Uncertainty-Aware Robust Learning on Noisy Graphs

Anonymous Author(s)
Affiliation
Address
email

Abstract

Graph neural networks have shown impressive capabilities in solving various graph learning tasks, particularly excelling in node classification. However, their effectiveness can be hindered by the challenges arising from the widespread existence of noisy measurements associated with the topological or nodal information present in real-world graphs. These inaccuracies in observations can corrupt the crucial patterns within the graph data, ultimately resulting in undesirable performance in practical applications. To address these issues, this paper proposes a novel uncertainty-aware graph learning framework motivated by distributionally robust optimization. The framework aims to alleviate the challenges by considering the distributional uncertainty associated with the graph data. Specifically, we use a graph neural network-based encoder to embed the node features and find the optimal node embeddings by minimizing the worst-case risk through a minimax formulation. Such an uncertainty-aware learning process leads to improved node representations and a more robust graph predictive model that effectively mitigates the impact of uncertainty arising from data noise. The learned Least Favorable Distributions (LFDs) also provide a means to quantify the predictive uncertainty, which is valuable in some uncertainty-sensitive scenarios where incorrect decisions can have severe consequence. In addition, we adopt the idea of differentiable optimization and develop an end-to-end learning algorithm that seamlessly integrates graph learning and distributionally robust optimization. Our experimental result shows that the proposed framework achieves superior predictive performance compared to the state-of-the-art baselines under various noisy settings.

1 Introduction

The field of graph learning has witnessed significant advancements in recent years, fueled by the remarkable performance of graph neural networks (GNNs) [1, 2]. In general, GNNs leverage message-passing techniques to enable efficient information exchange among nodes in a graph, leading to improved graph embeddings. Among the various graph-based learning tasks, node classification, especially in a semi-supervised setting, stands out as a prominent and widely applicable problem that has greatly benefited from GNNs. The objective of semi-supervised node classification is to learn high-quality node embeddings and make predictions on unlabeled nodes on a graph that has only a small subset of nodes labeled.

However, similar to other deep neural networks, GNNs are susceptible to the influence of noisy input, including both noisy node features and topological structure [3, 4, 5] of the graph. This issue becomes even more critical in the low-data setting, where the performance can be greatly impaired by noisy observations when only a limited number of labeled nodes are available [3, 4, 6]. For example, in a social network, new users may inconsistently engage with content that aligns with their actual interests or express their preferences due to limited options available at that time. Such noises would introduce uncertainty and incorrect information about the user's preferences into the training data.

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

With only a few labeled samples available, the GNN heavily relies on the noisy graph to learn the user's preferences and generalize them to make less accurate recommendations. Hence, the GNN's capacity to capture the underlying patterns of different nodes can be significantly compromised when the input graph is noisy. Therefore, it is challenging but imperative to achieve robust learning on noisy graphs, especially for the problem of semi-supervised node classification.

In this paper, we propose a novel uncertainty-aware graph learning framework based on the idea of distributionally robust optimization (DRO) [7, 8, 9, 10]. The proposed *Distributionally Robust Graph Learning* (DRGL) framework significantly improves node embedding quality and enhances model performance, particularly in the presence of noisy node features and topological structure that hinder the direct observation of key patterns. To tackle the challenge, we first allow the underlying node feature patterns to vary within a pre-defined family of distributions and seek the least favorable distribution (LFD) that minimizes the risk in the worst-case scenario, as illustrated by Figure 1. It is important to note that the uncertainty arising from data noise is represented by an uncertainty set [9, 11, 12], characterized by a Wasserstein ball [9, 11, 13, 14]. Then, by minimizing the risk associated with this worst-case distribution, we can uncover the most robust node embeddings that effectively mitigate the impact of uncertainty stemming from the noisy observation. To seamlessly integrate this minimax formulation into the graph learning process, we also leverage differentiable optimization techniques [15, 16] and develop a tractable end-to-end learning algorithm. Hence, the resulting minimax solution also provides a means to estimate the potential uncertainty associated with predictions, allowing decision-makers to assess the uncertainty of different outcomes. This information is particularly valuable in some high-stakes scenarios where incorrect decisions can have severe consequences. In summary, the contributions of this paper are threefold:

- We study the problem of robust graph learning under a challenging context, where both node attributes and topological structure are noisy.
- We present a model-agnostic graph learning framework – *Distributionally Robust Graph Learning* (DRGL), which improves the reliability of GNNs against noisy graph inputs, especially when labeled data is limited.
- We conduct extensive experiments that demonstrate the proposed framework significantly enhances the quality of node embeddings and improves the predictive performance of GNNs on semi-supervised node classification.

2 Related Work

Graph neural networks (GNNs) have emerged as powerful models aiming to learn expressive representations for graph-structure data [1, 2, 17, 18]. While these approaches proved to be successful [1, 2, 19], they are highly sensitive to the quality of the given graphs in terms of either node feature or graph structure [3, 5].

