Boosting GNN's Generalization via Graph Data Augmentation

Siddhant Patil, Shivam Patel, Dipti Ranjan Sahu, Nischal Reddy Chandra

Outline

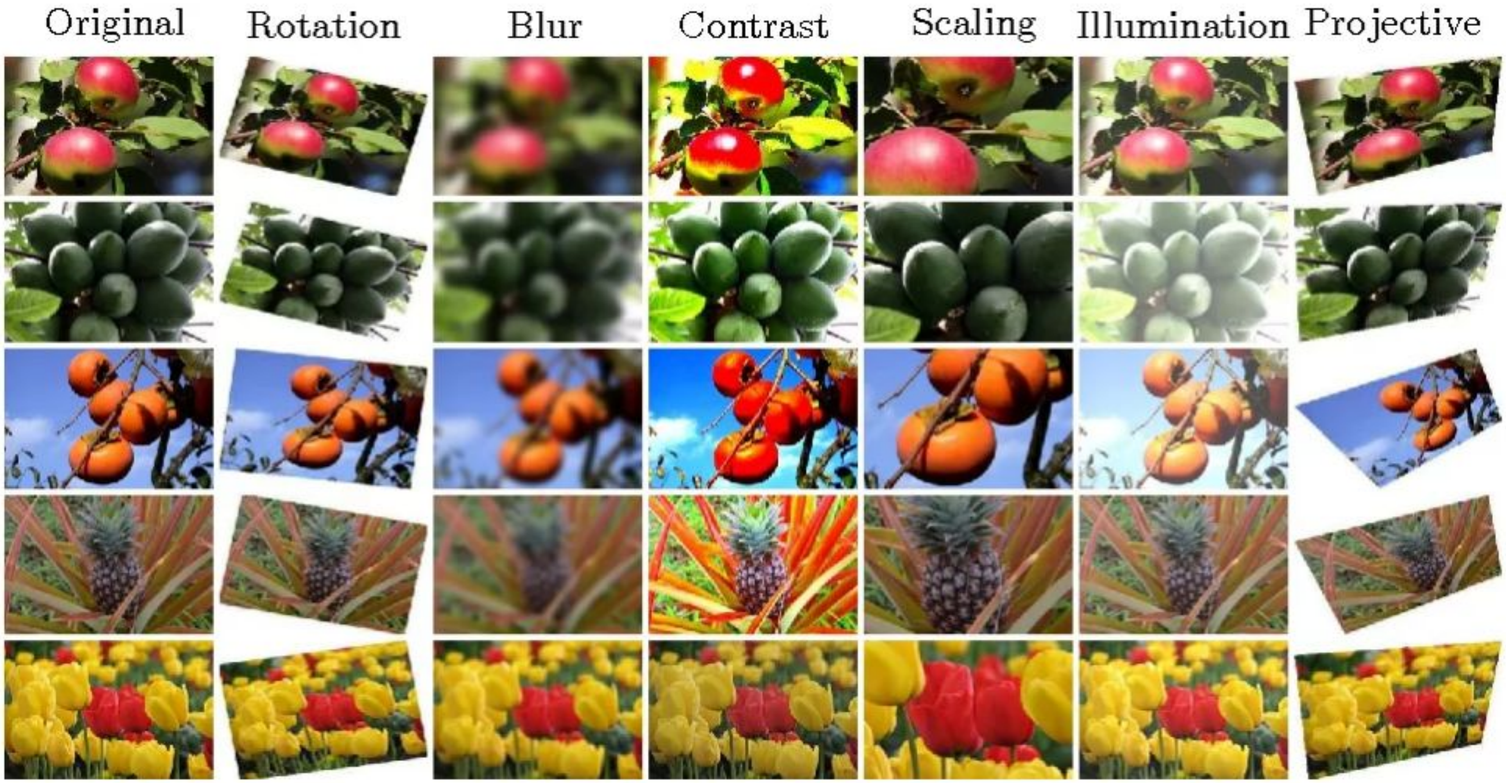
- Introduction
- Datasets and Evaluation Metrics
- Proposed Methods
- Experimental Results
- Conclusion
- Future Work

