Graph Embeddings for Street Network Analysis

Patrick DeMichele, Pablo Santos, and Isaac Scheinfeld

December 2019

Abstract

The field of street network analysis has not yet benefited from much of the recent work on graph machine learning. We extend a framework for large-scale analysis of OpenStreetMap data with recently released traffic data from Uber, and apply node embedding techniques to study the street networks of New York City and San Francisco. We present results for unsupervised clustering-based role discovery and supervised models for predicting speeds and a proxy for vulnerability to congestion.

1 Introduction

In this paper, we model the street networks of New York and San Francisco with data from OpenStreetMap (OSM) [19], an open source collection of real world street data. This allowed us to model street networks as graphs with nodes representing street intersections and edges representing streets. Using features from OSM, we construct embedding vectors for streets which can then be used in a variety of unsupervised and supervised models.

We examine the results of unsupervised role discovery on these embeddings as a first approach to interpreting what information they encode. Local, recursively generated features seem to cluster according to road type, while random-walk based node2vec features are similar within neighborhoods.

Modeling traffic flow in cities such as New York and San Francisco requires congestion-aware models since both cities suffer from frequent traffic jams. For each of these two networks, we experiment with different models to predict the mean speed – provided by Uber Movement [2] in a recently released dataset – as well as the "congestion" (according to a metric we define) of a given street. The relative success of these models demonstrates the power of graphical properties alone in explaining congestion and mean speed, indeed many of the models we train do not use any features besides network topology.

2 Literature Review

2.1 Street Network Analysis

There is extensive literature related to modeling street networks [18], with approaches inspired by fields as varied as statistical physics [12] and economics [17]. Much of the traffic modeling literature is concerned with traffic simulation at different scales of detail. Microscopic queue-based models such as SUMO [15] model individual vehicles based on demand and decision models. Such models require extensive data both for configuring and validation, even requiring traffic light sequences for accurate modeling of intersections. Macroscopic traffic models, which model traffic as flows rather than discrete vehicles, require less extensive configuration data but validating them is still a challenge. [14]

These challenges are compounded since most street network analysis has suffered from small sample sizes and data robustness constraints. Until recently there was no standardized data source and approach for analyzing street networks at scale. However, in 2018, Geoff Boeing conducted a systematic analysis of street networks across the United States using OpenStreetMap (OSM) [19] data. [6] This survey validated the OSMnx Python library which [7] was developed to handle conversion and preprocessing of OSM data. We build on this data pipeline in our work.

2.2 Traffic Congestion

One prominent problem in street network analysis is modeling traffic congestion. This has applications from government transit funding to route planning. As with street network analysis more broadly, studies of congestion tend to be situation-specific. For example, models exist for congestion due to people looking for curbside parking [3], commuters' decisions during rush hour [4], and congestion's effect on the climate [5]. All of these models are highly specialized to their domains.

In recent years, there has been work to address traffic congestion using machine learning models. Graph convolutional networks (GCNs) have recently been used to tackle the problem of modeling complex spatial and temporal dependencies of street networks. This is significant because previous statistical methods attempting to model street networks were linear and failed to model these more complicated dependencies. Yu Yol Shin et al. [21] present such a GCN framework that is applied to street networks. This framework considers different street attributes such as distance, speed limit, street angle, etc. to predict the actual speed at which cars travel in these streets.

In a working paper from Park et al. published in November of 2019, the authors use a Graph Attention Network (GAT) trained on graph representing spatio-temporal data of a traffic network in order to predict traffic speed in the short term. The data used for predictions is contemporaneous spatio-temporal data (includes both graph structure and recent speed measurements measurements) and the goal of the predictor is to output accurate estimates of road speeds in the near future [20]. Park et al's problem setup is nearly identical to that of a paper by Chen et al also released this past November which tests a wider variety of different deep learning architectures to tackle the problem [9].

3 Experiments

We describe here a series of experiments intended to validate a machine-learning first, embedding based approach to street network analysis and traffic modeling. We combine the OSM pipeline of Boeing 2018 [6] with features derived from the Uber Movement dataset [2] for our supervised models.

