electrek express

Creating Optimal Transportation Networks

Josh Kim 101153583

Rajessen Sanassy 101147410

Shrish Mohapatra 101141938

Carleton University

COMP 4107 Winter 2023

Matthew Holden

April 12th, 2023

Introduction

STATEMENT OF CONTRIBUTIONS

Each group member contributed a significant and equal amount to ensure the success of this project. The following table provides an overview of the each member's contributions:

Josh Kim	Simulation Logic, Genetic Algorithm Application, Model Training
Rajessen Sanassy	Model Training, Experiments, Data Sample Generation
Shrish Mohapatra	React Visualization, Flask API Server, Model Training

BACKGROUND

Subway systems are one of the most popular forms of transportation in large cities, as they are fast, efficient, and environmentally friendly. However, designing an optimal subway system that can deliver the maximum number of passengers while minimizing path cost and the number of lines is a complex optimization problem. Traditional optimization methods may not be sufficient to handle such complex problems, and therefore, new approaches are required.

Genetic algorithms can be useful for creating efficient subway networks since they can optimize complex problems that are difficult to solve using traditional methods [1]. For instance, a study conducted by Chung-Ang university involved applying genetic algorithms for energy optimizations of subway networks, with promising results when tested on Milan's subway system [2].

OBJECTIVE

The objective of the project is to create optimal design of subway lines by using a neural network trained with a genetic algorithm. The aim is to create a subway network that is efficient and maximizes passenger flow, while minimizing the number of lines used and total path cost. The project will involve generating sample data for passenger location frequency and station configurations, then using this data as input for the genetic neural network. The network will be trained to generate optimal subway lines that can deliver the maximum number of passengers while minimizing the number of missed passengers and total path cost. The project has the potential to improve the efficiency and convenience of public transportation systems, making them more attractive and accessible to commuters.

Methodology

DATASET

The first step for solving this problem was to develop a method for representing station and passenger data into a format which can be interpreted by a neural network (NN). This was done by creating a 2D matrix, where each row represented a station, and each column represented features about a station. These features include the station's shape, x & y coordinates, and number of passengers (triangles, squares, circles). A generator was developed to create data samples of various random station configurations, with seeding enabled to ensure consistency across different NN configurations.

APPROACH

A Tensorflow model was developed to take station data samples as an input and produce outputs corresponding to the edges the NN predicts. For example, if the NN output was [1, 0, ...] then the NN is predicting a line from station 1 and 2, but no line between stations 1 and 3. The Tensorflow model was composed of dense layers for each feature in the input.

After creating a Tensorflow model, a genetic algorithm was used to train the NN.

1. Create a fixed number of NN instances with different weights, referred to as *solutions*. Initially these are random, for subsequent generations the weights are inherited.
2. For each solution, generate predictions for a set of data inputs.
3. Determine the fitness for each solution based on a simulation which determines each passenger's path cost to reach their destination. A score is generated which accounts for passenger path cost and line length. More detail on this in the *Evaluation* section.
4. Select the top solutions based on their fitness for the next generation. A breeding process along with mutations is used to develop weights for children solutions.
5. Repeat 1-4 for a fixed number of generations.

BENEFITS & DRAWBACKS

One of the benefits of this approach is that procuring data samples becomes easier since ground truth values do not need to be provided as the NN learns through simulations instead. This is ideal for this problem since the "optimal subway configuration" may not be known beforehand, allowing the NN to explore different approaches to minimize network cost.

A drawback of this approach is the training time. Genetic algorithms can require many iterations before reaching optimal results, as opposed to the use of backpropagation which can efficiently adjust the NN weights to reach optimal solutions faster. However, with experiments conducted it was observed that near-optimal results were reached within 20 generations.

EVALUATION STRATEGY

A core component of genetic algorithms is the fitness function which is used to evaluate solutions. A good fitness function must measure how well a solution solves a problem, such that better solutions have better fitness score values [3]. Measuring the fitness of a subway network involves accounting for various factors including number of passengers delivered, number of lines used, length of lines, and passenger's path cost. The following formula was used to calculate the fitness score. Other fitness functions were tested during experiments (see *Results* for results of these experiments).

$$score = (10 \ \#passengers) + (-1 \cdot \#lines) + (-2 \cdot path\ length) + (-1 \cdot line\ length)$$

Results

QUALITATIVE RESULTS

A React application was developed to visualize the subway networks generated by the best performing neural network.