Problem Statement

Data Augmentation

- Data augmentation involves creating new plausible variations from the original training data
- Improves the generalization of machine learning models on images and natural language
- Reduces cost of data acquisition and labelling, enables rare prediction events

Image Augmentation



Text Augmentation

	Sentence
Original	The quick brown fox jumps over the lazy dog
Synonym (PPDB)	The quick brown fox climbs over the lazy dog
Word Embeddings (word2vec)	The easy brown fox jumps over the lazy dog
Contextual Word Embeddings (BERT)	Little quick brown fox jumps over the lazy dog
PPDB + word2vec + BERT	Little easy brown fox climbs over the lazy dog

Augmentation in Graphs

- Graphs are complex and have non-Euclidean structure, which limits possible manipulation operations
- The nodes in graphs are described by some (high-dimensional) characteristics while an image pixel is just scalars
- Sampling, edge manipulation and Mixup are amongst the techniques we use to augment
- Downstream task we evaluate our model on is *node classification*

Datasets & Evaluation Metrics

Datasets

	Cora	CiteSeer	PubMed	BlogCatalog
# Nodes	2,708	3,327	19,717	10,312
# Edges	5,278	4,552	44,338	333,983
# Features	1,433	3,703	500	64
# Classes	7	6	3	38
# Training Nodes	140	120	60	519
# Validation Nodes	500	500	500	1,039
# Test Nodes	1,000	1,000	1,000	3,638

Statistics and experimental setup for the four evaluation datasets

Datasets

	Cora	CiteSeer	PubMed	BlogCatalog
# Nodes	2,708	3,327	19,717	10,312
# Edges	5,278	4,552	44,338	333,983
# Features	1,433	3,703	500	64
# Classes	7	6	3	38
# Training Nodes	140	120	60	2578
# Validation Nodes	500	500	500	2578
# Test Nodes	1,000	1,000	1,000	5156

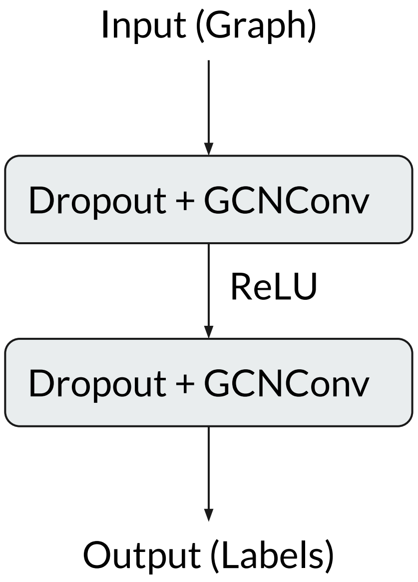
Statistics and experimental setup for the four evaluation datasets

Evaluation Metrics

- **Accuracy** - Number of nodes that are predicted correctly
- Other Metrics - AUC, F1 score
- Loss Function: **Cross Entropy** - Measure of the difference between two probability distributions for a given random variable

