Binary Social Graph Classification using GNN

Advanced Topics in Machine Learning and Optimization

Francesco Ferrini
francesco.ferrini@studenti.unitn.it
224010

ABSTRACT

This report describes the methodology used to solve the task of binary classification of graphs in the domain of social interactions. In particular the data used to develop the solution were taken by [6] that is a an interdisciplinary research collaboration which studies social dynamics and human activity. The framework used in this project is Pytorch Geometric. The first section of the report describes the dataset, the method, training and evaluation along with different experiments; the second part instead will concentrate mostly on the explanation of the obtained result.
Github Link

1 INTRODUCTION

Graph Neural Networks (GNNs), which generalize deep neural networks with graph-structured data, have achieved considerable attention and achieved state-of-the-art performance in several tasks. The reason for that is the fact that many situations and domains can be translated to a graph domain. In this report are described the techniques implemented to perform binary graph classification in a school domain where the 2 classes represents the interactions in a high school and in a primary school. In order to improve the classification tasks are also calculated some graph statistics (node features and edge features) that have shown to help the network learning better. While the first part of the report will discuss about the dataset and the network implemented for this task, the second will introduce some explainability, showing the reasons why some graphs have been putted in one class with respect to the other one.

2 DATASET

As said in the introductory part, the dataset for this project were collected by the SocioPatterns collaboration, using an infrastructure based on wearable sensors that exchange radio packets, detecting close proximity (less than or equal to 1.5 m) of people wearing the device. The temporal resolution of the data is 20 s. The first dataset, used in [2] and [7], describes the interaction between children and teachers in a primary school and is composed by 242 nodes while the second one, [1], contains the temporal network of contacts between students in a high school in Marseilles, France and is formed by 126 nodes. Since the datasets given by Sociopatterns are in a csv format, in order to make use of them, firstly they are transformed in graphs using the networkx framework [3]. In the next paragraphs will be shown the different modalities in which those graphs will be sampled from the csv raw files.

Sliding window. With the term Sliding Window (SW) we mean that each sampled graph contains the interactions of people within t seconds. In the experiments you will see that we used different sliding windows in order to see the differences in the learning. In particular the SW used are of 60, 120, 180, 240 and 300 seconds. In order to make an example, if we use a SW of 60 sec, the first graph will contain all the interactions between 0 and 60 seconds and so on.

Temporal repetition. In order to have more graph for our training, since the temporal resolution of the data is 20 seconds, we sample graphs every 20 seconds. That means, making an example, that if we have a SW of 60 seconds, the first graph is containing the interactions from 0 to 60 seconds, the second graph is containing the interactions from 20 to 80 seconds and so on. This methodology allows to both have more graphs during the training and testing phase and also to have always the same number of graphs even with different SWs.

Balanced Dataset. In order to have a balanced dataset, the same number of graphs is sampled from the first and second files.

Node features. In order to study the effectivness of using graph metricsas node features in the training, in the report are shown the results of the network, both using and not using real node features, calculated on the graphs. The used node features are:

- Node degree: the number of edges adjacent to the node
- Clustering: the clustering coefficient for nodes
- Degree centrality: for a node is the fraction of nodes it is connected to
- Number of cliques: the number of maximal cliques for each node
- Core number: the core number for each vertex
- Square Clustering: the square clustering coefficient for each node that is the fraction of possible squares that exists at the node
- Average Neighbor Degree: the average degree of the neighborhood of each node
- Degree assortativity Coefficient: the degree assortativity which measures the similarity of connections in the graph with respect to the node degree
- Average Clustering: the average clustering coefficient for the graph. It will be the same for each node

In the case we don't use real node features, are appended as node features some random small float numbers.

Edge features. The edge features are used only in the Graph Attention Network (GAT). In this case, the only edge feature is the number of times the same interaction will be reapeted within the same sliding window. For example, whether node 1 and node 2 interact for 60 seconds in the same SW, the feature for that edge will be an integer value of 3.

2.1 Graphs

In this subsection are reported some graphs in order to visually and conceptually see the differences between them when changing the temporal length of the sliding window. What is clear even from these images is that augmenting the temporal factor of the sliding windows we have more interactions in both the classes of graphs.

