Graph-MLP

Ke Wan

December 27, 2023

Background

- **Effective Graph Learning with GNNs**:
  - Message passing-based Graph Neural Networks (GNNs) are highly effective for graph learning tasks.
- **Significance in Large-Scale Graphs**:
  - The application of GNNs is particularly significant in the context of large-scale graphs, offering practical benefits in various domains.
- **Computational Intensity on Large-Scale Graphs**:
  - GNNs require substantial computational resources when applied to large-scale graph datasets, posing challenges in efficiency and scalability.

High Level Abstract

- **Innovative Node Representation Learning**:
  - This study introduces a novel node representation learning method, leveraging the power of Multilayer Perceptrons (MLP) and contrastive learning.
- **Successful Node Prediction**:
  - Demonstrates the ability to accomplish node prediction tasks efficiently, even under conditions unfavorable for information passing.

Framework

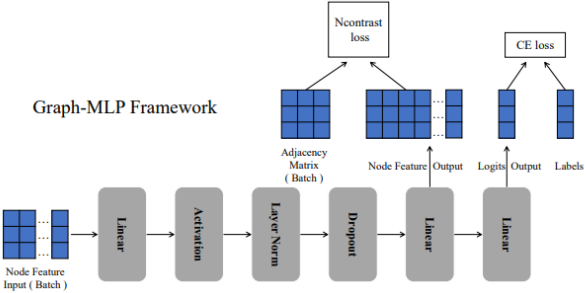

Figure 2: The framework of Graph-MLP.

Figure: The Framework of the Proposed Method

Revisiting GNNs

Graph Convolutional Networks (GCN) combine MLPs and node connections to propagate features across the graph.

$$X^{(l+1)} = \sigma\left(\hat{A}X^{(l)}W^{(l)}\right) \tag{1}$$

where $X^{(0)} = X$ is the input feature matrix, $\hat{A}$ is the normalized adjacency matrix, $W^{(l)}$ is the layer's weights, and σ is the activation function.

$$\hat{A} = D^{-\frac{1}{2}}(A + I)D^{-\frac{1}{2}} \tag{2}$$

A is the adjacency matrix, I is the identity matrix, and D is the degree matrix of $A + I$.

Graph-MLP's Structure

The Graph-MLP architecture processes node features through an MLP to produce node predictions. The model includes linear layers, activation functions, and normalization, structured as follows:

$$X^{(1)} = \text{Dropout}\left(\text{LN}\left(\sigma\left(XW^{(0)}\right)\right)\right) \tag{3}$$

$$Z = X^{(1)}W^{(1)} \tag{4}$$

$$Y = ZW^{(2)} \tag{5}$$

Here, σ denotes the activation function, LN is layer normalization, and Dropout is used to prevent overfitting. $W^{(0)}, W^{(1)}, W^{(2)}$ represent the weights at each stage of the model.

Question

Understanding Graph Structure in Graph-MLP

How does Graph-MLP capture the structure of a graph without message passing?

Contrastive Loss

- **Capturing Graph Structure**:
  - The structure of the graph is characterized through contrastive learning, an approach that distinguishes between similar (positive) and dissimilar (negative) samples.
- **Defining Positive and Negative Samples**:
  - For each node, its r-hop neighbors are considered as positive samples—nodes that share a similar context or role within the graph.
  - Non-neighboring nodes are treated as negative samples, representing dissimilar or unrelated contexts.

Neighbouring Contrastive Loss

The Neighbouring Contrastive Loss (NContrast loss) encourages positive samples to be closer in the feature space and negative samples to be further away. It is defined for the i-th node as follows:

$$\ell_i = -\log \frac{\sum_{j=1}^{B} 1_{\{y_i=y_j\}} \exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{B} 1_{[k \neq i]} \exp(\text{sim}(z_i, z_k)/\tau)} \tag{6}$$

where sim denotes the cosine similarity between nodes, τ is the temperature parameter, and γ_{ij} is an indicator function where $\gamma_{ij} = 0$ if node j is the r-hop neighbor of node i, and $\gamma_{ij} = 1$ otherwise.

Final Loss Function

The final loss function of Graph-MLP combines the Neighbouring Contrastive Loss with the standard cross-entropy loss:

$$\text{loss}_{\text{NC}} = \frac{1}{B} \sum_{i=1}^{B} \ell_i \tag{7}$$

$$\text{loss}_{\text{final}} = \text{loss}_{\text{CE}} + \alpha \cdot \text{loss}_{\text{NC}} \tag{8}$$

Here, loss_{CE} is the cross-entropy loss for node classification, and α is the weighting coefficient balancing the two loss terms.

Experimental Results

Graph-MLP was tested against baseline models on three citation network datasets. The results are as follows:

Table: Test accuracy (%) on citation network datasets.

Model	Cora	Citeseer	Pubmed
DeepWalk	70.7	51.4	76.8
AdaLNet	80.4	68.7	78.1
LNet	79.5	66.2	78.3
GCN	81.5	70.3	79.0
GAT	83.0	72.5	79.0
DGI	82.3	71.8	76.8
SGC	81.0	71.9	78.9
MLP ($\alpha = 0$)	57.8	54.7	73.3
Graph-MLP	**79.5**	**73.1**	**79.7**

Result Analysis on Cora Dataset

- Graph-MLP demonstrates a **lower performance on the Cora dataset**.
- This outcome suggests that the **smaller graph size** of Cora may not provide adequate supervisory signals needed for effective contrastive learning.

Conclusion: Smaller graphs might not fully leverage the benefits of contrastive learning due to limited contextual diversity.

Result Analysis on Citeseer and Pubmed Datasets

- On **Citeseer and Pubmed** datasets, Graph-MLP **outperforms** the baseline models.
- The **larger sizes** of these graphs likely offer **more opportunities** for contrastive learning to exploit the supervisory signal.

Conclusion: Larger graphs, with their rich and diverse contexts, are more suitable for contrastive learning approaches.

Experimental Results: Inference Speed

One of the significant advantages of Graph-MLP is its faster inference time compared to GCN, as demonstrated in the testing time comparison:

Table: Testing time (s) for Graph-MLP and GCN.

Model	Cora	Citeseer	Pubmed
GCN	0.000917	0.001918	0.010107
Graph-MLP	0.000276	0.000262	0.000262

Graph-MLP shows significantly lower testing times across all datasets.

Visualization of Embeddings

To understand the impact of NContrast loss on vanilla MLP, we visualize the feature embeddings using t-SNE:

- α values range: [0.0, 1.0, 10.0]
- Hidden dimension (d): 256

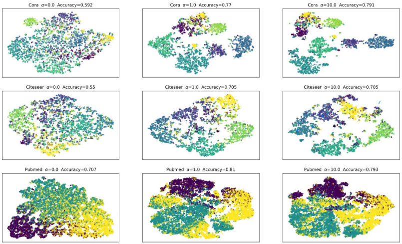

Figure 5: t-SNE visualization of feature embedding of citation networks. We plot the whole nodes feature produce from the converged model.

Figure: t-SNE visualization of feature embeddings with varying α

Thank You

Thank you for your attention!

Questions?