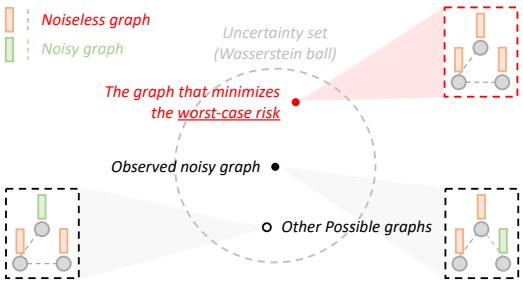


Figure 1: An illustration of the uncertainty set in our proposed framework. The goal is to search for the graph distribution that minimizes the worst-case risk.

To address the robustness limitations of GNNs in the presence of noises in node features and edges, researchers have explored different methods to purify the noisy graph [3, 4, 20]. For example, graph structure learning (GSL) focuses on the joint learning of an optimized graph structure and its corresponding representations. GSL methods typically leverage GNNs as graph encoders and employs specialized graph structure learners to refine and recover accurate graph structures, thereby enhancing the robustness of GNNs. [21, 22, 23, 24]. Another line of research focuses on improving the robustness of GNNs against inaccurate or missing node features [25, 26, 27] through developing noise-resistant aggregation and propagation techniques.

However, a significant limitation of the these methods is that they require a task-specific design of structure learning or regularization term tailored to address specific types of noise. This inherent

task-specific nature limits their applicability in more general application scenarios. For instance, [5, 28] propose parametric learners that aim to denoise graph through pruning edges, but such methods cannot handle noise due to inaccuracies in node features. In contrast, our method handles both edge and feature noise directly in the GNN embedding space, offering a more unified and direct approach for robust graph representation learning.

Another related research area is defense against graph attacks, which often leverages graph properties observed in real-world graphs to regularize the learning process. Notable methods in this area include GCN-SVD [29], which preprocesses GCN with the low-rank approximation of the perturbed graph. This approach is inspired by the observation that only the high-rank (low-valued) singular components of the graph are susceptible to perturbation. Additionally, ProGNN [24] is designed to guide the propagation process by intrinsic graph properties such as low-rank and sparsity in the graph structure, as well as the tendency for adjacent nodes to exhibit similar features. RGCN [30] models node representations as Gaussian distributions to mitigate adversarial attacks. However, these methods are primarily designed to handle various types of attacks, including random perturbations, in graphs with ample samples. To the best of our knowledge, they have not been extensively tested in low-data scenarios against random graph noise, which distinguishes our setting from this body of research.

In contrast to the aforementioned methods, our approach handles noisy graphs based on the DRO [7, 8], which aims to enhance the robustness of GNNs by minimizing the worst-case risk in the presence of data noise. Only a handful of recent research attempts, such as [11, 31], have embraced similar concepts and investigated the potential applications of DRO in the context of graph-structured data. For instance, Sadeghi et al. [31] use GNNs to encode the input graph and employ DRO for model training. However, their work is mainly focused on the noisy node features, while we consider a more general scenario involving both noisy features and edges. Furthermore, their method solves for worst-case distribution and updates model parameters separately at each iteration [31], while our approach offers an end-to-end approach updating parameters with gradient-based learning techniques. Additionally, our approach provides uncertainty quantification under Least Favorable Distributions (LFDs). This capability has meaningful applications in uncertainty-sensitive scenarios [32, 33], such as molecular classification tasks in graphs [34, 35].

3 Uncertainty-aware Graph Learning

3.1 Problem Setup

Consider an attributed graph $(\mathcal{I}, \mathcal{E})$, where $\mathcal{I} = \{i = 1, \ldots, n\}$ represents the set of n nodes, and $\mathcal{E} = \{(i, j), i, j \in \mathcal{I}\}$ represents the set of edges connecting the nodes. Each node i is associated with a d-dimensional feature vector $x_i \in \mathbb{R}^d$. The collection of all node features is denoted as $X = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{n \times d}$. To describe the graph more generally, we can represent it as $\mathcal{G} = (X, A)$, where $A = (a_{ij}) \in \{0, 1\}^{n \times n}$ is the adjacency matrix, which provides information about the connectivity between nodes. If $(i, j) \in \mathcal{E}$, then $a_{ij} = 1$; otherwise, $a_{ij} = 0$. Each node can be assigned one of the discrete labels $y \in \{m = 1, \ldots, M\}$. In the context of semi-supervised node classification, we have access to labels for only a subset of nodes, which we denote as $y_o = [y_1, \ldots, y_{n'}]^\top$, where n' represents the number of observed nodes. The remaining nodes have no assigned labels and are denoted as $y_u = [y_{n'+1}, \ldots, y_n]$. In our setting, the assigned labels are assumed to be correct, but there might be errors or inaccuracies in the observed edges or node features due to noisy measurements. Our objective is to accurately predict the labels y_u for these unobserved nodes in the graph.

3.2 Node Embedding

We use a graph encoder to embed the node features, including both the nodal information and the corresponding graph structure. We emphasize that our framework is not tied to any specific graph model and offers flexibility in selecting graph encoders. In this study, we use Graph Convolutional Networks (GCNs) [1, 36] and Graph Attention Networks (GATs) [2] to encode both the node feature and the graph topology. For each node i, the graph encoder functions as a nonlinear transformation, denoted as ϕ_θ, taking the nodal features and the corresponding graph topology as input and returning their node embeddings, denoted by $\xi \in \Xi$. Formally,

$$\phi_\theta(\cdot, \mathcal{G}) : \mathcal{I} \to \Xi.$$

Here $\theta \in \Theta$ denotes the parameters specific to the model used in our framework.