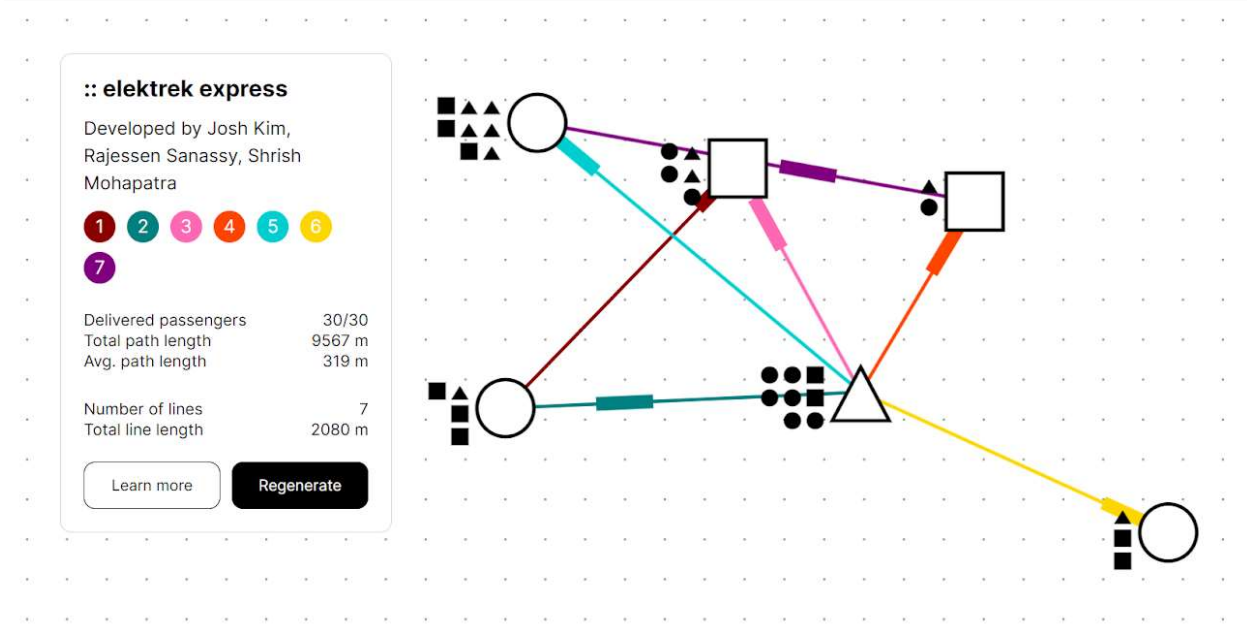

Figure 1: Screenshot from application

The qualitative results of our subway neural network are demonstrated through the frontend application developed with React. The network optimizes passenger delivery based on passenger frequency, displaying the most efficient routes for each passenger and minimizing their total travel distance. The effectiveness of the neural network in optimizing passenger delivery shown in Figure 1 is evident through the direct routes allocated to high-traffic areas such as the triangle passengers in the subway network. Each respective station is provided with a direct route to a central station where passengers can transfer to another station. Shapes of high demand at a station may also be given a direct route, if the current path cost is too high, resulting in a lower path cost and efficient transportation, reducing overcrowding and delays. This approach improves the overall passenger experience by reducing travel time and minimizing the need for transfers, making the subway system a more reliable and convenient transportation option. By using a neural network to optimize passenger delivery, the subway system becomes a reliable and convenient transportation option, improving the satisfaction of passengers.

QUANTITATIVE RESULTS

Experiments were conducted to observe the behavior of modifying the model's hyperparameters and its influence on the following factors: missed passengers, num lines, path length, and line length.

Experiment 1: Modifying Layers

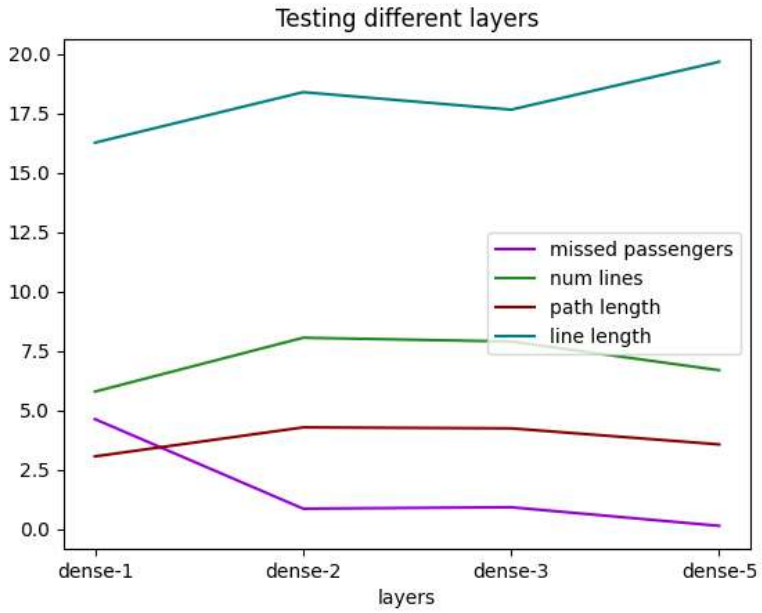

Figure 2: Number of dense layers experiment

The effectiveness of the neural network with a specific number of layers depends on the complexity of the problem being solved and the amount of data available for training. In some cases, a single-layer neural network may be sufficient for a simple problem; less number of stations with a small dataset. In the experiments we tested with 1,2,3 and 5 layers that included an architecture of 18 neurons in the first layer, 12 in each hidden layer, and 15 in the output layer. The results show that a 3-layer neural network is optimal for the dataset and problem because it strikes a balance between complexity and overfitting.

