University of Bergen
Department of Informatics

Solving Maximum Weighted Matching problem using Graph Neural Networks

Author: Nikita Zaicev
Supervisor: Fredrik Manne

UNIVERSITETET I BERGEN
Det matematisk-naturvitenskapelige fakultet

January, 2024

Abstract

In this work we tried to train a Graph Neural Network (GNN) to solve the Maximum Weighted Matching problem on graphs.

Acknowledgements

I want to thank Fredrikk Manne, Kenneth Langedal and Johannes Langguth for helping me with this work.

Nikita Zaicev
Tuesday 16th January, 2024

Contents

List of Figures

List of Tables

Listings

Chapter 1

Introduction

This chapter is dedicated to the general introduction of the problem at hand and previous work and research that is relevant for this project.

1.1 Goal

Machine Learning and Neural Networks have shown to be extremely potent and versatile in solving vast variety of problems across different fields. One research field that has been popular and challenging in the last few years is Combinatorial Optimization (CO). CO includes problems such as Maximal Indepndent Set (MIS) and Maximum Weighted Matching (MWM). Such problems can be viewed in the context of graphs. GNN is a subclass of Neural Networks designed specifically for solving problems related to graphs, but there are some problems that have not yet been solved efficiently with GNNs. The goal of this project is: "To find out whether a GNN can outperform greedy algorithms at solving MWM.

1.2 Background

Many researches have been done related to GNNs in the past years.

Lorenzo Brusca and Lars C. P. M. Quaedvlieg et al. [2] showed a self-training GNN for MIS.

Schuetz et. al. made an unsupervised GNN [4] for solving MIS and Angelini and Ricci-Tersenghi [1] compared its performance to greedy algorithms and reported some problems with GNNs performance.

The reason MIS solving is mentioned so much here is because the problems are to some degree related as algorithm probmels often are. There will be concrete examples in the Methodology and Data chapter. MIS is also more often a subject to research and researches on MWM are not that common. Bohao Wu and Lingli Li tried to solve MWM using deep reinforcement learning [6] and reporting that "Experimental results show that L2M outperforms state-of-the-art algorithms."

1.3 Project structure

The rest of this document has the following structure:

1. Methodology and Data:

   This Chapter will focus on how the research was done and also explain core concepts of glsnns and the general procedure of training a glsnn. Then the specifics of this case will be discussed and main challenges mentioned. The progress made step by step and the reasoning behind changes and choices made along the way will be shown. The chapter will also analyze the data used for training the model and evaluating results along with justification for the chosen data. The temporary results are also included in this Chapter but not the final results.

2. Results:

   This Chapter will focus on analyzing the final results and comparing them with the expactations.

3. Conclusion:

   Here the conclusion for this project will be drawn regarding whether the results gave any meaningfull insight and what future work can be done for improvements.

1.3.1 Listings

You can do listings, like in Listing 1.1

Listing 1.1: Look at this cool listing. Find the rest in Appendix A.1

```
$ java -jar myAwesomeCode.jar
```

You can also do language highlighting for instance with Golang: And in line 6 of Listing 1.2 you can see that we can ref to lines in listings.

Listing 1.2: Hello world in Golang

```go
1 package main
2
3 import "fmt"
4
5 func main() {
6     fmt.Println("hello world")
7 }
```

