Graph Neural Network Learning: Integration of U-Net and Graph Convolutional Layers

Yang Zhang

Abstract – Convolutional neural networks (CNNs) have presented a great success in the application of artificial intelligence, specifically on the Euclidean structure data. With the development of CNNs, Graph Neural Networks (GNNs) as an extension of the convolutional method have been proposed with a promising potential to deal with non-Euclidean data, e.g. graph data. The property of graph data shows more flexibility and strong presentations in high dimensions that fastens the process of feature captures in deep learning models, which also benefits the image segmentation tasks. In this project, we explored the concept of GNNs from various previous researches and applied them to segment the breast cancer tumor by using ultrasonic images.

Keyword – graph neural network, ultrasound image data, image processing

1. Background

Due to the flexibility, generalizability and potential efficiency, graph data has gained great popularity in many fields, such as physics, economics, biology (B. Knyazev, 2019). In general, graph data has been used as media to describe connections or similarities among the objectives in networks. These properties of graph data make it more flexible and informative for the training of deep learning neural models, which might obtain a better performance under the different experimental scenarios. Mathematically speaking, a graph is composed of a set of vertices/nodes and edges, where the vertices/nodes represent the points or objects, and the edges represent the links connecting the vertices (Tutorialspoint, n.d.). Thus, the graph data can be also extended to more modern fields of interest like social network study, and city transportation study, sports team study or any studies involving interacting objects in dynamic systems (B. Knyazev, 2019).

The successful application of neural networks is building on the massive data size requirements. Nevertheless, the boost in computer science makes it practical and realistic to deal with such heavy loads of data. Further, graph neural networks would benefit the models with less requirement on the data size. Garcia et al. proposed a graph neural network model that only requires a few shots of the learning process (2018), which drastically decreases the amount of input data. Compared with many other traditional machine learning algorithms, even deep learning models display better performances to deal with big datasets. However, the limitation of data size sometimes failed the application of deep learning models. Graph Neural Network (GNN) has been proposed to combine the neural network with the knowledge/information of the graphical structures of datasets. The GNN is a newly-developed deep learning model that has been presented in various current research fields, especially applied in the image processing domain. Unlike typical traditional neural networks (CNN, RNN) are rigorously restricted with minimal-required data size, GNN extends the possible extraction of the structural information (dependency) from data by constructing a graph network into non-Euclidean space. In this case, a higher accuracy, less data size requirement and efficiency of the deep learning model might be achieved. Moreover, the nodes of the graph are not limited to restrictive data forms, which include images, pixels, voxels and even image classes based on different research purposes. As we know, U-Net is one of the most classical and typical architectures for image segmentation tasks. In this project, it is combined with the concept of graph network to segment breast cancer tumor images.

The target of this paper is to understand and implement GNN, which is combined with U-Net architecture for ultrasonic image tumor segmentations. In specific, the objectives are to:

1) implement the U-Net model in PyTorch;
2) understand and implement the GNNs in PyTorch;
3) develop convolutional graph U-Net.

2. Literature Review

There are many papers that discuss the GNN concept and its applications. Some of them have been reviewed in this project to construct a proper graph neural network model for image segmentation. I first reviewed a panel of papers that have been written by Scarselli, Gori, et al. (2009), Wu, Pan, et al. (2019), Zhou, Cui, et al. (2019), which gives me a general understanding of graph neural networks and its applications. Then I focused on the possible option of building a graph neural network by few-shot learning, which is inspired by V. Garcia et al. (2018). However, the model was proposed in that paper is not suitable for our image segmentation purpose. Hamilton, Ying, et al. present a learning process through the edges in a graph network that embeds node information to their local neighbors (2018), it can be realized by using a Pytorch package "torch_geometric.nn". Other options to build a decent graph neural network, in the paper of Defferard, Bresson, et al., the authors presented a simple way to define the adjacency matrix by computing the distance of nodes, which updates the prior weights before models (2016). It can be understood as a preprocessing step or adding the initial weights step to improve training time and decrease the data size requirements. In another paper proposed by Kipf and Welling, it presented a semi-supervised classification model with graph convolutional networks, which calculated the weights of convolutional layers from graph topology level (2017). It is also an option that can be used to create a GNN. In the end, a famous paper outlined a novel model in 2019, 'Convolutional Graph U-Net' that can be perfectly used for image segmentation under the construction of a graph neural network (Anonymous, 2019).

3. Data Description

In the field of breast cancer tumor study, tumor segmentation is key to analyze breast cancer image data. To understand and leverage GNN for reducing the manual radiological work, I have applied the proposed method on the Lumpectomy project data. I consulted with the data owner, Dr. Parvin Mousavi, and then I was granted with access to the tumor ultrasound image data of four cases. The compressed image file was sent to me with 961 images contained for analytical work, and each image is sized 128x128. All the data pre-processing work has been performed by Dr. Parvin Mousavi's lab. An example of the ultrasound image attached below(see Figure 1).

