rho}{K-1} & 1-\rho & \frac{\rho}{K-1} & \cdots & \frac{\rho}{K-1} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \frac{\rho}{K-1} & \cdots & \ddots & 1-\rho & \frac{\rho}{K-1} \\ \frac{\rho}{K-1} & \frac{\rho}{K-1} & \cdots & \frac{\rho}{K-1} & 1-\rho \end{bmatrix} \tag{1}$$

2.2 Class Conditional Noise

In Class Conditional Noise (CCN) [13, 4], the probability with which the label is changed depends on both y and y'. The probability of a node of class m being reassigned to class n is given by ρ_{mn} ($P(y'=n|y=m)=\rho_{mn}$), where $m \neq n$. So, a node with label m is flipped with probability $\rho_m = \sum_{i=1,i\neq m}^{K} \rho_{mi}$ and the label is retained with the probability $1-\rho_m$. The transition probability matrix (Q_{ccn}) is given by

$$Q_{ccn} = \begin{bmatrix} 1-\rho_1 & \rho_{12} & \rho_{13} & \cdots & \rho_{1K} \\ \rho_{21} & 1-\rho_2 & \rho_{23} & & \rho_{2K} \\ \vdots & & \ddots & \ddots & \vdots \\ \rho_{(K-1)1} & & & 1-\rho_{K-1} & \rho_{(K-1)K} \\ \rho_{K1} & \rho_{K2} & \cdots & \rho_{K(K-1)} & 1-\rho_K \end{bmatrix}$$

Pairwise Noise: Pairwise Noise (PWN) [13] is a special class of CCN. The motivation behind Pairwise Noise is that one is more likely to mislabel two similar classes. For Pairwise Noise $\rho_1 = \rho_2 = \ldots = \rho_K = \rho$, and the label is flipped to the next label (with probability ρ). The transition probability matrix Q_{pwn} is given by

$$Q_{pwn} = \begin{bmatrix} 1-\rho & \rho & 0 & \cdots & 0 \\ 0 & 1-\rho & \rho & & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & & & 1-\rho & \rho \\ \rho & 0 & \cdots & 0 & 1-\rho \end{bmatrix} \tag{2}$$

Many label noise robust algorithms for graphs have been proposed to tackle existing node label noise. DGNN [6] employs backward loss correction. PIGNN [7] leverages pairwise interactions (PI) between nodes for noise-resistant learning. RNCGLN [8] uses pseudo-labeling within a self-training framework to correct noisy labels. NRGNN [9] connects unlabeled nodes with high feature similarity to labelled nodes for better pseudo-labelling. RTGNN [10] enhances information flow by bridging labelled and unlabeled nodes while employing dual GNNs for noise mitigation. CRGNN [11] combines contrastive learning and dynamic cross-entropy loss to encourage robust feature learning. CGNN [5] integrates graph contrastive learning and a sample selection strategy based on the homophily assumption to filter noisy labels. DeGLIF [14] uses the influence function to identify and relabel noisy nodes in the graph. In our work, we would check how these algorithms perform in the presence of structure-dependent noise like EDN.

Some works, such as [15, 16], introduce structural noise into the graph by dropping edges. In contrast, our work focuses on node label noise propagated through the edges, rather than modifying the graph structure itself. Next, [17] and [18] explore structurally motivated adversarial or noisy label settings on graphs. However, they primarily focus on label flips and structural changes that mislead GNN's training, whereas our EDN model introduces a fundamentally different approach where label noise arises due to edge connectivity, making it structure-dependent and degree-sensitive.

3 A Novel Edge-Dependent Noise Model (EDN)

Assume $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, is an undirected graph, having m *nodes.* $\mathcal{X} = \{x_1, \ldots, x_m\}$ and $\mathcal{Y} = \{y_1, \ldots, y_m\}$ are the set of feature vectors and the set of true labels associated with corresponding nodes, respectively. We propose a node-label noise model for graphs (called edge-dependent noise model (EDN)) where the noisy labels of connected nodes are correlated as these noisy labels depend on the edge connecting them. Similar to existing noise models, edge-dependent noise models inject noise into the graph in two steps: 1. Selecting nodes whose labels should be changed, 2. Deciding new labels for selected nodes. What differentiates EDN from existing label noise models is that in EDN, the selection of nodes for label change depends on their structural information.

3.1 Selecting nodes to change their labels

In the proposed EDN model, we sample each edge with **fixed probability $\boldsymbol{\rho}$**; these sampled edges are called noisy edges. These noisy edges suggest that the labels of nodes on both sides of the noisy edge should be changed. For a node with degree 1, the label is changed if the edge incident to that node is noisy. For nodes with a degree $n > 1$, incident edges may have conflicting opinions on changing the label of the node. Based on how these opinions are aggregated, we have three variants:

Majority vote (MV): The label of a node v is changed if more than or equal to half of the edges incident to v are noisy. If the degree of v is $deg(v)$, the probability of a node getting selected to change its label is hence given by $q(deg(v), \rho)$

$$q(deg(v), \rho) = \sum_{i=\lceil \frac{deg(v)}{2} \rceil}^{deg(v)} \binom{deg(v)}{i} \rho^i (1-\rho)^{deg(v)-i}. \tag{3}$$

Veto power (Veto): The label of a node v is changed if at least one of the incident edges to v is noisy. The probability of the label of a node v with degree $deg(v)$ getting changed is hence given by $r(deg(v), \rho)$

$$r(deg(v), \rho) = 1 - \binom{deg(v)}{0}(1-\rho)^{deg(v)} = 1 - (1-\rho)^{deg(v)}. \tag{4}$$

Sequential flipping (seq): In this variant, the label of a node sequentially evolves. For a node v, we consider all its *noisy incident edges*. The first noisy incident edge changes the label to a new class. The second noisy edge changes the label from this new class to another (with a possibility of reverting to the original label). This process continues for all noisy edges. The probability with which a node v with degree $deg(v)$ is flipped depends on how new labels are assigned when observing a noisy edge and is discussed in Section 3.2.

3.2 Assigning noisy labels to selected nodes

For Majority vote and Veto power, after selecting nodes whose labels are to be changed, we use SLN and Pairwise noise [13, 4] to assign a new label. The difference between the existing and the EDN variants is that existing models have the same probability of selecting every node, whereas, in our noise model, the *probability of selecting a node is dependent on the degree of the node.* This is demonstrated by Equation 3,4,5 and 6. Existing noise models have the same transition probability matrix for all nodes, whereas EDN has different transition probability matrices for nodes with different degrees. In sequential flipping, the assignment of a new label due to a noisy edge follows SLN and Pairwise Noise. Both these sub-variants lead to different probabilities of flipping a node.

