Node-Aligned Graph Convolutional Network Implementation Experiment

Liu Jing Xuan
McMaster University
Hamilton, ON, Canada
liu1819@mcmaster.ca

ABSTRACT

In this essay, I attempt to reproduce and validate the method originally proposed by Guan, Yonghang[1], which introduces a hierarchical global-to-local clustering strategy for constructing a Node-Aligned Graph Convolutional Network (NAGCN). This innovative approach aims to represent Whole Slide Images (WSIs) with enriched local structural information and a broader global distribution. Whole Slide Images, pivotal in computational pathology, pose significant challenges due to their gigapixel size and the intricate details they contain. By implementing a global clustering operation on the instance features of the dataset, I establish correspondence across different WSIs. This is followed by executing a local clustering-based sampling strategy, allowing for the selection of typical instances from each cluster within a WSI. Ultimately, employing graph convolution enables me to obtain a comprehensive representation. My graph construction strategy facilitates the generation of a consistent representation for varying WSIs, streamlining the process for subsequent classification tasks. This essay will detail my reproduction process, compare my findings with the original experimental results on cancer subtype classification datasets.

PVLDB Reference Format:
Liu Jing Xuan. Node-Aligned Graph Convolutional Network Implementation Experiment. PVLDB, 14(1): XXX-XXX, 2020.
doi:XX.XX/XXX.XX

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at URL_TO_YOUR_ARTIFACTS.

1 EXPERIMENT

In this section, I will present my work about reproduce the method from the paper. All the experiment are implemented using PyTorch[3] and a personal PC with Nvidia RTX2070 GPU and Intel i5 9_gen CPU.

1.1 Dataset

In this experiment, I selected the Breast Histology Images dataset[2] available on Kaggle, which comprises 279 whole slide images of Invasive Ductal Carcinoma (IDC) within Breast Cancer (BCa) specimens, scanned at a magnification of 40x. From these, a total of 277,524 patches, each measuring 50x50 pixels, were extracted, including 198,738 IDC-negative and 78,786 IDC-positive samples. Notably, each patch's filename encodes crucial information, including the patient ID and the classification, where "0" denotes non-IDC and "1" indicates IDC presence.

1.2 Implementation Step

The step of implementation will describe by each subsection below in order, and also illustrated in Figure 1.

1.2.1 Target Label. Categorize the severity of Invasive Ductal Carcinoma (IDC) in the dataset, I devised a method based on the proportion of IDC-positive (label 1) to IDC-negative (label 0) patches within each patient's Whole Slide Image (WSI). Specifically, I calculated the ratio of positive to negative labels for the patches associated with each patient's WSI. If the proportion of IDC-positive patches exceeded 50%, the case was classified as 'severe' (represented by a label of 1). Conversely, if the proportion was equal to or less than 50%, the case was classified as 'mild' (denoted by a label of 0).

1.2.2 Feature Encode. Initially, I organized the dataset by mapping image data to corresponding labels and patient IDs, make easy to track the relation for each data during the experiment. And standardized image patches to a 224x224 pixels to ensure uniformity dimension required for model input. Then employed a pre-trained ResNet-50 model for feature extraction, focusing on extract data from third layer of model to encode each image into a 1024-dimensional feature vector.

1.2.3 Codebook Generation. The process of codebook generalization encompasses two pivotal stages: global clustering and inner clustering. Initially, the entirety of the dataset is segmented into k distinct clusters via Kmean global clustering, the choice of k was evaluate by silhouette coefficients over 0.4, and principal component analysi PCA was used to reduce the feature dimension to obtain a better clustering performance. Each cluster, denoted as k_i, embodies a 'visual word' within the codebook lexicon. For the purpose of generating these codebooks, a subset constituting 50% of the data was used for training. The residual dataset was then subjected to clustering using the trained clusterer, which effectively applying the codebook for direct categorization. After formation of the codebook, data of the codebooks were systematically associated with their respective patient IDs by the map, that ach patient have their own codebook.