As a foundation, we explore a variety of graph-based techniques for creating road embeddings. Working with the line graphs of street networks, we use recursive features [13] to build up street representations that take into account local structural features as well as features available in OSM data. We also explore node2vec embeddings which have the potential to encode features of the graph which are less local [10].

We then leverage these embeddings to experiment with unsupervised street role discovery, as well as supervised problems related to traffic and congestion modeling.

3.1 Data

For street network data, we follow the data methodology of Boeing 2018 [6], leveraging OSMnx to download drivable street networks for San Francisco (including 10km of surrounding land) and New York City. The topology of these street networks is then simplified by removing nodes not at intersections or dead-ends, as OSM data has nodes along roads as well as at intersections. The original OSM data is loaded with a 0.5km buffer around each region, and nodes and edges outside this buffer are removed after simplification.

These networks are represented as directed graphs where edges represent roads and nodes represent intersections. Again, these graphs are directed, so a two way road will have two edge entries between the intersections they run through. After simplifying both graphs, we were left with a directed graph for New York with 55,356 nodes and 141,233 edges, or 141,233 roads and 55,356 intersections. For San Francisco, we have 32,074 nodes and 85,223 edges,

or 85,223 roads and 32,074 intersections. See Figure 1 for plots of the in and out degrees of both cities. As expected, the plots look similar for the in and out degrees as roads are typically two ways.

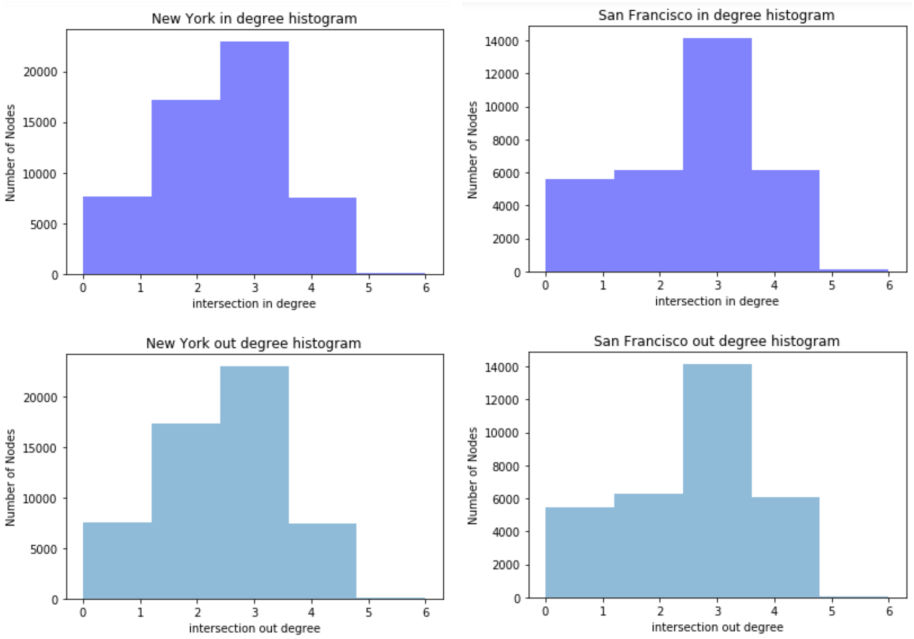


Figure 1: In and out degree histograms for New York and San Francisco

For traffic data, we worked with the recently released Uber Movement speed data [2], specifically that from Q2 of 2019, with statistics calculated for each hour of the day. The quarterly Uber data contains "the average, standard deviation, 50th percentile, and 85th percentile speeds aggregated by hour of day across all days in the specified quarter," where data is only provided for road segments with sufficient Uber traffic during the period. [2]

In Figure 2, we can see the mean speed profiles of 50 random streets in New York and of 50 random streets in San Francisco respectively.