Proposed Methods
&
Experimental Results

Classifier - Baseline



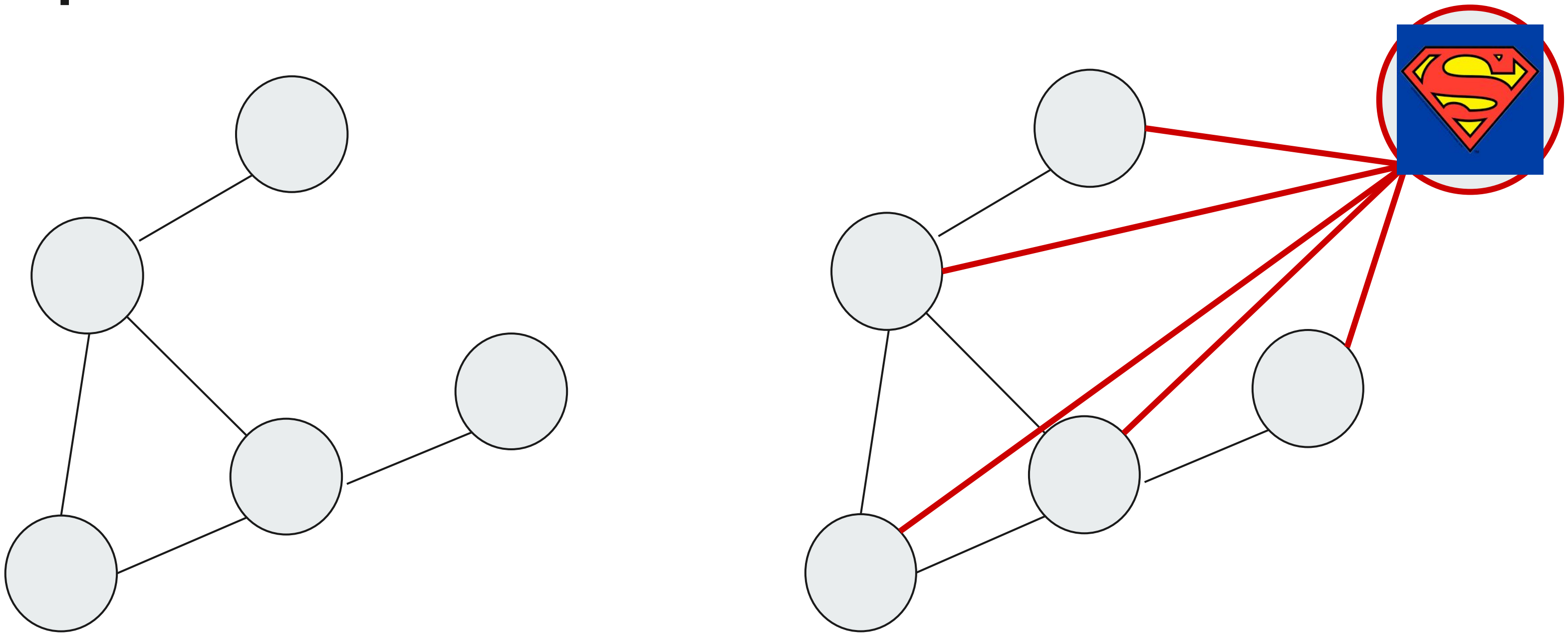

```python
class GCN(torch.nn.Module):
    def __init__(self, in_channels, hidden_channels, out_channels):
        super().__init__()
        self.conv1 = GCNConv(in_channels, hidden_channels, cached=True,
                             normalize=not args['use_gdc'])
        self.conv2 = GCNConv(hidden_channels, out_channels, cached=True,
                             normalize=not args['use_gdc'])

    def forward(self, x, edge_index, edge_weight=None):
        x = F.dropout(x, p=args['dropout'], training=self.training)
        x = self.conv1(x, edge_index, edge_weight).relu()
        x = F.dropout(x, p=args['dropout'], training=self.training)
        x = self.conv2(x, edge_index, edge_weight)
        return x
```

