Uncertainty-Matching Graph Neural Networks to Defend Against Poisoning Attacks

Uday Shankar Shanthamallu[1]**, Jayaraman J. Thiagarajan**[2]**, Andreas Spanias**[1]
[1]Arizona State University
[2]Lawrence Livermore National Labs
ushantha@asu.edu, jjayaram@llnl.gov, spanias@asu.edu

Abstract

Graph Neural Networks (GNNs), a generalization of neural networks to graph-structured data, are often implemented using message passes between entities of a graph. While GNNs are effective for node classification, link prediction and graph classification, they are vulnerable to adversarial attacks, i.e., a small perturbation to the structure can lead to a non-trivial performance degradation. In this work, we propose Uncertainty Matching GNN (UM-GNN), that is aimed at improving the robustness of GNN models, particularly against poisoning attacks to the graph structure, by leveraging epistemic uncertainties from the message passing framework. More specifically, we propose to build a surrogate predictor that does not directly access the graph structure, but systematically extracts reliable knowledge from a standard GNN through a novel uncertainty-matching strategy. Interestingly, this uncoupling makes UM-GNN immune to evasion attacks by design, and achieves significantly improved robustness against poisoning attacks. Using empirical studies with standard benchmarks and a suite of global and target attacks, we demonstrate the effectiveness of UM-GNN, when compared to existing baselines including the state-of-the-art robust GCN.

Introduction

Representation learning methods, in particular deep learning, have produced state-of-the-art results in image analysis, language modeling and more recently with graph-structured data (Torng and Altman 2019). In particular, graph neural networks (GNNs) (Kipf and Welling 2017; Hamilton, Ying, and Leskovec 2017) have gained prominence due to their ability to effectively leverage the inherent structure to solve challenging tasks including node classification, link prediction and graph classification (Wu et al. 2020).

Despite their wide-spread use, GNNs are known to be vulnerable to a variety of adversarial attacks, similar to standard deep models. In other words, a small imperceptible perturbation intentionally designed in the graph structure can lead to a non-trivial performance degradation as seen in (Zügner, Akbarnejad, and Günnemann 2018). This limits their application to high-risk and safety critical domains. For example, the popular graph convolutional networks (GCN), which rely on aggregating message passes from a node's neighborhood, are not immune to poisoning attacks, wherein an attacker adds fictitious edges to the graph before the model is trained. Though there exists a vast literature on adversarial attacks on images (Goodfellow, Shlens, and Szegedy 2014; Szegedy et al. 2013) and their countermeasures (Ren et al. 2020; Chakraborty et al. 2018), designing attack strategies for graphs is a more recent topic of research. In general, designing graph attacks poses a number of challenges: (i) the adversarial search space is discrete; (ii) nodes in the graphs are non-i.i.d., and (iii) lack of effective metrics to measure structural perturbations. Following the progress in graph adversarial attacks, designing defense mechanisms or building robust variants of GNNs have become critical (Zhu et al. 2019).

In this paper, we propose a new approach UM-GNN aimed at improving the robustness of GNN models, particularly against challenging poisoning attacks to the graph structure. Our approach jointly trains a standard GNN model (implemented using GCN) and a surrogate predictor, which accesses only the features, using a novel uncertainty matching strategy. Through a systematic knowledge transfer from the GNN model, the surrogate demonstrates significantly improved robustness to challenging attacks. The key contributions of this work are summarized as follows:

- A novel architecture for semi-supervised learning, UM-GNN, that can be built upon any existing GNN model and is immune to evasion attacks by design;
- An uncertainty matching-based knowledge transfer strategy for achieving robustness to structural perturbations;
- Across a suite of global poisoning attacks, UM-GNN consistently outperforms existing methods including the recent Robust GCN (Zhu et al. 2019);
- UM-GNN achieves significantly lower misclassification rate ($> 50\%$ improvement) against targeted attacks.

Problem Setup

In this paper, we are interested in building graph neural networks that are robust to adversarial attacks on the graph structure. We represent an unweighted graph using the tuple $\text{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, v_2, \cdots, v_N\}$ denotes the set of nodes with cardinality $|\mathcal{V}| = N$, $\mathcal{E}$ denotes the set of edges and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. The edges in the graph may be alternately represented using an adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$. In addition, each node v_i may be endowed with a d-dimensional node attribute vector $\mathbf{x}_i \in \mathbb{R}^d$. We use the matrix $\mathbf{X} \in \mathbb{R}^{N \times d}$ to denote the features from all nodes. We focus on a transductive learning setting, where the goal is to perform node classification. In particular, we assume that we have access to labels for a subset of nodes $\mathcal{V}_L \subset \mathcal{V}$ and we need to predict the labels for the remaining nodes $(v \in \mathcal{V} \setminus \mathcal{V}_L)$ in G. Each node v_i is associated with a label $y_i \in \mathcal{Y} = [1, \cdots, K]$.