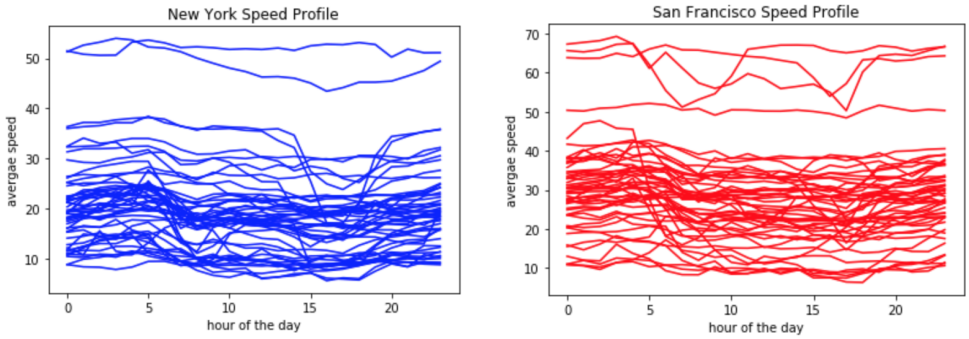
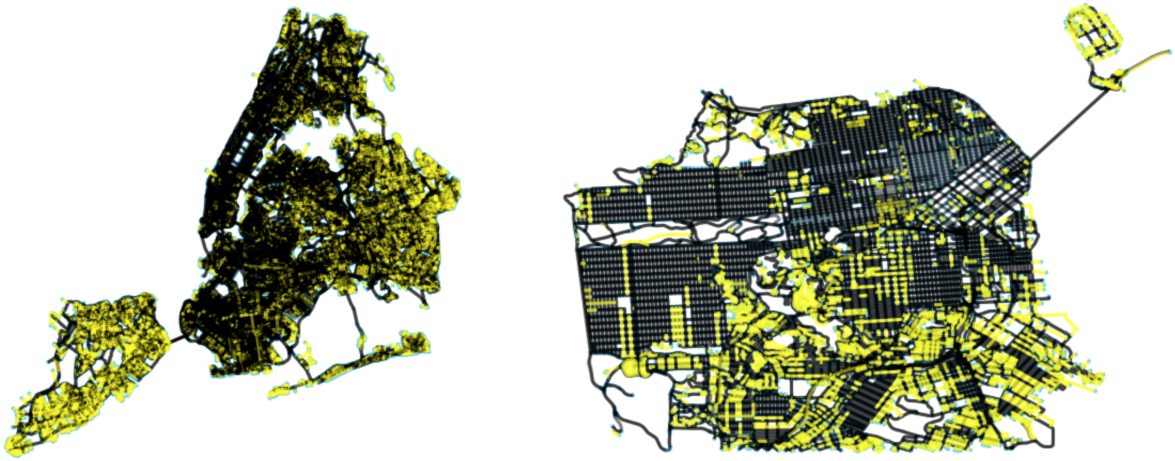


Figure 2: Average Speed Profiles for New York and San Francisco

As expected, we can see some drop offs in mean speeds during typical morning commute hours and afternoon commute hours.

Combining these datasets is possible since Uber Movement data is keyed to OSM data. However, due to the constantly changing nature of OSM data (as the data is improved, node and edge IDs change) and differences in how Uber and OSM define road segments (called Ways in OSM) we can only recover speed data for 70% of streets in both datasets. This is visualized in Figure 3.



(a) New York

(b) San Francisco

Figure 3: Black streets have Uber speed data, yellow streets do not.

For all of our analysis, we work with the line graph G_L of the street network G, where each node in the line graph corresponds to a street in the street network and streets are (directionally) connected if you can drive from one to the other at some intersection. This allows us to use node-centric models while preserving street features as node features in the line graph.

3.2 Embeddings

3.2.1 Basic Features

The simplest feature vectors for nodes in our line graph are composed of features already present in OSM data for individual streets and local graph properties of the node. Although many such features exist in the standard, only a few are available for most streets in the network. Thus, we restrict ourselves to using street length as our only OSM feature, as well

as in and out degrees for each node. Thus, our most basic embedding is a vector

$$e_v = [speed, d_{in}, d_{out}] \in \mathbb{R}^3, v \in G_L$$

.