Classifier - Baseline

	Cora	CiteSeer	PubMed
Baseline	0.8170	0.7108	0.7916

Super-Nodes



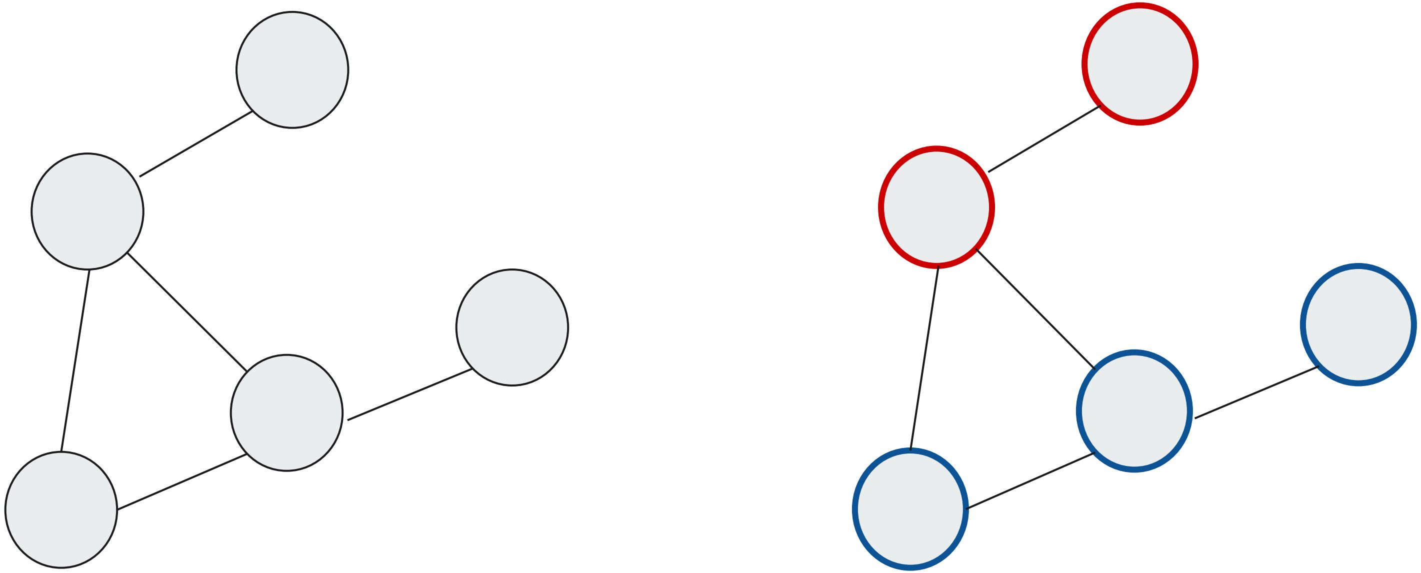
Super-Nodes

- Allow information to travel long distances during the propagation phase
- “Neural Message Passing for Quantum Chemistry” introduces such super-nodes, however
  - Special edge
  - Different Feature Dimension
  - Separate Weights

Super-Nodes

	Cora	**CiteSeer**	**PubMed**
Baseline	0.8170	0.7108	**0.7916**
Super-Node	**0.8264**	**0.7154**	0.7894