While a variety of approaches currently exist to solve this semi-supervised learning problem, we restrict our study to the recently successful solutions based on graph neural networks (GNNs). A recurring idea in many existing GNN models is to utilize a message passing mechanism to aggregate and transform features from the neighboring nodes. Implementing a GNN hence involves designing a message function P and an update function U, i.e.,

$$m_i = \sum_{j \in \mathcal{N}_i} \mathrm{P}(\mathbf{h}_i, \mathbf{h}_j, e_{ij}); \quad \mathbf{h}_i = \mathrm{U}(\mathbf{h}_i, m_i), \tag{1}$$

where $\mathcal{N}_i$ denotes the neighborhood of a node v_i and $\mathbf{h}_i$ its feature representation (in the input layer $\mathbf{h}_i = \mathbf{x}_i$). For example, in a standard graph convolutional network (GCN),

$$\mathbf{h}_i = \psi \left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{h}_j \mathbf{W} \right). \tag{2}$$

Here, the message computation is parameterized by α_{ij}, which can be a symmetric normalization constant (Kipf and Welling 2017) or a learnable attention weight (Veličković et al. 2018). The update function U is parameterized using the learnable weights $\mathbf{W}$ and applies a non-linearity ψ.

As discussed earlier, our goal is to defend against adversarial attacks on the graph structure. Formally, we assume that an adversary induces structural perturbations to the graph, i.e., $\hat{\mathrm{G}} = (\hat{\mathbf{A}}, \mathbf{X})$ such that $\|\mathbf{A} - \hat{\mathbf{A}}\|_0 \leq \Delta$. Here, Δ is used to ensure that the adversarial attack is imperceptible. Note that, one can optionally also consider the setting where the features $\mathbf{X}$ are also perturbed. While different classes of attacks currently exist (see Section), we focus on *poisoning* attacks, wherein the graph is corrupted even before the predictive model is trained. This is in contrast to *evasion* attacks, which assume that the model is trained on clean data and the perturbations are introduced at a later stage. We consider different popular poisoning attacks from the literature (see Section) and study the robustness of our newly proposed UM−GNN approach.

Proposed Approach

In this section, we present the proposed approach, Uncertainty Matching-GNN (UM−GNN), and provide details on the model training process.

While there exist very few GNN formulations for specifically defending against adversarial attacks, the recent robust GCN (RGCN) approach (Zhu et al. 2019) has been the most effective, when compared to standard GCN and GAT models. At its core, RGCN relies on using the *aleatoric* uncertainties in the graph structure to weight the neighborhood. Since there exists no *a* priori knowledge about the structural uncertainties, in practice, simple priors such as the normal distribution (zero mean, unit variance) are placed on the node features and propagated through the network to estimate uncertainties at the output of each layer. Finally, a modified message passing is utilized, wherein neighboring nodes with low feature variance are emphasized during message computation to produce robust features. Despite its empirical benefits, this approach suffers from three main challenges: (i) the choice of the prior is critical to its success; (ii) since the estimated uncertainties are not calibrated, the fidelity of the uncertainty estimates themselves can be low, thus leading to only marginal improvements over GCN in practice; (iii) the model (*epistemic*) uncertainties are not considered, which can impact the generalization of the inferred parameters to the test nodes. In order to alleviate these challenges, we propose UM−GNN, a new GNN formulation that uses an uncertainty matching-based knowledge transfer strategy for achieving robustness to graph perturbations. In contrast to RGCN, UM−GNN utilizes *epistemic* uncertainties from the GNN, and does not require any modifications to the message passing module. As we will show in our empirical studies, our approach provides significant improvements in defending against well-known poisoning attacks.

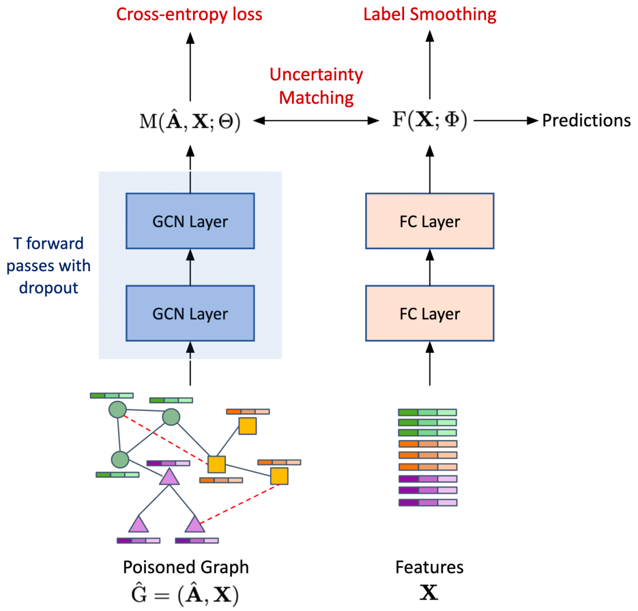

Cross-entropy loss
Label Smoothing
Uncertainty Matching
$\mathrm{M}(\hat{\mathbf{A}}, \mathbf{X}; \Theta)$
$\mathrm{F}(\mathbf{X}; \Phi)$
Predictions
T forward passes with dropout
GCN Layer
GCN Layer
FC Layer
FC Layer
Poisoned Graph
$\hat{\mathrm{G}} = (\hat{\mathbf{A}}, \mathbf{X})$
Features
$\mathbf{X}$

Figure 1: An illustration of the proposed UM−GNN, which constructs a surrogate model F and through an uncertainty matching strategy achieves robustness to poisoning attacks. After the model is trained, we use the surrogate model F to make predictions for the unlabeled nodes.