Figure 1: Highschool graph - 60 seconds SW

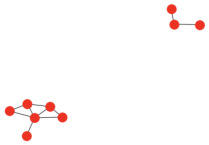

Figure 2: Highschool graph - 120 seconds SW

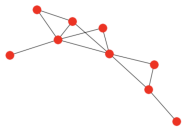

Figure 3: Highschool graph - 180 seconds SW

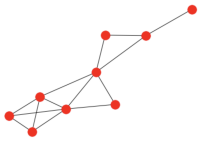

Figure 4: Highschool graph - 240 seconds SW

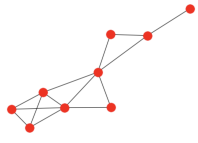

Figure 5: Highschool graph - 300 seconds SW

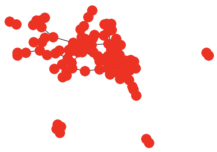

Figure 6: Primary school graph - 60 seconds SW

Figure 7: Primary school graph - 120 seconds SW

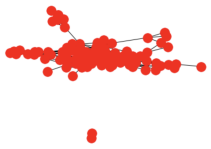

Figure 8: Primary school graph - 180 seconds SW

Figure 9: Primary school graph - 240 seconds SW

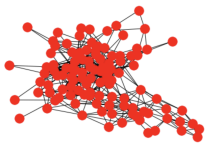

Figure 10: Primary school graph - 300 seconds SW

3 GRAPH CLASSIFICATION

In this part of the report will be discussed the Network implementation details. In particular we'll have a look at how to apply Graph Neural Networks to the task of binary graph classification that is the problem of classifying entire graphs given a dataset of graphs. So we want to embed entire graphs in such a way so that they are linearly separable.

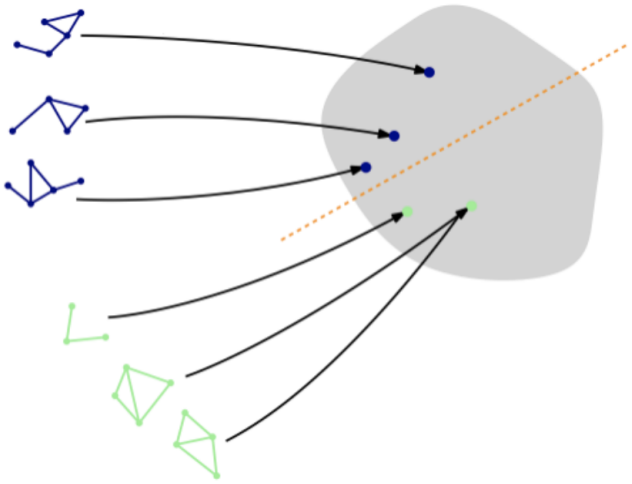

Figure 11: Binary Graph Classification Task

3.1 Training a Graph Neural Network

In order to train a GNN for graph classification the next procedure is usually followed:

- Embed each node by performing several message passing rounds
- Using a readout layer in order to aggregate node embeddings into an unified graph embedding
- Train the final classifier

The most used readout layer is the one that performs the average of node features across the node dimension. Having a single graph, its output is computed by:

$$r_i = \frac{1}{N_i} \sum_{n=1} x_n \tag{1}$$

It takes the node embeddings of all nodes in the batch and computes the graph embedding.

3.2 Network Structure

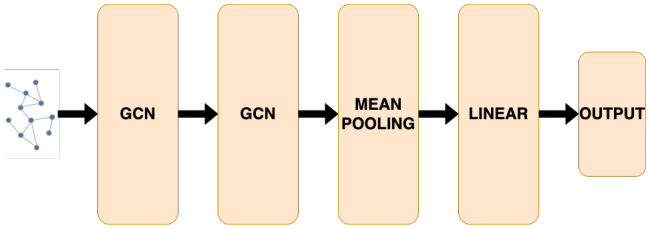

Figure 12: Network Structure

The Network Structure used for this project is reported in Figure 12. As you can see, the input is a graph represented by its adjacency matrix and node feature matrices. As first 2 layers in the image we have Graph Convolutional Layers but in the experiments we tried 3 different types of layers (with each one of them having 64 units and relu activation function), in particular:

- Graph Convolutional Layers (GCN) from [5]
- Graph Sage Layers (SAGE) from [4]
- Graph Attention Layers (GAT) from [8]. In this case we also made use of edge features.