1.3.2 Figures

Example of a centred figure

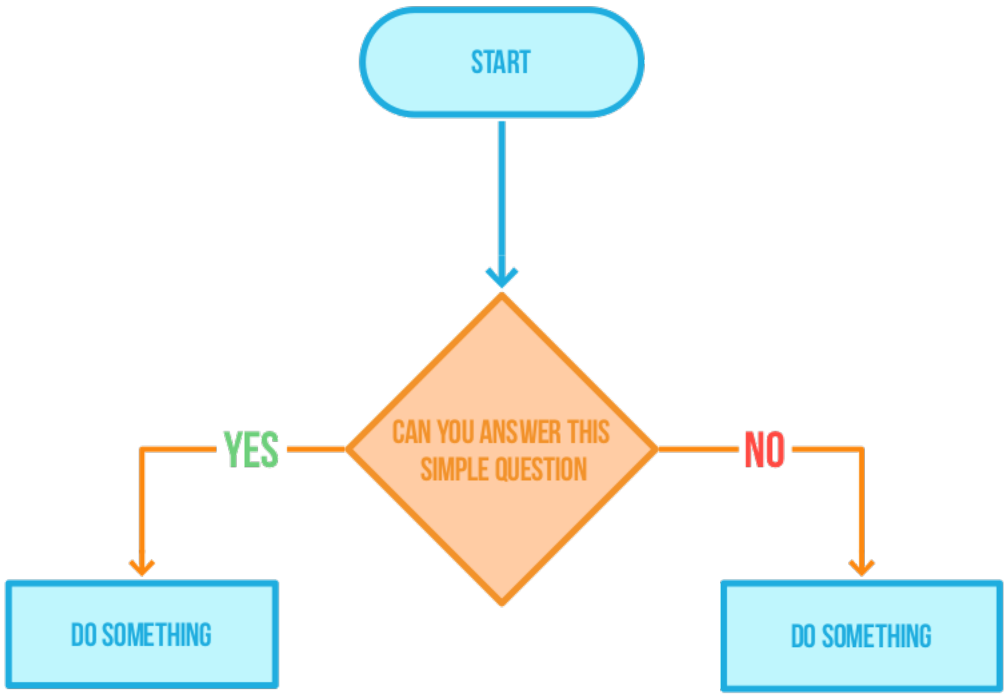

Figure 1.1: Caption for flowchart

Credit: Acme company makes everything `https://acme.com/`

1.3.3 Tables

We can also do tables. Protip: use `https://www.tablesgenerator.com/` for generating tables.

Table 1.1: Caption of table

Title1	Title2	Title3
data1	data2	data3

1.3.4 Git

The whole project can be seen here: `https://github.com/nikitazaicev/Master`

Chapter 2

Research Methodology and Data

This chapter focuses on describing which approaches were chosen for the task at hand and why, as well as shows how the work progressed.

2.1 Core Concepts

To reiterate the current problem is to find out if GNN can compete at solving the MWM problem compared to other options such as greedy algorithm.

Before properly describing the choices made for approaching this problem, it is worth going through basic concepts of GNNs and neural networks in general. It is common to think of neural networks as of a black box where you give some data to this box and it gives back an answer. In this case the data is a graph consisting of vertices connected by edges that have weights and the expected answer should be pairs of vertices that were matched together. This is however not as straight forward as the more common examples of classification where each data item needs to be classified.

Supervised learning has one side effect

2.2 Challenges

As mentioned the idea behind supervised learning is to give a model the correct answers so it can by trial and error learn from it. Theese answers are not included the initial datasets so the optimal solutions need to be calculated. Blossom algorithm implemented by Joris van Rantwijk in Python was used [5]. This does however point out an important weakness of the supervised approach. Obviously a model needs to be able to handle graphs of different sizes and it is the large ones that are most interesting. Finding an optimal MWM for the large graphs can be time consuming, at the same time a model need as much data as possible to learn leading to multiple large graphs consuming to much time. This poses a question whether it is possible for the model to learn on small and medium sized graphs that are not as time consuming and transfer learned patterns to solve larger graphs.

2.3 Data

The model should be capable of solving any graph relevant to MWM problem. A relevant graph can be difined by following characterisitcs Ideally the model should be able to handle any kind of a graph with weighted edges.