Figure 1 provides an illustration of UM−GNN, which jointly trains a GNN model $\mathrm{M}(\Theta)$ and a surrogate model $\mathrm{F}(\Phi)$ that is trained solely using the features $\mathbf{X}$ without any knowledge of the graph structure. Here Θ and Φ denote the learnable model parameters. Since we expect the graph structure to be potentially corrupted (though severity or type of corruption is unknown), the predictions from the GNN model could be unreliable due to the presence of noisy edges. We reformulate the problem of making M robust into systematically transferring the most reliable knowledge to the surrogate F, so that F can make robust predictions. When compared to existing regularization strategies such as GraphMix (Verma et al. 2019), we neither use the (solely) feature-based model F to regularize the training of M nor are the weights shared between the networks. Instead, we build a surrogate predictor that selectively extracts the most reliable information from the "non-robust" M with the hope of being more robust to the noise in the graph structure. Interestingly, by design, the model F does not rely on the graph structure and hence is oblivious to evasion attacks. As showed in the figure, after training, we only use the surrogate F to obtain the predictions for unlabeled nodes.

Bayesian Uncertainty Estimation

Quantifying the prediction uncertainties in the graph neural network M is at the core of UM-GNN. We propose to utilize Bayesian Neural Networks (BNNs) (Blundell et al. 2015), in particular its scalable variant based on Monte Carlo dropout (Srivastava et al. 2014). In general, dropout variational inference is used to estimate the epistemic uncertainties as follows: A deep network is trained with dropout and even at test time the dropout is used to generate samples from the approximate posterior through Monte Carlo sampling. Interestingly, it was showed in (Gal and Ghahramani 2016) that the dropout inference minimizes the KL divergence between the approximated distribution and the posterior of a deep Gaussian process. The final prediction can then be obtained by marginalizing over the posterior, using Monte Carlo integration. In our formulation, the node classification task is transductive in nature and does not require test-time inferencing. Hence, we propose to leverage the prediction uncertainties in the training loop itself. More specifically, we obtain the prediction for each node v_i as

$$p(y_i = k; \mathbf{x}_i, \mathbf{A}) = \texttt{Softmax}\left(\frac{1}{T}\sum_{t=1}^{T} \mathrm{M}(\mathbf{x}_i, \mathbf{A}; \tilde{\Theta})\right).$$

Here we make T forward passes for $\mathbf{x}_i$ with different masked weights $\tilde{\Theta}$ (using dropout inference) and compute the final prediction using a sample average. Note, we assume that the predictive model produces logits, i.e., no activation in the final prediction layer and hence compute the Softmax of the average predictions. We then use the *entropy* of the resulting prediction $p(y_i = k; \mathbf{x}_i, \mathbf{A})$ as an estimate of the model uncertainty for node v_i.

$$\begin{aligned}\texttt{Unc}(v_i) &= \texttt{Entropy}\Big(p(y_i = k; \mathbf{x}_i, \mathbf{A})\Big) \\ &= -\sum_{k=1}^{K} p(y_i = k)\log p(y_i = k) \qquad (3)\end{aligned}$$

Algorithm

We now present the algorithm to train an UM-GNN model given a poisoned graph $\hat{\mathrm{G}} = (\hat{\mathbf{A}}, \mathbf{X})$. As described earlier, our architecture is composed of a graph neural network $\mathrm{M}(\Theta)$ and a surrogate model $\mathrm{F}(\Phi)$ that takes only the features $\mathbf{X}$ as input. While we implement M using graph convolution layers as defined in eqn.(2), it can be replaced using any other message passing strategy, e.g, graph attention layers (Veličković et al. 2018). Given that all datasets we consider in our study contain vector-values defined at the nodes, we implement F as a fully connected network. The optimization problem used to solve for the parameters Θ and Φ is given below:

$$\underset{\Theta,\Phi}{\text{minimize}}\ \mathcal{L}_{ce} + \lambda_m \mathcal{L}_m + \lambda_s \mathcal{L}_s. \qquad (4)$$

Here, the first term $\mathcal{L}_{ce}$ corresponds to the standard cross entropy loss over the set of labeled nodes computed using the predictions from the GNN model M.

The second term $\mathcal{L}_m$ is used to align the predictions between the surrogate and GNN models so that the resulting classifiers are consistent. Directly distilling knowledge from the GNN model enables F to actually make meaningful predictions for the nodes, even without accessing the underlying graph structure. However, using a poisoned graph to build M can lead to predictions with high uncertainties. Such noisy examples may lead to unreliable gradients, thus making the knowledge transfer unstable. Hence, we propose to attenuate the influence of samples with high prediction uncertainty. We refer to this process as uncertainty matching and implement it using the KL divergence. However, this can be readily replaced using any general divergence or the Wasserstein metric. Mathematically,

$$\mathcal{L}_m = \sum_{i=1}^{N} \beta_i \texttt{KLDiv}(\mathrm{M}(\mathbf{x}_i, \mathbf{A}; \Theta), \mathrm{F}(\mathbf{x}_i; \Phi)), \qquad (5)$$

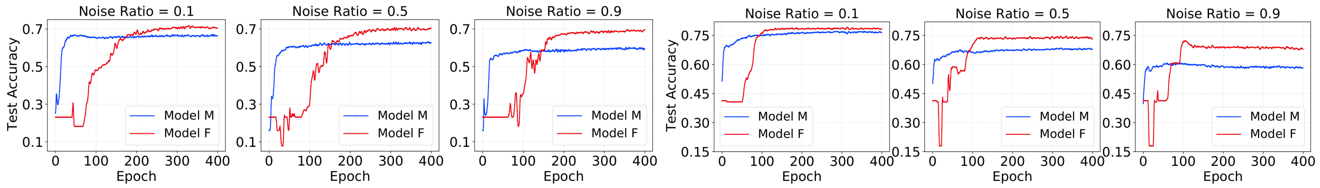
where the weight β_is are computed as

$$\beta_i = \frac{\exp(-\alpha_i)}{\sum_j \exp(-\alpha_j)};\ \text{where } \alpha_i = \log\frac{1}{1 + \texttt{Unc}(v_i)}. \qquad (6)$$