It's important to note that the node embeddings obtained through the minimization of standard loss functions (*e.g.*, cross-entropy loss) may not accurately capture the key feature patterns of the graph in our problem setting [3, 4, 5]. The presence of noise in the observed graph $\mathcal{G}$, compounded by the limited number of labeled nodes, can result in the model fitting too closely to the noise, being misled by incorrect gradients, and generalizing poorly. Such models can be vulnerable to outliers, which might compromise the quality of the learned representations.

3.3 Distributionally Robust Graph Learning

In light of these challenges, one of the goals is to enhance the robustness of graph encoders directly within their embedding space. This robustification process ensures that the embeddings better align with the underlying graph structure, even in the presence of noise and perturbations.

We develop a novel graph learning framework that improves the node embeddings, resulting in more robust predictive performance, particularly when confronted with noisy data. Figure 2 summarizes the architecture of the proposed model.

Our approach assumes that node embeddings share the same label m adhere to an underlying distribution ($\xi_i \sim P_m \in \mathcal{P}_m, \forall i : y_i = m$) within an uncertainty set $\mathcal{P}_m$ encompassing all potential distributions P_m. However, obtaining a direct observation of this distribution is challenging due to inaccuracies in the nodal or topological information, and any changes in node embeddings will influence their corresponding distributions. The key idea of our proposed framework is to find the most robust node embeddings, parameterized by θ, that minimize the worst-case risk over the uncertainty set $\mathcal{P}_m$ in the probability simplex $\Delta_M = \{\pi \in \mathbb{R}_+^M : \sum_{m=1}^M \pi_m = 1\}$. This results in the definition of our distributionally robust minimax problem:

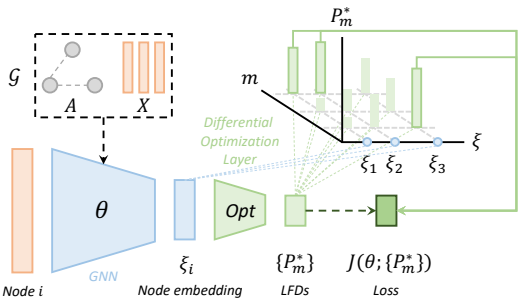


Figure 2: The architecture of the proposed framework consists of two cohesive modules: (1) a graph encoder parameterized by θ, which produces the node embedding ξ given the graph information $\mathcal{G}$; (2) a differential optimization layer, which generates the corresponding least favorable distributions (LFDs) $\{P_m^*\}$ for ξ by solving the convex optimization defined in (4).

$$\min_{\pi \in \Delta_M} \max_{\substack{P_m \in \mathcal{P}_m \\ 1 \le m \le M}} \Psi\left(\pi; P_1, \ldots, P_M\right), \tag{1}$$

where Ψ is the risk function of a classifier π. We define the risk function as the summation of error probabilities under each class, *i.e.*, $\Psi\left(\pi; P_1, \ldots, P_M\right) := \sum_{m=1}^M \mathbb{E}_{\xi \sim P_m}\left[1 - \pi_m(\xi)\right]$. We note that the optimal solution $P_1^*, \ldots, P_M^*$ to the inner maximization problem is known as the *least favorable distributions* (LFDs) in statistics literature [9, 37]. The risk associated with these distributions is considered the worst-case risk [9].

As shown in Figure 3, we choose the uncertainty set $\mathcal{P}_m$ to be a Wasserstein ball of radius ϑ_m centered at the empirical distribution $\widehat{P}_m$:

$$\mathcal{P}_m := \{P_m \in \mathscr{P}(\Xi) : \mathcal{W}_1(P_m, \widehat{P}_m) \le \vartheta_m\}, \tag{2}$$

where $\mathscr{P}(\Xi)$ denotes the set of all probability distributions on Ξ. The Wasserstein distance of order one, $\mathcal{W}_1$, is defined as $\mathcal{W}_1\left(P, P'\right) := \min_\gamma \mathbb{E}_{(\xi, \xi') \sim \gamma}\left[c\left(\xi, \xi'\right)\right]$, where $c(u, v)$ is some cost function transferring from u to v, $c(u, v) \ge 0$. The empirical distribution $\widehat{P}_m$ is represented by the Dirac point mass, denoted as:

$$\widehat{P}_m := \frac{1}{|\{i : y_i = m\}|} \sum_{i=1}^n \delta_{\xi_i} \mathbb{1}\left\{y_i = m\right\}, m = 1, \ldots, M, \tag{3}$$

Here, δ refers to the Dirac delta function, $|\cdot|$ represents the cardinality of a set, and $\mathbb{1}$ denotes the indicator function.