Figure 1: An example of breast cancer tumor ultrasound image data from the Lumpectomy project.

4. Methodology

For many reasons, I ultimately decided to create a graph convolution layer that later combined with U-Net architectures for my final graph neural network model. Ideas are inspired by reviewing papers written by Defferard, Bresson, et al. (2016), Bronstein, Bruna, et al. (2017) and Kipf and Welling (2017) to create a graph convolutional layer for the model training.

The idea of graph neural networks is how to update the parameter weights for each variable to have a less loss value in forward and backward propagation through the model. It is different from the regular convolutional neural network to update the weight, graph neural network considering the connections or similarities between nodes, which can be present as follows:

$$X^{i+1} = A * X^i * W^i .$$

Where X^{i+1} represents the outputs at i+1 iteration, and X^i represents the variable at the ith iteration, weights for the parameters are W^i and the adjacency matrix A shows the difference between graph network model and convolutional network model. As mentioned previously, many methods can be applied to compute matrix A. Two approaches have been used here to calculate the value of the adjacency matrix.

First, using a fixed Gabor filter to convert pixel image data into the adjacency matrix, this is a simple and typical way to get it.

Second, I used the normalized graph Laplacian to compute the adjacency matrix, the equation is defined as follow:

$$L = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}, \text{ where}$$

$A \in R^{N \times N}$ represents a binary or weighted adjacency matrix,

$D_{ii} = \sum_j A_{ij}$ represents the degree matrix.

Normally, in order to create graph neural network models, people use Linear or Dense functions to link the adjacency matrix and input feature data as inputs, and global pool (sum pool) or mean pool for the fully connected layer for the final outputs. However, the above case only works well on classification tasks, not efficient for segmentation. I thus combined a convolutional graph layer (convert data into graph data) with U-Net model (for the training process) for the segmentation problem.

5. Learning Process

The learning process has been carried out in three steps. I first tested a simple GNN model by using the PyTorch package in Python on a public MNIST dataset. Pytorch also takes me lots of time to learn and adjust the environment for the remote platform I've been using (It turns out Pytorch is not mature for graph network development, such as half decision problems and so on). The reason I have to learn Pytorch is that most graph network models are built in Pytorch, since it is easier to customize the weights by adding adjacency and has its own graph network development package, however, the package is not compatible with my remote working platform. Then, I expanded my learning to image segmentation and applied the U-Net approach on the Transmission Electron Microscopy (ssTEM) dataset of the Drosophila first instar larva ventral nerve cord. I experimented on the U-Net realization in PyTorch but struggled due to the version conflict and environmental incompatibility. Finally, I combined a convolutional graph layer with U-Net to Karas with the purpose of performing graph networks using GNNs to complete my understanding of graph GNNs.

6. Results

I obtained a decent accuracy rate from both the training and validation datasets by using the Convolutional Graph U-Net model. The dice_coef_loss function is utilized to compute error for each training process and 64 features were used for the first convolutional layer. Figure 2 demonstrates the accuracy and loss curves for the Convolutional GraphUnet model. Compared with the U-Net model, both models have good accuracy, while the Convolutional GraphUnet model converges much faster than regular U-Net. It can be explained by the fact that convolutional graph layers give prior evidence or pre trained weight based on the pixel positions. Graph data shows more concentrated feature maps in high dimensions.

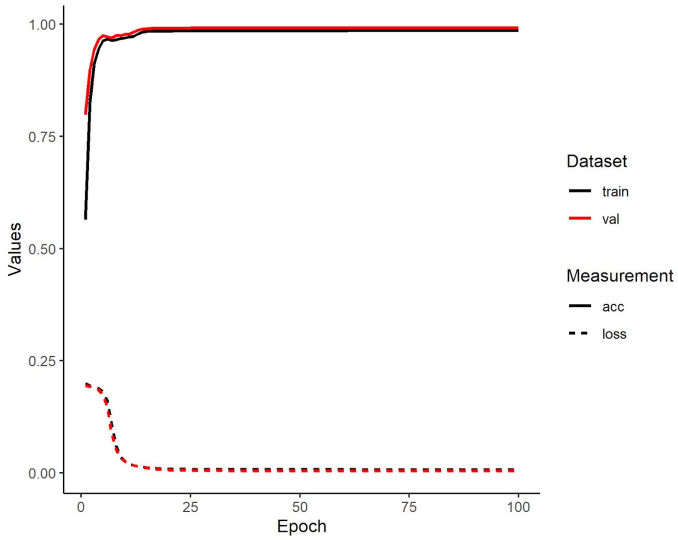

Figure 2. The plot of convolutional graph U-Net results by using ultrasonic breast cancer images. The black curves represent the curves of the training set, and red curves represent the curves of the validation set; the solid curves represent the algorithm accuracy, and dash curves represent the loss function of the algorithm.