Sequential flipping + SLN: Each edge incident to node v is noisy with probability ρ. So, due to a single edge, the label of node v is unchanged with probability $1-\rho$, and the label changes to a different class with probability ρ. In the case of sequential flipping + SLN, when a noisy edge alters the label, the new label is selected according to the SLN model. This means that each possible class is equally likely, with a probability of $\frac{\rho}{K-1}$. The transition probability matrix associated with the noise caused by a single edge is given by Q_{SLN} from Equation 1. If v has degree n, then the transition probability matrix for Sequential flipping + SLN is Q^n_{SLN} (relabelling n times, every time starting with a new label). Q_{SLN} is a symmetric matrix, using the diagonalization property of symmetric matrix (Spectral theorem) [19], we derive Q^n_{SLN} as follows:

$$Q^n_{SLN} = \frac{1}{K}\begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & 1 & 1 & & 1 \\ \vdots & & \ddots & \ddots & \vdots \\ 1 & & & 1 & 1 \\ 1 & 1 & \dots & 1 & 1 \end{bmatrix} + \left(1 - \frac{K\rho}{K-1}\right)^n \begin{bmatrix} \frac{K-1}{K} & -\frac{1}{K} & -\frac{1}{K} & \dots & -\frac{1}{K} \\ -\frac{1}{K} & \frac{K-1}{K} & -\frac{1}{K} & & -\frac{1}{K} \\ \vdots & & \ddots & \ddots & \vdots \\ -\frac{1}{K} & & & \frac{K-1}{K} & -\frac{1}{K} \\ -\frac{1}{K} & -\frac{1}{K} & \dots & -\frac{1}{K} & \frac{k-1}{K} \end{bmatrix}$$

Detailed derivation for Q^n_{SLN} is available in Appendix A. Using Q^n_{SLN}, for a node v starting with the true label y, the probability of the label being changed to a specific class is given by:

$$s\_sc(deg(v), \rho) = \frac{1}{K}\left(1 - \left(1 - \frac{K\rho}{K-1}\right)^{deg(v)}\right).$$

For the node v, when using Sequential flipping + SLN model, the probability of its label being flipped is $(K-1) \times s\_sc(n)$ and is hence given by

$$s_{sln}(deg(v), \rho) = \frac{K-1}{K}\left(1 - \left(1 - \frac{K\rho}{K-1}\right)^{deg(v)}\right). \tag{5}$$

Sequential flipping + PWN: In sequential flipping with PWN, label reassignment due to a single edge follows the pairwise noise model. The corresponding transition probability matrix is given by Q_{pwn} from Equation 2. If v has degree n, then the transition probability matrix for Sequential flipping with the pairwise noise is Q^n_{pwn} (relabelling n times, every time starting with a new label). Observe that in Q_{pwn}, each row is a rightward cyclic shift of the previous row, which makes it a circulant matrix [20, 19]. We use the eigendecomposition of the circulant matrix [20, 19] to obtain Q^n_{pwn}. Since Q^n_{pwn} is a product of circulant matrices, it remains circulant [20, 19], meaning the entire matrix is characterised by its first row. The first row of Q^n_{pwn} is given by:

$$Q^n_{pwn}[0, j] = \sum_{m=0}^{n} \binom{n}{m} \rho^m (1-\rho)^{n-m} \delta_{m-j \mod K}.$$

Detailed derivation for Q^n_{pwn} is in the Appendix B. For the node v, when using the Sequential flipping + PWN model, the probability of its label being flipped is

$$s_{pwn}(deg(v), \rho) = \sum_{j=1}^{K-1} \sum_{m=1}^{deg(v)} \binom{deg(v)}{m} \rho^m (1-\rho)^{deg(v)-m} \delta_{m-j \mod K} \tag{6}$$

In all three variants, the probability of a node's label changing *depends on its degree.* The relationship between node degree and this probability is illustrated in Fig. 1. We observe that for a fixed ρ, in the majority vote variant, the probability $q(deg(v), \rho)$ decreases with slight fluctuations as the node degree increases. In the other two variants, both the probabilities $r(deg(v), \rho)$ and $s_{sln}(deg(v), \rho)$ monotonically increase and saturate at 1 and $\frac{K-1}{K}$ respectively. We summarize this as Theorem 1, with its proof provided in Appendix C:

Theorem 1. *Three variants of EDN satisfy the following properties:*

1. *For a fixed ρ, $r(deg(v), \rho)$ is an increasing function of $deg(v)$. Also, $s(deg(v), \rho)$ is an increasing function of $deg(v)$ for $\rho < \frac{K-1}{K}$.*

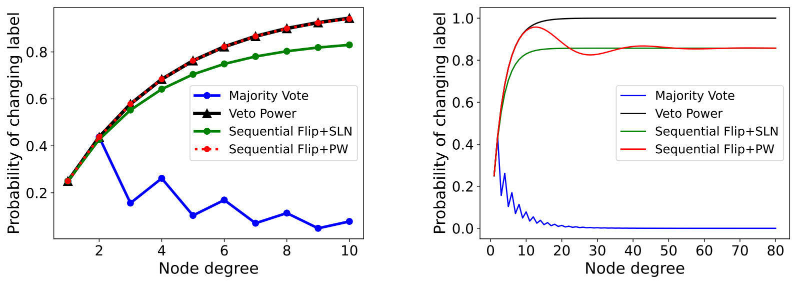

Fig. 1: Node degree vs probability of label change for three variants of EDN. We use $\rho = 0.25$ (the probability of an edge being noisy) and use $K = 7$ for sequential flipping.

2. *$r(deg(v), \rho) \geq q(deg(v), \rho)\ \forall\ deg(v)$ and fixed ρ.*
3. *If $\rho < \frac{K-1}{K}$, then $s_{sln}(deg(v), \rho) < \frac{K-1}{K}$ and $s_{sln}(n) = \frac{K-1}{K}$ iff $\rho = \frac{K-1}{K}$.*

The curve for s_{pwn} initially follows veto power and then oscillates to converge to seq+SLN. For $deg(v) \leq K-1$, we have $s_{pwn}(deg(v), \rho) = r(deg(v), \rho)$. In Figure 1, we observe that s_{pwn} forms a cycle of alternating increasing and decreasing phases. For a node v_i, without loss of generality, assume that its label $y_i = 1$. If $deg(v) = 1$, its label gets changed to class 2 with some probability, but never to any other class. If $deg(v) = 2$, v, moving from Q_{pwn} to Q^2_{pwn}, if the node has been already assigned label 2, it can not return to the original label and can either remain in 2 or get reassigned to class 3. If v_i was not assigned label 2, it now has the probability ρ of getting assigned to class 2. This implies $s_{pwn}(2, \rho) > s_{pwn}(1, \rho)$; meaning, s_{pwn} starts with an increasing phase. If $deg(v) = K-1$, then it can be reassigned to any class, and s_{pwn} is an increasing function of node degree up to this point. At $deg(v) = K$, if v_i was reassigned to class K by its first $K-1$ edges, then there is a small probability it gets reassigned to label 1. When this probability exceeds that of class 1 reassigned to class 2 (which occurs at $n = 13$ for $K = 7$), then it starts to decrease. and the decreasing phase will continue until another cycle is completed again, explaining the alternating pattern.

Recall Q_{pwn} is the transition probability matrix of a discrete-time Markov chain. This Markov chain is aperiodic and has only one communicating class, and $\pi = [1/K, 1/K, \ldots, 1/K]$ is the unique stationary distribution for Q_{pwn}. Hence, Q^n_{pwn} converges to a matrix Q^*, where every row of Q^* equals π [12]. So, $s_{pwn}(deg(v), \rho)$ converges to $(K-1) \times \frac{1}{K}$, this value is same as the upper bound for s_{sln}. We summarize this discussion about s_{pwn} as:

Theorem 2. *$s_{pwn}(deg(v), \rho)$, the probability with which the label of node v is changed in presence of Sequential flipping+PWN model, satisfies the following:*

1. *s_{pwn} forms a cycle of alternating increasing and decreasing phases. It initially increases, followed by a period of decrease, and this pattern continues.*

2. *For any fixed ρ we have $s_{pwn}(deg(v), \rho) = r(deg(v), \rho)$, for $deg(v) \leq K - 1$.*
3. *$s_{pwn}(deg(v), \rho)$ converges to $\frac{K-1}{K}$ as $deg(v) \to \infty$.*

4 Experiments and Results

4.1 Datasets and their splits

We test the impact of EDN on existing GNNs and Noise-robust algorithms. using Citeseer [21], Cora [21], and Amazon photo [22], with splits similar to DeGLIF [14]. Details about dataset statistics are in Table 1. These datasets were selected as they vary in node count, feature dimensions, and average degree.

