Node Classification on PubMed using Graph Convolutional Networks

Emil Thorsbjerg (thorsemi)[1]

[1]**Graph Neural Networks(NI-GNN)**

ABSTRACT

In this project, I investigate the use of Graph Convolutional Networks (GCN) for node classification on the PubMed citation network. I train a two-layer GCN to predict the subject category of scientific publications. The model achieves a test accuracy of approximately 79%, and I visualize the learned embeddings, classification performance, and examples of predictions. The findings confirm that GCNs can capture graph structure effectively even with limited node features.

Keywords: Graph Neural Networks, GCN, PubMed, Node Classification, Machine Learning

1. INTRODUCTION

Graph-structured data is common in many domains such as citation networks, social networks, and molecular structures. This project focuses on applying Graph Convolutional Networks (GCNs) to classify scientific articles in the PubMed citation graph based on their subject category. The project builds upon techniques covered in the Graph Neural Networks (NI-GNN) course and uses one dataset and a complete GCN implementation and evaluation pipeline.

2. DATASET

I use the PubMed dataset from the Planetoid benchmark suite. Each node represents a scientific article with 500 sparse binary features, and edges represent citation links. The classification task is to predict one of three subject areas.

3. TASK

The task of this project is to train a Graph Convolutional Network (GCN) on the PubMed citation dataset to classify scientific articles into three different categories. The goal is to evaluate how well the model performs by validating it through various techniques such as accuracy metrics, a confusion matrix, and a t-SNE visualization of the learned node embeddings.

4. METHOD

I implement a two-layer Graph Convolutional Network using the PyTorch Geometric library. The model consists of:

- **Input:** 500-dimensional node features
- **Hidden layer:** 16 units with ReLU and dropout
- **Output layer:** 3 units with log-softmax

The loss function is negative log likelihood, and the optimizer is Adam with a learning rate of 0.01 and weight decay of 5e-4. The model is trained for 200 epochs. During training, the loss fluctuates between 0.07 and 0.15, which suggests stable convergence despite local variation.

5. RESULTS AND VISUALIZATIONS

5.1 Accuracy

The model achieved the following results on the PubMed dataset:

- **Training Accuracy:** 100%
- **Validation Accuracy:** 80.6%
- **Test Accuracy:** 79.0%

This indicates that the model learns effectively and generalizes reasonably well.

5.2 Training Loss Curve

The model's loss over the course of 200 training epochs is shown below. The curve fluctuates slightly, but overall remains low and supports the model's ability to generalize.

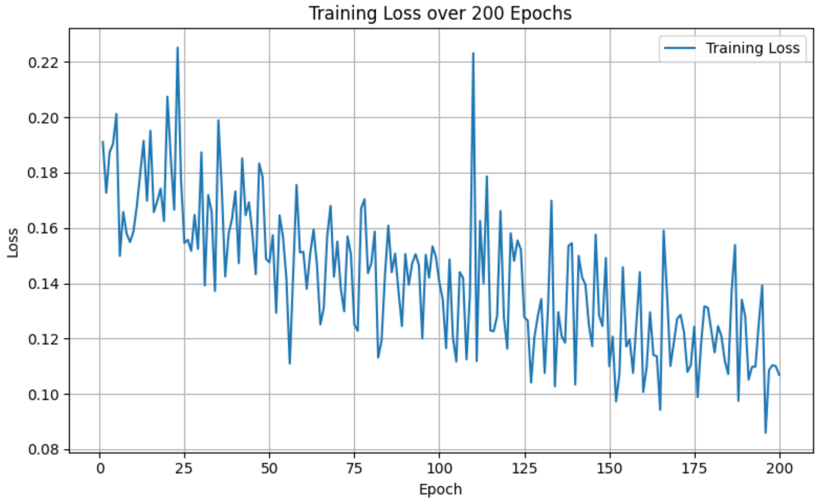

Figure 1. Training loss over 200 epochs.

The training loss over 200 epochs shows a generally decreasing trend, although with some fluctuations. This is expected due to the stochastic nature of gradient descent. The overall downward slope indicates that the model is learning to minimize the negative log-likelihood loss. The loss stabilizes at a relatively low level after around 150 epochs, which suggests that the model has reached a reasonable convergence point.

5.3 Embedding Visualization (t-SNE)

To gain insight into the learned node representations, I visualize the hidden embeddings from the first GCN layer using t-SNE. The resulting plot shows class separation in the embedding space.

