PPOL 563 - Data Visualization

Vince Egalla

2021/12/16

Final Report

## Executive Summary

Since the beginning of this semester, I’ve been building towards a graph visualization using D3.js. The culmination of this work is a scrolling website, featuring a force-simulated graph, various graph representations of the DC Metro system, and graph machine learning predictions. Prior to this class, my only experience with graphs came from my summer internship; however, developing this web page increased both my understanding of this data structure and how to use D3.js for customized visualization. I realized in the end that public transportation systems aren’t an ideal method for graph machine learning, which rendered the goals I initially set at the beginning of the semester unfeasible. I haven’t predicted where a new DC metro station should be located with these methods, but I hope to have built a useful demonstration of this graphs and graph machine learning.

<https://vegalla.github.io/ppol_563_final/>

## Data

The primary data of this project is a manual creation of graph records from the WMATA metro map. This data is modelled after the raw data used in a graph learning demonstration by Junya (openjy) on Kaggle.com

Example:

|  |  |  |  |
| --- | --- | --- | --- |
| Start | Stop | Line | Color |
| Shady Grove | Rockville | 1 | red |
| Rockville | Twinbrook | 1 | red |
| Twinbrook | White Flint | 1 | red |
| … | … | … | … |

The records reflect connections between two stations, along an arbitrarily assigned line number and the station’s line color. Normally, a graph record explicitly mentions a relationship, but all relationships are consistent for this graph of “X is connected to Y along this line color.”

After review, I noticed that I’m missing one station from my visualization, Cleveland Park along the red line and between Van Ness and Woodley Park. This oversight highlights the primary issue of user-created data. Given the impending deadline, I have chosen not to make a correction to this issue.

For the geographic representation of the graph, I use coordinates from the Intermodal Passenger Connectivity Database (IPCD), published by the US Department of Transportation (DOT) Bureau of Transportation Statistics (BTS).

For the WMATA map representation of the graph, I use pixel coordinates that I manually collected from reviewing the image.

For both representations, I had scaled the values to between -1 and 1 for convenience.

Data: <https://github.com/vegalla/vegalla.github.io/tree/master/ppol_563_final/data/raw>

## Methods

For visualization, I am using D3.js to draw the network graph, the loss chart, and animations between text sections of the web page. D3.js handles the initialization of the graphs, movement between circles (nodes) and lines (edges), as well as color changes. Other functions, such as the scrolling, are implemented through JavaScript, particularly functions written by Cuthbert Chow. The webpage and formatting are implemented through HTML and CSS.

Visualization code: <https://github.com/vegalla/vegalla.github.io/blob/master/ppol_563_final/sections.js>

For data processing and machine learning predictions, I use Python and the packages: numpy, pandas, NetworkX, DGL, and torch. NetworkX converts the raw data into a graph object, consisting of a list of nodes and a list of edges (source/target). DGL handles the machine learning using a graph convolutional neural network with a pytorch backend. Semi-supervised machine learning is used to predict nodes’ station color while relying on knowing only a few of the nodes’ true colors. Of all 89 the nodes, only 12 are provided labels. The predictions currently use a simple neural network with two layers, as more sophisticated examples of deeper learning have depreciated since their publishing. For visualization, I export the graph into a JSON object and the prediction losses into a CSV.

Jupyter notebook: <https://github.com/vegalla/vegalla.github.io/tree/master/ppol_563_final/data_processing>

## Highlights

The primary highlights of this project are the personal gains I’ve made in terms of D3.js visualization and graph machine learning. The web page is built around teaching the fundamental idea of graph machine learning. It’s narrative structure generally follows these questions:

* What is a graph?
* How can I analyze a graph?
* Is this application of prediction (upon transportation systems) useful?

The visualizations I added to the web page enhance a reader’s experience and maintains engagement with animated graphics across scrolling text, a shift from abstract to familiar concepts, and optional selections to read more when necessary.

## Updates compared to the ML Visualization Assignment

* Supplemental information on graph theory and neural networks have been moved to a selectable pop-up window. This improves the page structure by reducing text sections without animation.
* Loss chart has been recreated with D3.js. Previously, this was an embedded static image created with matplotlib. Creating this chart in D3.js allowed me to sync the machine learning epochs between both visualizations.
* Final graph layout now reflects the WMATA map closely. In response to Gloria’s feedback, I added a final layout that should allow readers to better compare a familiar metro map with the graph representation of it.
* Minor tweaks on visualization sizes and text formats.

## Final Remarks

I could have spent more time writing efficient code, particularly the draw5 function in sections.js. However, I only started learning JavaScript in this project and will improve efficiency as I learn more. Lastly, I realized after I animated the synchronized loss chart that it does strain the eyes to keep track of both animated visuals. In the future, I would take more careful consideration of overloading animated information.