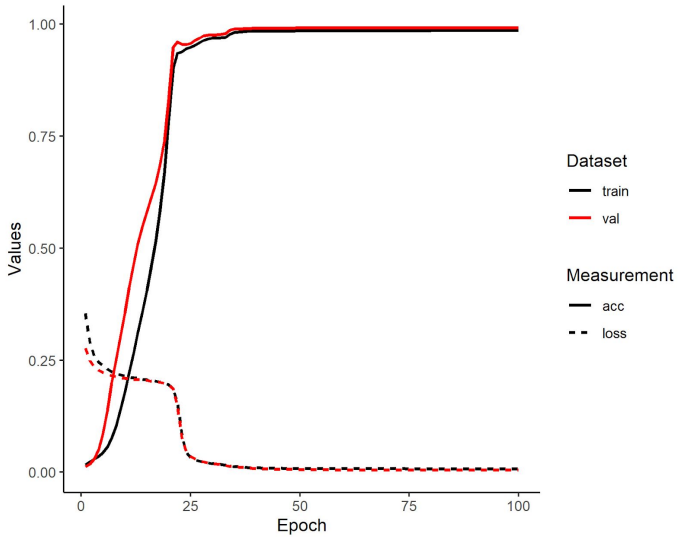

Figure 3. Results_unetwithVal
The black curves represent the curves of the training set, and red curves represent the curves of the validation set; the solid curves

represent the algorithm accuracy, and dash curves represent the loss function of the algorithm.

7. Discussion

This project is very time-consuming and complicated in terms of not only understanding the graph network, but also the process of implementation from scratch. There is very little source code that we can learn for the graph neural network. Also, the lack of experience in deep learning model development gives me a tough time at the beginning of coding. Even though the model achieves a good accuracy rate, there is still a knowledge gap as to why and how it works. I've also tried to plot the predicted results compared with original masks and figures. However, Convolutional graph U-Net presents messy outputs, it is likely that the adjacency matrix shrunk the value scale of inputs that needs to be fixed after we got the prediction results. In the future, the gPool and upgPool function can be implemented from paper (Anonymous, 2019), which basically uses the rank way to choose the most informative nodes from the highest projection direction. In conclusion, this project still needs a huge amount of effort and time to be completed throughout.

8. References

Anonymous. (2019). Graph U-Net. ICLR. https://doi.org/10.1007/978-3-030-00928-1_48

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., & Vandergheynst, P. (2017). Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, *34*(4), 18-42.

Defferrard, M., Bresson, X., & Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. Neural Information Processing Systems.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2019). A Comprehensive Survey on Graph Neural Networks. LaTex Class Files, XX(Xx), 1–21. https://doi.org/10.1109/tnnls.2020.2978386

Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. 5th International Conference on Learning Representations, 1–14.

Knyazev, B. (2019). *Tutorial on Graph Neural Networks for Computer Vision and Beyond (Part 1)*. https://medium.com/@BorisAKnyazev/tutorial-on-graph-neural-networks-for-computer-vision-and-beyond-part-1-3d9fada3b80d

Kung-Hsiang, H. (2019). A Gentle Introduction to Graph Neural Networks (Basics, DeepWalk, and GraphSage). Medium. https://towardsdatascience.com/a-gentle-introduction-to-graph-neural-network-basics-deepwalk-and-graphsage-db5d540d50b3

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., & Monfardini, G. (2009). The graph neural network model. IEEE Transactions on Neural Networks, 20(1), 61–80. https://doi.org/10.1109/TNN.2008.2005605

Tutorialspoint. (n.d.). *Data Structure - Graph Data Structure*. Retrieved April 16, 2020, from https://www.tutorialspoint.com/data_structures_algorithms/graph_data_structure.htm

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2019). A Comprehensive Survey on Graph Neural Networks. LaTex Class Files, XX(Xx), 1–21. https://doi.org/10.1109/tnnls.2020.2978386

Xu, K., Hu, W., Leskovec, J., & Jegelka, S. (2019). How Powerful are GCN. Int. Conf. on Learning Representations, 1–17. https://openreview.net/forum?id=ryGs6iA5Km

Zhou, J., Cui, G., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., & Sun, M. (2019). Graph Neural Networks: A Review of Methods and Applications. 1–22. http://arxiv.org/abs/1812.08434