Algorithm 1 Learning algorithm of DRGL

Input: $\mathcal{G} = (X, A)$; $y_o = [y_1, \ldots, y_{n'}]^\top$; θ_0;
Output: θ_T
for $t \leftarrow [0 \ldots \mathrm{T}]$ **do**
  $\mathcal{L}(\theta_t) = 0$;
  for *each mini-set* **do**
    Compute the node embeddings $\{\xi\}$ given θ_t for all labeled nodes in the mini-set;
    Calculate $\widehat{P}_1, \ldots, \widehat{P}_M$ given $\{\xi\}$ and y_o using (3);
    Obtain LFDs $P_1^*, \ldots, P_M^*$ by solving (4) with DO given $\{\xi\}$ and $\widehat{P}$;
    $\mathcal{L}(\theta_t) \leftarrow \mathcal{L}(\theta_t) + J(\theta_t; P^*)$ using (5);
  end for
  $\theta_{t+1} \leftarrow \theta_t - \alpha \nabla \mathcal{L}(\theta_t)$ (α is the learning rate);
end for
return θ_T

As the original problem (1) entails an intractable infinite dimensional functional optimization, we follow [9, 10] and introduce the proposition below. This proposition reformulates the problem into a computationally tractable convex optimization problem due to our careful selection of the risk function and uncertainty sets. We note that this selection exploits the structure of the least favorable distributions yielding from Wasserstein uncertainty sets [9].

Proposition 1 *For the uncertainty sets defined in* (2)*, the least favorable distribution of problem* (1) *can be obtained by solving the following problem:*

$$\begin{aligned} \min_{p_1,\ldots,p_M \in \mathbb{R}_+^n} & \sum_{i=1}^{n} \max_{1 \le m \le M} p_m^i \\ \textit{subject to } & \sum_{i=1}^{n}\sum_{j=1}^{n} \gamma_m^{i,j}\, c(\xi_i, \xi_j) \le \vartheta_m \\ & \sum_{i=1}^{n} \gamma_m^{i,j} = \widehat{P}_m(\xi_j), \quad \sum_{j=1}^{n} \gamma_m^{i,j} = p_m^i, \\ & \forall 1 \le i, j \le n, 1 \le m \le M. \end{aligned} \tag{4}$$

The decision variable $\gamma_m \subset \mathbb{R}_+^{n \times n}$ can be viewed as a joint distribution on n empirical points with marginal distributions $\widehat{P}_m$ and P_m, represented by a vector $p_m \in \mathbb{R}_+^n$. The inequality constraint controls the Wasserstein distance between P_m and $\widehat{P}_m$.

Remark 1 *The maximization in* (4) *measures the margin between the maximum likelihood of ξ_i among all classes and the likelihood of the m-th class. Thus, the objective can be equivalently rewritten as the minimization of the total margin. When $M = 2$, the total margin reduces to the total variation distance.*

3.4 Model Estimation

The proposed learning method for robust node embeddings can be carried out in an end-to-end fashion. To propagate the error backward through the convex optimization problem described in (4) to the graph encoder, we adopt the idea of differentiable optimization (DO) [15, 16].

This approach enables us to differentiate through certain subclasses of convex optimization problems. In other words, the convex solver can be seen as a function that maps the data of the problem to its corresponding solutions, making it amenable to gradient-based learning techniques. Therefore, the learning objective can be jointly written as:

$$J(\theta; P_1^*, \ldots, P_M^*) := \sum_{i=1}^{n'} \max_{1 \le m \le M} P_m^*(\phi_\theta(i, \mathcal{G})), \tag{5}$$

where $P_m^*(\cdot)$ can be regarded as the output layer of our model, which takes the node embeddings ξ_i as input and returns their LFDs by solving (4) with DO.

We apply the mini-batch stochastic gradient descent as summarized in Algorithm 1. It is necessary that each mini-batch, which is provided to the convex solver, includes at least one training sample from every class to maintain the integrity of the optimization and its ability to generate valid solutions.

4 Experiments

4.1 Experimental Setup

In our experiments, we use three widely-used data sets, including Cora [38], Citeseer [39], and Pubmed [40]. These data sets consist of citation networks among $2,708$, $3,327$, and $19,717$ scientific publications, respectively [41]. In these networks, each node represents a text document, and its feature vector corresponds to a bag-of-words representation. We primarily focus on few-shot learning tasks where each data set contains M-class and K training samples per class. For each class, we randomly select K labeled nodes for a K-shot low-data setting.

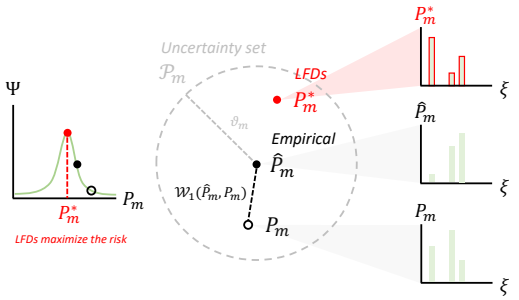


Figure 3: The minimax problem (1) aims to find the least favorable distributions (LFDs) by searching the optimal P_m in the uncertainty set $\mathcal{P}_m$ that maximizes the risk Ψ. The uncertainty set is defined by Wasserstein distance.

To assess the robustness of our method, we introduce random noise into these data sets in the following two ways: (1) We add Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma)$ to the node feature matrix X, where σ is set proportionally to the standard deviation of the bag-of-words representation of all nodes in each graph; (2) We randomly remove or add a certain percentage r of edges in the graph. We also test the performance of `DRGL` in K-shot low-data setting without adding noise to the original graphs. We repeat each test three times with three different seeds and calculate the average accuracy.