Clustering + Super-Nodes

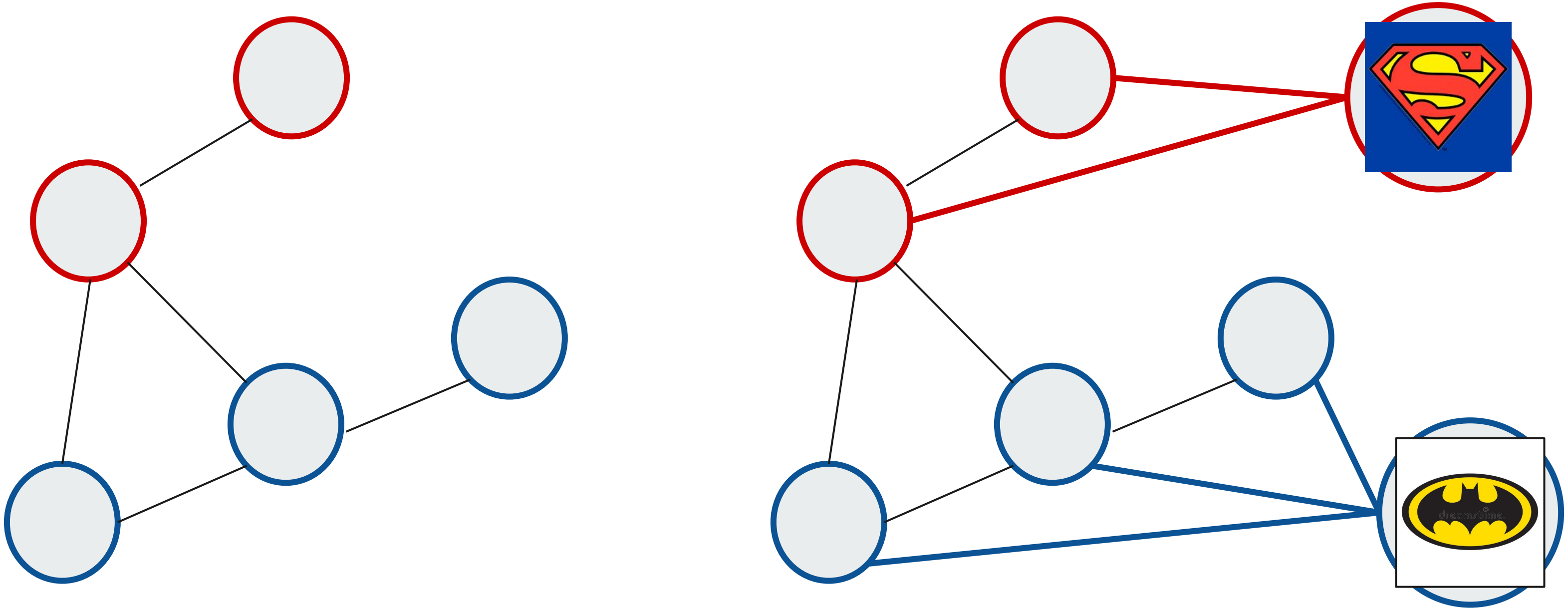


Clustering + Super-Nodes

Structure-based Clustering - METIS

Feature-based Clustering - K-Means

Clustering + Super-Nodes

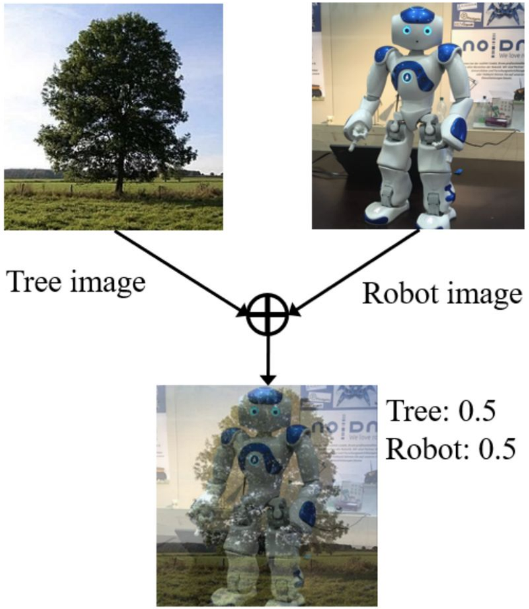


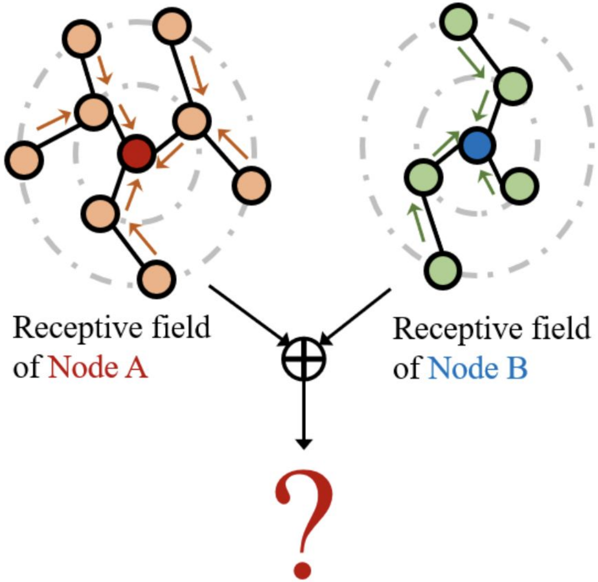
Clustering + Super-Nodes

	Cora	**CiteSeer**	**PubMed**
Baseline	0.8170	0.7108	0.7916
Super-Node	0.8264	0.7154	0.7894
K-Means	0.819 (5)	**0.7234** (100)	**0.8016** (3)
METIS	**0.8282** (20)	0.7196 (20)	0.801 (5)