When the prediction uncertainty for a sample is low, it is given higher attention during matching. Note that, this loss is evaluated using both labeled and unlabeled nodes, since it does not need access to the true labels. Finally, the third term $\mathcal{L}_s$ corresponds to a label smoothing regularization that attempts to match the predictions from F to an uniform distribution (KL divergence). This is included to safeguard the surrogate model from being misguided by the graph network, when the latter's confidences are not well-calibrated due to the poisoned graph. In all our experiments, we set $\lambda_m = 0.3$ and $\lambda_s = 0.001$. Figure 2 illustrates the behavior of UM-GNN for two different datasets under varying levels of poisoning. As the severity of the corruption increases, the surrogate model achieves significantly higher test performance when compared to the graph-based model M. In cases where no explicit node attributes are available, F may be implemented as a GNN and the uncertainty matching strategy will still be applicable and this is part of our future work.

Poisoning Attacks Used for Evaluation

While there exists a broad class of adversarial attacks that are designed to be applied during the testing phase of the



(a) Citeseer with random attack

(b) Pubmed with DICE attack

Figure 2: Illustration of the behavior of UM−GNN for two datasets under varying types and levels of poisoning attacks. In each case, we show the test accuracy curves across the training epochs from both the GNN and surrogate models. As the noise severity increases, the surrogate model F demonstrates improved robustness.

model, we focus on the more challenging poisoning attacks. Poisoning attacks are intended to disrupt the model training itself by injecting carefully crafted corruptions to the training data. In particular, it is well known that they are highly effective at degrading the performance of GNNs. More importantly, existing robust modeling variants such as RGCN provide only marginal improvements over the standard GNN models, when presented with poisoned graphs. Hence, we evaluate the proposed UM−GNN using several widely-adopted poisoning attacks. Here, we briefly describe those attacks and provide our implementation details.

Random Attack This is a purely black-box attack, where the attacker has no knowledge of ground truth labels or the model information. More specifically, in this attack, new edges are randomly introduced between two nodes that were not previously connected. Though being simple, this attack is known to be effective, particularly at higher noise ratios and sparse graphs. For our experiments, we varied the ratio of noisy edges between 10% and 100% of the total number of edges in the original graph.

DICE Attack (Waniek et al. 2018) This is a gray-box attack where the attacker has information about the node labels but not the model parameters. This attack uses a modularity-based heuristic to Disconnect Internally (nodes from the same community) and Connect Externally (DICE) (nodes from different communities). For a given budget, an attacker randomly deletes edges that connect nodes from the same class; and adds edges between randomly chosen node pairs of samples from different classes. Similar to the random attack, we varied the perturbation ratio between 10% and 100% of the total number of existing edges.

Meta-Gradient Attack (Mettack) (Zügner and Günnemann 2019) This is a more challenging gray-box attack where the attacker utilizes the graph structure and labels to construct a surrogate model, which is then utilized to generate the attacks. More specifically, Mettack formulates a bi-level optimization problem of maximizing the classification error on the labeled nodes after optimizing the model parameters on the poisoned graph. In other words, the graph structure is treated as the hyper-parameter to optimize, and this is solved using standard meta-learning strategies. Since the surrogate model is also designed based on GCNs (similar architectures as our predictive model) and trained with the entire graph (transductive setting), this gray-box attack is very powerful in practice. Hence we used lower noise ratios for our experiments, i.e., between 1% to 10% of the total existing edges, when compared to Random and DICE attacks.

Dataset	# Nodes	# Edges	# Features	# Classes
Cora	2708	5278	1433	7
Citeseer	3327	4614	3703	6
Pubmed	19717	44325	500	3

Table 1: Summary of the three benchmark citation datasets used in our experments.

Projected-Gradient Attack (PGD) (Xu et al. 2019) PGD is a first-order topology attack that attempts to determine the minimum edge perturbations in the global structure of the graph, such that the generalization can be maximally affected. Since PGD cannot access the true model parameters, we use a surrogate GNN model to generate the attacks. Similar to Mettack, we varied the perturbation ratio between 1% and 10% in this case as well.

Fast Gradient Attack (FGA) (Chen et al. 2018) FGAs are created based on gradient information in GNNs and they belong to the category of targeted attacks. The goal of a targeted attack is to mislead the model into classifying a target node incorrectly. In FGA, the attacker adds an edge between node pairs that are characterized by largest absolute difference in their gradients. We choose FGA to show the superior performance of UM−GNN even against targeted attacks.

The implementations for Mettack, PGD and FGA were based on the publicly available DeepRobust (Jin et al. 2020) library. Due to the lack of computationally efficient implementations, we could not generate these attacks on large-scale graphs such as Pubmed.

Empirical Evaluation

In this section, we evaluate the robustness of UM−GNN against the graph poisoning methods discussed in the previous section. As mentioned in Section , non-targeted poisoning attacks are far more challenging and pose a more realistic threat to graph-based models.