As our framework is designed to be GNN-agnostic, we chose to conduct experiments using Graph Convolutional Networks (`GCNs`) [1] and Graph Attention Networks (`GATs`) [2]. The purpose of adopting two basic GNNs is to test the generalizability of our `DRGL` framework. The performance of these basic GNN models aslo serves as a reference point for evaluating the robustness enhancements achieved by `DRGL` in our experiments. Specifically, for GCN, we employed two graph convolutional layers with hidden dimensions of 16 to learn 16-dimensional node representations. In the case of GAT, we utilized two graph attentional operator layers, with the first layer producing 8 attention heads, and ultimately obtaining 16-dimensional node embeddings.

The implementation of `DRGL` was carried out in `PyTorch`. Implementing `DRGL` for node classification tasks involved two stages. In the first stage, `DRGL` is used to enhance a pre-trained GNN encoder for robustness, following the algorithm detailed in Algorithm 1. In the second stage, we train a classification model based on either the learned embeddings ξ or the corresponding Least Favorable Distributions (LFDs) P^* of ξ. Kernel density estimation methods or k-nearest neighbors (k-NN) could also be employed to estimate the predicted probability based on the LFDs of the node embeddings. In our experiments, we evaluated our framework using two classifiers: a shallow (2-layer) neural network with Softmax output based on the learned embedding ξ and a weighted k-NN based on LFDs.

4.2 Robust GNN Baselines

To assess the effectiveness of `DRGL`, we conducted a comparative evaluation against baseline GNNs and state-of-the-art defence methods. An overview of these defense methods is provided below:

Table 1: Model performances with Gaussian noise in node features ($K = 5$).

Models	Cora (M = 7)		Citeseer (M = 6)		Pubmed (M = 3)	
σ	0.112	0.224	0.091	0.182	0.018	0.036
LP	47.70	47.70	21.73	21.73	28.90	28.90
GCN+ Softmax	66.17	52.43	37.83	30.35	63.43	57.83
GAT+ Softmax	<u>66.48</u>	<u>59.03</u>	**65.90**	<u>60.43</u>	63.73	58.70
ProGNN	63.53	53.60	43.57	43.90	OOM	OOM
RGCN	61.43	52.87	45.93	36.13	<u>64.10</u>	59.03
GCN-SVD	55.27	55.27	33.57	31.53	49.80	49.80
GCN_{DRGL}+ Softmax	66.20	54.40	39.80	34.90	63.20	<u>59.70</u>
GAT_{DRGL}+ Softmax	**67.83**	**59.40**	<u>65.60</u>	**61.27**	**64.57**	**59.77**

Table 2: Model performances with random edge removal ($K = 5$).

Models	Cora (M = 7)		Citeseer (M = 6)		Pubmed (M = 3)	
r	20%	50%	20%	50%	20%	50%
LP	41.67	32.10	17.83	11.60	26.53	23.53
GCN+ Softmax	<u>67.30</u>	50.83	39.25	35.70	<u>66.07</u>	<u>62.07</u>
GAT+ Softmax	67.28	<u>57.08</u>	<u>64.07</u>	<u>60.13</u>	64.80	61.96
ProGNN	63.70	56.60	46.73	46.73	OOM	OOM
RGCN	62.76	56.33	47.00	43.77	63.93	58.20
GCN-SVD	32.53	27.80	32.53	27.80	47.40	44.37
GCN_{DRGL}+ Softmax	66.70	52.15	40.15	41.80	**66.63**	**64.30**
GAT_{DRGL}+ Softmax	**69.02**	**58.88**	**65.93**	**61.20**	65.93	60.53

GCN-SVD [29]: This approach is a preprocessing method that suggests robustifying GCN with the low-rank approximation of the perturbed graph. It was originally proposed to defend against the *nettack* attack [42], which includes perturbations in both node features and edges;

RGCN [30]: RGCN adopts a strategy of modeling node representations as Gaussian distributions to mitigate the impact of adversarial structural attacks. Additionally, it employs an attention mechanism to penalize nodes with high variance, enhancing robustness;

ProGNN [24]: ProGNN jointly learns a structural graph and a robust GNN model guided by intrinsic graph properties such as low-rank and sparsity in the graph structure, as well as the tendency for adjacent nodes to exhibit similar features. Since ProGNN [30] was originally introduced as a defense mechanism against graph structural attacks and noises, it can be readily extended and tested under our low-data graph noise scenarios.

4.3 Main Results

We evaluated the node classification accuracy of various methods in different noisy scenarios, including random noise, random edge removal or addition, and the standard low-data case. The best-performing method is highlighted in bold in the result tables, while the second-best is underlined. We note that the ProGNN method encountered out-of-memory (OOM) issues when trained with Pubmed using the official implementation, despite running on a 24GB RTX 4090 GPU. In such cases, its results are marked as OOM in the tables. Based on the experimental results, we make the following observations regarding the performance of DRGL in different settings:

Random Gaussian noise in node features Table 1 presents the average classification accuracy, revealing that GCN and GAT models trained with DRGL consistently outperforms the standard models using the same classifiers and baseline robust graph methods across almost all scenarios. These findings underscore the effectiveness of DRGL in improving the embedding's robustness when confronted with noise in node features within low-data settings.