Proceedings of the VLDB Endowment, Vol. 14, No. 1 ISSN 2150-8097.
doi:XX.XX/XXX.XX

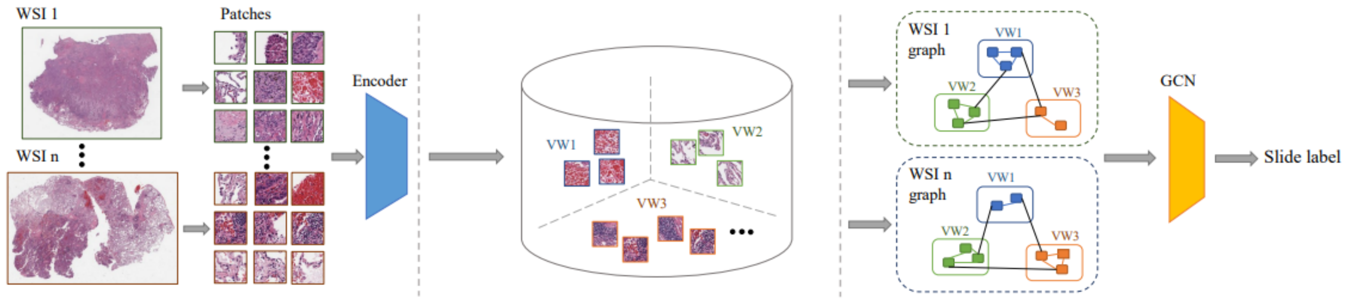

Figure 1: Overview of the implementation process: WSI was divided into patches (which was already done in my dataset); Encode each patches into feature vector; Generate codebook through global and inner clustering; Construct the graph; Training the GCN model.

Then I go through each patient's codebook data. Within each cluster k_i, a Kmean inner clustering was performed, during which one representative feature sample were selected randomly from each sub-cluster sk_i. After this step, each patient have a data represent by a vector of dimensions $[k * sk, 1024]$. For those k_i do not have enough individuals, I embedded zero vector and record the those clusters as missing node for optimize training purpose.

1.2.4 Graph Construction. To constructing a graph representation for each patient's feature data. Considering each global cluster k_i as a sub-bag sub_k. The graph was represent by a dimensions nxn adjacent matrix, where $n = k * sk$. I use euclidean distance to calculate the distance between each feature samples, $matrix[i, j] = 1$ if i_{th} and j_{th} samples belong to same sub_k or they have a decent distance value, otherwise, $matrix[i, j] = 0$, also the samples marked in the miss will consider as 0.

1.2.5 GCN Train and Evaluate. A graph convolution neural network with three graph convolution with each layer followed by a ReLU activation and a classify net with two layer was created for the task. Binary Cross Entropy with sigmod layer and Adam optimizer were adopted to optimize the model. The graph convolution layer generate a slide-level representation of the WSI feature like showed in Figure 2, I set output feature dimension of this layer to 16, this give me a $16 * k$ dimension vector as ouput. The output from the convolution layer was used in a classify layer to perform binary classification.

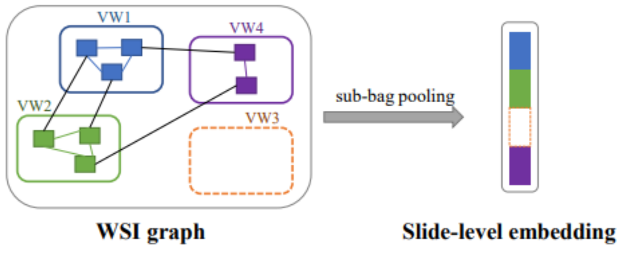

Figure 2: An illustration of graph convolution layer encode WSI graph.

The train loss, train accuracy and validation accuracy was used as metrics to evaluate the performance of classification.