Experiment 2: Modifying Number of Training Samples

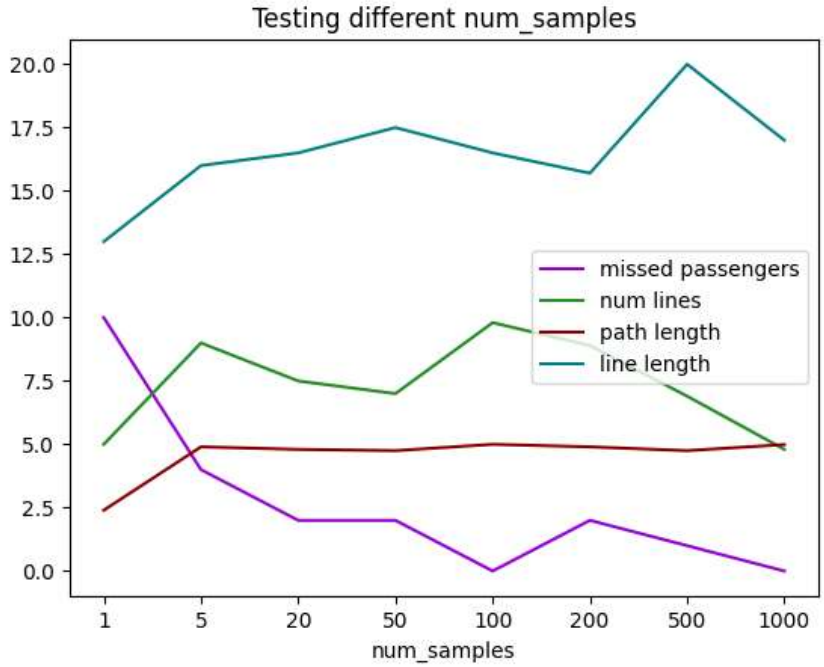

Figure 3: Number of samples experiment

The number of samples used to train a neural network for a subway system can also have a significant impact on the network's performance. Increasing the number of training samples can help to reduce overfitting by providing the model with more information about the patterns in the data. As seen in Figure 3, the models with lower sample size performed worse on the validation set as compared to the models with larger sample sizes.

Experiment 3: Modifying Scoring Functions

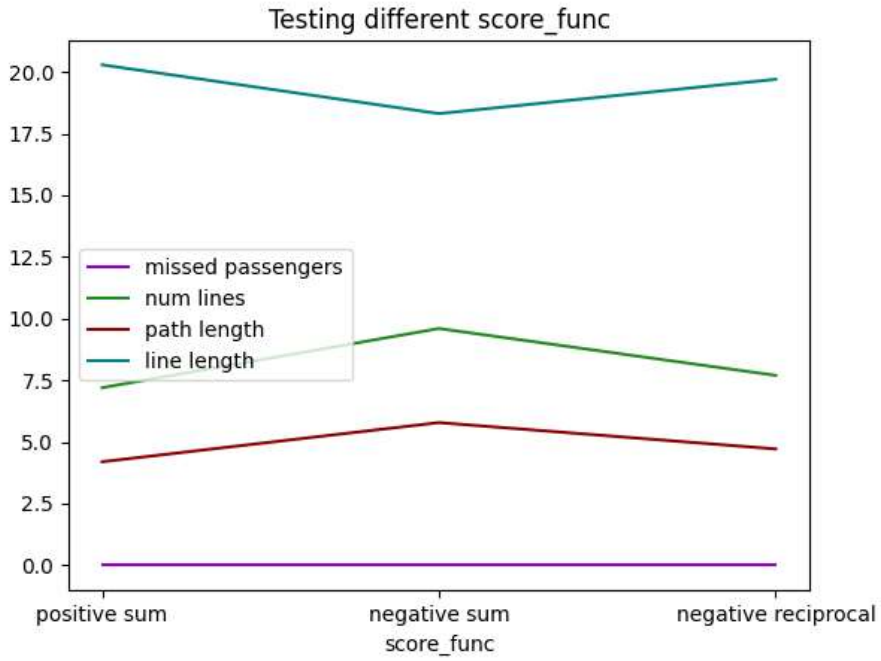

Figure 4: Scoring functions experiment

Determining fitness scores effectively is an integral component for a genetic algorithm. The following fitness functions were used for these experiments:

$$pos\_sum = 10 * passengers - lines - 2 * path\_length - line\_length$$
$$neg\_sum = -(10 * missed\_passengers + lines + 2 * path\_length + line\_length$$
$$neg\_reciprocal = \frac{1}{10 * missed\_passengers + lines + 2 * path\_length + line\_length}$$

Based on these results it seems that either of these fitness functions yields near-optimal results, with the $negative\_sum$ function performing slightly worse by creating more lines resulting in larger passenger path cost.