Then next layer is a mean pooling layer where the learned node representations are summarized to have a graph representation. After that the graph representation is inputted to the final linear classifier passing through a dropout layer.

3.3 Training specs

In this subsection are reported the various training specs used in the project:

- Training-Validation Split: It's used 80 percent of the graphs in the dataset as training and 20 percent as validation.
- Number of graphs: the total number of graphs is 6198
- Optimizer: As optimizer it's used SGD with a learning rate of 0.001, a weight decay of 0.01 and a momentum of 0.9
- Loss function: As loss it's used the Binary Cross Entropy Loss since the task is binary classification
- Regularization: As regularizer it's implemented Early Stopping with a patience of 10.

3.4 Results and Experiments

In this section it's shown the results of the Networks in terms of loss and accuracy using graph metrices as node features.

Table 1: Accuracy Results - graph metrics as node features

NN Layers	60 sec	120 sec	180 sec	240 sec	300 sec
GCNConv	0.97	0.96	0.95	0.96	0.92
SAGEConv	0.98	0.97	0.96	0.97	0.97
GATConv	0.96	0.94	0.95	0.95	0.96

Looking at Table 1 and Table 2 it's clear that using graph metrices as node features we have quite interesting results. The networks show a good behaviour in the classification of graphs outputting

Table 2: Loss Results - graph metrics as node features

NN Layers	60 sec	120 sec	180 sec	240 sec	300 sec
GCNConv	0.0011	0.0012	0.0017	0.0016	0.0014
SAGEConv	0.0011	0.0012	0.0015	0.0016	0.0014
GATConv	0.0016	0.0025	0.0023	0.0025	0.0018

high accuracy and low loss. In order to make also experiments without using graph metrics as node features, we tried to fill them with a very small number to see whether the network is capable of learning. Using this setting resulted in an accuracy of 0.5 that means the network is not learning anything. This demonstrates clearly the fact that using graph metrics as node features helps the network to learn the graph structures. Since, for the Exlainability section, it's needed a model, it will be used the best one in terms of accuracy and loss and so the one with SAGEConv layers and 60 seconds of temporal sliding window. From Figure 13 to 16 are reported the graphs showing both loss and accuracy during training and validation phase. You can see tat during training we have a better loss and accuracy but also in validation they remains competitive.