3.2.2 Recursive Features

Next, we extend these features using the recursive method presented in the RolX role discovery algorithm. [13]. A street represented by a node v in the line graph with embedding e_v can recursively be assigned embedding

$$\left[e_v \,||\, \frac{1}{|N(v)|} \sum_{w \in N_{\text{in}}(v)} e_w \,||\, \frac{1}{|N(v)|} \sum_{w \in N_{\text{out}}(v)} e_w \right]$$

where $N_{\text{in}}(v)$ and $N_{\text{out}}(v)$ are the in and out neighborhoods of v. Note that this is a natural generalization from the approach defined for undirected graphs in [13]. Mean aggregation performed well on downstream tasks. We generated embeddings with one and two recursive steps, giving

$$\begin{aligned} e_v &\in \mathbb{R}^3 \\ R^1(e_v) &\in \mathbb{R}^9 \\ R^2(e_v) &\in \mathbb{R}^{27} \end{aligned}$$

3.2.3 Node2Vec

Alternatively, we used Node2vec as developed by Grover and Leskovec to generate street embeddings directly from the graph topology [11]. Node2Vec takes in a graph $G = (V, E)$ (optionally, weighted edges, but we ignore this functionality for our purposes), a dimension d, and returns an embedding $f : V \to \mathbb{R}^d$. The goal of Node2Vec is to optimize

$$\max_f \sum_{u \in V} \log Pr(N_S(u)|f(u))$$

where $N_S(u)$ is a *network neighborhood* of u defined through some sampling strategy – a set of nodes which generalizes the idea of a node's neighbors. Intuitively, Node2Vec searches for an embedding that maximizes the likelihood of the true graph network neighborhoods assuming a probability function $Pr(N_S(u)|f(u))$ of an embedded node having a certain embedded network neighborhood. This probability $Pr(N_S(u)|f(u))$ is modeled (via an assumption of conditional independence) as

$$Pr(N_S(u))|f(u) = \prod_{n_i \in N_S(u)} Pr(n_i|f(u))$$

and each $Pr(n_i|f(u))$ is modeled by

$$Pr(n_i|f(u)) = \frac{\exp(f(n_i) \cdot f(u))}{\sum_{v \in N_S(u)} f(v) \cdot f(u))}$$

which reflects that $Pr(n_i|f(u))$ should be smaller when $f(n_i), f(u)$ are closer to orthogonal in $\mathbb{R}^d$. Node2Vec creates $N_S(u), u \in V$ via a series of biased random walks that combine elements of BFS and DFS, then performs stochastic gradient descent on the objective function to return the embeddings f. Note that Node2Vec does not take in precomputed node features, so our use of Node2Vec in this project is unsupervised, and does not consider any data about street networks besides their graphical properties.

Our Node2Vec embeddings were generated using the Node2Vec repo in the examples directory of SNAP. We generated embeddings with 24 and 48 dimensions, using the defaults for all other parameters.

$$f_{24}(v) \in \mathbb{R}^{24}$$
$$f_{48}(v) \in \mathbb{R}^{48}$$

3.3 Unsupervised Role Discovery

The first task to which we applied our node embeddings was role discovery through unsupervised clustering. Although there is extensive research on unsupervised clustering using embeddings [22, 13], we picked a simple approach which could be applied across different types of embeddings. After standardizing each feature of our embeddings to zero mean, unit variance, we clustered the resulting vectors using K-means to find the three most significant clusters. Standardization is necessary since K-means assumes the embedding space is invariant under rotation and translation. [16]

Interpretation of unsupervised models is always a challenge, but here we were able to leverage the roughly planar nature of street networks to allow us to visualize the resulting roles directly, see Figure 4. The most visually interpretable results arose from the one and two-level recursive embeddings $R^1(e_v)$ and $R^2(e_v)$, which seem to group roads into highways and smaller/larger city streets. Clustering node2vec embeddings, while not especially interpretable, shows how node2vec gives geographically close streets similar embeddings resulting in geographically clustered streets. This matches with the intuition that random walks starting in the same neighborhood have roughly the same distribution.