Table 1: Dataset Statistics

Dataset	# Nodes	# Edges	Feature dim	# Classes
CiteSeer	3,327	9,104	3,703	6
Cora	2,708	10,556	1,433	7
Amazon Photo	7,650	238,162	745	8

Datasets split details: We use split similar to [14]. For the Cora dataset, we use 172 nodes per class for training, 500 nodes for validation, and 1000 nodes for testing. For the Citeseer dataset, we use 250 randomly sampled nodes per class for training, 500 nodes for validation, and 1000 nodes for testing. For the Amazon Photo dataset, we use 54 nodes per class for training, 500 nodes in total for validation, and the rest of the nodes for testing. All datasets have been fetched from the PyTorch Geometric library, with feature normalisation being true.

4.2 Injecting EDN noise

The same value of ρ can lead to different levels of noise using different variants of EDN noise models. Also, as variants of EDN are degree-dependent, graphs with different degree distributions can have different noise levels for the same ρ. Let $d(n)$ represent the degree distribution of a graph, then the expected noise level in the graph is given by $\sum_{i=1}^{maxdegree} l(i)d(i)$ where l is q, r or s. To make a fair comparison with the existing noise model and among different variants of EDN, we choose different ρ for each variant and each dataset so that the expected noise level in the graph is the same. A ready-to-refer value of ρ for different datasets, corresponding to overall noise levels in the graph ranging from 5% to 50% in increments of 5%, is available in Table 6 of the supplementary material.

4.3 Experimental Setup

We add different types of noise to data, where the noise level is between 5% and 50% in increments of 5%. GCN, GraphSAGE, GAT and Graph Transformer have been implemented using Pytorch Geometric using GCNConv, SAGEConv, GATConv and Graph Transformer respectively. For GCN and GraphSAGE we use 1 hidden layer of size 16 and relu activation. For GAT and Graph Transformer, we use 1 hidden layer with 8 heads of size 8. For GIN we use implementation by [13]. It is worth mentioning that the comparison is not between GNNs but between different noise models, so using slightly different architectures for different GNNs strengthens our experiments. We use implementation by [13] for all noise-robust algorithms except for DeGLIF for, which we use implementation by [14]. Each experiment is repeated 10 times, with mean $\pm$ standard deviation reported. Models are trained on a 24 GB Nvidia RTX 4090 GPU.

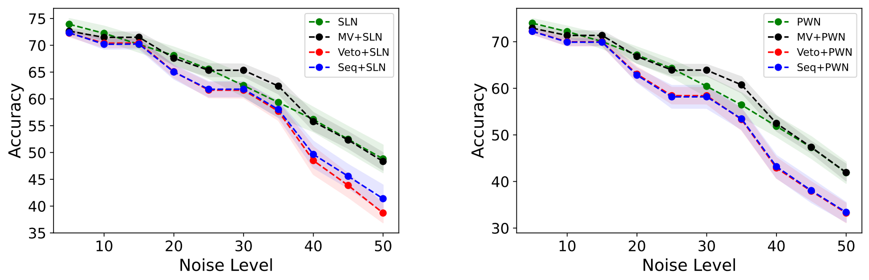

Fig. 2: GCN accuracy at same noise level but different noise models on Citeseer dataset.

4.4 Computational Results

In this section, we attempt to empirically understand the impact of EDN on existing graph learning algorithms. To do so we try to answer following questions:

Q1. How does GCN architecture perform in the presence of EDN? GCN [23] is one of the widely used GNN architecture, it is also used as a backbone for many noise robust algorithms for graph [9, 10, 14]. We test GCN under different noise variants of EDN, as well as SLN and PWN. Results for the Citesser dataset are reported in Fig 2. Results for all datasets at 5%, 25%, and 45% noise level is presented in Table 2. Due to the large size of tables and lack of space, results for all datasets at all noise levels are provided in Tables 7,8,9 in the supplementary material. GCN performance in the presence of the Majority vote variant is comparable to existing noise models, SLN and PWN. GCN shows more degradation in performance when injected with Veto Power and Sequential flipping variants of EDN as compared to the existing node label noise model. At low noise levels (less than 15%) the gap is low. However, at higher noise levels, veto power and sequential flipping gap widen, and they degrade GCN performance the most.

Table 2: Comparison of noise model variants across GNN architecture. Reported values are accuracy±std of 10 repetitions.

