Yash Avinash Patole

CWID: 10460520

CS-541 Artificial Intelligence
Mini Project

Title:
Environment Setup

Under the Guidance of:

Prof. Abdul Rafae Khan

Stevens Institute of Technology

Title : The Logical Expressiveness of Graph Neural Networks.

Authors: Pablo Barcelo´ (IMC, PUC & IMFD Chile), Egor V. Kostylev (University of Oxford), Mikael Monet ¨ (IMFD Chile), Jorge Perez ´ DCC, UChile & IMFD Chile Juan Reutter DCC, PUC & IMFD Chile Juan-Pablo Silva DCC, UChile

Name: Yash A. Patole

StevensId: ypatole@stevens.edu

CourseName: CS 541 Artificial Intelligence

Abstract: Graph neural networks are a type of neural network that has lately gained popularity for a variety of structured data applications, including as molecular classification, knowledge graph completion, and Web page ranking. The main idea of the authors implementing in this paper is to experiment and understand the connection between neurons which are not arbitrary but still they reflect in the structure of the input data. In this paper it states that the capacity of graph neural networks (GNNs) to differentiate nodes in graphs has recently been measured using the Weisfeiler-Lehman (WL) graph isomorphism test. However, this definition does not resolve the question of which Boolean node classifiers may be described by GNNs. In this paper it tackles the problem by using various formulas such as the logic of FOC2, a well-studied first-order logic fragment. FOC2 is linked to the WL test and, as a result, to GNNs. They have focused on explaining the class of GNNs where this class shows and updates the features of each node in the graph. But here they show that this class of GNNs is too weak to capture all FOC2 classifiers, and we give a syntactic description of the greatest subclass of FOC2 classifiers that AC-GNNs can capture. This subclass corresponds to a widely used logic in the knowledge representation community. The paper then shows demonstrate on synthetic data complying to FOC2 formulae, AC-GNNs fail to fit the training data, whereas ACR-GNNs can generalize even to graphs of sizes not encountered during training.

Github link of the research paper: https://github.com/juanpablos/GNN-logic

Data:

The graphs used in the paper are in the zip-file datasets.zip. Just unzip them to src/data/datasets.

The script expects 3 folders inside src/data/datasets named p1, p2 and p3. These folders contain the datasets for the properties α_1, α_2 and α_3 described in the appendix F on data for the experiment with classifier $\alpha_i(x)$ in equation.

In the code there these datasets are accepted parsed and there is a small description of the arguments in generate_dataset.

While there are .txt in file in each p1, p2 and p3 which has test and train data respectively. Size of the data are as follows:

P1: 1) size of train data is 230087. 2) size of test data is 51377.

P2: 1) size of train data is 230232. 2) size of test data is 51198.

P3: 1) size of train data is 229788. 2) size of test data is 51239.

Link to the dataset: https://github.com/juanpablos/GNN-logic

We can find the link here and also I will attach the zip file of the dataset with this submission.

Tools & Technologies: The details of the software/library packages tqdm==4.35.0

scipy==1.3.1

numpy==1.17.1

scikit-learn==0.21.3

matplotlib==3.1.1

networkx==2.3

torch==1.2.0

torch-cluster==1.4.4

torch-scatter==1.3.1

torch-sparse==0.4.0

torch-spline-conv==1.1.0

torch-geometric==1.3.1

requests==2.22.0

For this project we will use Google Colab and it's technologies.

Following links will be required to save:

1) /content/drive/MyDrive/GNN-logic/src/logging/results/p1-0-0-acgnn-aggA-readA-combT-cl0-L1.log
2) /content/drive/MyDrive/GNN-logic/src/logging/results/p1-0-0-acgnn-aggA-readA-combT-cl0-L1.log.train
3) /content/drive/MyDrive/GNN-logic/src/logging/results/p1-0-0-acgnn-aggA-readA-combT-cl0-L1.log.test

Hardware Requirements: This project requires GPU (cuda and other torch packages). My GPU is Nvidia RTX3080 and processor is AMD Ryzen 5000.

Data:

P1	**230087**	**51377**
P2	**230232**	**51198**
P3	**229788**	**51239**

For data Pre-Processing this project uses two files named argparser_real_data.py and datasets.py to parse and get the inputs ready for processing epochs.