Note: (n) means n clusters

Mixup for Node Classification





Mixup for Node Classification

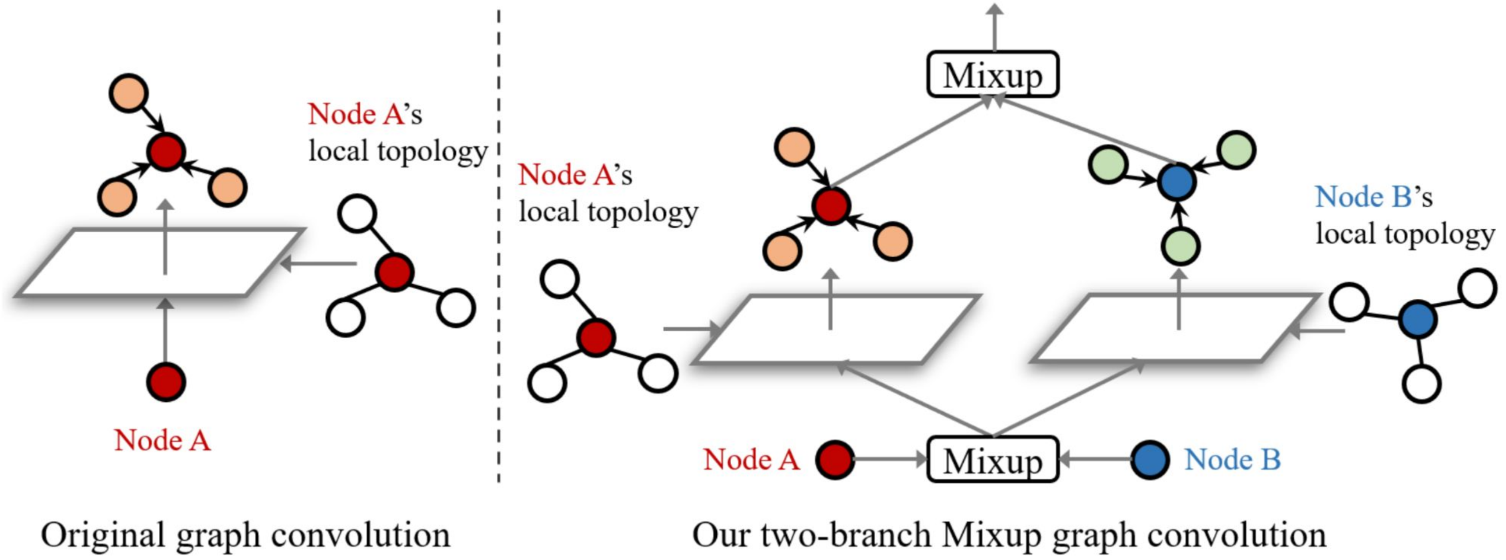
$$\tilde{x} = \lambda x_i + (1 - \lambda) x_j,$$

$$\tilde{y} = \lambda y_i + (1 - \lambda) y_j,$$

Data-Augmentation and Mixup - Cora

DataSet	Mixup	Data_Aug_Type	Test Accuracy
Cora	Yes	Single Node	0.811
	Yes	K-Means, 20	0.808
	Yes	Metis, 20	**0.818**
	Yes	N/A	0.799
	No	N/A	0.78

Data-Augmentation and Mixup - Cora



Data-Augmentation and Mixup - Cora

DataSet	Mixup	Data_Aug_Type	Test Accuracy
Cora	Yes	Single Node	0.811
	Yes	K-Means, 20	0.808
	Yes	Metis, 20	**0.818**
	Yes	N/A	0.799
	No	N/A	0.78