	GNN Architecture	Noise Level	SLN	MV+ SLN	Veto+ SLN	Seq+ SLN	PWN	MV+ PWN	Veto+ PWN	Seq+ PWN
	GCN	5%	73.92±1.1	72.64±0.5	72.27±0.7	72.34±0.6	73.99±1	72.91±0.6	72.24±0.6	72.23±0.58
		25%	65.55±1.74	65.34±1.20	61.64±1.43	61.79±1.42	64.29±1.91	63.93±1.22	58.42±1.76	58.17±2.43
		45%	52.47±2.39	52.33±1.55	43.86±2.21	45.57±2.21	47.30±2.34	47.36±1.25	37.93±2.02	38.05±2.56
C	GIN	5%	71.92±4.50	69.52±1.93	68.59±2.76	67.76±3.35	67.99±3.27	68.56±3.01	68.78±2.22	68.58±1.95
I		25%	60.12±5.71	63.39±3.58	54.51±4.01	55.42±6.26	54.22±9.29	61.47±3.51	49.34±7.72	49.98±4.16
T		45%	42.29±8.80	47.87±7.42	37.98±5.40	39.28±5.53	39.19±3.55	46.07±6.53	34.25±5.86	33.50±1.65
E	GraphSAGE	5%	75.95±0.96	74.96±0.51	74.69±0.49	74.61±0.62	75.97±0.88	75.07±0.56	74.00±0.53	75.69±1.30
S		25%	70.96±1.49	70.70±1.07	68.42±1.36	68.99±0.91	68.93±1.92	68.33±1.07	63.27±1.43	65.70±2.16
E		45%	59.61±1.99	59.97±1.79	52.89±2.11	54.17±2.08	49.71±2.85	49.19±1.56	41.19±2.29	44.32±4.10
E	GAT	5%	76.46±1.53	74.75±0.67	74.33±0.89	74.36±0.67	76.62±1.49	74.93±0.85	74.39±0.66	76.61±1.39
R		25%	74.59±1.61	73.06±1.04	71.81±1.06	72.16±1.39	73.31±1.84	71.32±1.50	68.13±1.45	71.42±2.64
		45%	70.88±1.64	70.45±1.81	66.60±2.33	67.51±1.87	55.00±3.75	53.93±2.64	43.41±3.18	45.97±6.50
	Graph Transformer	5%	76.28±0.86	75.64±0.52	76.28±0.50	76.54±0.54	76.10±1.40	75.50±0.14	76.16±0.35	75.50±0.95
		25%	70.24±1.63	70.36±0.60	68.84±0.68	68.68±0.86	68.86±1.03	67.48±1.12	64.94±0.68	65.45±2.00
		45%	58.10±1.24	59.22±1.42	52.38±0.60	53.58±0.94	49.04±2.25	48.94±2.43	43.00±1.01	44.09±2.77
	GCN	5%	84.73±0.94	85.00±0.44	85.19±0.74	85.09±0.73	84.27±0.98	85.36±0.48	85.36±0.50	84.58±0.54
		25%	76.46±1.48	77.46±1.99	74.49±2.09	74.65±2.03	71.97±1.77	74.84±2.58	68.85±2.30	76.35±1.44
		45%	62.29±2.29	63.16±2.94	57.12±2.54	58.06±2.54	50.52±2.76	52.86±3.87	44.28±2.62	50.18±3.92
	GIN	5%	80.95±2.01	81.11±0.88	80.45±2.63	82.11±2.60	80.35±2.38	83.34±1.55	80.77±2.21	79.36±1.99
		25%	79.02±2.77	77.87±5.65	74.90±2.73	76.57±2.59	73.66±3.74	73.32±4.28	64.84±3.21	67.62±2.51
C		45%	71.02±4.38	75.17±2.36	66.79±9.06	65.95±11.8	48.40±5.01	53.90±4.37	42.55±10.1	48.36±3.33
O	GraphSAGE	5%	83.10±1.19	83.38±0.68	83.37±1.18	83.41±1.09	82.84±1.31	83.56±0.46	83.22±1.13	84.43±0.81
R		25%	70.72±1.85	72.39±2.37	68.83±1.91	69.02±2.12	67.69±2.50	69.64±2.76	66.53±1.93	71.6±2.48
A		45%	55.27±2.50	53.93±3.23	51.14±2.74	51.35±2.62	47.20±3.15	49.40±3.33	46.21±2.12	47.77±5.17
	GAT	5%	79.50±1.80	80.05±1.02	80.39±1.14	80.45±1.20	79.35±1.67	79.43±1.51	79.45±1.37	79.06±1.5
		25%	70.33±2.16	69.58±2.69	70.77±2.57	70.48±1.44	65.87±3.48	65.64±2.34	64.80±2.47	66.09±2.37
		45%	57.46±3.68	57.14±3.53	55.49±2.94	55.56±3.14	44.76±3.42	46.71±3.98	43.82±3.07	45.25±5.46
	Graph Transformer	5%	84.42±0.84	84.78±0.29	84.70±0.41	84.64±0.67	84.06±0.71	84.70±0.14	84.96±0.54	84.12±0.99
		25%	75.30±0.91	76.56±0.70	77.18±0.67	77.50±0.87	72.12±1.05	72.18±1.08	72.88±0.95	70.33±2.01
		45%	61.16±1.75	62.16±1.86	61.32±1.56	60.62±2.15	50.08±2.15	51.32±0.92	51.28±1.44	48.11±2.4
	GCN	5%	86.78±1.43	83.65±6.28	85.05±4.56	83.55±6.18	86.66±1.46	84.65±5.59	85.35±4.21	84.85±2.89
A		25%	83.96±2.99	78.28±9.38	81.87±4.08	80.58±7.92	77.00±5.47	74.15±9.42	70.81±7.69	66.82±8.03
M		45%	73.45±5.67	75.36±8.82	66.92±11.98	67.85±11.36	44.15±4.85	49.57±7.92	40.74±2.65	40.03±5.61
A	GIN	5%	80.24±4.32	81.32±2.95	67.86±17.91	64.96±15.31	66.24±17.36	78.02±3.71	77.10±5.15	34.24±6.73
Z		25%	50.58±19.81	67.24±16.88	35.04±2.90	33.62±8.99	50.16±10.96	69.38±14.52	41.78±10.62	33.94±7.74
O		45%	34.38±7.71	40.28±4.80	24.74±4.46	27.04±7.18	33.52±17.39	48.84±17.31	25.14±3.94	22.58±7.66
N	GraphSAGE	5%	90.56±0.86	90.41±0.61	90.29±0.82	90.20±0.64	90.39±1.01	90.64±0.50	90.29±0.79	89.93±1.16
		25%	81.73±2.29	84.06±1.34	82.15±1.89	82.15±2.09	77.96±2.87	80.25±2.78	77.73±3.13	74.08±4.06
P		45%	65.88±3.47	68.72±2.34	61.36±4.87	62.66±4.42	50.78±4.01	55.60±4.85	47.78±4.23	43.61±6.1
H	GAT	5%	78.20±1.79	79.07±1.89	77.75±1.95	77.29±1.60	77.36±2.45	78.79±1.52	76.30±1.64	76.81±1.79
O		25%	63.20±3.03	66.08±2.52	63.13±2.32	62.52±2.51	61.62±3.64	63.45±3.42	61.21±1.97	58.14±3.82
T		45%	47.79±3.67	48.55±3.52	46.59±3.19	46.70±3.37	45.01±2.82	47.40±4.53	42.69±3.05	40.29±4.36
O	Graph Transformer	5%	85.57±0.83	80.87±8.55	84.13±1.88	84.25±1.36	85.48±0.95	85.80±0.72	84.32±1.22	84.94±0.61
		25%	74.34±3.35	76.04±2.34	71.73±2.25	72.54±3.45	71.27±4.41	73.05±2.94	72.28±1.72	69.8±1.86
		45%	58.77±2.13	58.33±4.13	57.54±2.01	57.76±1.11	54.38±2.85	49.03±6.71	54.25±3.26	49.8±4.64

Q2. How do other GNN architectures perform in the presence of EDN? Maybe GCN is not robust enough for EDN, what about other GNN architectures, we test common GNN architectures GIN[24], GraphSage[25], GAT[26], and Graph Transformer[27]. Result for Citeseer with 40% noise is pictorially reported in Figure 3. For all datasets at $5\%, 25\%$, and 45% noise levels, result is presented in Table 2. Detailed results for all datasets at all noise levels are in Tables 7,8,9. We observe that, similar to GCN, other GNN architectures' performance is also degraded more in the presence of Veto power and the Sequential flipping variant of EDN as compared to the existing noise model and majority vote variant.

Q3. How does the existing label noise robust algorithm perform in the presence of EDN? Next, we evaluate noise-robust algorithms DGNN [6],

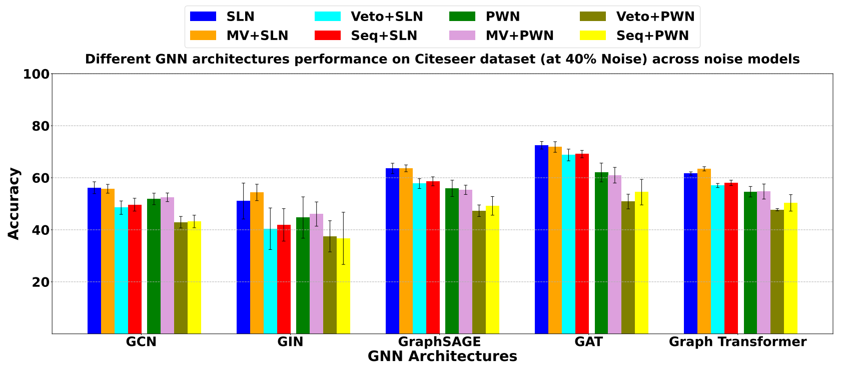

Fig. 3: Comparison of noise model variants across GNN architectures. A cluster is for an architecture, and coloured bars show the accuracy of the corresponding noise type.