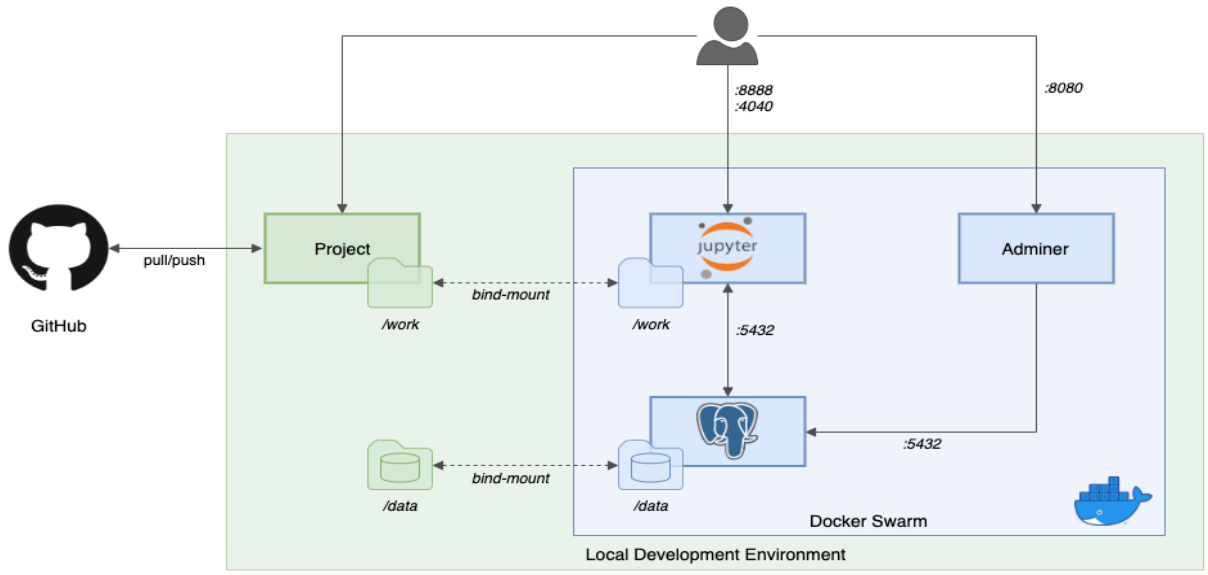
Experiments: The code helps to get logical expressiveness of graph neural networks. In this code it considers the simple FOC2 formula: (x):= Red(x) y Every red node in a graph must satisfy the condition Blue(y) if the graph contains at least

one blue node. The code first considers half of the graphs in each set (training and testing) containing no blue nodes, and 50% containing at least one blue node (about 20% of nodes are blue every set has a true class). As my initial test I will provide following training data outcomes which explains the learning rate, epochs of the our training dataset.

train_loss,test1_loss,test2_loss,train_micro,train_macro,test1_micro,test1_macro,test2_micro,test2_macro

0.6901471019, 0.6904880404, 0.6916720867, 0.53859858, 0.53905009, 0.53607746, 0.53637073, 0.52487661, 0.52607605

Following is similar network architecture that will be required where instead of postgres in this project I am using google colab and jupyter lab to get this project onboarded and modules are installed respectively.



Results: Following are preliminary results of this project that I managed to execute on my computer successfully. Following outputs contains 20 epochs for dataset P1. More testing, including datasets P2 and P3 are needed to be done.