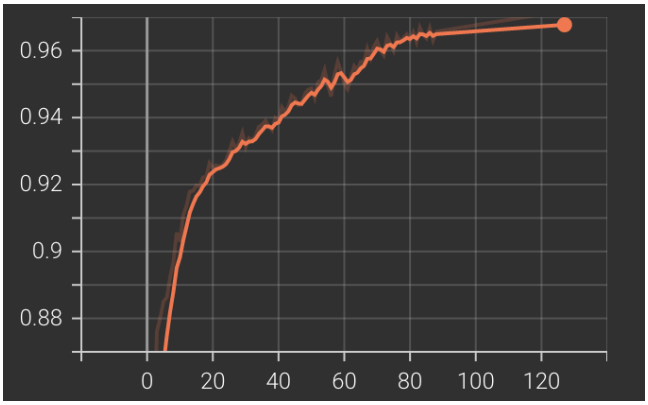

Figure 13: Training accuracy

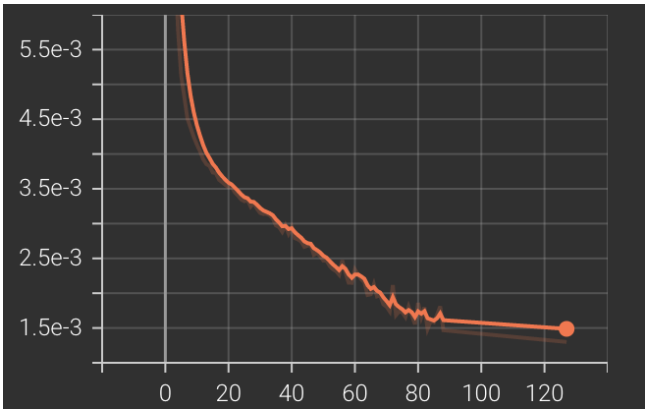

Figure 14: Training loss

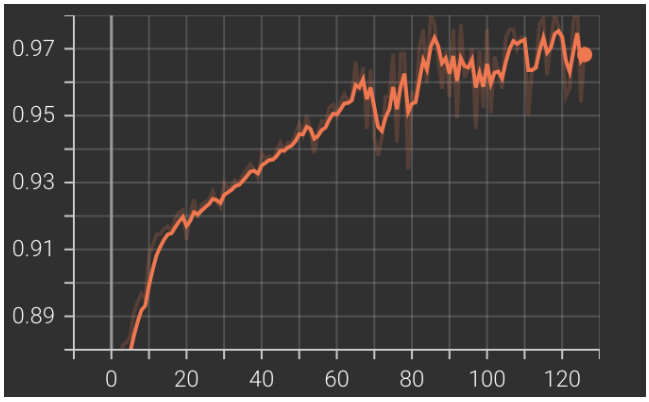

Figure 15: Validation accuracy

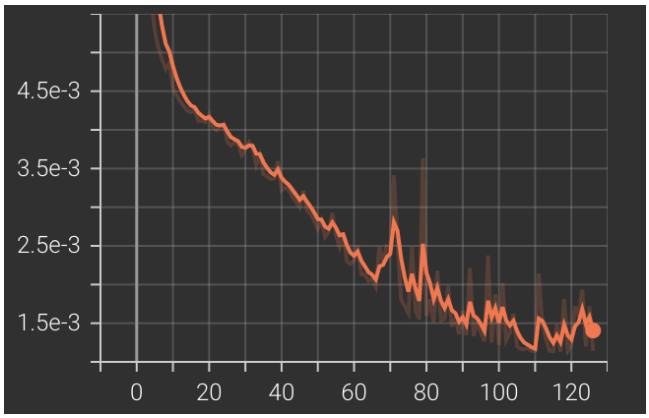

Figure 16: Validation loss

In Figure 17 is reported the confusion matrix in order to have a more compact visualization of the results.

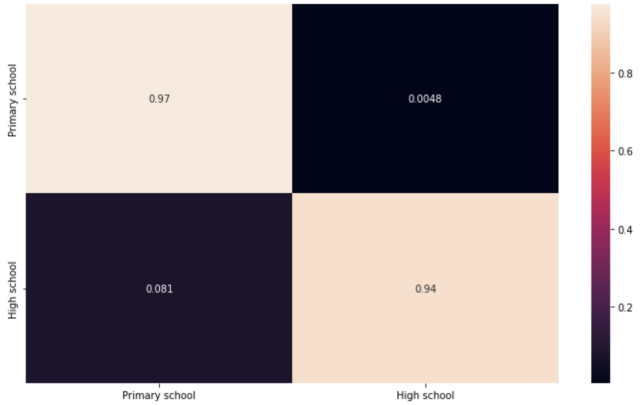

Figure 17: Confusion matrix

4 EXPLAINABILITY

This section is used to make some inference for what regards the results of the experiments. In particular all the three models have shown a quite high capacity in distinguishing between the two classes of graphs and so they seem to learn quite well. The question that now arise is: what are they learning? And which information are they using to distinguish between high school and primary school graphs? If we can explain this, we probably can understand the different types of relations that are present in different contexts like different types of school.

4.1 GNN Explainer

In order to study the obtained results, we made use of the GNN Explainer from [9] which is the first approach for providing interpretable explanations for predictions of any GNN-based model. GNNExplainer is capable of identifying compact subgraph structures and a small subset of node features that have a crucial role in prediction so, it is capable in reducing the redundant information in a graph which does not directly impact the decisions. In order to explain a graph we need to know which are the important features and structures in the graph that affect the decisions of the network. Whether a feature is crucial, then the prediction should be altered largely by removing it or replacing it with something else.

The objective for a GNNExplainer is to create a minimal graph that explains the decision for a node or for a graph. To achieve this goal the problem can be defined as finding a subgraph in the computation graph that minimizes the difference in the prediction scores using the whole computation graph and the minimal graph. In the paper the process is described as maximizing the mutual information (MI) between the minimal graph Gs and the computation graph G with this formulation:

$$max_{G_s} MI(Y, (G_s, X_s)) = H(Y) - H(Y|G = G_s, X = X_s) \quad (2)$$

The approach for the explanation is a mask approach where the learning of a minimal graph G_s is done by learning a mask for edges and a mask for features. So for each edge in the computation graph there will be a value in the edge mask which determines the importance of the edge and similarly for the features.