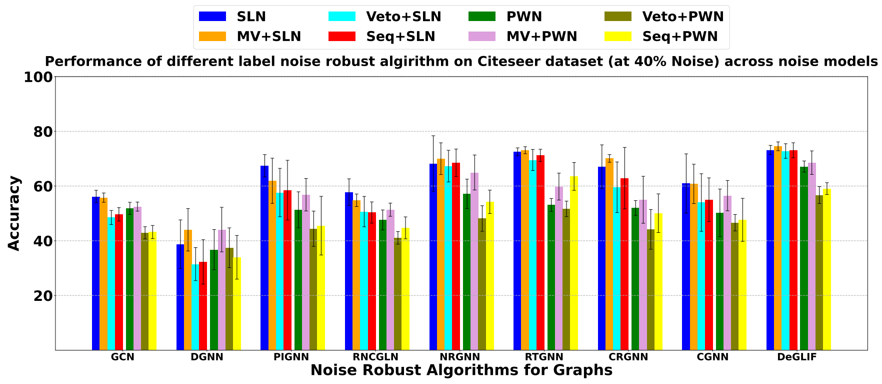

Fig. 4: Comparison of noise model variants across graph label noise-robust algorithms. A cluster is for an algorithm, and coloured bars show the accuracy of a noise type.

PIGNN [7], RNCGLN [8], NRGNN [9], RTGNN [10], CRGNN [11], CGNN [5], and DeGLIF [14] under different noise variants of EDN, as well as SLN and CCN. Graphical representation of the result for the Citeseer dataset with 40% noise is in Figure 4. For all datasets at 5%, 25%, and 45% noise level, result is presented in Tables 3 and 4. Detailed results for all datasets are in Tables 10,11,12 in the supplementary material. At low noise levels, these algorithms give comparable results across all noise models. At higher noise levels, most algorithms (Fig. 4) show trends similar to GCN. Performance under the majority vote variant is comparable to or better than SLN or CCN. However, under the Veto Power and sequential variants, almost all algorithms struggle to learn as robustly as they do with SLN or CCN.

The Majority Vote (MV) algorithm penalizes low-degree nodes, making its performance decline similar to SLN or PW. On the other hand, Veto power and Sequential flipping have higher noise levels for higher degrees (Fig 3). Variants of EDN leading to a greater performance drop suggest that the structure of nodes plays a critical role in how noise impacts performance, making such EDN variants valuable for robust evaluations.

Table 3: Comparison of noise model variants across graph label noise robust algorithms for Citeseer and Cora datasets. Reported values are accuracy±std of 10 repetitions.

	Noise Robust Methods	Noise Level	SLN	MV+ SLN	Veto+ SLN	Seq+ SLN	PWN	MV+ PWN	Veto+ PWN	Seq+ PWN
	DGNN	5%	66.46±2.84	65.15±2.37	62.17±2.41	60.28±3.58	64.04±2.22	66.06±1.90	61.92±8.79	58.12±13.30
		25%	53.07±4.92	51.18±10.06	45.02±6.99	44.70±6.52	49.78±5.92	53.62±6.91	46.24±6.07	42.28±12.04
		45%	41.89±6.73	43.71±4.72	29.48±6.39	27.27±9.29	32.62±6.84	44.71±5.86	32.83±4.41	34.20±7.13
	PIGNN	5%	76.58±2.04	72.83±3.60	71.34±5.36	71.61±5.36	74.02±2.20	73.60±1.74	71.63±5.38	73.14±2.17
		25%	71.61±3.67	69.62±3.46	66.79±6.75	66.11±8.15	66.19±5.57	68.11±4.49	62.79±4.89	63.62±7.72
		45%	60.79±11.05	56.18±9.03	56.32±3.70	58.49±7.80	44.47±7.21	49.97±7.33	38.14±4.95	44.48±9.20
	RNCGLN	5%	72.15±3.13	69.00±3.88	68.33±1.98	67.54±1.51	69.86±3.23	68.05±2.46	68.87±2.75	72.08±3.06
		25%	65.09±4.12	62.51±2.37	64.76±4.86	62.41±4.41	58.22±2.80	61.01±2.82	56.50±2.01	58.88±3.38
C		45%	51.68±5.01	51.01±3.35	44.24±4.52	45.94±4.38	41.87±2.96	46.81±3.73	37.95±3.20	40.72±1.63
I	RTGNN	5%	73.98±4.38	74.31±1.04	74.26±1.53	73.95±1.35	74.18±0.80	74.07±1.15	74.12±0.97	75.08±1.32
T		25%	72.47±1.78	72.81±1.55	72.82±1.97	71.95±3.23	65.87±2.81	71.07±2.65	66.09±2.46	70.22±3.38
E		45%	70.25±1.97	71.09±2.30	64.59±5.95	69.16±2.81	42.61±4.55	53.90±3.99	45.36±3.73	54.05±3.09
S	NRGNN	5%	75.76±0.99	74.19±1.47	74.22±1.19	74.14±1.15	71.69±3.75	73.56±2.35	74.53±1.63	76.58±3.30
E		25%	73.78±1.02	72.68±2.15	70.86±4.00	70.40±4.18	67.06±2.90	72.08±2.89	66.07±6.16	69.92±1.67
E		45%	70.80±2.43	70.73±5.18	65.17±6.89	64.09±8.24	46.98±3.99	55.79±5.34	40.30±5.00	49.60±10.83
R	CRGNN	5%	76.34±2.41	74.74±1.86	74.64±1.79	74.24±2.41	75.09±1.07	74.78±2.31	74.30±1.48	74.88±3.26
		25%	73.22±1.63	70.35±6.12	71.21±3.16	72.50±1.97	65.98±4.25	70.15±1.75	58.29±9.09	65.34±3.45
		45%	64.84±4.24	64.16±8.81	57.43±7.86	63.37±5.93	45.19±3.63	52.46±3.34	40.44±3.00	42.56±6.28
	CGNN	5%	77.80±0.83	74.20±1.74	73.69±1.85	73.11±3.40	72.67±4.49	73.51±2.24	74.21±1.59	76.60±4.10
		25%	69.86±4.46	70.54±3.82	69.04±5.15	70.09±4.03	61.63±9.43	67.52±6.32	60.02±7.17	63.08±9.39
		45%	58.62±8.03	55.96±11.66	50.57±11.89	50.62±10.95	44.70±6.39	50.49±6.26	41.45±3.50	46.02±4.30
	DeGLIF	5%	77.58±1.10	77.36±1.24	77.64±1.71	77.50±1.80	77.92±1.24	77.34±1.41	77.64±1.42	77.74±1.2
		25%	76.10±1.21	76.24±1.48	75.86±2.28	76.04±1.96	74.86±2.20	75.94±2.44	73.40±1.72	74.4±2.95
		45%	70.58±1.44	72.38±2.06	71.18±2.99	71.98±2.90	59.48±3.76	62.02±3.25	47±5.34	52.48±8.28
	DGNN	5%	78.32±4.81	82.68±2.81	79.80±1.65	78.08±4.58	74.66±9.15	82.50±0.95	78.38±7.58	79.18±1.75
		25%	74.90±5.87	79.38±2.39	65.20±11.22	65.38±13.62	61.78±9.81	77.36±3.91	53.28±12.29	66.54±7.27
		45%	62.34±12.99	60.08±15.94	40.26±8.56	40.10±8.99	46.64±11.84	45.02±11.65	30.94±6.26	52.36±8.54
	PIGNN	5%	80.19±3.16	81.48±2.16	81.81±2.22	81.71±2.15	80.05±3.50	81.58±1.97	81.79±1.88	81.48±2.84
		25%	79.84±2.30	80.11±2.09	80.66±2.37	78.41±5.17	76.54±3.43	75.71±3.57	72.85±3.56	72.87±8.57
		45%	75.68±2.69	75.74±2.92	74.95±6.39	74.61±5.72	49.64±10.92	46.89±9.64	47.60±7.82	57.05±10.22
	RNCGLN	5%	83.82±5.00	84.06±4.05	85.32±2.00	87.04±2.01	83.78±2.71	82.08±3.42	88.32±2.02	81.38±0.61
		25%	80.26±5.05	80.24±3.99	71.54±6.55	70.50±4.53	71.94±5.37	77.00±9.12	64.98±3.93	70.60±1.39
		45%	56.52±5.79	61.22±6.57	51.68±5.66	55.40±5.21	52.40±8.32	63.90±8.12	42.58±2.93	49.72±0.79
C	RTGNN	5%	73.54±2.02	74.42±2.12	74.64±1.78	75.04±1.70	75.29±3.09	73.91±2.63	74.28±1.90	73.95±1.58
O		25%	75.67±3.00	72.20±3.70	74.12±3.52	73.16±5.07	70.44±3.29	62.89±6.02	55.39±9.03	68.75±4.56
R		45%	66.44±7.61	64.95±8.76	68.27±6.98	71.45±5.01	57.38±6.77	51.81±3.87	40.51±8.34	42.77±7.32
A	NRGNN	5%	75.13±3.77	75.47±1.73	75.65±2.62	73.19±2.98	75.40±2.21	74.81±1.82	76.12±2.36	74.99±2.20
		25%	75.03±4.14	73.52±1.88	75.46±2.96	75.05±1.90	69.89±7.82	64.88±7.89	64.97±6.98	65.47±7.14
		45%	71.63±7.77	65.53±9.80	69.96±7.19	70.07±5.05	50.58±13.34	40.34±9.91	40.91±10.40	49.96±6.07
	CRGNN	5%	84.10±1.86	83.99±1.48	84.18±1.72	84.36±1.53	84.26±1.64	84.28±0.92	83.70±1.58	82.16±4.84
		25%	78.25±1.88	76.40±2.57	76.43±5.53	77.58±2.64	75.71±2.56	74.05±2.41	67.48±4.64	69.94±6.77
		45%	65.02±10.42	60.58±12.98	62.98±6.41	60.63±7.09	48.32±9.40	49.27±6.13	46.61±7.83	44.22±7.37
	CGNN	5%	83.70±3.67	82.43±4.47	83.15±3.11	82.81±3.26	83.57±2.19	83.15±2.76	78.94±11.48	83.27±2.92
		25%	79.46±3.45	74.93±5.98	77.06±4.52	76.29±4.45	75.82±3.25	70.34±9.60	70.54±3.38	71.37±10.88
		45%	67.76±6.19	63.57±9.94	65.63±13.02	65.15±13.00	51.20±7.78	47.45±6.24	46.39±8.02	47.56±10.42
	DeGLIF	5%	88.79±2.60	88.77±2.79	87.04±6.67	89.41±1.99	88.20±2.12	88.73±2.32	88.86±2.41	84.46±0.8
		25%	87.33±2.74	85.71±6.37	86.17±3.31	85.68±5.04	86.63±1.84	87.60±2.01	73.27±16.10	80.74±1.15
		45%	84.85±2.21	83.60±2.00	75.33±7.08	78.58±6.30	60.11±3.78	63.50±6.69	44.94±2.33	61.2±6.07