Random edge perturbations In the edge removal setting, Table 2 illustrates that both GCN and GAT exhibit significant improvements when enhanced with DRGL, particularly when a higher percentage of edges are removed. Their performance also consistently surpasses that of benchmark robust baselines, highlighting the effectiveness of DRGL in strengthening GNNs against missing edges. In the case of

Table 3: Model performances with random edge addition ($K = 5$).

Models	Cora (M = 7)		Citeseer (M = 6)		Pubmed (M = 3)	
r	20%	50%	20%	50%	20%	50%
LP	47.70	47.70	21.73	21.73	28.90	28.90
GCN+ Softmax	61.73	55.03	**47.67**	39.83	59.27	**60.07**
GAT+ Softmax	**68.93**	**63.00**	47.20	**44.27**	**63.70**	59.53
ProGNN	66.63	56.60	43.00	38.00	OOM	OOM
RGCN	57.23	50.97	42.63	39.20	61.76	56.00
GCN-SVD	47.80	44.37	32.06	31.30	51.90	50.07
GCN_{DRGL}+ Softmax	62.63	55.90	**47.67**	40.26	60.56	59.80
GAT_{DRGL}+ Softmax	67.60	59.13	47.30	43.97	62.30	59.03

Table 4: Model performances without graph noise.

Models	Cora (M = 7)		Citeseer (M = 6)		Pubmed (M = 3)	
	K = 5	K = 10	K = 5	K = 10	K = 5	K = 10
LP	47.70	52.80	21.73	28.33	28.90	38.63
GCN+ k-NN	36.20	66.50	28.50	52.47	45.57	68.27
GCN+ Softmax	63.47	70.67	45.37	57.83	67.10	71.27
ProGNN	**71.20**	**75.80**	45.10	54.53	67.70	OOM
RGCN	66.13	71.23	**50.00**	53.00	66.43	70.53
GCN-SVD	52.67	60.43	33.57	34.34	52.17	57.43
GCN_{DRGL}+ k-NN	44.60	68.23	34.07	53.30	51.10	68.67
GCN_{DRGL}+ Softmax	66.13	72.60	49.83	**59.33**	**67.83**	**72.20**

edge addition, while Table 3 demonstrates a more moderate impact of DRGL on GATs, it still improves performance of GCN in most settings and outperforms robust methods.

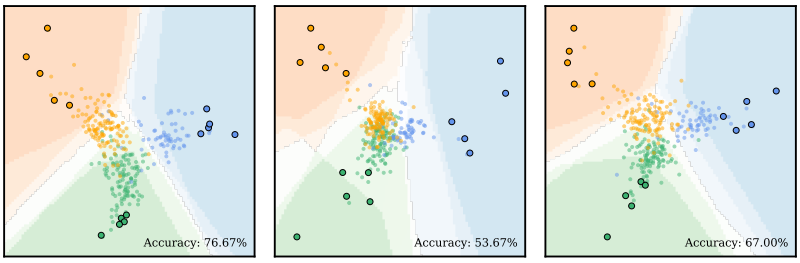
Standard few-shot setting In this setting, each node class is provided with a limited number of labeled samples, along with the original features and edges. The results presented in Table 4 demonstrate DRGL improves GCN in across all settings. This suggests that while DRGL is primarily designed to enhance robustness in noisy graph scenarios, it also enhances model performance in standard few-shot scenarios. We complemented the Softmax classifier with a k-NN classifier to further support this observation, where DRGL demonstrates similar improvement.

4.4 Learned Embeddings and Uncertainty

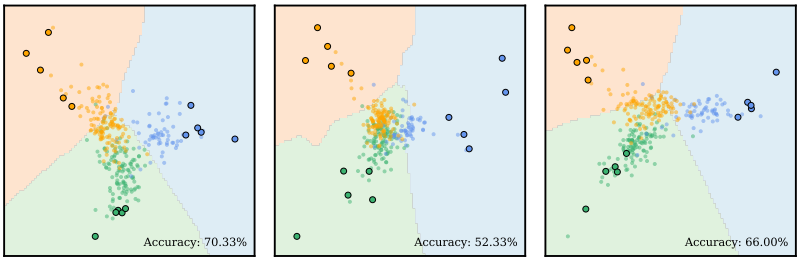
To gain a more intuitive understanding of the learned embedding space produced by DRGL, we conducted an additional ablation study. In this study, we set the output of the learned node embeddings to be two-dimensional and and visualize them as scatter plots. Figure 4 and Figure 5 give real examples using three classes of nodes from Cora data set [38]. In these figures, the training data points are represented by large dots, while the testing data points are represented by small dots. The color of the dots corresponds to their true categories, providing a visual reference. The color depth of the regions suggests the likelihood of a sample being classified into the predicted category.

Figure 4 presents a comparison between the learned node embeddings and those generated by the baseline GCN used for DRGL in various noisy scenarios. We can observe that the between-class distance demonstrates a slight increase in contrast to the vanilla GCN. Conversely, the intra-class distance shows a minor reduction, as denoted by the denser distribution of dots with matching colors. It is worth mentioning that such differences in distribution can become more prominent in higher dimensional embedding spaces. Although seemingly subtle, this shift in the distribution of embeddings due to DRGL plays a pivotal role in significantly improving the accuracy of the classification outcomes, indicated by the accuracy displayed in the lower right corner and results in Section 4.3.