2.4 Expected Results

2.4.1 Accuracy and total weight

There are not that many researches specifically for MWM, but problems like MIS that are relatively close to MWM can indicate similar results for this case as well. As discussed in the Background section, there are researches that show that GNNs are capable of solving CO problems and beating greedy algorithms, while other rather indicate that improvements are still needed for it to be worth using. Therefore it is hard to forecast any results based on previous work. Nothing stands in the way of being optimistic however, additionally to the fact that greedy algorithm is relatively simple and Neural Network should be able the recognise a more complex pattern it can use to achieve better results. It is absolutely not expected for the model to be able to find optimal solution since any

Neural Network is a heuristic. The margin by which GNN can surpass the greedy solution is expected to be rather small, since from the data analysis it was observed that for the majority of graphs greedy algorithm preforms rather well with above 80% of the optimal possible weight.

2.4.2 Time

GNN model is a heuristic solver and gives an approximate answer. Therefore model should be noticeably faster than exact algorithm, otherwise it would not be worth it. The time a model takes to solve one instance of a problem should be closer to that of a greedy algorithm and probably slightly longer due to preproccessing required such as augmenting data with adittional features. Naturaly, time will also depend on the depth and the width of the network.

2.5 Model Architecture

2.6 Line Graph Approach

Line graph approach was the first attempt at using a simple GNN, which later turned out to be too time consuming for larger graphs to be worth further expriements. It did however give some usefull insight as well as a showed to be a proof of concept. In the context of graphs line graph is a complement of the original graph that turns each edge to a vertex and connects the vertices is they shared a vertex in the original graph.

Nouranizadeh et. al. showed anothing approach at solving MIS [3].

Examaple of a graph and its line graph convertion

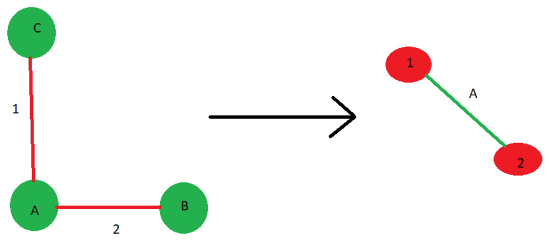

Figure 2.1: Original graph (left) and its line graph (right)

2.6.1 Progress

1. bare bones 2. class weights -¿ all or nothing 3. optimizer learning rate, network depth and width, 4. skip connections 5. extra features

2.7 Edge Classification Approach

A more natural way of approaching this problem is for each pair of vertices that are conneced by an edge, ask the model if given pair should be a part of the matching. In other words model is classifying an edge.

2.7.1 Progress

2.8 Result Validation

1. Time - how long an algorithm took to produce an answer.
2. Correctness - is the answer correct. In case of MWM the total weight aquired would be the measurement of how correct the solution is. It is unlikely that GNN can find an optimal solution for more complex problems so it is reasonable to look at how close GNN comes to the optimal solution.
3. Memory - how much memory is needed. However in this project there is less of a focus on memory

All the experiments have been done on the same machine with: 11th Gen Intel(R) Core(TM) i7-11700K 3.60GHz 8-core CPU, 16 GB RAM and NVIDIA RTX 3080Ti graphics card.

Chapter 3

Results

trained on a 1000 graphs for now

RESULTS ON ONE DATAEXAMPLE MNIST 1. 0.55 of OPT 2 After adding features: one by one -¿ 0.15 / 0.14 / 0.5 / 0.12 deg / weight diff / max / relative diff -¿ -¿ 0.75 / 0.85 / 0.91 / 0.92

Line graph vs normal vs normal with features (compared to standard greedy)

MNIST: 0.92 vs 0.77 vs 0.88 Normal greedy remainder LOW vs LOW vs LOW

GOOD CASE data/Pajek/GD98b/GD98b.mtx: 1.43 vs 0.00 vs 1.00 Normal greedy remainder LOW vs LOW vs LOW

NOT line NN graph MNIST weights normalized vs MNIST weights ones vs MNIST weights 1-2 vs MNIST weights 1-5 0.88 vs 1.00 vs 0.83 vs 0.60