Table 4: Comparison of noise model variants across noise robust algorithms for graphs for the Amazon Photo dataset. Reported values are accuracy±std of 10 repetitions.

	Noise Robust Methods	Noise Level	SLN	MV+ SLN	Veto+ SLN	Seq+ SLN	PWN	MV+ PWN	Veto+ PWN	Seq+ PWN
	DGNN	5%	78.32±4.81	82.68±2.81	79.80±1.65	78.08±4.58	74.66±9.15	82.50±0.95	78.38±7.58	61.44±27.08
		25%	74.90±5.87	79.38±2.39	65.20±11.22	65.38±13.62	61.78±9.81	77.36±3.91	53.28±12.29	55.50±5.54
		45%	62.34±12.99	60.08±15.94	40.26±8.56	40.10±8.99	46.64±11.84	45.02±11.65	30.94±6.26	35.96±4.80
	PIGNN	5%	88.9±0.4	89.56±0.58	90.72±0.31	92.42±0.59	89.74±1.26	88.02±0.96	89.96±0.58	90.06±0.88
		25%	86.8±3.4	88.96±1.72	90.30±0.65	90.52±0.50	85.22±2.21	87.52±2.15	74.56±2.17	82.10±2.06
A		45%	82.2±4.2	87.22±2.37	79.86±7.22	80.64±3.36	60.44±8.51	62.38±9.04	42.86±6.44	63.52±2.81
M	RNCGLN	5%	83.82±5.00	84.06±4.05	85.32±2.00	87.04±2.01	83.78±2.71	82.08±3.42	88.32±2.02	84.48±3.53
A		25%	80.26±5.05	80.24±3.99	71.54±6.55	70.50±4.53	71.94±5.37	77.00±9.12	64.98±3.93	69.96±6.36
Z		45%	56.52±5.79	61.22±6.57	51.68±5.66	55.40±5.21	52.40±8.32	63.90±8.12	42.58±2.93	48.34±1.06
O	RTGNN	5%	80.8±5.3	81.93±2.17	81.46±2.44	84.14±2.19	82.24±0.86	83.45±1.10	81.95±1.42	82.04±1.53
N		25%	82.9±4.9	82.63±3.85	82.93±3.32	81.96±2.17	83.17±3.79	84.88±3.15	69.78±3.62	76.19±6.53
		45%	86±1.3	84.98±1.62	75.71±7.21	74.46±5.80	60.86±8.45	58.03±12.32	47.17±8.88	60.56±3.57
P	NRGNN	5%	69±8	87.52±0.90	87.34±1.68	88.40±2.77	89.74±1.26	87.08±1.67	86.56±1.54	85.90±2.81
H		25%	55.1±5.4	86.92±2.84	85.92±0.64	86.08±0.90	85.22±2.21	85.52±2.32	72.50±4.41	81.24±5.63
O		45%	54.5±6.2	86.70±1.72	73.78±4.98	81.42±2.51	60.44±8.51	66.20±8.31	49.90±3.87	58.78±1.89
T	CRGNN	5%	59.04±12.73	44.60±12.81	54.28±14.68	54.52±14.41	54.80±12.30	52.82±12.77	47.36±15.24	37.54±46.21
O		25%	45.02±10.86	37.82±11.13	35.76±11.64	35.22±14.65	47.68±20.70	31.22±7.03	41.32±7.76	32.74±39.74
		45%	29.48±15.05	31.20±5.21	23.12±5.72	18.82±4.97	33.14±13.04	37.74±8.31	27.42±4.28	19.78±21.92
	CGNN	5%	39.08±28.12	28.70±12.19	23.32±6.22	33.60±21.41	33.82±19.18	25.10±7.71	32.08±11.53	56.84±25.81
		25%	34.08±23.35	22.70±8.20	24.30±5.93	25.88±24.09	31.26±18.21	22.48±3.59	29.08±7.08	47.86±14.05
		45%	16.62±9.35	17.40±9.27	20.84±7.90	22.96±11.08	22.74±12.74	24.44±7.80	25.78±7.39	36.52±9.30
	DeGLIF	5%	88.79±2.60	88.77±2.79	87.04±6.67	89.41±1.99	88.20±2.12	88.73±2.32	88.86±2.41	89.09±2.13
		25%	87.33±2.74	85.71±6.37	86.17±3.31	85.68±5.04	86.63±1.84	87.60±2.01	73.27±16.10	79.08±4.45
		45%	84.85±2.21	83.60±2.00	75.33±7.08	78.58±6.30	60.11±3.78	63.50±6.69	44.94±2.33	52.92±11.73