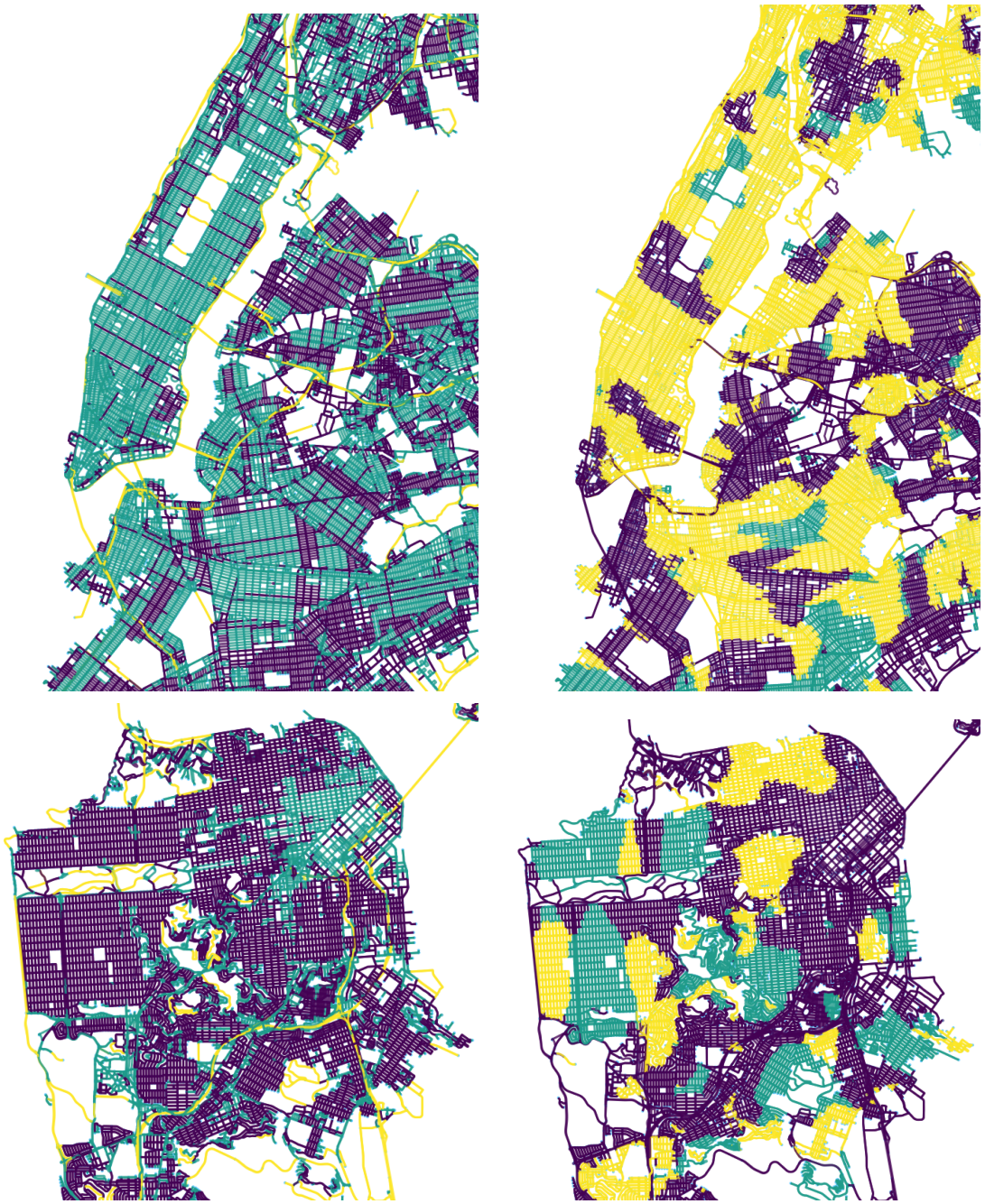


Figure 4: Unsupervised clustering using recursive features and node2vec. (Left) K-Means with 3 clusters on $R^1(e_v) \in \mathbb{R}^9$ where $e_v = [length, d_{in}, d_{out}]$. (Right) K-Means with 3 clusters on node2vec embeddings $f(v) \in \mathbb{R}^{24}$ generated from the graph topology.

3.4 Supervised Traffic Modeling

3.4.1 Mean Speed Prediction

Our primary objective was developing a machine learning approach to traffic modeling. The simplest problem we could address in this domain is predicting mean speeds for each street at a specific hour of day. This can be posed as a supervised problem where streets are represented by the feature vectors described in Section 3.2 and the target speeds are modeled using a multi-variate predictor. We tested linear models as well as random forests as a robust non-linear predictor [8] on each of the embeddings as well as combinations thereof.