3.1 Performance on unseen data

Algorithm	Time	Weight
GNN	1	6
Greedy	2	7

How well the final model solves the matching problem

Chapter 4

Conclusion

Results show that GNNs are capable of solving MWM, but the approaches presented in this work showed worse performance overall comapared to a simple standard greedy algorithm. The margin between the results was not big enough to indicate that the approach was completely sensless. Compared to the greedy algorithm model showed some level of "understanding" of the task at hand and in some special cases even manages to beat the greedy algortihm. It is worth mentioning that theese cases were manualy chosen because of they abuse the naiveness of the greedy algorithm, but it does still indicate that such cases do exist and therefore there is value in using GNN instead of a greedy algorithm.

4.1 Future work

The fact that GNN in this work underperformed does not neccessary mean that the GNNs are in general unfit for MWM problem. There several potential improvements at hand. A deeper or wider model can be trained, meaning adding more layers as well as neurons to each layer to potentialy improve models ability to recognise complex patterns at the cost of longer training and prediction times. However for this particular architecture adding more layers showed little to no effect. Theres also a variety of different architectures that can be tryed out. An unsupervised approach is a good potential candidate where precomputing the optimal solution would not be needed. Instead the model can try to find the best solution by incentivising it to get as high weight sum as possible, in a way resembeling a game.

4.2 Final words

Glossary

Artificial Intelligence Artificial Intelligence is a field of study regarding intelligence simulated by computers. Where intelligence is meant in context of human intelligence.

Git Git and GitHub is a Version Control System (VCS) for tracking changes in computer files and coordinating work on those files among multiple people.

Machine Learning Machine Learning studies algorithms that learn general patterns and make predictions based on some form of input. It is one form of Artificial Intelligence.

Neural Network Neural Network is one of many technologies used inMachine Learning.

List of Acronyms and Abbreviations

CO Combinatorial Optimization.

GNN Graph Neural Network.

MIS Maximal Indepndent Set.

MWM Maximum Weighted Matching.

VCS Version Control System.

Bibliography

[1] Maria Chiara Angelini and Federico Ricci-Tersenghi. Modern graph neural networks do worse than classical greedy algorithms in solving combinatorial optimization problems like maximum independent set. *Nature Machine Intelligence*, 5(1):29–31, December 2022. ISSN 2522-5839. doi: 10.1038/s42256-022-00589-y.
URL: `http://dx.doi.org/10.1038/s42256-022-00589-y`.

[2] Lorenzo Brusca, Lars C. P. M. Quaedvlieg, Stratis Skoulakis, Grigorios G Chrysos, and Volkan Cevher. Maximum independent set: Self-training through dynamic programming, 2023.

[3] Amirhossein Nouranizadeh, Mohammadjavad Matinkia, Mohammad Rahmati, and Reza Safabakhsh. Maximum entropy weighted independent set pooling for graph neural networks. *CoRR*, abs/2107.01410, 2021.
URL: `https://arxiv.org/abs/2107.01410`.

[4] Martin J. A. Schuetz, J. Kyle Brubaker, and Helmut G. Katzgraber. Combinatorial optimization with physics-inspired graph neural networks. *Nature Machine Intelligence*, 4(4):367–377, April 2022. ISSN 2522-5839. doi: 10.1038/s42256-022-00468-6.
URL: `http://dx.doi.org/10.1038/s42256-022-00468-6`.

[5] Joris van Rantwijk. Weighted maximum matching in general graphs, 2008.
URL: `https://github.com/ageneau/blossom/blob/master/python/mwmatching.py`.

[6] Bohao Wu and Lingli Li. Solving maximum weighted matching on large graphs with deep reinforcement learning. *Information Sciences*, 614:400–415, 2022. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2022.10.021.
URL: `https://www.sciencedirect.com/science/article/pii/S0020025522011410`.

Appendix A

Generated code from Protocol buffers

Listing A.1: Source code of something

```
System.out.println("Hello Mars");
```