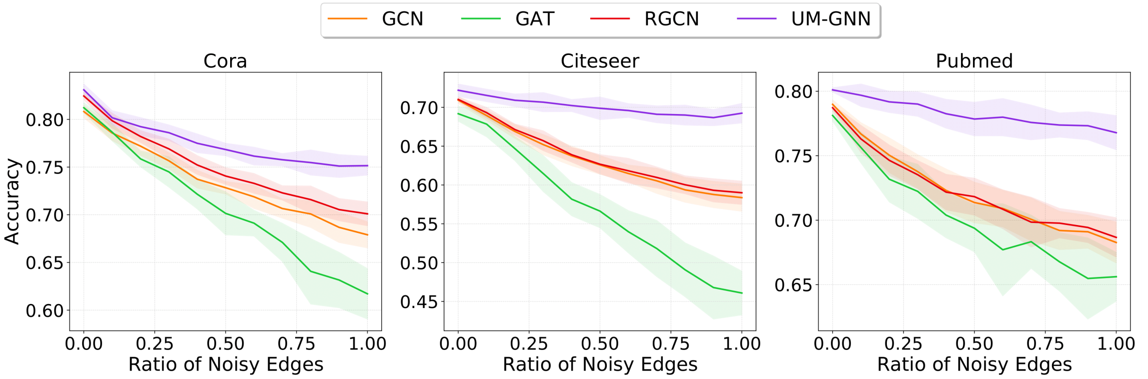


Figure 3: *Random attack*: UM−GNN achieves robustness to random attacks, providing over $5-10\%$ improvements in the test accuracy, even when the noise ratio is 1.0.

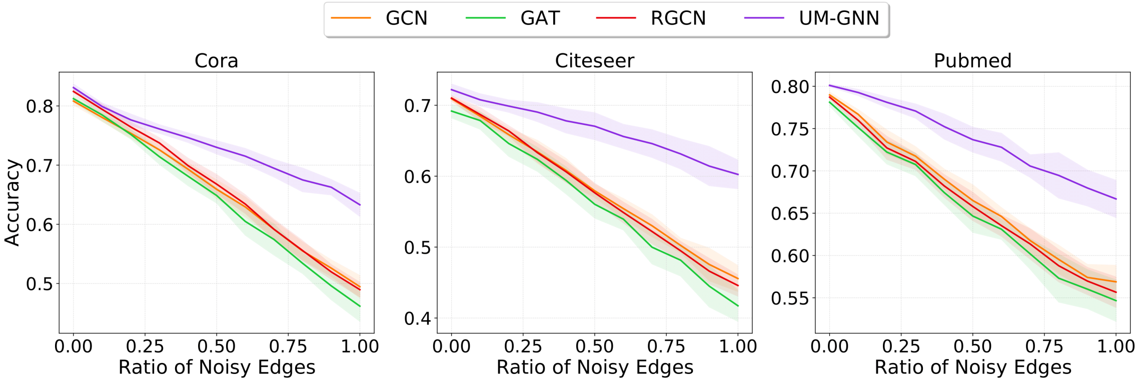


Figure 4: *DICE attack*: For all datasets, UM−GNN is consistently more robust in this challenging scenario, where the attacker both adds and deletes edges. The performance improvement with UM−GNN is as high as $\approx 15\%$ (Citeseer).

Datasets We consider three benchmark citation networks extensively used in similar studies: Cora, Citeseer, and Pubmed (Sen et al. 2008). The documents are represented by nodes, and citations among the documents are encoded as undirected edges. We follow the typical transductive node classification setup (Kipf and Welling 2017; Veličković et al. 2018), while using the standard train, test, and validation splits for our experiments (see Table 1).

Baselines We compare the proposed approach with three important baseline GNN models, which adopt different message passing formalisms and have been successfully used in semi-supervised node classification tasks. Note that the performance of a feature-only classifier (MLP) which ignores the graph structure produces trivial performances with the following accuracies: 55.1% for Cora, 46.5% for Citeseer, and 71.4% for Pubmed.
GCN: We use the GCN model, proposed by Kipf & Welling, based on the message passing formulation in eqn. (2).
GAT (Veličković et al. 2018): This model uses a multi-head attention mechanism to learn the hidden representations for each node through a weighted aggregation of features in a closed neighborhood where the weights are trainable.
RGCN (Zhu et al. 2019): This is a recently proposed approach that explicitly enhances the robustness of GCNs. RGCN models node features as distributions as opposed to deterministic vectors in GCN and GAT models. It employs a variance-based attention mechanism to attenuate the influence of neighbors with large variance (potentially corrupted). Following (Zhu et al. 2019), we set hidden dimensions at 16 and assume a diagonal covariance for each node.

For all baselines, we set the number of layers (2 layers) and other hyper-parameter settings as specified in their original papers. We set the number of hidden neurons to 16 for both GCN and GAT baselines. In addition, we set the number of attention heads to 8 for GAT. We implemented all the baselines and the proposed approach using the Pytorch Deep Graph Library (version 0.5.1) (Wang et al. 2019). In our implementation of UM−GNN, the GNN model M was designed as a $2-$layer GCN similar to the baseline and the surrogate F was a $3-$layer FCN with configuration $32-16-K$, where K is the total number of classes.