Each model was trained on mean speed data for the hour 12-1, for Q2 2019. The data was split with 80% of data available for the given city for training, with the remaining 20% withheld for testing. The optimal maximum depth of the random forest was determined through cross-validation for each embedding type (up to the embedding size), and each forest was an ensemble of 100 trees. See Figure 5 for the resulting test R^2 values for each combination of embedding, model, and city.

Embedding	OLS	Random Forest
$e_v \in \mathbb{R}^3$	0.267 / 0.059	0.350 / 0.200
$R^1(e_v) \in \mathbb{R}^9$	0.310 / 0.114	0.475 / 0.327
$R^2(e_v) \in \mathbb{R}^{27}$	0.320 / 0.180	0.490 / 0.385
$f_{24}(v) \in \mathbb{R}^{24}$	0.027 / 0.032	0.493 / 0.437
$f_{48}(v) \in \mathbb{R}^{48}$	0.072 / 0.061	0.491 / 0.439
$e_v \|\| f_{24}(v) \in \mathbb{R}^{27}$	0.262 / 0.089	0.553 / 0.488
$R^1(e_v) \|\| f_{24}(v) \in \mathbb{R}^{33}$	0.332 / 0.164	0.599 / **0.514**
$R^2(e_v) \|\| f_{24}(v) \in \mathbb{R}^{51}$	0.357 / 0.204	**0.604** / 0.494

Figure 5: Test R^2 for mean-speed prediction, New York City / San Francisco, with the best performing models bolded.

We see that the linear model is barely able to predict speed from the non-linear node2vec embeddings, and achieves only poor performance on even the concatenated embeddings. However, the non-linear random forest is able to leverage the node2vec embeddings and achieves surprisingly strong results given that so many variables predictive of speed (such as demand, number of lanes, road grade, etc.) are not directly available to the model.

3.5 Congestion Prediction

We apply the same models from the previous section to learn a metric other than mean speed which is more relevant to congestion. Metrics for congestion in traffic models in academic papers are varied and often difficult to generate easily from the data as provided by Uber. We designed our metric to approximate the "Travel Time Index" – the ratio of actual time traveled versus time traveled under ideal conditions – presented as a metric of congestion in a report from the US Department of Transportation. [1]

For a street s, we define the metric m_s by

$$m_s = \frac{\text{speed85}_{s,12}}{\min_{0 \le h \le 23} \text{mean speed}_{s,h}}$$

where $\text{speed85}_{s,12}$ is the 85% of speed measurements taken on s between noon and 1 p.m. over all days in the period (this was included in the Uber data to serve as a reasonable estimate for free flow speed on s), and mean $\text{speed}_{s,h}$ is the mean speed of measurements on s taken between hour h and $h+1$ over the period. This ratio roughly reflects the ratio between the ideal speed of vehicles on s to its lowest speed under regular congestion conditions.

Performance on this problem is slightly worse across the board, see Figure 6. However, the peak performance achieved is still surprising since it is trained on just the node2vec embeddings, which only have access to the graph topology. Thus, our model is able to explain significant variance in street congestion directly from the directed graph structure of a city's streets.

Embedding	OLS	Random Forest
$e_v \in \mathbb{R}^3$	0.014 / 0.04	0.030 / 0.086
$R^1(e_v) \in \mathbb{R}^9$	0.027 / 0.056	0.066 / 0.130
$R^2(e_v) \in \mathbb{R}^{27}$	0.048 / 0.057	0.109 / 0.172
$f_{24}(v) \in \mathbb{R}^{24}$	0.047 / 0.088	**0.375** / **0.388**
$f_{48}(v) \in \mathbb{R}^{48}$	0.072 / 0.081	0.368 / 0.368
$e_v \| \| f_{24}(v) \in \mathbb{R}^{27}$	0.051 / 0.109	0.357 / 0.371
$R^1(e_v) \| \| f_{24}(v) \in \mathbb{R}^{33}$	0.061 / 0.112	0.346 / 0.356
$R^2(e_v) \| \| f_{24}(v) \in \mathbb{R}^{51}$	0.071 / 0.106	0.340 / 0.365

Figure 6: Test R^2 for predicting congestion m_s, New York City / San Francisco, with the best performing models for each city bolded.