4.5 Hypothesis Testing

All experiments to check the impact of EDN on GNN architectures and noise robust algorithms for graphs are repeated 10 times. For a particular noise level, Let $\{x^s_1, \dots x^s_{10}\}$, denote accuracy values obtained by GNN when noise is injected using SLN. Similarly let $\{x^v_1, \dots, x^v_1\}$ denote accuracy values obtained by GNN when noise is injected using Veto Power + SLN model. We want to use the theory of hypothesis testing to check if some of the variant of EDN really lead to more degradation in performance as compared to existing noise models. To do so, let, $d_i = x^s_i - x^v_i$. Then $\bar{d} = \bar{x}^s - \bar{x}^v$. We assume that d_i are sampled from a normal distribution with an unknown mean μ_d and unknown variance σ^2_d. We define hypothesis test as follows [28, 29]:

Null Hypothesis $H_0 : \mu_d \leq 0$
Alternate Hypothesis $H_1 : \mu_d > 0$

We would like to mention that the normal distribution has a domain of $(-\infty, \infty)$, but the possible values of d_i are restricted to $[-100, 100]$. Empirically, almost all observed values of d_i lie within $[-10, 10]$ and exhibit a small standard deviation. As a result, the probability of a well-fitted normal distribution assigning values beyond $[-100, 100]$ is extremely low. Therefore, it is reasonable to assume that d_i are sampled from a normal distribution. The estimate for a mean of d_i is given by $\bar{d}_i$. The pooled estimator S^2_d for variance of d_i is given by

(assuming that population variance of SLN and Veto+SLN are the same)

$$S_d^2 = \frac{(n-1)S_s^2 + (n-1)S_v^2}{2n-2} = \frac{S_s^2 + S_v^2}{2}$$

where, S_s^2 and S_v^2 are sample variance for SLN and Veto+SLN varaints respectively. Then, the test statistic is given by

$$T = \frac{\bar{d} - 0}{\sqrt{S_d^2\left(\frac{1}{n} + \frac{1}{n}\right)}} = \frac{\sqrt{n}\bar{d}}{\sqrt{S_s^2 + S_v^2}}$$

The significance level α test is to *reject* H_0 *if* $T \geq t_{\alpha,2n-2}$; not reject H_0, otherwise. Here $t_{\alpha,2n-2}$ denotes t-distribution with degree of freedom $2n-2$. Significance level α means that the probability of H_0 getting rejected when it is actually true is never greater than α. In our experiment, we expect the data to support the alternative hypothesis H_1, but do not want to make the assertion unless the data really gives convincing support. So, we have set up the test so that the alternate hypothesis is the one that we expect data to support and we hope to prove. Alternate hypothesis in such setup is also called the research hypothesis. By choosing a small α as the significance level, we minimize the risk of incorrectly concluding that the data supports the research hypothesis when it is actually false [29]. Common choices for α are 0.1, 0.05, and 0.005; here, we choose $\alpha = 0.05$.

Illustrative examples (A) Let us look at a few examples. For Citeseer data set, GCN architecture in presence of 5% noise, we have $\bar{x}^s = 73.92$, $\bar{x}^v = 72.27$, $S_s = 1.1$, $S_v = 0.7$ (from Table 2), and thus, $T = \frac{\sqrt{10}\times(73.92-72.27)}{\sqrt{1.1^2+0.7^2}} = 4.00184$. For significance level $\alpha = 0.05$, the value of $t_{\alpha,18} = 1.734$. As, $T > t_{0.05,18}$ so we reject H_0. It means for the Citeseer dataset at 5% noise, GCN performs worse in the presence of Veto + SLN as compared to SLN, and the probability of incorrectly concluding this is at most 5%. In fact, in this case, $T > t_{0.0005,18} = 3.921643$, so the probability of incorrectly concluding rejection of H_0 is at most 0.05%.

(B) For Citeseer data set, GIN architecture in presence of 45% noise, we have $\bar{x}^s = 42.29$, $\bar{x}^v = 37.98$, $S_s = 8.8$, $S_v = 5.4$ (from Table 2), and hence, $T = 1.32$. This means, $T < t_{0.05,18}$ and we accept H_0.

Analysis of Hypothesis Tests: We perform hypothesis testing for all datasets, for all GNN architectures and all noise robust algorithms and the results are presented in Table 5. We divide noise levels into three subclass, Low (5% − 15% noise levels), Medium (20%−35% noise levels), and High (40%−50% noise levels). Overall represents aggregate across all noise levels. We revisit questions asked in Section 4.4 through the lens of hypothesis testing. For GCN architectures, we have 10 noise levels × 3 datasets × 2 sub-variants (SLN and PWN) = 60 hypothesis tests. Out of 60, we were able to reject 32 null hypotheses with

$\alpha = 0.05$. As in (B), accepting H_0 at a low significance level does not always mean that the null hypothesis is true, but it means that we are unable to reject it with high confidence. In example (B) clearly $\bar{d} > 0$, but as the variances are high, we are not confident if $\bar{d} > 0$ is due to a change in noise model or due to inherent randomness in the learning process (randomness in training set sampling, stochastic gradient descent, etc). So, when we are able to reject H_0, we can say with high confidence that the decrease in performance is due to a change in the noise model (for example, a change from SLN to Veto+SLN).

Table 5: Summary of all Hypothesis Tests: The fraction of cases in which we rejected the null hypothesis at significance level $\alpha = 0.05$. Even algorithms designed to handle graph label noise experience greater performance degradation due to EDN compared to the existing noise models. We can say this with high confidence for 34% cases under Veto Power and 22% cases under Sequential Flipping. The same is true for 41% cases for GNN architectures.