Results

We evaluated the classification accuracy on the test nodes for the datasets against each of the attacks, under varying levels of perturbation. For random and DICE attacks, we varied the

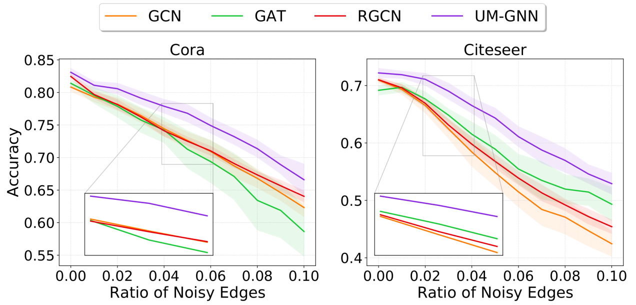


Figure 5: *Mettack* - This gray-box attack is known to be highly effective at causing performance degradation in GNNs. However, UM−GNN consistently provides $3-5\%$ improvements in the test accuracy over the baselines.

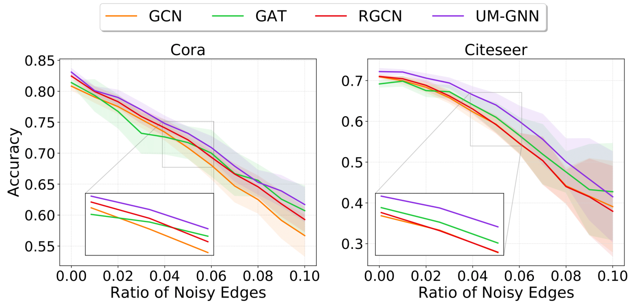


Figure 6: *PGD Attack* - This is comparatively very severe, since it uses gradients from a GCN model (same architecture as M). While the accuracy improvements are still non-trivial $(1\% - 2\%)$, the more interesting observation is the reduced variance of UM−GNN across trials.

ratio of noisy edges to clean edges between 0.1 and 1. Since Mettack and PGD attacks are more powerful, we used noise ratios in the range $(0.01, 0.1)$. For all the 4 global attacks, we repeated the experiment for 20 random trials (different corruption) for each noise ratio, and report the expected accuracies along with their standard deviations.

(i) Random Attack: The results for random attacks for all three datasets are shown in Figure 3. As discussed earlier, RGCN provides only a marginal improvement over the vanilla GCN and GAT. However, UM−GNN consistently outperforms the baselines by a large margin even when the ratio of noisy edges to clean edges is high. In addition, UM−GNN has the least variance in performance compared to the baselines. In comparison, GAT appears to be the most sensitive to random structural perturbations and its low performance strongly corroborates with the findings in (Zhu et al. 2019).

(ii) DICE Attack: In this challenging attack, where the attacker can both delete and add edges, all baseline methods suffer from severe performance degradation, when compared to random attacks. Surprisingly, UM−GNN is significantly more robust and achieves performance improvements as high as $\approx$ 15% (Figure 4, Citeseer, noise ratio = 1.0). This clearly evidences the ability of UM−GNN to infer the true modular structure, even when the graph is poisoned.

(iii) Mettack Attack: Since mettack uses a surrogate model and its parameters to generate attacks, it is one of the more challenging attacks to defend. Nevertheless, UM−GNN consistently outperforms all the baselines by a good margin, as illustrated in Figure 5. Interestingly, under this attack, both GCN and RGCN perform poorly when compared to the GAT model. However, the large variance makes GAT unreliable in practice, particularly when the attack is severe.

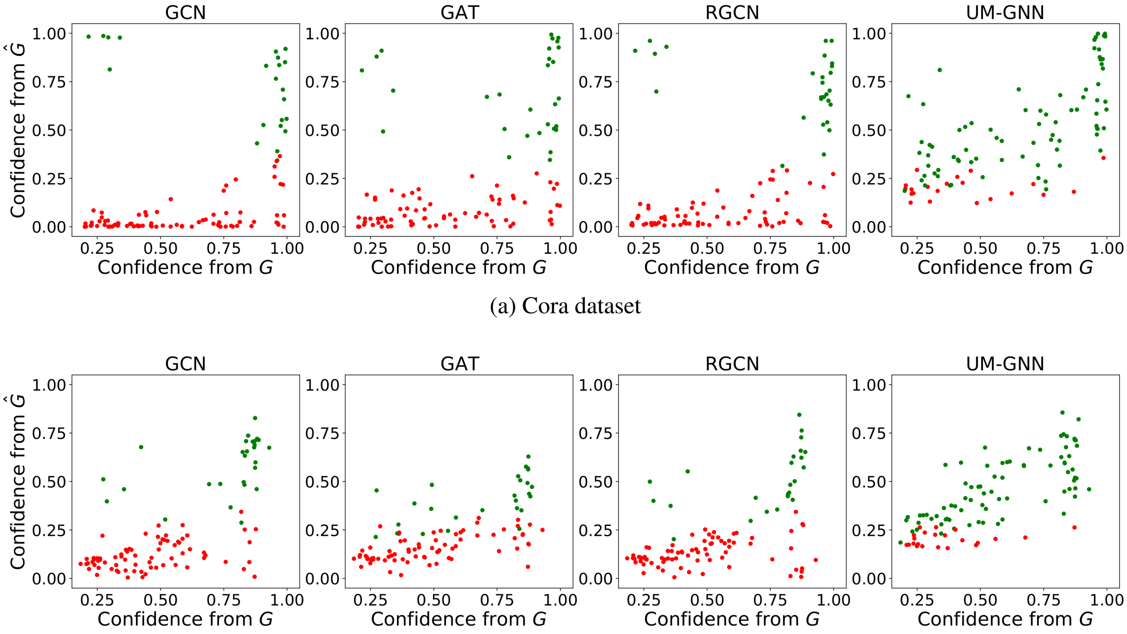
(iv) PGD Attack: This is comparatively the most severe, since the GCN model used to generate the attack has the same architecture as our model M, thus in actuality making it a white-box attack. From Figure 6, we observe $1\% - 2\%$ improvements in mean performance over the baselines. More importantly, the lower variance of UM−GNN across trials makes it a suitable choice for practical scenraios.