4.2 Edge mask

In order to understand the importance of the edges, the easiest way is to look at some graphs that have been classified as Primary school graphs and High school graphs.

Figure 18: High school graph 1

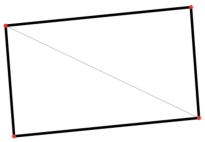

Figure 19: High school graph 2

Figure 20: High school graph 3

Figure 21: High school graph 4

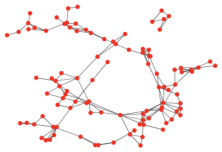

Figure 22: Primary school graph 1

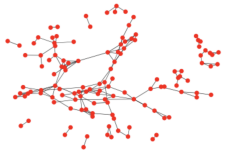

Figure 23: Primary school graph 2

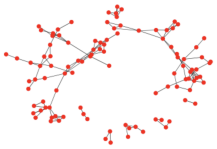

Figure 24: Primary school graph 3

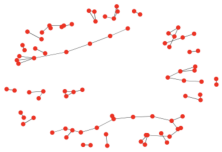

Figure 25: Primary school graph 4

Looking at all these results you can clearly see the differences in the two type of graphs also visually speaking. The GNNExplainer help us to understand the differences in the structures of both of them and we can try to inference something from this in the Results section.

4.3 Node feature mask

In order to better understand the importance of the features, we can use simply the results of the explainer.

Table 3: Node feature explanation results

Feature name	High school	Primary school
Node degree	0.24	0.67
Clustering	0.15	0.13
Degree centrality	0.52	0.16
Number of cliques	0.24	0.75
Core number	0.38	0.19
Square clustering	0.14	0.14
Avg neighbor degree	0.15	0.14
Degree assortativity	0.15	0.15
Avg clustering	0.60	0.26

As you can see from Table 3 this is how it looks like an average explanation for every feature. It shows that, averaging speaking, the network made use mostly of the features Degree centrality and Core number to understand that a graph belongs to the High school graphs; on the other hand the same network made use more of the features Node degree and Number of cliques to learn that a graph is a Primary school graph.

4.4 Results

Looking at the several graphs examples we can clearly understand why our network has a so high accuracy in predictions. In particular look at Figures from 18 to 25. You can see that we have much more complex (in terms of number) relations in primary school graphs. This can be explained thinking at the fact that probably in primary school the interactions between children are several but with a short duration. On the other hand in high school we have a smaller number of interactions that have a higher duration. The higher duration is represented in Figures from 18 to 21 by the fact that edges are ticker with respect to the ones from Figure 22 to 25.

REFERENCES

[1] Julie Fournet and Alain Barrat. 2014. Contact Patterns among High School Students. *PLoS ONE* 9, 9 (09 2014), e107878. https://doi.org/10.1371/journal.pone.0107878

[2] Valerio Gemmetto, Alain Barrat, and Ciro Cattuto. 2014. Mitigation of infectious disease at school: targeted class closure vs school closure. *BMC infectious diseases* 14, 1 (Dec. 2014), 695. https://doi.org/10.1186/PREACCEPT-6851518521414365

[3] Aric Hagberg, Pieter Swart, and Daniel S Chult. [n. d.]. Exploring network structure, dynamics, and function using networkx. ([n. d.]). https://www.osti.gov/biblio/960616

[4] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. *CoRR* abs/1706.02216 (2017). arXiv:1706.02216 http://arxiv.org/abs/1706.02216

[5] Thomas N. Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. *CoRR* abs/1609.02907 (2016). arXiv:1609.02907 http://arxiv.org/abs/1609.02907

[6] SocioPatterns. 2008. http://www.sociopatterns.org

[7] Juliette Stehlé, Nicolas Voirin, Alain Barrat, Ciro Cattuto, Lorenzo Isella, Jean-François Pinton, Marco Quaggiotto, Wouter Van den Broeck, Corinne Régis, Bruno Lina, and Philippe Vanhems. 2011. High-Resolution Measurements of Face-to-Face Contact Patterns in a Primary School. *PLOS ONE* 6, 8 (08 2011), e23176. https://doi.org/10.1371/journal.pone.0023176

[8] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[9] Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. 2019. GNN Explainer: A Tool for Post-hoc Explanation of Graph Neural Networks. *CoRR* abs/1903.03894 (2019). arXiv:1903.03894 http://arxiv.org/abs/1903.03894