Data-Augmentation and Mixup - CiteSeer

DataSet	Mixup	Data_Aug_Type	Test Accuracy
CiteSeer	Yes	Single Node	0.661
	Yes	K-Means, 20	**0.704**
	Yes	Metis, 20	0.507
	Yes	N/A	0.69
	No	N/A	0.655

Data-Augmentation and Mixup - PubMed

DataSet	Mixup	Data_Aug_Type	Test Accuracy
PubMed	Yes	Single Node	0.768
	Yes	K-Means, 20	0.764
	Yes	Metis, 20	**0.775**
	Yes	N/A	0.757
	No	N/A	0.76

Class Imbalance

- Though GCNs have achieved state-of-the-art performance on node classification, they only deal where node samples for different classes are balanced.
- Many real world scenarios where some classes may have very less representation in the population. Example: BlogCatalog, Twitter - fake account detection
- Existing methods like interpolation between node and its nearest neighbor may not be directly applicable. The synthesized node attributes may be of low quality. Also, majority classes may dominate the loss function.

Imbalanced Data Classification

Feature extractor

$$\mathbf{h}_v^1 = \sigma(\mathbf{W}^1 \cdot CONCAT(\mathbf{F}[v,:], \mathbf{F} \cdot \mathbf{A}[:,v]))$$

Synthetic Node Generation

$$nn(v) = \underset{u}{\operatorname{argmin}} \|\mathbf{h}_u^1 - \mathbf{h}_v^1\|, \quad \text{s.t.} \quad \mathbf{Y}_u = \mathbf{Y}_v$$

$$\mathbf{h}_{v'}^1 = (1-\delta) \cdot \mathbf{h}_v^1 + \delta \cdot \mathbf{h}_{nn(v)}^1$$

Edge Generator

$$\mathbf{E}_{v,u} = softmax(\sigma(\mathbf{h}_v^1 \cdot \mathbf{S} \cdot \mathbf{h}_u^1))$$

$$\mathcal{L}_{edge} = \|\mathbf{E} - \mathbf{A}\|_F^2$$

GNN Classifier

- Same as baseline

Optimization Objective: $\min_{\theta,\phi,\varphi} \mathcal{L}_{node} + \lambda \cdot \mathcal{L}_{edge}$

Class Imbalance - Experimentation

- Synthetic class imbalance generation for the Cora dataset.
  - Majority classes: 20 samples per class for training
  - Minority classes: 3 randomly selected classes with 20 x imbalance_ratio samples per class for training
- BlogCatalog is already imbalanced.
- Used GCN, GCN + Oversampling, GCN + Reweight and GCN + GraphSMOTE to train a classifier on Cora and BlogCatalog Dataset.
- Calculate the accuracy, AUC-ROC and F1 Score.

Results (Class Imbalance)

Model \ Data	Cora*			BlogCatalog		
	Accuracy	AUC-ROC	F1 Score	Accuracy	AUC-ROC	F1 Score
GCN	63.64%	0.8858	0.6434	13.68%	0.4973	0.0065
GCN + Oversampling	66.75%	0.9096	0.6774	13.74%	0.5019	0.0077
GCN + Reweight	66.49%	0.9122	0.6731	13.78%	0.5015	0.0076
GCN + GraphSMOTE	**67.68%**	**0.9172**	**0.6804**	**14.48%**	0.5013	**0.0081**