(v) FGA Attack: For this targeted attack, we selected 100 test nodes with correct predictions in a baseline GCN as our targets. Out of the 100 target nodes, 25 nodes were those with the highest margin of classification, 25 nodes were those with the lowest margin, and the remaining 50 were chosen randomly. Further, we set the number of perturbations allowed on each target node to be equal to its degree (so that it is imperceptible). The FGA attack was generated for each target node independently, and we checked if the targeted attack was defended successfully or not, i.e., whether the targeted node was classified correctly using the poisoned graph. The overall misclassification rates for the different models are shown in Table 2. We find that UM−GNN provides dramatic improvements in defending against FGA attacks, through its systematic knowledge transfer between the GNN M and the surrogate F. In Figure 7, we plot the prediction probabilities for the true class (indicates a model's confidence) for all target nodes obtained using the original and poisoned graphs G and $\hat{\mathrm{G}}$ respectively. As it can be observed, UM−GNN improves the confidences considerably for all samples, while the baseline methods demonstrate vulnerability to FGA.

Related Work

Semi-supervised learning based on graph neural networks (GNNs) enables representation learning using both the graph structure and node features (Wu et al. 2020). While GNNs based on spectral convolutional approaches (Bruna et al. 2013; Defferrard, Bresson, and Vandergheynst 2016; Kipf and Welling 2017) have been widely adopted, there also exists models that implement convolutions directly using spatial neighborhoods (Duvenaud et al. 2015; Atwood and Towsley 2016; Hamilton, Ying, and Leskovec 2017). The vulnerability of GNNs to adversarial attacks was first studied in (Zügner, Akbarnejad, and Günnemann 2018). Since then, several graph adversarial attacks have been proposed (Jin et al. 2020; Sun et al. 2018). Adversarial attacks on graphs can be broadly categorized as follows:

(i) *Attacker knowledge*: based on the level of access an attacker has to the model internals, namely white-box (Xu et al. 2019; Wu et al. 2019), gray-box (Zügner, Akbarnejad, and Günnemann 2018; Zügner and Günnemann 2019)



(a) Cora dataset

(b) Citeseer dataset

Figure 7: Results from FGA attacks on two benchmark datasets - On the x-axis, we plot the prediction probabilities for the true class obtained using GCN on the clean graph G. On the y-axis, we show the prediction probabilities obtained after the targeted attack. Note, for each method, we show the misclassified nodes in red and the correct predictions in green.

Model	Cora	Citeseer
GCN	0.78	0.73
GAT	0.71	0.74
RGCN	0.73	0.76
UM−GNN	**0.21**	**0.23**

Table 2: Misclassification rates from 100 target nodes with FGA attack. A lower value implies improved robustness.

and black-box attacks (Bojchevski and Günnemann 2019).
(ii) *Attacker capability*: based on whether the attacker perturbs the graph before (Liu et al. 2019) or after (Dai et al. 2018) the model is trained.
(iii) *Attack strategy*: based on whether the attacker corrupts the graph structure or node features. While structural perturbations can be induced by deleting, adding or re-wiring edges; new nodes could also be injected into the graph (Shanthamallu, Thiagarajan, and Spanias 2020).
(iv)*Attacker's goal*: based on whether the attack is aimed at degrading the model's overall performance (Waniek et al. 2018) or targeting specific nodes either directly or indirectly for their misclassification (Chen et al. 2018).

Graph Adversarial Defense As graph adversarial attacks continue to be studied, efforts aimed at designing suitable defense strategies have emerged recently. For example, Feng *et al.* adapted the conventional adversarial training approach to the case of graphs in order to make GNNs more robust (Goodfellow, Shlens, and Szegedy 2014; Feng et al. 2019). On the other hand, methods that rely on graph preprocessing have also been proposed – for example, in (Wu et al. 2019), edges with low Jaccard similarity between the constituent nodes were removed prior to training a GNN. Similarly, in (Jin et al. 2019), explicit graph smoothing was performed by training on a family of graphs to defend against evasion attacks. Entezari *et al.* obtained a low rank approximation of the given graph and showed that it can defend against specific types of graph attack (Zügner, Akbarnejad, and Günnemann 2018). Recently, Zhu *et al.* (Zhu et al. 2019) introduced a robust variant of GCN based on a variance-weighted attention mechanism, and showed it to be effective against different types of attacks.

Conclusions

In this work, we presented UM−GNN an uncertainty matching-based architecture to explicitly enhance the robustness of GNN models. UM−GNN utilizes epistemic uncertainties from a standard GNN M and does not require any modifications to the message passing module. Consequently, our architecture is agnostic to the choice of GNN to implement M. By design, the surrogate model F does not directly access the graph structure and hence is immune to evasion-style attacks. Our empirical studies clearly evidenced the effectiveness of UM−GNN in defending against several graph poisoning attacks, thereby outperforming existing baselines. Furthermore, we showed dramatic improvements on defense against targeted attacks (FGA). Future work includes studying the performance bounds of UM−GNN and developing extensions for inductive learning settings.

Acknowledgements

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. This work was also partially supported by the ASU SenSIP Center.

References

Atwood, J.; and Towsley, D. 2016. Diffusion-convolutional neural networks. In *Advances in neural information processing systems*, 1993–2001.