Noise Model	Noise Level	GCN	GNN Architectures	Existing Noise Robust Algorithms
Veto Power	Low	$6/18 \approx 0.33$	$24/90 \approx 0.27$	$26/144 \approx 0.18$
	Medium	$10/24 \approx 0.42$	$45/120 \approx 0.38$	$54/192 \approx 0.28$
	High	$16/18 \approx 0.89$	$55/90 \approx 0.61$	$85/144 \approx 0.59$
	Overall	$32/60 \approx 0.53$	$124/300 \approx 0.41$	$165/480 \approx 0.34$
Sequential Flipping	Low	$7/18 \approx 0.38$	$23/90 \approx 0.26$	$26/144 \approx 0.18$
	Medium	$9/24 \approx 0.38$	$41/120 \approx 0.34$	$41/192 \approx 0.21$
	High	$14/18 \approx 0.78$	$59/90 \approx 0.66$	$39/144 \approx 0.27$
	Overall	$30/60 \approx 0.5$	$123/300 \approx 0.41$	$106/480 \approx 0.22$

From Table 5, we observe that Veto Power and Sequential Flipping cause greater performance degradation than traditional noise models across different noise levels. GCN is the most affected by EDN, while other GNN architectures show slightly more robustness. However, none of the GNN architecture performs similarly to the existing noise model and EDN across noise levels. Existing noise robust algorithms also fail to completely tackle two variants of EDN. Overall, at $\alpha = 0.05$ significance level, in 34% cases, Veto power degrades the performance of noise robust algorithm more as compared to existing noise models; sequential flipping degrades more performance in 22% of cases. This number increases to 41% for GNN architectures and increases to 50% for GCN. Also, we observe that the impact of EDN becomes prominent with an increase in noise levels.

5 Conclusion

In this work, we introduce a novel noise model for graph data called Edge-Dependent Noise (EDN). Unlike existing noise models used for graph data that were originally designed for i.i.d. data, EDN captures the impact of connections among nodes, on node label noise. We propose three variants of EDN - Majority Vote, Veto Power, and Sequential Flipping. In all three variants, the probability of a node's label being flipped is directly determined by its degree, making implementation feasible even for large graphs. This degree-dependency is a distinguishing feature of EDN. We theoretically compare the probabilities of label flipping as a function of the node degree for various EDN variants that we propose. Experiments followed by hypothesis testing on results reveal that EDN, especially the Veto power and Sequential flipping variants, leads to more significant performance degradation compared to existing noise models like SLN and CCN. This highlights the critical role of node in understanding the impact of noise on GNN performance, making EDN a valuable tool for robust evaluations of GNNs. This underscores the need for further research into developing noise-robust algorithms specifically designed to handle the complexities of edge-dependent noise in graph data. As the differences in accuracies that we consider are in the interval $[-100, 100]$, one can pursue the hypothesis testing approach in a more principled way without resorting to normal approximations or assuming equal population variance. Our hypothesis testing framework is Berhens-Fisher problem and currently has no completely satisfactory solution (Sec. 8.4.3 of [28]).

References

1. Shunxin Xiao, Shiping Wang, Yuanfei Dai, and Wenzhong Guo. Graph neural networks in node classification: survey and evaluation. *Machine Vision and Applications*, 33, 2021.
2. Jie Zhou, Ganqu Cui, Z. Zhang, Cheng Yang, Zhiyuan Liu, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI open*, 2020.
3. Wei Ju, Siyu Yi, Yifan Wang, Zhiping Xiao, Zhengyan Mao, Hourun Li, Yiyang Gu, Yifang Qin, Nan Yin, Senzhang Wang, Xinwang Liu, Xiao Luo, Philip S. Yu, and Ming Zhang. A survey of graph neural networks in real world: Imbalance, noise, privacy and ood challenges. *ArXiv*, abs/2403.04468, 2024.
4. Sandhya Tripathi and N. Hemachandra. Label noise: Problems and solutions. *Tutorial at IEEE DSAA*, 2020.
5. Jingyang Yuan, Xiao Luo, Yifang Qin, Yusheng Zhao, Wei Ju, and Ming Zhang. Learning on graphs under label noise. *ICASSP*, 2023.
6. Hoang NT, Choong Jun Jin, and Tsuyoshi Murata. Learning graph neural networks with noisy labels. *arXiv preprint arXiv:1905.01591*, 2019.
7. Xuefeng Du, Tian Bian, Yu Rong, Bo Han, Tongliang Liu, Tingyang Xu, Wenbing Huang, Yixuan Li, and Junzhou Huang. Noise-robust graph learning by estimating and leveraging pairwise interactions. *Trans. Mach. Learn. Res.*, 2021.
8. Yonghua Zhu, Lei Feng, Zhenyun Deng, Yang Chen, Robert Amor, and Michael Witbrock. Robust node classification on graph data with graph and label noise. In *AAAI Conference on Artificial Intelligence*, 2024.
9. Enyan Dai, Charu Aggarwal, and Suhang Wang. Nrgnn: Learning a label noise resistant graph neural network on sparsely and noisily labeled graphs. In *Proceedings of the 27th ACM SIGKDD*, pages 227–236, 2021.

10. Siyi Qian, Haochao Ying, Renjun Hu, Jingbo Zhou, Jintai Chen, Danny Ziyi Chen, and Jian Wu. Robust training of graph neural networks via noise governance. *ACM International Conference on Web Search and Data Mining*, 2023.
11. Xianxian Li, Qiyu Li, De Li, Haodong Qian, and Jinyan Wang. Contrastive learning of graphs under label noise. *Neural networks*, 2024.
12. James R. Norris. *Markov Chains*. Cambridge University Press, 1997.
13. Zhonghao Wang, Danyu Sun, Sheng Zhou, Haobo Wang, Jiapei Fan, Longtao Huang, and Jiajun Bu. Noisygl: A comprehensive benchmark for graph neural networks under label noise. *NeurIPS Dataset Benchmark*, 2024.
14. Pintu Kumar and Nandyala Hemachandra. DeGLIF for Label Noise Robust Node Classification using GNNs. *arXiv*, abs/2506.00244, 2025.
15. Zhaoliang Chen, Zhihao Wu, Ylli Sadikaj, Claudia Plant, Hong-Ning Dai, Shiping Wang, and Wenzhong Guo. Adedgedrop: Adversarial edge dropping for robust graph neural networks. *ArXiv*, abs/2403.09171, 2024.
16. Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. In *International Conference on Learning Representations*, 2019.
17. Xuanqing Liu, Si Si, Xiaojin Zhu, Yang Li, and Cho-Jui Hsieh. A unified framework for data poisoning attack to graph-based semi-supervised learning. In *Neural Information Processing Systems*, 2019.
18. Mengmei Zhang, Linmei Hu, Chuan Shi, and Xiao Wang. Adversarial label-flipping attack and defense for graph neural networks. *2020 IEEE International Conference on Data Mining (ICDM)*, 2020.
19. Gilbert Strang. Linear algebra and its applications, 2000.
20. P.J. Davis. *Circulant Matrices*. Monographs and textbooks in pure and applied mathematics. Wiley, 1979.
21. Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. *ArXiv*, abs/1603.08861, 2016.
22. Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*, 2018.
23. Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.
24. Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *ArXiv*, abs/1810.00826, 2018.
25. William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NIPS*, 2017.
26. Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio', and Yoshua Bengio. Graph attention networks. *ArXiv*, abs/1710.10903, 2017.
27. Yunsheng Shi, Zhengjie Huang, Wenjin Wang, Hui Zhong, Shikun Feng, and Yu Sun. Masked label prediction: Unified massage passing model for semi-supervised classification. *IJCAI*, 2021.
28. Sheldon M. Ross. *Introduction to Probability and Statistics for Engineers and Scientists*. Academic Press, 6th edition, 2020.
29. George Casella and Roger L. Berger. *Statistical Inference*. Taylor & Francis, 2nd edition, 2024.