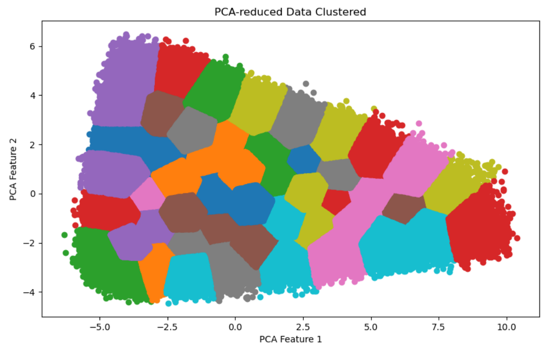

Figure 3: global clustering when $k = 50$.

2 RESULTS

2.1 Model Performance

In the experiment, the different number of combination for global and inner cluster was used for training. The current best setting is $k = 50$ and $sk = 15$, the global clustering visualization is showed in Figure 3. During the training and testing. The dataset batch size was 32, I set the learning rate as 5×10^{-5}, L2-regularization 5×10^{-3} and dropout rate 0.4 to mitigate model over-fitting, the dataset was split to 80% for train and 20% for validation. And the result is showed in table 1.

Table 1: GCN Performance

Epochs	Train Loss	Train Acc	Valid Acc
1	0.7340	0.7175	0.6786
20	0.4457	0.7892	0.7500
40	0.2117	0.8520	0.8321
60	0.0522	0.9203	0.8114
80	0.0179	0.9596	0.8857
100	0.0123	0.9776	0.8993

From the result, the training process reflected a successful learning trajectory. The loss decreased to 0.0123, the train accuracy obtained 97% and the model seems managed to mitigate over-fitting effectively, as evidenced by the consistent increase in validation accuracy and obtained a 89% at later epochs.

3 DISCUSSIONS

3.1 Problems Encountered and Solutions

In the initial phases of my experiments, I did not track missing nodes in the inner cluster, but directly remove the patient that has a cluster that did not meet the criteria. This approach inadvertently led to a significant reduction in the dataset's size, adversely impacting data diversity and promoting overfitting, it only take few epochs to reach the 100% train accuracy. This was solved by adopted a strategy of marking insufficient clusters as missing nodes and representing the corresponding missing features with zero vectors. Through this strategy, This adjustment enabled us to retain the original dataset's integrity while conveying the absence of data to the model effectively. Furthermore, a mask matrix was introduced to inform the model about missing nodes, facilitating more nuanced weight adjustments during training.

Due to the involvement of multiple model training in this experiment, in order to facilitate data processing and reduce training workload, the output data will be saved locally after feature encoding and graph construction, so that it can be directly called in subsequent steps to save training time.

Although the result shows that my model can achieve good result on the dataset, there are still several issues that need to be point out. When trying different numerical combinations of k and sk, it have a significant impact on the model result, especially lower values can lead to severe overfitting. Unfortunately, this issue may be caused by the diversity and integrity of the dataset. An intuitive solution could be to use a more extensive array of original Whole Slide Imaging (WSI) data, but this requires substantial storage capacity and computational resources, as evidenced by memory exhaustion errors encountered during our experiments. Perhaps leveraging cloud server or employing a more advanced workstation would be a better choice for this experiment.

3.2 Conclusion

In this work, I successfully reproduced the implementation described in the original paper. The approach outlines a graph construction strategy for large-scale and complex images. This strategy not only preserves the local structural information and global distribution but also aligns different Whole Slide Images (WSIs), thereby circumventing the need for unordered pooling to obtain bag-level representations. The experiment further demonstrated that the model's performance heavily depends on the effectiveness of the codebook construction, which in turn requires a high-quality dataset. Future work will likely focus on improvements in feature engineering.

REFERENCES

[1] Yonghang Guan, Jun Zhang, Kuan Tian, Sen Yang, Pei Dong, Jinxi Xiang, Wei Yang, Junzhou Huang, Yuyao Zhang, and Xiao Han. 2022. Node-Aligned Graph Convolutional Network for Whole-Slide Image Representation and Classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 18813–18823.

[2] Paul Timothy Mooney. n.d.. Breast Histopathology Images. https://www.kaggle.com/datasets/paultimothymooney/breast-histopathology-images. Accessed: 2024-04-05.

[3] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. https://pytorch.org/