Figure 5 showcases the visual representation of the LFDs generated by DRGL. The shades of grey in the visualization are obtained using kernel density estimation, where darker shades indicate a higher level of uncertainty between the different categories. These LFDs serve as a valuable tool for uncertainty quantification (UQ), enabling us to pinpoint the data points that are susceptible to the greatest impact in worst-case scenarios. Let $\tilde{p}^* = [\tilde{p}_1^*, \ldots, \tilde{p}_M^*]$ denote the predicted probabilities of a classifier built on learned LFDs of a graph. The predictive uncertainty can be expressed using

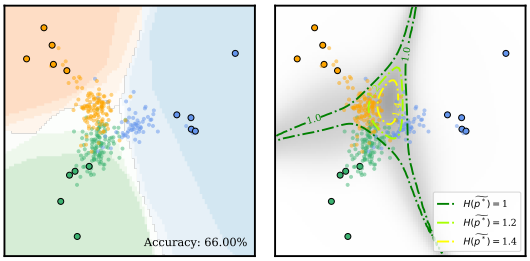


(a) Raw (DRGL) (b) 2σ noise (DRGL) (c) 20% edge (DRGL)



(d) Raw (GCN) (e) 2σ noise (GCN) (f) 20% edge (GCN)

Figure 4: The impact of noise on the learned feature spaces. (a) and (b) show the embeddings from graphs without noise; (b) and (e) show the embeddings when the graphs have nodal features with 2σ noise; and (c) and (f) present the representations from graphs where 20% of the edges are removed.



(a) DRGL + GCN (b) UQ of DRGL + GCN

Figure 5: The learned embeddings and the uncertainty.

entropy [43, 44, 45]:

$$H(\tilde{p}^*) = -\sum_{m=1}^{M} \tilde{p}_m^* \log \tilde{p}_m^* \tag{6}$$

The result highlights the improved capability of our approach in capturing and quantifying uncertainty compared with the vanilla GCN method.

5 Conclusion

To address the challenges posed by noisy graphs, we proposed a novel GNN-agnostic framework, DRGL, that enhances the robustness of node embeddings and predictive performance. Our proposed framework improves model robustness by accounting for the uncertainties arising from data noise within the graph, leading to substantial improvements over state-of-the-art baselines on different benchmark datasets in various few-shot noisy graph settings. Extending this framework to other graph representation models beyond GCN and GATs and citation network data sets will be left to our future work.

References

[1] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[2] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018.

[3] Jintang Li, Bingzhe Wu, Chengbin Hou, Guoji Fu, Yatao Bian, Liang Chen, Junzhou Huang, and Zibin Zheng. Recent advances in reliable deep graph learning: Inherent noise, distribution shift, and adversarial attack. *arXiv e-prints*, pages arXiv–2202, 2022.

[4] Yanqiao Zhu, Weizhi Xu, Jinghao Zhang, Yuanqi Du, Jieyu Zhang, Qiang Liu, Carl Yang, and Shu Wu. A survey on graph structure learning: Progress and opportunities. *arXiv e-prints*, pages arXiv–2103, 2021.

[5] Dongsheng Luo, Wei Cheng, Wenchao Yu, Bo Zong, Jingchao Ni, Haifeng Chen, and Xiang Zhang. Learning to drop: Robust graph neural network via topological denoising. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, WSDM '21, page 779–787, New York, NY, USA, 2021. Association for Computing Machinery.

[6] Kaize Ding, Elnaz Nouri, Guoqing Zheng, Huan Liu, and Ryen White. Toward robust graph semi-supervised learning against extreme data scarcity. *arXiv preprint arXiv:2208.12422*, 2022.

[7] John C Duchi and Hongseok Namkoong. Learning models with uniform performance via distributionally robust optimization. *The Annals of Statistics*, 49(3):1378–1406, 2021.

[8] Shai Shalev-Shwartz and Yonatan Wexler. Minimizing the maximal loss: How and why. In *International Conference on Machine Learning*, pages 793–801. PMLR, 2016.

[9] Rui Gao, Liyan Xie, Yao Xie, and Huan Xu. Robust hypothesis testing using wasserstein uncertainty sets. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[10] Shixiang Zhu, Liyan Xie, Minghe Zhang, Rui Gao, and Yao Xie. Distributionally robust weighted k-nearest neighbors. *Advances in Neural Information Processing Systems*, 35:29088–29100, 2022.

[11] Xiang Zhang, Yinfei Xu, Qinghe Liu, Zhicheng Liu, Jian Lu, and Qiao Wang. Robust graph learning under wasserstein uncertainty, 2021.

[12] Rizal Fathony, Ashkan Rezaei, Mohammad Ali Bashiri, Xinhua Zhang, and Brian Ziebart. Distributionally robust graphical models. *Advances in Neural Information Processing Systems*, 31, 2018.

[13] Matthew Staib and Stefanie Jegelka. Distributionally robust deep learning as a generalization of adversarial training. In *NIPS workshop on Machine Learning and Computer Security*, volume 3, page 4, 2017.