Blundell, C.; Cornebise, J.; Kavukcuoglu, K.; and Wierstra, D. 2015. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424* .

Bojchevski, A.; and Günnemann, S. 2019. Adversarial attacks on node embeddings via graph poisoning. In *International Conference on Machine Learning*, 695–704. PMLR.

Bruna, J.; Zaremba, W.; Szlam, A.; and LeCun, Y. 2013. Spectral networks and locally connected networks on graphs. *arXiv:1312.6203* .

Chakraborty, A.; Alam, M.; Dey, V.; Chattopadhyay, A.; and Mukhopadhyay, D. 2018. Adversarial attacks and defences: A survey. *arXiv:1810.00069* .

Chen, J.; Wu, Y.; Xu, X.; Chen, Y.; Zheng, H.; and Xuan, Q. 2018. Fast gradient attack on network embedding. *arXiv:1809.02797* .

Dai, H.; Li, H.; Tian, T.; Huang, X.; Wang, L.; Zhu, J.; and Song, L. 2018. Adversarial attack on graph structured data. *arXiv:1806.02371* .

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, 3844–3852.

Duvenaud, D. K.; Maclaurin, D.; Iparraguirre, J.; Bombarell, R.; Hirzel, T.; Aspuru-Guzik, A.; and Adams, R. P. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in neural information processing systems*, 2224–2232.

Feng, F.; He, X.; Tang, J.; and Chua, T.-S. 2019. Graph adversarial training: Dynamically regularizing based on graph structure. *IEEE Transactions on Knowledge and Data Engineering* .

Gal, Y.; and Ghahramani, Z. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, 1050–1059.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *arXiv:1412.6572* .

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, 1024–1034.

Jin, M.; Chang, H.; Zhu, W.; and Sojoudi, S. 2019. Power up! robust graph convolutional network against evasion attacks based on graph powering. *arXiv:1905.10029* .

Jin, W.; Li, Y.; Xu, H.; Wang, Y.; and Tang, J. 2020. Adversarial Attacks and Defenses on Graphs: A Review and Empirical Study. *arXiv:2003.00653* .

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR*.

Liu, X.; Si, S.; Zhu, X.; Li, Y.; and Hsieh, C. 2019. A unified framework for data poisoning attack to graph-based semi-supervised learning. *arXiv:1910.14147* .

Ren, K.; Zheng, T.; Qin, Z.; and Liu, X. 2020. Adversarial attacks and defenses in deep learning. *Engineering* .

Sen, P.; Namata, G.; Bilgic, M.; Getoor, L.; Galligher, B.; and Eliassi-Rad, T. 2008. Collective classification in network data. *AI magazine* 29(3): 93–93.

Shanthamallu, U. S.; Thiagarajan, J. J.; and Spanias, A. 2020. A Regularized Attention Mechanism for Graph Attention Networks. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 3372–3376.

Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15(1): 1929–1958.

Sun, L.; Dou, Y.; Yang, C.; Wang, J.; Yu, P. S.; and Li, B. 2018. Adversarial attack and defense on graph data: A survey. *arXiv:1812.10528* .

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2013. Intriguing properties of neural networks. *arXiv pre. arXiv:1312.6199* .

Torng, W.; and Altman, R. B. 2019. Graph convolutional neural networks for predicting drug-target interactions. *Journal of Chemical Information and Modeling* 59(10): 4131–4149.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. *International Conference on Learning Representations* URL https://openreview.net/forum?id=rJXMpikCZ.

Verma, V.; Qu, M.; Lamb, A.; Bengio, Y.; Kannala, J.; and Tang, J. 2019. Graphmix: Regularized training of graph neural networks for semi-supervised learning. *arXiv:1909.11715* .

Wang, M.; Yu, L.; Zheng, D.; Gan, Q.; Gai, Y.; Ye, Z.; Li, M.; Zhou, J.; Huang, Q.; Ma, C.; Huang, Z.; Guo, Q.; Zhang, H.; Lin, H.; Zhao, J.; Li, J.; Smola, A. J.; and Zhang, Z. 2019. Deep Graph Library: Towards Efficient and Scalable Deep Learning on Graphs. *ICLR Workshop on Representation Learning on Graphs and Manifolds* URL https://arxiv.org/abs/1909.01315.

Waniek, M.; Michalak, T. P.; Wooldridge, M. J.; and Rahwan, T. 2018. Hiding individuals and communities in a social network. *Nature Human Behaviour* 2(2): 139–147.

Wu, H.; Wang, C.; Tyshetskiy, Y.; Docherty, A.; Lu, K.; and Zhu, L. 2019. Adversarial examples on graph data: Deep insights into attack and defense. *arXiv:1903.01610* .

Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Philip, S. Y. 2020. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems* .

Xu, K.; Chen, H.; Liu, S.; Chen, P.-Y.; Weng, T.-W.; Hong, M.; and Lin, X. 2019. Topology attack and defense for graph neural networks: An optimization perspective. *arXiv:1906.04214* .

Zhu, D.; Zhang, Z.; Cui, P.; and Zhu, W. 2019. Robust graph convolutional networks against adversarial attacks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1399–1407.

Zügner, D.; Akbarnejad, A.; and Günnemann, S. 2018. Adversarial attacks on neural networks for graph data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2847–2856.

Zügner, D.; and Günnemann, S. 2019. Adversarial attacks on graph neural networks via meta learning. *arXiv:1902.08412* .