Discussion

ANALYSIS OF RESULTS

The neural network was designed with the goal of creating optimal lines in the system, taking into consideration factors such as the number of passengers delivered, the number of lines created, the length of the lines, and the path cost of each passenger. Among these factors, the number of passengers delivered was identified as the most important. As a result, the model has been successful in creating lines which deliver all passengers. However, there have been instances where unnecessary lines were created (such as the square to square line in figure 1) and less optimal routes were taken due to the

prioritization of passenger count over other factors. Moreover, based on the experiments conducted, the best performing model had the following hyperparameters: 3 dense layers with 18, 12, 15 neurons, trained with 500 samples on 50 generations, and used the *positive_sum* fitness function.

LIMITATIONS

The current approach has a limitation in that the input size is fixed, currently supporting 6 stations in the system. This constraint may impact the network's ability to create optimal lines for any other number of stations. As mentioned earlier, factors such as the number of lines and path costs of each passenger can counteract each other. For example, when passengers have more options (i.e., more lines) to choose from to reach their respective goals, they may be able to select shorter paths. This creates biases in the fitness function, potentially favoring certain aspects more than others, which could be a contributing factor to the network's suboptimal performance. Addressing this limitation and fine-tuning the fitness function to better balance these factors could lead to improved results in the future.

IMPLICATIONS

The project has shown that employing a combination of genetic algorithms and neural networks can be a promising approach for designing optimal subway networks which maximize passenger delivery and minimize network cost. This approach is also cost effective due to the simplicity in acquiring data samples for training, as opposed to traditional reinforcement learning which requires providing ground truth values which the NN can learn from. Moreover, due to the exploratory nature of genetic algorithms, this approach can provide ways to discover new configurations solely based on the passenger data.

FUTURE IMPROVEMENT

In order to further improve the approach, one could consider incorporating reinforcement learning principles into the training of the neural network after the initial training using genetic algorithms. Consider a model that has been trained with a fixed number of generations using the approach specified in the project. Afterwards, transportation engineers could review a subset of subway line predictions from the model, and provide alternative configurations which are more efficient (ie. use less lines but satisfy more passengers). These alternative line configurations could then be used as ground truth values which the model could learn from to fine-tune its weights and optimize the subway network design even further.

Another improvement could be to use a graph neural network (GNN) instead of the feed forward neural network used in the project. Graph neural networks provide a better method of encoding information regarding nodes and edges, which is more suitable for transportation problems (ie. stations considered as nodes, lines between stations considered as edges) [4]. The approach could leverage GNN to predict more properties about lines, for instance creating lines spanning multiple stations and sharing the same subway train.

Development Notes

The project's training process was implemented using Python, Tensorflow, and a genetic algorithm library known as PyGAD [5]. The model is accessible via a REST API server developed with Flask. This server is used by the visualization application developed using React. The source is available on a private Github repository, feel free to contact the authors of this report for access.

References

[1] M. Jiang, “Improved Genetic Algorithm Based Passenger Flow Control in Subway,” *Lecture Notes in Electrical Engineering*, pp. 701–708, 2018, doi: https://doi.org/10.1007/978-981-10-7989-4_71.

[2] M. Brenna, F. Foiadelli, and M. Longo, “Application of Genetic Algorithms for Driverless Subway Train Energy Optimization,” *International Journal of Vehicular Technology*, vol. 2016, Feb. 2016, doi: https://doi.org/10.1155/2016/8073523.

[3] V. Mallawaarachchi, “How to define a Fitness Function in a Genetic Algorithm?,” *Medium*, Nov. 10, 2017. https://towardsdatascience.com/how-to-define-a-fitness-function-in-a-genetic-algorithm-be572b9ea3b4

[4] J. Zhou *et al.*, “Graph neural networks: A review of methods and applications,” *AI Open*, vol. 1, no. 8, pp. 57–81, 2020, doi: https://doi.org/10.1016/j.aiopen.2021.01.001.

[5] A. Gad, “PyGAD - Python Genetic Algorithm! — PyGAD 2.16.1 documentation,” *pygad.readthedocs.io*, 2020. https://pygad.readthedocs.io/en/latest/