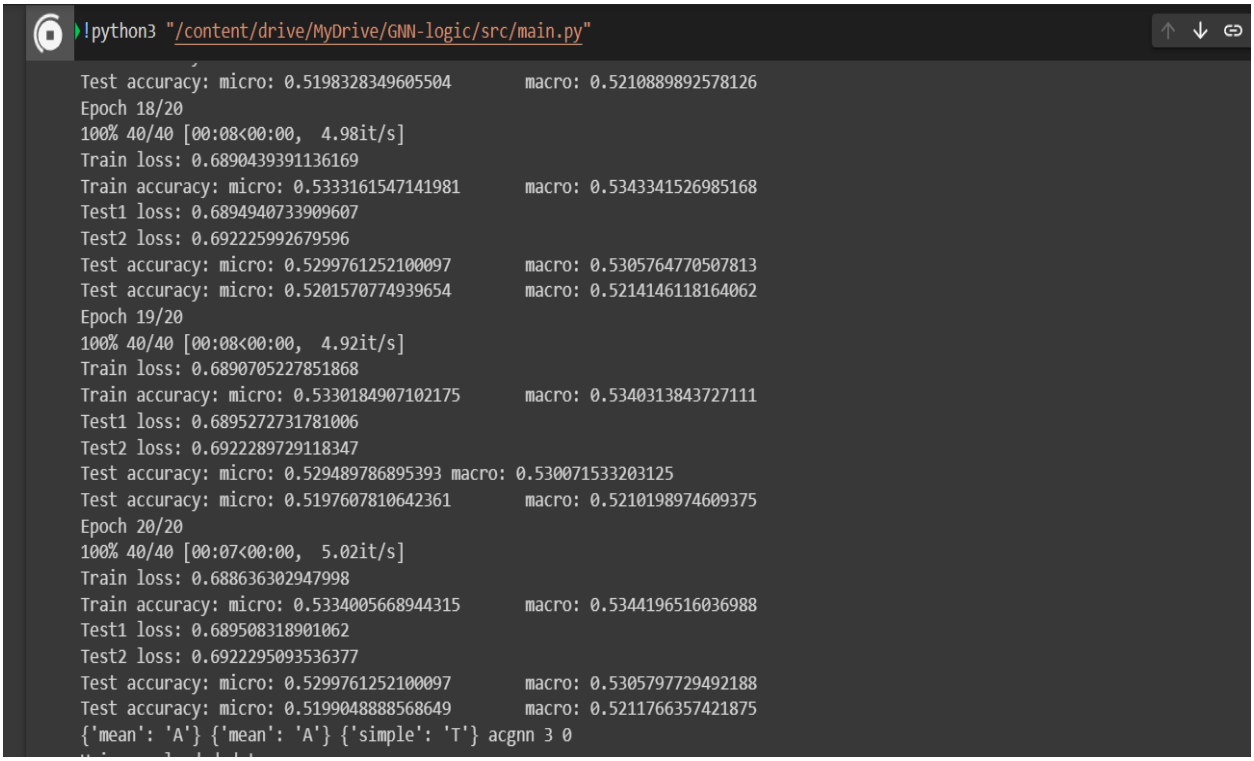
+ Code + Text

```
!python3 "/content/drive/MyDrive/GNN-logic/src/main.py"

Start running
Start for dataset datasets/p1/train-random-erdos-5000-40-50-datasets/p1/test-random-erdos-500-40-50-datasets/p1/test-random-erdos-500-51-60
Loading data...
#Graphs: 5000
#Graphs Labels: 2
#Node Features: 5
#Node Labels: 2
Loading data...
#Graphs: 500
#Graphs Labels: 1
#Node Features: 5
#Node Labels: 2
Loading data...
#Graphs: 500
#Graphs Labels: 1
#Node Features: 5
#Node Labels: 2
{'mean': 'A'} {'mean': 'A'} {'simple': 'T'} acgnn 1 0
Using preloaded data
/usr/local/lib/python3.7/dist-packages/torch_geometric/deprecation.py:12: UserWarning: 'data.DataLoader' is deprecated, use 'loader.DataLoader
  warnings.warn(out)
Epoch 1/20
100% 40/40 [00:02<00:00, 16.13it/s]
Train loss: 0.6920526623725891
Train accuracy: micro: 0.5314901859733613       macro: 0.5325788519382477
```

```
!python3 "/content/drive/MyDrive/GNN-logic/src/main.py"
Train accuracy: micro: 0.5314901859733613       macro: 0.5325788519382477
Test1 loss: 0.6921760439872742
Test2 loss: 0.6926807165145874
Test accuracy: micro: 0.5245821911751702        macro: 0.5252341918945312
Test accuracy: micro: 0.5167705443671866        macro: 0.5174842529296875
Epoch 2/20
100% 40/40 [00:02<00:00, 15.66it/s]
Train loss: 0.6914803981781006
Train accuracy: micro: 0.5275539127266912       macro: 0.5288044064044952
Test1 loss: 0.6918432712554932
Test2 loss: 0.6934691667556763
Test accuracy: micro: 0.5257317181006278        macro: 0.526685546875
Test accuracy: micro: 0.5122311488993767        macro: 0.5136141357421875
Epoch 3/20
100% 40/40 [00:02<00:00, 15.65it/s]
Train loss: 0.6913267970085144
Train accuracy: micro: 0.5274872715317701       macro: 0.5287326717376709
Test1 loss: 0.6918301582336426
Test2 loss: 0.6933726668357849
Test accuracy: micro: 0.5258201432487399        macro: 0.526787353515625
Test accuracy: micro: 0.512267175847534 macro: 0.5136484375
Epoch 4/20
100% 40/40 [00:02<00:00, 15.84it/s]
Train loss: 0.6912528872489929
Train accuracy: micro: 0.527549469980363        macro: 0.5287102447986602
Test1 loss: 0.6918671131134033
```

Executing (12m 25s) Cell > system() > _system_compat() > _run_command() > _monitor_process() > _poll_process()



Problems/Issues Faced: I faced few problems while setting this up on my local machine. First of all I tried to get this on VScode which did not work due to some version error. Then I decided to get this project on Jupyter and google colab. This step helped me setting up my environment successfully. While installing few libraries like torch-cluster, torch-scatter, scipy(numpy) I faced issues due to some errors. One of which included cannot get wheel setup for torch.