Programming Project – Graph Neural Networks and Other Techniques for Node Classification

CIS 5930, Graph Neural Networks (Summer 2022), Department of Computer Science, Florida State University

Points: 100
Maximum Team Size: 2
Due: 11:59pm on Tuesday, July 19th, 2022

Submission: You need to submit electronically via Canvas by uploading a) a pdf file (named "**lab-Lastnames.pdf**") as your report (including analysis and experimental results), and b) the program(s) you have created (named as "**lab-prog-Lastnames.???**"); if there are multiple program files, please zip them as a single archive. Here replace "Lastnames" using your last names of the group members in the file names. Only one submission is required for each group.

The main purpose of this assignment is to design and use graph neural networks, (deep) convolutional neural networks, and label propagation for node classification. Note that the focus of this programming project is to gain a deeper understanding of these techniques and having good recognition performance itself is not sufficient.

For this assignment, we use a handwritten digit dataset, consisting of a training set (http://www.cs.fsu.edu/~liux/courses/GNN/assignments/zip_train.txt, 2007 samples) and a test set (http://www.cs.fsu.edu/~liux/courses/GNN/assignments/zip_test.txt, 7291 samples). A description of the entire dataset is available at http://www.cs.fsu.edu/~liux/courses/deepRL/assignments/zip_info.txt. Very briefly, each row in the files is a sample, starting with the class name (0 to 9) and then 256 floating point numbers between -1 and 1. See the additional information near the end on how to visualize the digits.

You need to generate the graph by combining the samples in the training set and test set first, and then creating an edge between any two samples if one sample is among the seven nearest neighbors of the other sample; since nearest neighbors are not symmetric, you need to check both cases. In your report, you need to describe the graph generated, including a 10x10 "generalized" adjacency matrix for the super graph, where all the samples with the same label are grouped into one super node; a diagonal element of the generalized adjacency matrix shows the total number of edges within the super node and an off-diagonal element shows the total number of edges between the samples in the two super nodes.

Task I – Graph Neural Network Design

In the graph learning framework you have set up, design one convolutional neural network and two different graph neural networks for node classification.

(1) A convolutional neural network to classify the input images; the trained neural network will be used by the correct and smooth technique.
(2) A graph network for transductive node classification using one or more spectral-based graph filters; your network must have at least two layers for graph filtering.
(3) A graph network for transductive node classification using one or more spatial-based graph filters; your network must have at least two layers for graph filtering.

In your report, you need to describe your convolutional and graph neural networks clearly and justify your design choices.

Task II - Node Classification Techniques and Analyses

You need to complete the following tasks and perform the required analyses.

(1) **DL Method** - Train your convolutional neural network using the samples in the training set and then use the model to classify the samples in the test set. You need to report the loss and accuracy on the training and test set with respect to the number of training epochs and the confusion matrix for the test set of your final model.

(2) **C&S Method** - Initialize the probabilities for the test samples in the graph using the probabilities from your trained convolutional neural network and then run the correct and smooth algorithm to classify the samples in the test set. You need to report the accuracy on the test set with respect to the number of smoothing iterations and the confusion matrix for the test set of the final results.

(3) **Spectral GNN Method** - Train your spectral-based graph neural network and classify the test samples. You need to report the loss and accuracy on both the training and test set with respect to the number of training epochs and the confusion matrix for the test set of your final model.

(4) **Spatial GNN Method** - Train your spatial-based graph neural network and classify the test samples. You need to report the loss and accuracy on both the training and test set with respect to the number of training epochs and the confusion matrix for the test set of your final model.

Then compare and contrast the four different methods by analyzing the results. In particular, you need to compute and show relative confusion matrixes between each pair of the methods, where the confusion is relative to the results of the other method (i.e., using the classification results from the method as the ground truth). Then you need to analyze the differences among the four methods and provide plausible explanations for the observed differences.

Extra Credit Options

(1) **Nonlinear high-order label spreading algorithm**. Instead of using linear combination of neighbors for label smoothing, use the label spreading method proposed in this paper ("Nonlinear Higher-Order Label Spreading," by F. Tudisco et al., WWW, 2021). You need to show the confusion matrix relative to the ground truth labels and also the relative confusion matrix relative to the labels given by the original correct and smooth algorithm. You need to analyze the differences and provide plausible explanations.

(2) **Inductive node classification**. Instead of doing transductive node classification, train your spatial-based graph neural network on the training set using the seven nearest neighbor graph of the training samples only and then classify the test set using the trained network and the graph using the seven nearest neighbor graph of the test samples only. You need to show the confusion matrix relative to the ground truth labels and also the relative confusion matrix relative to the labels given by the original transductive classification results. You need to analyze the differences and provide plausible explanations.

(3) **Deep belief network (or deep restricted Boltzmann machine) for unsupervised feature learning**. In this case, you train a deep belief network (or deep restricted Boltzmann machine) in an unsupervised manner and then use the learned representations to compute the distances and therefore the nearest neighbors. You need to show how the learned representations change the graph structure and the node classification for the last three methods (C&S method, spectral GNN method, and spatial GNN method).

Grading

- **Report and analysis** – 20 points
  - Note that the focus is on understanding and you have to provide meaningful/insightful analyses and explain what you have observed in your experiments as explained in the tasks.
- **Correct implementation and experimental results** – 80 points
  - **Task I – 20 points**
    - **Correct architectures** – 15 points
    - **Explanations -** 5 points
  - **Task II – 60 points**
    - **10 points for each of the four methods**
    - **Analyses – 20 points**
- **Nonlinear high-order label spreading algorithm –** 10 points
- **Inductive node classification** – 10 points
- **Deep belief network (or deep restricted Boltzmann machine)** - 10 points

Additional information

Spectral-based graph filters are covered in Section 5.3.1 and spatial-based graph filters in Section 5.3.2 of the textbook. The Correct and Smooth algorithm is described in "Combining Label Propagation and Simple Models Out-performs Graph Neural Networks" by Q. Huang et al., available from https://arxiv.org/abs/2010.13993.

Deep belief networks and deep restricted Boltzmann machines for representation learning are covered in Chapter 20 of "Deep Learning," available from https://www.deeplearningbook.org.

The following shows the hand-written digits in the training set so that you know what the digits look like. If you like to show a digit, you need to scale the values from [-1 1] to [0 255] and reorganize the pixels from a 256-element vector to a 16×16 image; you may need to rotate the image.