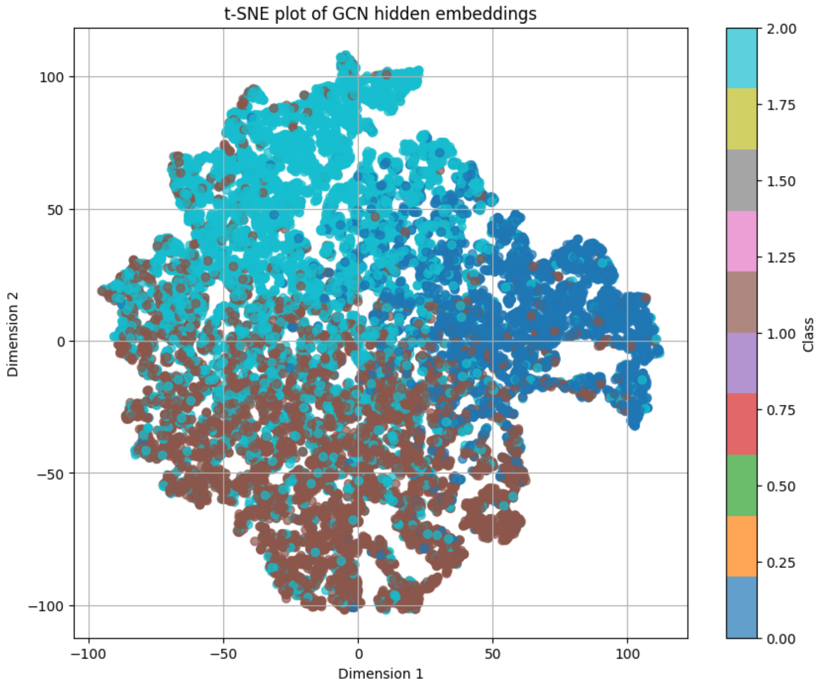

Figure 2. t-SNE plot of GCN node embeddings.

The t-SNE visualization of the hidden embeddings from the first GCN layer shows that nodes from different classes are somewhat clustered together. Although there is some overlap between the classes—especially between class 0 and 2—the embedding space reveals a clear underlying structure. This suggests that the GCN is learning meaningful representations based on the graph structure and feature information.

5.4 Confusion Matrix

The confusion matrix confirms that most misclassifications occur when class 0 or 2 is predicted as class 1. This aligns with the cluster overlaps seen in the t-SNE plot.

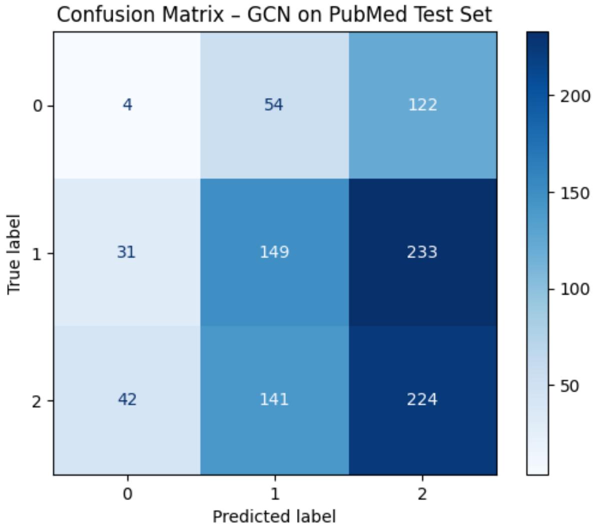

Figure 3. Confusion Matrix – GCN on PubMed Test Set. Most misclassifications occur when class 0 or 2 are predicted as class 1.

The confusion matrix shows that most misclassifications occur when nodes from class 0 or class 2 are predicted as class 1. This pattern suggests that class 1 acts as a ”central” class in the embedding space, absorbing borderline cases. The diagonal dominance of the matrix, especially for class 2, confirms that the model performs better on some classes than others, highlighting an imbalance in either feature separability or training signal.

5.5 Prediction Examples

Below are examples of both correct and incorrect predictions. Most errors occur when class 0 or 2 are misclassified as class 1.

Node ID	True Label	Predicted Label	Correct
572	2	1	No
271	2	1	No
138	2	1	No
102	1	2	No
994	0	2	No
362	2	2	Yes
584	1	1	Yes
224	1	1	Yes
425	1	1	Yes
260	1	1	Yes

Table 1. Example predictions: five incorrect (top) and five correct (bottom).

The table of prediction examples illustrates concrete cases of correct and incorrect classifications. Most incorrect predictions are consistent with the confusion matrix findings—nodes from class 0 or 2 are often misclassified as class 1. This supports the hypothesis that class 1 is less well separated. In contrast, correct predictions demonstrate that the model can confidently classify several nodes, particularly from class 1, when the features are well-aligned with the class prototypes.

6. CONCLUSION

The GCN model performs well on the PubMed citation graph, learning meaningful node representations despite sparse features. Visualizations such as t-SNE, confusion matrix, and prediction examples highlight the model’s strengths and limitations. The project confirms the effectiveness of GCNs for real-world graph-based classification tasks and provides insight into their behavior on citation networks.

7. FUTURE WORK

Although the model achieves relatively good test accuracy, there are clear signs of overfitting, especially after epoch 140, where training accuracy reaches 100% while validation accuracy stagnates. In future work, we would address this by implementing techniques such as early stopping, increased dropout, or stronger weight decay. Additionally, the model struggles to classify class 0 correctly due to strong class imbalance. Exploring class weighting or oversampling could help mitigate this issue. Finally, experimenting with alternative architectures like Graph Attention Networks (GAT) or GraphSAGE could improve the model’s ability to distinguish overlapping classes.

ACKNOWLEDGMENTS

This project was completed as part of the Graph Neural Networks course (NI-GNN) at CTU in Prague.