[14] Soroosh Shafieezadeh Abadeh, Peyman M Mohajerin Esfahani, and Daniel Kuhn. Distributionally robust logistic regression. *Advances in Neural Information Processing Systems*, 28, 2015.

[15] A. Agrawal, B. Amos, S. Barratt, S. Boyd, S. Diamond, and Z. Kolter. Differentiable convex optimization layers. In *Advances in Neural Information Processing Systems*, 2019.

[16] Brandon Amos and J. Zico Kolter. OptNet: Differentiable optimization as a layer in neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 136–145. PMLR, 2017.

[17] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs, 2014.

[18] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, 2016.

[19] Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. *ICLR (Poster)*, 2(3):4, 2019.

[20] Kaize Ding, Zhe Xu, Hanghang Tong, and Huan Liu. Data augmentation for deep graph learning: A survey. *ACM SIGKDD Explorations Newsletter*, 2022.

[21] Yu Chen, Lingfei Wu, and Mohammed Zaki. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 19314–19326. Curran Associates, Inc., 2020.

[22] Wei Jin, Yao Ma, Xiaorui Liu, Xianfeng Tang, Suhang Wang, and Jiliang Tang. Graph structure learning for robust graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 66–74, New York, NY, USA, 2020. Association for Computing Machinery.

[23] Wei Jin, Yaxing Li, Han Xu, Yiqi Wang, Shuiwang Ji, Charu Aggarwal, and Jiliang Tang. Adversarial attacks and defenses on graphs. *SIGKDD Explor. Newsl.*, 22(2):19–34, 1 2021.

[24] Wei Jin, Yao Ma, Xiaorui Liu, Xianfeng Tang, Suhang Wang, and Jiliang Tang. Graph structure learning for robust graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 66–74, 2020.

[25] L. Wang, W. Yu, W. Wang, W. Cheng, W. Zhang, H. Zha, X. He, and H. Chen. Learning robust representations with graph denoising policy network. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 1378–1383, Los Alamitos, CA, USA, 12 2019. IEEE Computer Society.

[26] Emanuele Rossi, Henry Kenlay, Maria I. Gorinova, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael Bronstein. On the unreasonable effectiveness of feature propagation in learning on graphs with missing node features, 2022.

[27] Bo Jiang and Ziyan Zhang. Incomplete graph representation and learning via partial graph neural networks. *arXiv preprint arXiv:2003.10130*, 2020.

[28] Cheng Zheng, Bo Zong, Wei Cheng, Dongjin Song, Jingchao Ni, Wenchao Yu, Haifeng Chen, and Wei Wang. Robust graph representation learning via neural sparsification. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11458–11468. PMLR, 7 2020.

[29] Negin Entezari, Saba A Al-Sayouri, Amirali Darvishzadeh, and Evangelos E Papalexakis. All you need is low (rank) defending against adversarial attacks on graphs. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 169–177, 2020.

[30] Dingyuan Zhu, Ziwei Zhang, Peng Cui, and Wenwu Zhu. Robust graph convolutional networks against adversarial attacks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1399–1407, 2019.

[31] Alireza Sadeghi, Meng Ma, Bingcong Li, and Georgios B. Giannakis. Distributionally robust semi-supervised learning over graphs, 2021.

[32] Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U. Rajendra Acharya, Vladimir Makarenkov, and Saeid Nahavandi. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 76:243–297, 2021.

[33] Edmon Begoli, Tanmoy Bhattacharya, and Dimitri Kusnezov. The need for uncertainty quantification in machine-assisted medical decision making. *Nature Machine Intelligence*, 1(1):20–23, 2019.

[34] Seongok Ryu, Yongchan Kwon, and Woo Youn Kim. A bayesian graph convolutional network for reliable prediction of molecular properties with uncertainty quantification. *Chemical science*, 10(36):8438–8446, 2019.

[35] Yao Zhang et al. Bayesian semi-supervised learning for uncertainty-calibrated prediction of molecular properties and active learning. *Chemical science*, 10(35):8154–8163, 2019.

[36] Qimai Li, Zhichao Han, and Xiao-ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), 4 2018.

[37] Erich Leo Lehmann, Joseph P Romano, and George Casella. *Testing statistical hypotheses*, volume 3. Springer, 1986.

[38] Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3:127–163, 2000.

[39] C Lee Giles, Kurt D Bollacker, and Steve Lawrence. Citeseer: An automatic citation indexing system. In *Proceedings of the third ACM conference on Digital libraries*, pages 89–98, 1998.

[40] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93, 9 2008.

[41] Zhilin Yang, William Cohen, and Ruslan Salakhudinov. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, pages 40–48. PMLR, 2016.

[42] Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. Adversarial attacks on neural networks for graph data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, page 2847–2856, New York, NY, USA, 2018. Association for Computing Machinery.

[43] C. E. Shannon. Communication theory of secrecy systems. *The Bell System Technical Journal*, 28(4):656–715, 1949.

[44] Alireza Namdari and Zhaojun Li. A review of entropy measures for uncertainty quantification of stochastic processes. *Advances in Mechanical Engineering*, 11(6):1687814019857350, 2019.

[45] Stephan Schwill. Entropy analysis of financial time series. *arXiv preprint arXiv:1807.09423*, 2018.