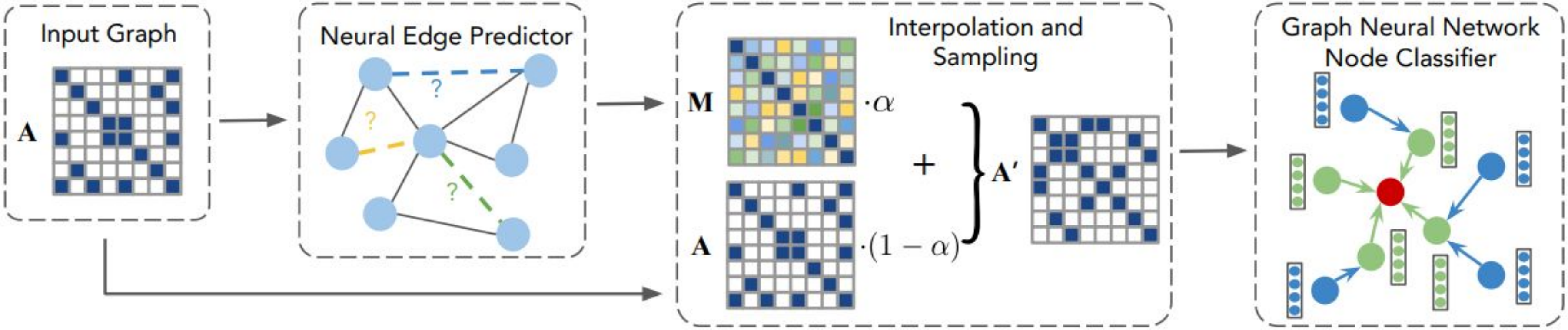
* - synthetically imbalanced data

Graph Augmentation by Modifying edges

- Variational Graph Auto-Encoder to calculate edge probabilities for all possible and existing edges
- Followed by normal GCN architecture for node classification

$$\mathbf{M} = \sigma\left(\mathbf{Z}\mathbf{Z}^T\right), \text{ where } \mathbf{Z} = f_{GCL}^{(1)}\left(\mathbf{A}, f_{GCL}^{(0)}\left(\mathbf{A}, \mathbf{X}\right)\right)$$

Graph Augmentation by Modifying edges



Results (Edge Manipulation)

	Cora	CiteSeer
Baseline	0.8170	0.7108
GAUG-M	**0.8343**	**0.7228**
GAUG-O	0.8318	0.7223

* GAUG-M uses the modified graph during inference
** GAUG-O uses the original graph during inference

Conclusion & Future Work

Conclusion

- Proposed different data augmentation frameworks to improve node classification performance. Also, plan to ensemble to all these methods into one
- GCN + GraphSMOTE works good for imbalanced data, while edge manipulation and clustering+mixup works good for other datasets
- Each model showed an absolute improvement of around **1-3% over the baseline** GCN model and is comparable to the state-of-the-art solution

Future Work

- Combine edge manipulation technique with data augmentation and mix-up (if possible); Proper organization and documentation of the github repo
- Other clustering techniques for clustering + super-nodes
- Try different model architectures (GAT, SplineCNN) to improve performance
- Implement proposed methods on new dataset such as Twitter - fake account detection dataset.
- **Project Extension**- Edge Prediction using Reinforcement Learning and rewire nodes which are 2-hop neighbors (more or less, preserves graph properties) based on the RL update rules

Thank you!

References

Tianxiang Zhao, Xiang Zhang, Suhang Wang. 2021. GraphSMOTE: Imbalanced Node Classification on Graphs with Graph Neural Networks. In Proceedings of the Fourteenth ACM International Conference on Web Search and Data Mining (WSDM ’21), March 8–12, 2021, Virtual Event, Israel. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3437963.3441720

Shorten, Connor, and Taghi M. Khoshgoftaar. "A survey on image data augmentation for deep learning." Journal of big data 6.1 (2019): 1-48.

Feng, Steven Y., et al. "A survey of data augmentation approaches for nlp." arXiv preprint arXiv:2105.03075 (2021).

Zhao, Tong, et al. "Graph Data Augmentation for Graph Machine Learning: A Survey." arXiv preprint arXiv:2202.08871 (2022)

Ding, Kaize and Xu, Zhe and Tong, Hanghang and Liu, Huan, “Data Augmentation for Deep Graph Learning: A Survey”, arXiv (2022)