3.6 Graph Neural Network

For our final experiment, we attempted to set up a Graph Neural Network Using the GNNStack code that was presented in class. We modified it by removing the final log softmax activation to turn the model into a scalar predictor instead of a classifier, changing the loss function to mean squared error correspondingly. We then attempted to use this model to predict the mean speed for a street in our SF and NY line graphs, but our model constantly defaulted to predict about the overall mean street speed of the entire city for every street. After a hyper parameter search (in particular, on choice of optimizer, learning rate curve, and activation functions), we did not see any improvements and decided to abandon this approach and focus on our other models.

4 Conclusion

Using local network features, recursive feature generation, and node2vec, we were able to create different embeddings for the nodes/streets in our networks. With these embeddings, we explored unsupervised clustering for role discovery, and trained supervised models to predict the mean speed and a congestion metric of individual streets. Overall, a big take-away from these experiments is that graph structural properties alone can explain a lot of the variation in physical attributes of street networks, in this case mean speed and the congestion metric of a street. This demonstrates the usefulness of graphical machine learning algorithms in predicting important properties of networks, even without much data besides graph structure.

5 Contributions

All three of us contributed evenly to the project.

6 Acknowledgements

Data retrieved from Uber Movement, (c) 2019 Uber Technologies, Inc., https://movement.uber.com.

References

[1] Traffic Congestion and Reliability: Trends and Advanced Strategies for Congestion Mitigation.

[2] Uber Movement: lets find smarter ways forward, together.

[3] Richard Arnott and Eren Inci. An integrated model of downtown parking and traffic congestion. *Journal of Urban Economics*, 60(3):418–442, 2006.

[4] Richard Arnott and Kenneth Small. The economics of traffic congestion. *American scientist*, 82(5):446–455, 1994.

[5] Matthew Barth and Kanok Boriboonsomsin. Real-world carbon dioxide impacts of traffic congestion. *Transportation Research Record*, 2058(1):163–171, 2008.

[6] Geoff Boeing. A multi-scale analysis of 27,000 urban street networks: Every US city, town, urbanized area, and zillow neighborhood. page 2399808318784595.

[7] Geoff Boeing. OSMnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks. 65:126–139.

[8] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[9] Weiqi Chen, Ling Chen, Yu Xie, Wei Cao, Yusong Gao, and Xiaojie Feng. Multi-range attentive bicomponent graph convolutional network for traffic forecasting, 2019.

[10] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks, 2016.

[11] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864. ACM, 2016.

[12] Dirk Helbing, Ansgar Hennecke, Vladimir Shvetsov, and Martin Treiber. Master: macroscopic traffic simulation based on a gas-kinetic, non-local traffic model. *Transportation Research Part B: Methodological*, 35(2):183–211, 2001.

[13] Keith Henderson, Brian Gallagher, Tina Eliassi-Rad, Hanghang Tong, Sugato Basu, Leman Akoglu, Danai Koutra, Christos Faloutsos, and Lei Li. Rolx: Structural role extraction & mining in large graphs. In *Proceedings of the 18th ACM SIGKDD*

International Conference on Knowledge Discovery and Data Mining, KDD '12, pages 1231–1239, New York, NY, USA, 2012. ACM.

[14] J Holm, T Jensen, SK Nielsen, A Christensen, B Johnsen, and G Ronby. Calibrating traffic models on traffic census results only. *Traffic Engineering & Control*, 17(4), 1976.

[15] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. Microscopic traffic simulation using sumo. In *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018.

[16] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[17] Sven Maerivoet and Bart De Moor. Transportation planning and traffic flow models. version: 1.

[18] William R McShane and Roger P Roess. *Traffic engineering*. 1990.

[19] OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org, 2017.

[20] Cheonbok Park, Chunggi Lee, Hyojin Bahng, Taeyun won, Kihwan Kim, Seungmin Jin, Sungahn Ko, and Jaegul Choo. Stgrat: A spatio-temporal graph attention network for traffic forecasting, 2019.

[21] Yu Y Shin and Yoonjin Yoon. Incorporating dynamicity of transportation network with multi-weight traffic graph convolution for traffic forecasting.

[22] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis.