THESIS

TOWARDS INTERACTIVE BETWEENNESS CENTRALITY ESTIMATION FOR TRANSPORTATION NETWORK USING CAPSULE NETWORK

Submitted by

Abdul Matin

Department of Computer Science

In partial fulfillment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Fall 2022

Master's Committee:

Advisor: Sangmi Lee Pallickara

Shrideep Pallickara
Aditi S. Bhaskar

ABSTRACT

TOWARDS INTERACTIVE BETWEENNESS CENTRALITY ESTIMATION FOR TRANSPORTATION NETWORK USING CAPSULE NETWORK

The node importance of a graph needs to be estimated for many graph-based applications. One of the most popular metrics for measuring node importance is betweenness centrality, which measures the amount of influence a node has over the flow of information in a graph. However, the computation complexity of calculating betweenness centrality is extremely high with large-scale graphs. This is especially true when analyzing the road networks of states with millions of nodes and edges, making it infeasible to calculate their betweenness centrality (BC) in real-time using traditional iterative methods. The application of a machine learning model to predict the importance of nodes provides opportunities to address this issue. Graph Neural Networks (GNNs), which have been gaining popularity in recent years, are particularly well-suited for graph analysis. In this study, we propose a deep learning architecture RoadCaps to estimate the BC by merging Capsule Neural Networks with Graph Convolutional Networks (GCN), a convolution operation based GNN. We target the effective aggregation of features from neighbor nodes to approximate the correct BC of a node. We leverage patterns capturing the strength of the capsule network to effectively estimate the node level BC from the high-level information generated by the GCN block. We further compare the model accuracy and effectiveness of RoadCaps with the other two GCN-based models. We also analyze the efficiency and effectiveness of RoadCaps for different aspects like scalability and robustness. We perform one empirical benchmark with the road network for the entire state of California. The overall analysis shows that our proposed network can provide more accurate road importance estimation, which is helpful for rapid response planning such as evacuation during wildfires and flooding.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Natural disasters cause substantial disruptions to a community's transportation network, and natural phenomena extremes are predicted to increase both in their frequency and intensity [1]. When sections of roads are flooded, covered by debris, or suffer structural damage, the number of accessible roads and intersections are reduced, which contributed to a further reduction in the number of available routes. Temporary or permanent changes to road networks lead to unexpected traffic spikes over the remainder of the network. The consequences are even more severe for areas with limited number of routes (such as rural areas). An immediate and comprehensive understanding of the impact of infrastructure loss is critical for planning timely responses and recovery.

Traditionally, the impact of fast-evolving disrupted road networks and its complex influences have been analyzed with diverse methodologies such as stochastic optimization processes [2], demand and supply models [3,4], and traffic analysis [5]. Recently, network analysis has been applied to road networks to analyze the influence of various natural disasters such as earthquakes [6–8], and flooding [9–12].

Road networks can be represented as a planar graph that is a graph embedded in the plane with the graph's constituent edges representing physical road connections [13]. Road networks are distinguished from other networks, such as social networks. Since each vertex and edges are physically anchored to their geospatial locations and their network topology is relatively limited in terms of the number of long-range edges and number of edges associated with a single node [14, 15]. Therefore, instead of degree-based metrics, metrics that can provide non-local, higher-level information such as network centralities have been widely adopted in road network analysis [16]. Betweenness centrality (BC) is one of the well-studied centrality measures, and there are several road network analyses based on betweenness centrality [17–19]. Betweenness centrality measures the importance of a road based on the amount of flow at a location. Betweenness centrality is a path-based measure calculated based on the number of shortest paths within a planar graph

that passes through the vertex (e.g., intersection) [20]. However, the calculation of betweenness centrality measures over large-scale complex road systems in real-time poses critical computational challenges. First, computing betweenness centrality over highly complex large road networks is prohibitively expensive. Brandes' algorithm [21] for computing the betweenness centrality has a time complexity of $O(nm+n^2logn)$ and the space complexity is $O(n+m)$, where n and m are the number of vertices and edges in a graph, respectively. With the complexity and abundant data of modern road networks (for e.g., the road system of in the state of California comprises more than 4.45 million roads or edges and 2.67 million intersections) computing the betweenness centrality measures in real-time is infeasible [22]. Second, since betweenness centrality measures depend on the number of shortest paths flowing through the target location, betweenness centrality values are easily influenced by the partial changes within the networks. Removing one edge may require recalculation of betweenness centrality for a substantial area around the removed edge. Finally, for a large road network, the computation over a subarea may cause significant inaccuracies for nodes close to the boundary of an area. This boundary effect, in particular, introduces challenges for distributed approaches to calculation of betweenness centrality over a large spatial extent. In this study, we propose a deep learning-based approach, RoadCaps to calculate weighted BC measures with sub-second latencies over large and complex transportation networks. We combine aspects of Graph Neural Networks [23] and Capsule Networks to accomplish this. Topological characteristics and geospatial features of the surrounding area are extracted and factored into the model to achieve higher model generalization to support varying levels of complexity over the road network and locations of the target intersections. Sub-second inference latencies supported by our network are suitable for applications that need faster turnaround times.

1.1 Research Questions

In this study, we explore the following research questions.

RQ-1: How can we estimate betweenness centrality accurately and rapidly at scale to support applications with interactive explorations of road importance while providing reliable accuracy? Achieving robust accuracy across topological locations is important to avoid boundary effects.

RQ-2. How can we incorporate geospatial characteristics at a given location with topological information to improve accuracy of the estimations?

RQ-3. How can a system estimate the betweenness centrality of a node with limited computing resources? Estimating betweenness centrality should not trigger calculating betweenness centrality for the entire road network. Also, each computation must be lightweight enough to be portable.

1.2 Approach Summary

In this study, we propose a deep network, RoadCaps, that estimates accurate betweenness centrality measures over complex road networks at sub-second latencies. RoadCaps captures nonlinear relationships between the weighted BC values and topological characteristics of the surrounding area combined with area-specific structural road characteristics. RoadCaps leverages capsules to capture hierarchical structural relationships between target intersection(s) and their proximate intersections. CapsNets have been successfully applied in computer vision and graph theory and demonstrably outperform traditional convolutional layers. To generate inputs to capsule layers that effectively snapshot topological and geospatial features, RoadCaps comprises multiple convolutional graph layers. Compared to existing GNNs that primarily target graph classification tasks [24], RoadCaps provides a novel regression capability that estimates 1 or more BC estimates for intersections. As part of this research, we constructed a topological graph representation of road networks and extracted highly relevant features. We introduced a feature for intersections, traffic tendency that encapsulates traffic capacity for a road segment. We have also designed a novel space-efficient data structure, GeoDensityMap, that tracks the complexity of a large road system. We have evaluated our methodology with a road network dataset for the state of California in the U.S. RoadCaps demonstrates a mean absolute error of 2.054 on average, which represents a 31.08% improvement in accuracy compared to both model GCN and model GCNFCL. We per-

formed a variogram analysis to evaluate RoadCaps's capability to address boundary effects that arise. RoadCaps demonstrated a consistently stable model performance across the state of California. On average, our model estimates single point BC in 7.5 milliseconds and 500 points BC in 24.26 milliseconds.

1.3 Main Contributions

We have designed a scalable model that estimates the weighted betweenness centrality of intersections in a large road network. Our contributions include the following:

- Fast and accurate estimations of the weighted betweenness centrality of nodes in a large road network: Our model generates BC measures while accounting for the topological characteristics of proximate nodes and road network-specific features; crucially, this is performed at sub-second latencies.
- Highly generalizable estimations: Our model performance is robust to the topological variations of the road network.
- Wide applicability for other network centrality metrics: The proposed methodology is applicable for other net- work centrality metrics such as percolation centrality and eigenvector centrality.
- Light weight computing to accurately estimate betweenness centrality: Our system allows the users to estimate accurate BC without performing expensive computing tasks required in traditional BC calculations.

1.4 Thesis Organization

Section 2 describes the background and dataset. Our methodology is described in Section 3. Section 4 describes our empirical benchmarks alongside a discussion of the results. Section 5 describes related work. Finally, our conclusions and future work are described in Section 6.

Chapter 2

Background and Dataset

2.1 Betweenness Centrality Analysis for Road Networks

Centrality analysis is widely used to measure node importance at local and global spatial scales. Local centrality is measured between nodes within a given radius while global centrality calculates the distance between nodes within a whole system. The centrality index is useful to understand the operational impact in terms of the network flow tendencies based on topological characteristics, e.g., airline networks, road networks, power networks, and canal networks. Frequently used metrics to estimate network centrality include: betweenness, closeness, straightness, and degree. Closeness centrality is a way of detecting the capability of nodes to spread information efficiently by means of measuring the inverse distance to all other nodes [25]. The straightness index considers the degree of straightness of the path to determine the effectiveness of the connectivity [26]. The degree of centrality is based on the count of the total number of connecting edges to a node [27].

In road network analysis, betweenness centrality analysis has been widely used due to its close correlation to global traffic flows within the network [28]. If two areas are connected by a small number of links, the removal of these links will disable the high volume of traffic flowing between the two areas. Therefore, measuring BC is one of the primary interests of road network resilience to natural disasters [29]. The betweenness centrality is the total number of shortest paths at the target location divided by the total number of shortest paths that exist between two nodes (i and j) of a given radius (r). A node (k) would have a high betweenness centrality if it appears in many shortest paths for flows within the network.

$$Betweenness[k]^r = \sum_{i \neq j \neq k \epsilon d[i,j] \leq r_i}^{n} \frac{N_{d[i,j]}[K]}{N_{d[i,j]}} \tag{2.1}$$

Betweenness analysis has been applied to weighted graphs effectively. The first step of the estimation of betwenness centrality is the shortest path calculation. Shortest path between source and

destination points is determined as the path with smallest total weight. The weight of the edge is inversely related to the travel time and the number of lanes of a road. So, any intersection points will have high betweenness centrality if more paths with smaller weights pass through it.

2.2 Dataset and Study Areas

In this study, we have used transportation datasets provided by OpenStreetMaps [30] for the state of California, U.S. California is the third largest state in the U.S. by area (163,696 square miles) with diverse geographical landscapes, mountains, beaches, lakes, and large city areas. The road network in California contains more than 2.67 million intersections with more than 4.45 million miles of state and county highways. Other types of roads are primary, secondary, tertiary, trunk, service, pedestrian, bike, race, residential, and so on. We selected highways, primary, secondary, tertiary, and trunk roads to focus on land transportation, particularly for major roads that are used by auto vehicles.

The dataset also provides the latitude and longitude information of each intersection point, length, maximum speed, number of lanes, and direction of the road. A graph was made from the extracted road network: each intersection point becomes a vertex, and each road becomes an edge. For preserving geometry, OSM provides more than one intermediate point between two intersections. To reduce the complexity of graph all the intermediate points between two intersections were discarded. After the preprocessing, the graph becomes significantly smaller in size where the total nodes are 129289 and the edges are 281085.

Chapter 3

Methodology

In this study, we use weighted directional graph networks to model a large-scale road network. In section 3.1, we discuss our graph model that captures topological and geographical attributes effectively. We will also describe how we measure the corresponding weights for each of the graph's components. Based on this graph-based model, we propose a novel GNN based on Capsule Networks that estimates the betweenness score of a single/multiple intersections.

3.1 Modeling Road Networks using Graph Networks

Our graph networks reflect the unique set of attributes that comprises topological, physical, and regional characteristics of road networks.

3.1.1 Topological Characteristics

We represent the topological attributes of graph components using vertices and edges. The intersections and end points of roads are represented as vertices and the physical roads that connect a pair of vertices are represented as edges.

Topological Characteristics In our model, road networks are represented as a weighted directed graph, $G = (V, E)$, where the set of vertices, V, represents intersections and end points of the roads, and the set of edges, E represents the physical road between two vertices u and v, where $u, v \in V$. A vertex contains a unique identifier, vertex ID, properties such as geospatial coordinates (lat, lon), connected street count, and road type. An edge is composed of source and destination vertex IDs, and properties including the type of road, weights, length, maximum speed, number of lanes, and road direction. As depicted in Figure 1, the direction of an edge is determined based on the actual traffic flows in the road system. Therefore, if there is a one-way street connecting two intersections (Figure 3.1-(a)), it will be depicted as a single edge following the direction of

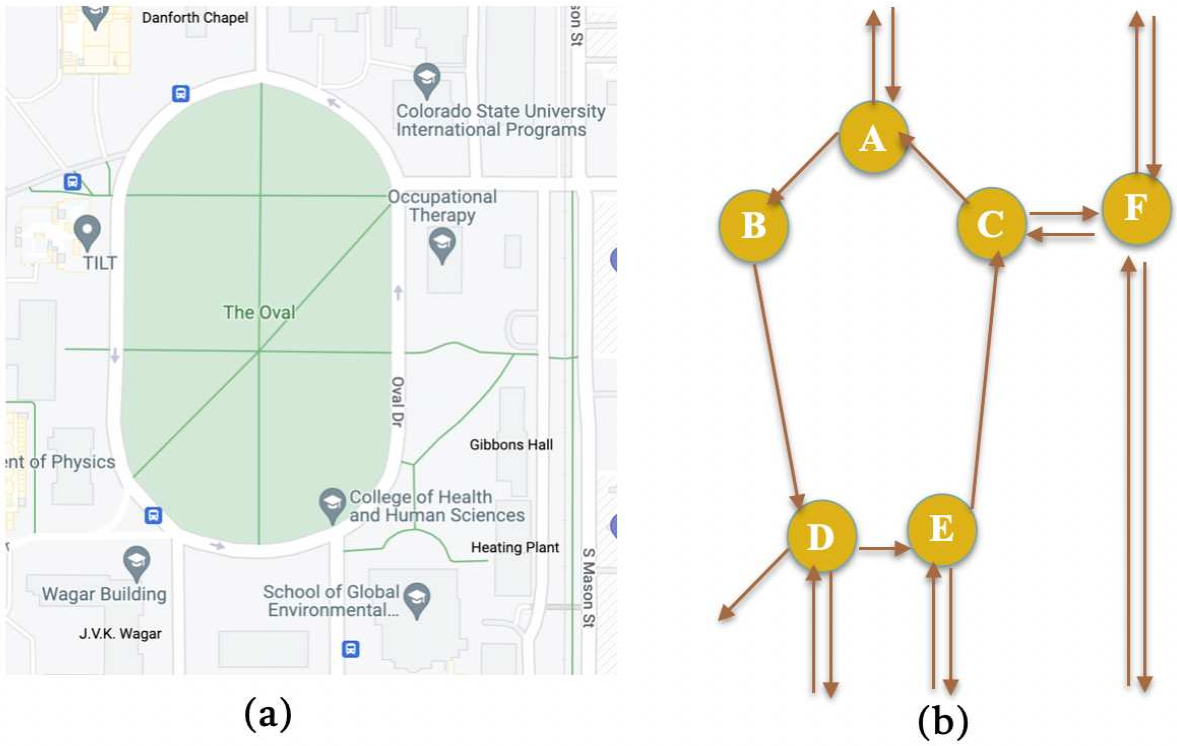

Figure 3.1: (a): The 701 oval drive (the oval in CSU campus) with One-way and Two-way portions (b): Graph representation of the streets depicted in (a)

the road (Figure 3.1-(b)). Our graph model considers the geospatial coordinates of the source and destination vertices only.

Weights with Physical Characteristics Our graph representation maintains a vector of weights for each vertex and edge. To reflect the tendency of the traffic flow within a road segment, we estimate the $traffic\_tendency$ for each edge $e \in E$. We calculate the traffic tendency $e_{traffic\_tendency}$ as follows.

$$e_{traffic\_tendency} = \frac{e_{number\_of\_lanes} \times e_{max\_speed}}{e_{length}} \tag{3.1}$$

,where $e_{number\_of\_lanes}$ is the number of lanes, e_{max_speed} is the speed limit, and e_{length} is the length of the edge. A high traffic tendency indicates that the road is designed for high traffic flows. Meanwhile, a low traffic tendency represents that low traffic flow has been expected. Since the shortest path calculation as a part of betweenness estimation gives priority to the path with smaller weight, we have used the inverse of $e_{traffic\_tendency}$ inversely for the edge weight. Besides the

traffic tendency, non-numeric properties such as the type of the road (one way or bidirectional), number of incoming roads as indegree, and number of outgoing roads as outdegree from any intersection point are also maintained.

3.1.2 Regional Characteristics

Unlike other networks such as social networks, road networks have limited in-degrees and out-degrees (edges connected to a single node) of vertices due to physical and topological constraints. This results in well-defined topological patterns across the networks. Therefore, a model cannot factor in the complexity of the regional road system effectively if it targets a smaller radius in the networks. However, inputting entire networks for each estimation would not be a feasible solution for the real-time BC analysis.

To strike a balance between the detecting regional complexity and computational effectiveness, we introduce a complexity measure based on the density of streets within a geospatial scope. *GeoDensity* Map is a gridded map with density of intersections within a geohash bounding box. Since the possible number of roads that can share the intersection is not highly variable (only 0.17% of our intersections are shared by 6-8 edges), we define the *GeoDensity* of a vertex as the density of intersections without considering the number of edges.

A geohash is a geospatial encoding system that generates a bounding box identified by a 5-bit character string [31]. This string identifier is often used as a part of spatial indexing scheme. The precision of the spatial bounding box is determined by the length of the string identifier. As a greater number of letters are used, the size of the bonding box is reduced. The geohash algorithm provides a hierarchical spatial data structure that preserves the proximity of spatial bounding boxes. We generate a geohash based map with the length of 5 that encompasses approximately 4.9 km^2 in our study areas. All the vertices in the same geohash bounding box share a GeoDensity value. A high GeoDensity value indicates that the given geohash box might be a part of highly complex road system, therefore more routes are to be expected. Otherwise, it belongs to a sparse road system

(a)

12	16	10	12	4
35	34	40	47	35
13	47	46	49	24
21	48	43	57	39
40	66	50	29	31
23	31	30	27	23
27	29	41	38	22

(b)

Figure 3.2: GeoDensity map of road network (a) and intersection count for each Geo-Hash(b)

and a relatively small number of paths may exist in the surrounding area. Figure 3.2 depicts an example of the GeoDensity map with different density of the intersections.

3.2 Network Architecture

Estimating weighted BC values involves multiple factors including patterns of connectivity around the target intersection(s) and structural characteristics such as the tendency of traffic flow. Our approach captures network connectivity and features by generating graph embedding using three layer of Graph Convolutional Network(GCN). The output from GCN is inputted to a Capsule Network layer to capture the hierarchical conceptual structure between the target node(s) and neighboring nodes. Figure 3.3 depicts the overview of the network architecture. To form the network architecture the first step is to form the graph structure which is described in the following sections.

3.2.1 Aggregating Road System Properties with the Graph Structure

Our GNN layer leverages graph embedding methods using Graph Convolutional Neural Network (GCN) to incorporate topological connectivity of graphs [32]. GCN combines the traditional

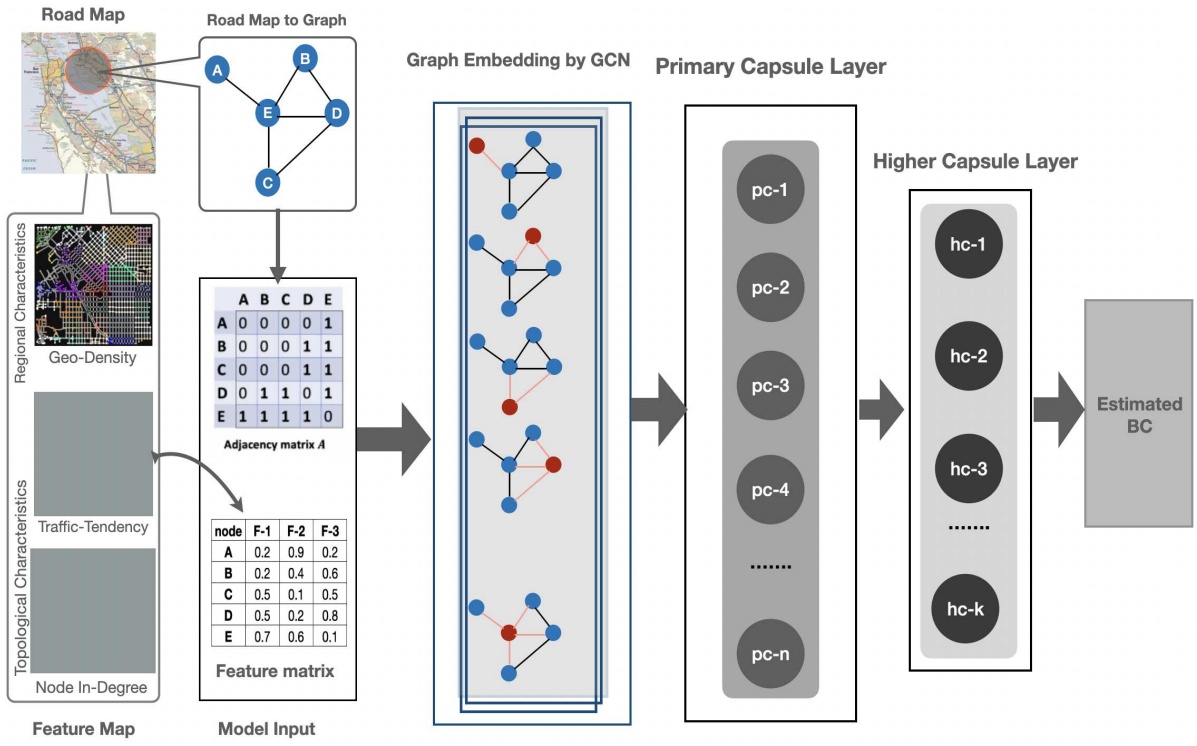

Figure 3.3: Proposed Model Architecture (RoadCaps)

graph encoding approach with Convolutional Neural Networks (CNN) to provide collectively aggregated information from graph while maintaining input and/or output comprising elements and their dependency. Our approach stacks 3 layers to generate graph embedding. At the end of the third layer, the output represents a vector of the extracted features considering all its adjacent nodes that are 3 steps away. In this study, we have selected the number of layers empirically. Figure 3.4 (a) contrasts the model performances with different number of layers, and with 3 layers of GCN, our model shows the better accuracy with less computation complexity.

The first layer of GCN works as the input layer, where the number of neurons equals the number of nodes in the input sample. In this study, we define the maximum number of neurons as 500. The number of neurons indicates the number of neighboring intersections considered within a single input dataset. Since the computing complexity of the BC analysis is closely related to the number of vertices considered for the computation, we have generated small scope of sub-networks to reduce the computing cost. The neighboring intersections are selected based on the

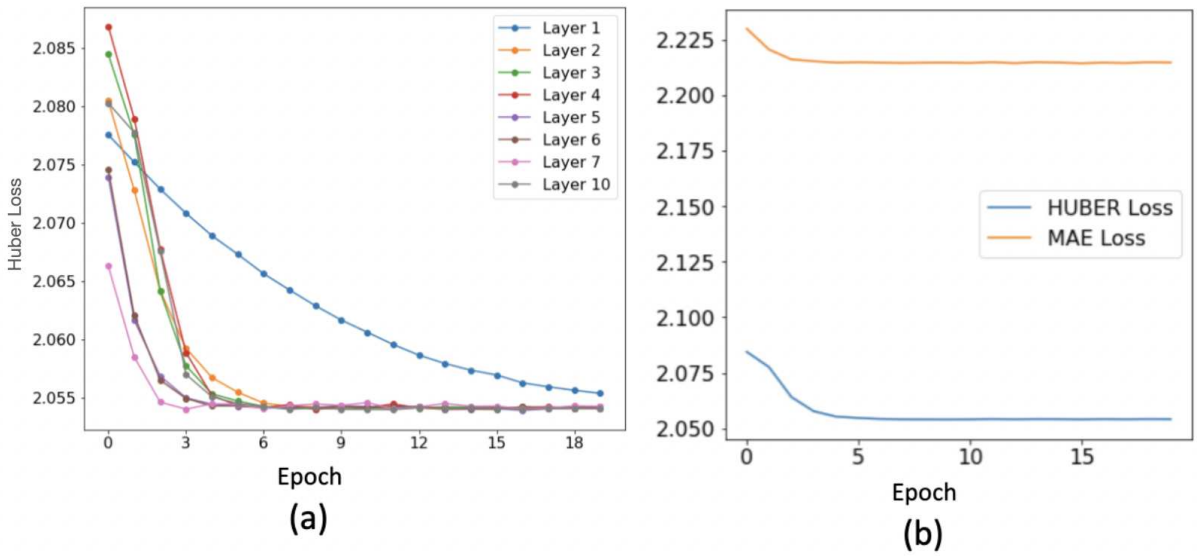

Figure 3.4: Model Performance with different number GCN layer (a) and with different loss functions (b)

distance to the intersection points. Our original dataset from OSM often include multiple points between intersections to preserve the geospatial shape of a road. However, we have simplified the OSM dataset to include only intersections. Therefore, the neighboring nodes are defined as the other connected intersection points around the intersection. Any pair of intersections without a path between them has not been considered for the BC analysis.

With this scope, it takes an adjacency matrix of size 500500. The other input is the feature vector for each node. If each node has five different features, the input features matrix will be 5005. Then, the very first layer will perform graph convolutions on the feature matrix following the adjacency matrix. Unlike classification or regression task with non-graph data structures (such as images or sequences), graph structures involve implicit patterns and behaviors that depend on the previous states of the traversals. The main objectives of GNN layers are to factor in the implicit scenarios stemming from the graph traversals to the model. In our road networks, we have aggregated features from the neighboring nodes. As depicted in Figure 3.3, the output of the first layer will be the aggregated value for each node in the form of a column vector of size 500. The input for the second layer is the single-column output and the original adjacency matrix. The same convolution operation is also done in the second layer following the original adjacency matrix.

The aggregated features from the original feature vector in each node are again aggregated in the second layer. So, the second layer can be thought of as operating two steps away from each node. Along with the direct neighbors, in each higher layer, values from the connected nodes of the first neighbors are also aggregated. The third layer has the same operations as the second layer, and all the following higher layers perform the same calculation. As we add more layers, farther nodes are involved in the convolution feature aggregation. Since adding more layers increases the computational complexity, we choose three GCN layers which provides reasonable accuracy for our task.

The final output dimension of the third layer of GCN maintains the same graph structure as the input sample. So, the size of the output becomes $C500$, where C is the number of output channels or filters. Performing efficient feature engineering and selecting effective features is one of the most challenging tasks for road network analysis. To maintain both topological and regional characteristics of the road network along with the road weight, we have also performed feature engineering for our model. The names of the features are traffic-tendency, GeoDensity, in-degree, and out-degree. From road length, maximum speed, and the number of lanes, the traffic tendency is calculated using the formula described above. Geo density is calculated from the density of connected nodes that share the same prefix of length five of Geo Hashes.

We also use Capsule Networks to efficiently map these low-level features to high-level features that can be used to accurately estimate BC. Capsule Networks are a neural network architecture that efficiently captures spatial relationships between objects and organizes them in a hierarchical fashion. These objects can represent any spatial pattern (e.g. intersections or a busy highway) and are represented in a vector (1D array) format known as a capsule. Each value in the capsule represents a different attribute of that object. For example a capsule representing an intersection might have one value representing the direction of an adjacent road and another value representing the number of nearby lanes. Which attributes are stored in a capsule is based purely on what increases the model's accuracy and can be virtually any object attribute. Each layer of a Capsule Network stores a set number of capsules, which are then routed to the next Capsule Network

layer with its own set of capsules. This dynamic routing process essentially predicts what low-level capsules (e.g. an intersection) from the first Capsule Network layer make up the high-level capsules (e.g. a grouping of intersections) in the next Capsule Network layer.

Capsule Networks have been widely applied in computer vision and graph theory out-performing traditional convolutional-based layers. This is due to the Capsule Network's dynamic routing process being more efficient at capturing information than the convolutional layer's pooling process, which loses a lot of information. A pooling process essentially takes the values of the surrounding area and applies a pooling operation such as taking the mean or maximum of the values. Like CNNs, GCNs also heavily utilize this pooling process, usually by taking the mean of each adjacent node's values. This loses a lot of information, especially when there are many adjacent nodes. Capsule Networks on the other hand combine these adjacent node values in a non-linear fashion (unlike pooling processes) preserving more spatial information and increasing the model's overall accuracy.

We implemented a Capsule Network in our model to increase its overall accuracy. This Capsule Network is placed directly after a few GCN layers. This may seem counterintuitive at first, but using only a few convolutional layers to initially encode the data is an efficient way to encode the lowest-level of data and is very much the norm for Capsule Networks. Capsule-based GNNs have been used with much success in recent years on both node-level and graph-level applications and can significantly outperform purely convolutional-based GNNs [33, 34].

3.3 Loss Functions and Hyper Parameters

Our proposed architecture is a single and multi-point regression model. We experimented with the three well-known loss functions for the regression model. These are Mean Squared Error (MSE), Mean Absolute Error (MAE), and Huber loss function. Due to the extensive range and irregularity of target values, MSE does not work well for our model. Both MAE and Huber loss functions fit well into our model. The loss plots of model training is shown in figure 3.4 (b)

We use the PyTorch L1-loss function that estimates the MAE between each element in the input x and target y. The loss can be described as:

$$\ell(x, y) = L = \{l_1, \ldots, l_N\}^\top, \quad l_n = |x_n - y_n| \tag{3.2}$$

$$\ell(x, y) = \begin{cases} \text{mean}(L), & \text{if reduction} = \text{‘mean’;} \\ \text{sum}(L), & \text{if reduction} = \text{’sum’.} \end{cases}$$

For most of the analysis, we use Pytorch HUBER Loss which utilizes a squared term if the absolute element-wise error drops below delta and a delta-scaled L1 term otherwise. This loss unites the benefits of both L1Loss and MSELoss; the delta-scaled L1 region makes the loss less susceptible to outliers than MSELoss, while the L2 part does smoothness over L1Loss near 0. The loss can be described as:

$$\ell(x, y) = L = \{l_1, ..., l_N\}^T \tag{3.3}$$

with

$$l_n = \begin{cases} 0.5(x_n - y_n)^2, & \text{if } |x_n - y_n| < delta \\ delta * (|x_n - y_n| - 0.5 * delta), & \text{otherwise} \end{cases}$$

If reduction is not none, then:

$$\ell(x, y) = \begin{cases} \text{mean}(L), & \text{if reduction} = \text{‘mean’;} \\ \text{sum}(L), & \text{if reduction} = \text{‘sum’.} \end{cases}$$

The hyperparameters are chosen based on both standard recommendations and empirical analysis. Adam is used as the optimizer with an initial learning rate 10^{-4} and a weight decay factor of 10^{-6}. Total GCN layers are 3, and 50 parallel filters are used for GCN layers. RELU is used as the activation function. The number of neurons in each GCN layer and capsules in the primary capsule layer equals the number of total nodes in each sample which is 500 for most experiments. The higher-level capsule layer contains 500 capsules if the target points are 500; otherwise, it equals the number of prediction points. The capsule dimension is set 5 empirically for both capsule layers.

3.4 Feature Extraction and Selection

The road network can be represented as a large graph where each intersection point of streets is a node, and each road is an edge. Other information is also important to traffic systems like maximum road speed, road direction, street count, and the number of lanes, and geospatial attributes. From the raw information provided, we extract road length, maximum speed, and the number of lanes to construct the feature named traffic tendency. From the topological information, we calculate the degree of incoming and outgoing connections of each node, boundary and non-boundary labeling for each node based on the edge cut in graph samples. We create the geospatial feature, GeoDensity from the geohashes. We consider the first five prefixes of the geohash value to form a group and then count the total number of the intersection points that fall in the same prefix group. The total count of a group is treated as the GeoDensity for all the nodes of that group. After empirical analysis, we select in-degree and out-degree, the traffic tendency, and GeoDensity as the features for each node to train our model. The performance and the time of the model training vary with each feature.

Chapter 4

Empirical Benchmarks and Performance Evaluation

In this section, we evaluate several aspects of our methodology.

4.1 Experimental Setup

The experiments of this work were performed on a 2021 Macbook Air with 16 GB RAM, Apple M1 chip (8-core CPU with 4 performance cores and 4 efficiency cores, 7-core GPU, 8-core GPU, 16-core Neural Engine). To extract the road network, OSM API v1.1.2 was used. Networkx 2.7.1 was used for preprocessing and graph sampling. Pytorch package 1.10.2 with Python 3.9.10 was used as the machine learning framework.

4.2 Model Implementation

The GCN model is used as the base machine learning model to estimate the improvements we found from our proposed dynamic routing-based learning and capsule-based computation model. To compare the robustness and model strength for multi-point prediction, we have used another model, GCNFCL by adding a linear layer on top of the GCN layer. Comparing the RoadCaps model to GCN model provides a good contrast of the changes in both performance and computational cost when we use capsule layers with GCN. Comparing the performance of GCNFCL and GCN model to RoadCaps gives the effectiveness of the capsules with GCN to the diverse road system. An analysis of the training and performance of these three models gives a clear indication of whether concatenating the capsule layer is more effective for road system analysis, especially in terms of accuracy and computational cost.

4.2.1 GRAPH CONVOLUTIONAL NETWORK (GCN)

GCN, a broadly used GNN structure, is picked out as one of the fundamental building blocks in our proposed RoadCaps model. The convolution process is enforced on each node and its neigh-

bors of each layer, and each node's new illustration is calculated through an activation function. We can write this operation as:

$$Z^{l+1} = f(TZ^lW^l) \tag{4.1}$$

Where $Z^l \in \mathbb{R}^{N\times d}$ expresses features of the nodes at the layer l, d denotes the number of channels of features, and $Z^0 = X, W^l \in R^{d\times d'}$ is a weight matrix that is trainable and acts as a channel filter, the nonlinear activation function is f, $T \in \mathbb{R}^{N\times N}$ is the information transform matrix, and it is computed from the adjacency matrix A for navigating the information streaming between nodes. An entire GNN usually stacks L layers to render final nodes embeddings Z^L. In the model architecture offered in paper [35], at the lth layer of GCN, the pulled features of each node take all its adjacent nodes within the l stages into consideration. So l can be thought of as the dimension of the node receptive field at this layer. This unique property encouraged us to use nodes features pulled from various layers to develop the graph

4.2.2 FULLY CONNECTED LAYER WITH GCN

The output from the graph convolutional layers illustrates high-level features in the data. While that output could be flattened and connected to the output layer, adding a fully-connected layer is a simple way of learning non-linear mixtures of these features. Essentially the convolutional layers deliver a significant, low-dimensional, and partially uniform feature space, and the fully connected layer is learning a (possibly non-linear) function in that space. After experimental analysis, one single linear layer is chosen to add after the three GCN layers for further evaluation.

4.2.3 CAPSULE NEURAL NETWORK

The idea of capsules was developed by Hinton's team [36] and employed lately by Hinton et al. [37] and Sabour et al. [38]. CapsNet is developed for image features extraction, and it is designed based on CNN. However, unlike conventional CNN, in which the existence of a feature is expressed with scalar value in feature maps, the features in CapsNet are characterized with vectors that are named capsules. In paper [38], the capsules direction reflects the particular properties of

the features, and the capsule length mirrors the probability of the existence of distinct features. The information dispatch between layers acts per the Dynamic Routing mechanism. Motivated by CapsNet, the capsule structure is embraced and combined with GCN in our proposed RoadCaps to develop primary and high-level capsules based on node capsules that are developed from GNN. Dynamic Routing is used to correct weights between capsules from one layer to the next layer so that the information captured by node capsules can be transmitted to suitable higher-level capsules. Different capsules reflect the properties of the graph from different aspects.

4.2.4 PRIMARY CAPSULE NEURAL NETWORK

The primary node features are aggregated with GCN. Even if no feature is used, the node in-degree and out-degree can be utilized as node attributes. Inspired by the use of capsules in paper [33] We use the capsule structure as the extractor of node features. This block has five sets of primary capsules, each set is Sn={pc1,pc2,.....pcn} where n is equal to the size of the sample and pc means primary capsule. The capsule embodies extracted features for each node. The operation is defined as:

$$Z_j^{l+1} = f(\sum_i \bar{D}^{-\frac{1}{2}} A \bar{D}^{-\frac{1}{2}} Z_i^l W_{ij}^l) \tag{4.2}$$

Where the trainable weights matrix is $W_{ij}^l \in \mathbb{R}^{d \times d'}$. It acts as the channel filters from one layer to the next layer. We use only the extracted features at the last layer of GCN to develop high-level capsules.

HIGH-LEVEL CAPSULE NEURAL NETWORK The capsules in the higher level are formed by applying a dynamic routing mechanism on the primar node capsules. The primary capsule sets Sn is the input to this block, and the output of this block is the higher level node capsule equals the size of prediction points. Each capsule mirrors the patterns of the graph from various factors. This block has five channels or dimensions, which are summed to a single channel to produce the prediction results. All the capsules' outputs are added to make a single point value for a single

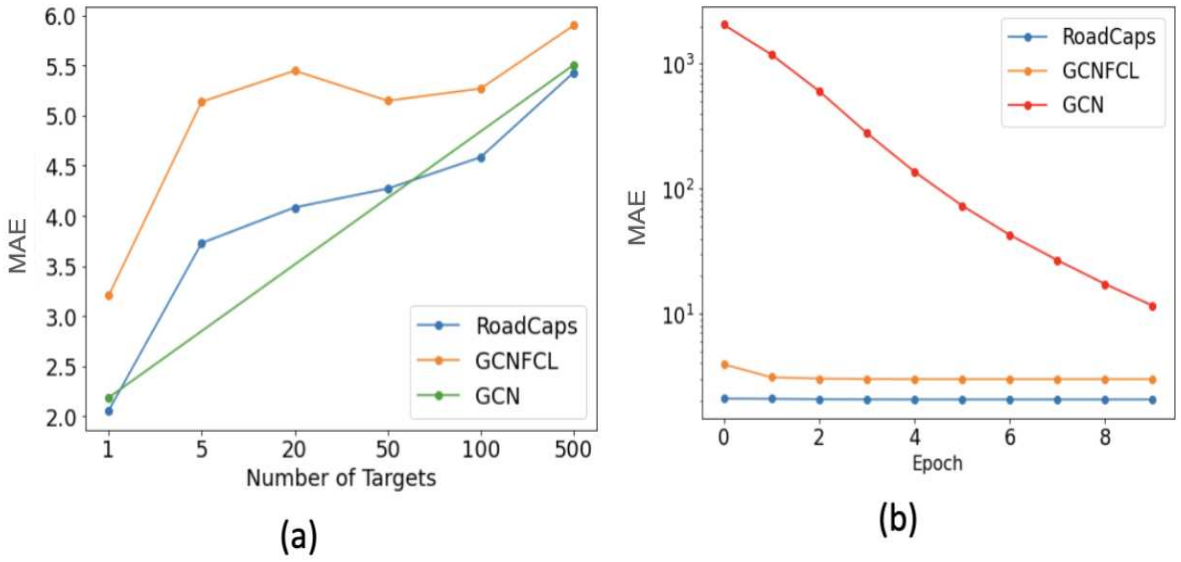

Figure 4.1: Model accuracy with different number of targets (a) and single target (b)

target point. Otherwise, each capsule generates the output value for each output node. So, for the target points of 500, there is a total of 500 capsules in this layer.

4.3 Model Performance Analysis

4.3.1 Comparisons between models: Model accuracy per model

The range of the ground truth values of the dataset is large enough and the values are not properly distributed from low to high range. The HUBER loss function is found less sensitive to the irregularity of such kind of dataset [39] and so for getting good model performance, we use this loss function for our experiments. We have compared the performance of our model with two other models. One is the base graph convolutional network (GCN) model, and the other model is the GCN with a fully connected layer.

For any size, n of sample graph, only two distinct types of prediction are possible by the base model GCN, target one, or target n. Keeping the same sample size, n if it is needed to predict centrality for any number of points between one and n, the only way is to add any layer on top of the outer GCN layer that reduces the dimensionality in the output layer. The fully connected layer

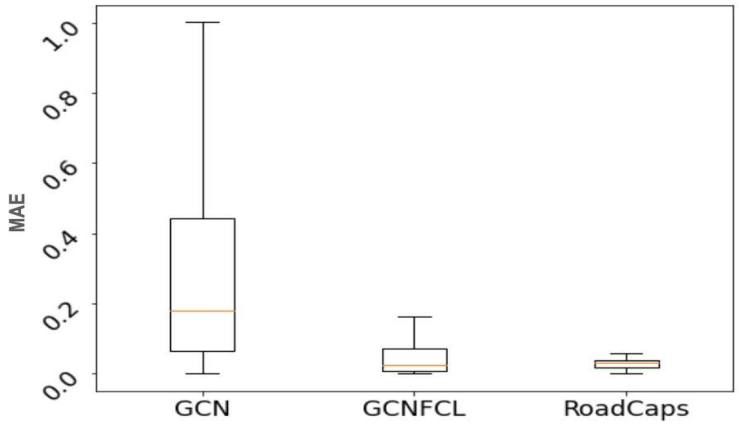

Figure 4.2: The accuracy of models

(FCL) in GCNFCL model and the higher level capsule layer in our proposed RoadCaps model can do this. The figure 4.1 (a) shows that our proposed model performs better than the other two models for all the distinct levels of target points. Our model is robust to work with any number of target points. RoadCaps leverages the information-capturing capability of the capsule network. The two layers of capsules following the GCN layers helps the model converge quickly to the optimal point. As depicted in figure 4.3, RoadCaps converges faster with better accuracy than the other two models.

4.3.2 Analysis of the importance of features

GCN is well known for its impressive performance in embedding the topological characteristics of graph structures and aggregation of node features which is also known as the message passing mechanism. We also leverage this advantage using GCN block prior to the Capsule layers. Also, we use road network specific features to enhance the model accurcy.Figure 4.4 shows that the model converges faster if the model is trained with features than the training without features. Both the GeoDensity and traffic tendency features play a significant role in the model training. Using these two features also increases the model accuracy. Table 4.1 shows that both the GeoDensity and traffic tendency features contribute to model accuracy and faster convergence.

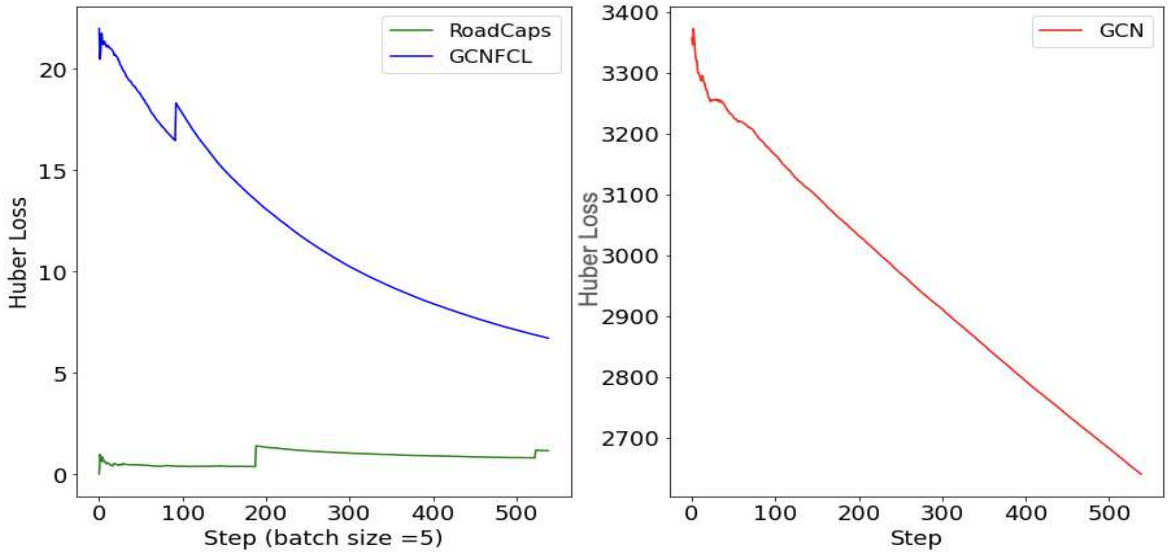

Figure 4.3: First epoch of model training

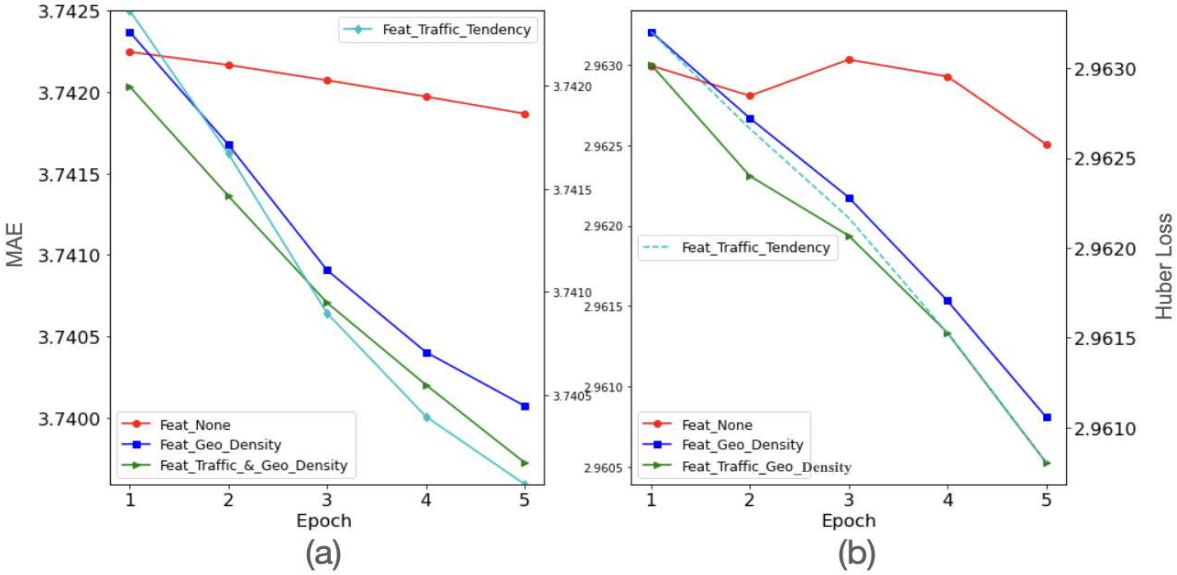

Figure 4.4: Model performance for different features

Table 4.1: Model Performance for different features after five epochs

Target Nodes	Without Feature-1 and 2		With Feature-1		With Feature-2		With Feature-1 and 2	
	Train	Test	Train	Test	Train	Test	Train	Test
1	1.145309	2.076185	1.122853	2.054372	1.122862	2.054366	1.124064	2.055077
5	2.962506	3.741867	2.961281	3.740422	2.960803	3.740068	2.960525	3.739724
50	4.256724	4.27026	4.252890	4.270248	4.252888	4.270246	4.252885	4.270243

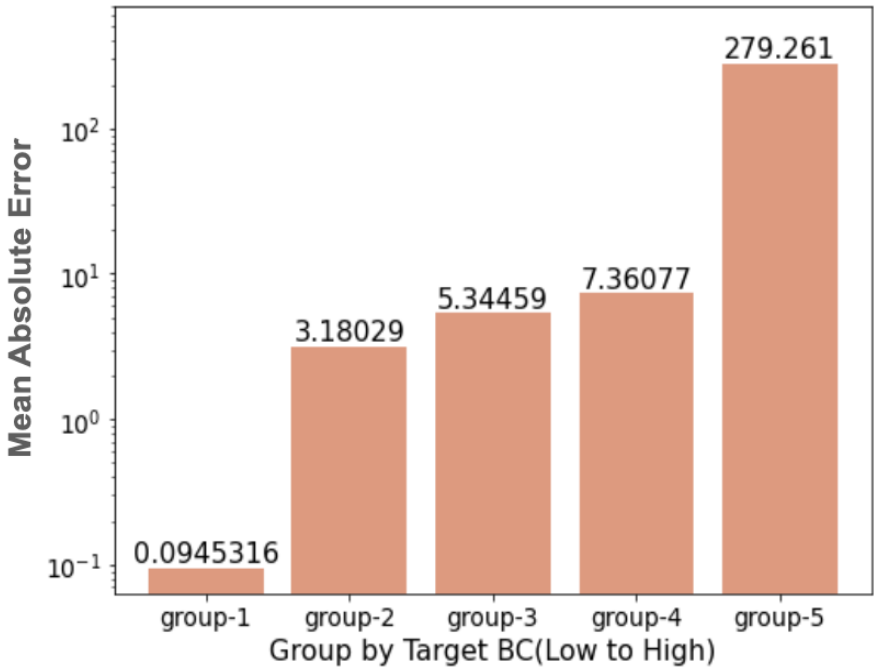

Figure 4.5: Model accuracy with different BC levels

4.3.3 Scalability Analysis

The motivation behind using a machine learning model is to estimate the road importance in real-time. At the same time, the model needs to be scalable. Our model performs well for multi-point regression. we observe the performance of the model by changing the number of target points and neighbor nodes. Figure 4.1 and 4.7 show that our model is highly robust to the changes in sample size and target points and can maintain better accuracy.

4.3.4 Model Performance per BC range

For the road network, it is quite common that the centrality distribution may not be normal or standard as it depends on the road connectivity. The road network dataset of California state also reflects that. Therefore, we have evaluated our model performance for different ranges of BC values. As depicted in figure 4.5, we grouped samples for 5 ranges (group-1(0>= BC <=3), group-2(3< BC <=5), group-3(5< BC <=7), group-4(7< BC <=9), group-5(9< BC <=1000)). Most

of samples have small BCs (group 1 and group 2), and the groups with larger BCs contain higher number of outliers.

4.4 Geospatial Analysis

4.4.1 High density vs Low density area

The prime part of this model network is the topological information, and the model training depends on the road structure first to estimate the centrality measure. To observe the model behaviors with varying road connection density, we have performed another experiment to measure the model accuracy for diverse levels of road density. To do that we depend on geospatial characteristics. Based on the GeoDensity, we have classified samples into two groups, urban and rural. Rural samples have lower GeoDensity than urban samples. The model performance for 50 samples from each group is shown in figure 4.6. Before plotting the line graph the loss values are sorted in ascending order for well understanding of the trend and importance of connected node density. From the experiment results it can be inferred that our model preserves the reasonable accuracy for both types of road network, but it can provide better accuracy for comparatively higher dense road network.

The betweenness centrality estimation is related to the road connection orientation. An intersection point can have high betweenness centrality only if the node becomes the cause of shortening the distance or traveling time between lots of source and destination points. The chances increase if it connects with many other neighboring nodes. Considering this fact, we did empirical analysis of the importance of neighboring nodes to train the model. We can choose any number of neighboring nodes when we do the graph sampling, but two matters need to be considered. First, the number of neighbor nodes must be the same or more than the number of target nodes. Secondly, if the sample size is increased, more computation is needed to process all the features and thus the computation complexity of the model is also increased. We did an experiment for a single target node with five different neighbor nodes. The model test loss is shown in figure 4.7 (b) as box and whisker plot without outliers and in figure 4.7(b) as line graph plot. From the experiment, it can be

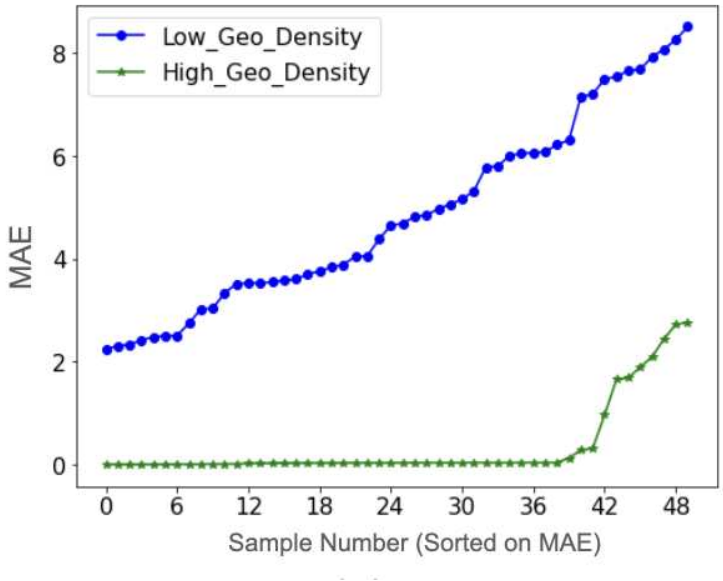

Figure 4.6: HUBER Loss of High and Low GeoDensity Region

observed that even for a single target value, sample with at least 50 neighbor nodes can give good accuracy.

4.4.2 Model Performance for different type of roads

The open street map API provides all types of roads including motorways, highways to bike and pedestrian roads. For our work, we have considered the first five types of roads which are motorways, primary, secondary, tertiary and trunk roads. As each type of road has distinct types of

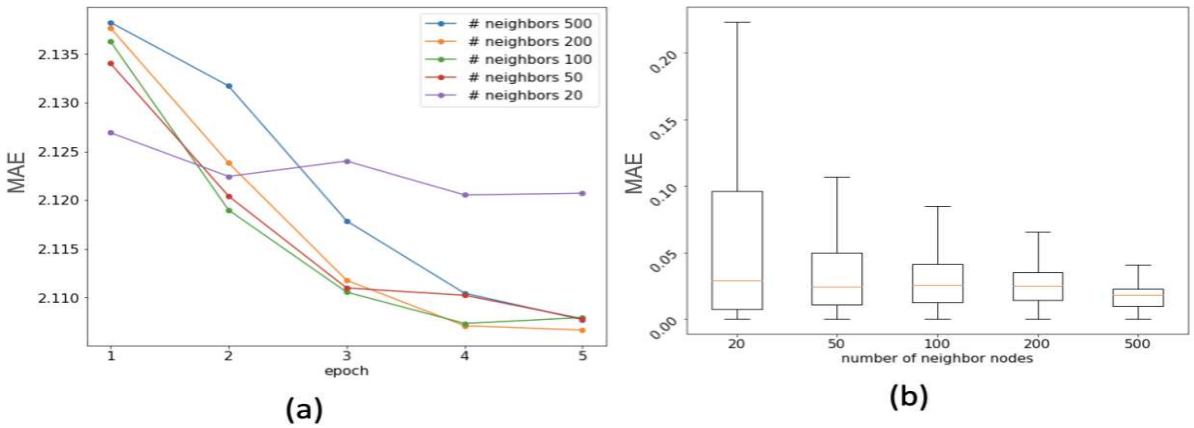

Figure 4.7: Model performance with different number of neighbors

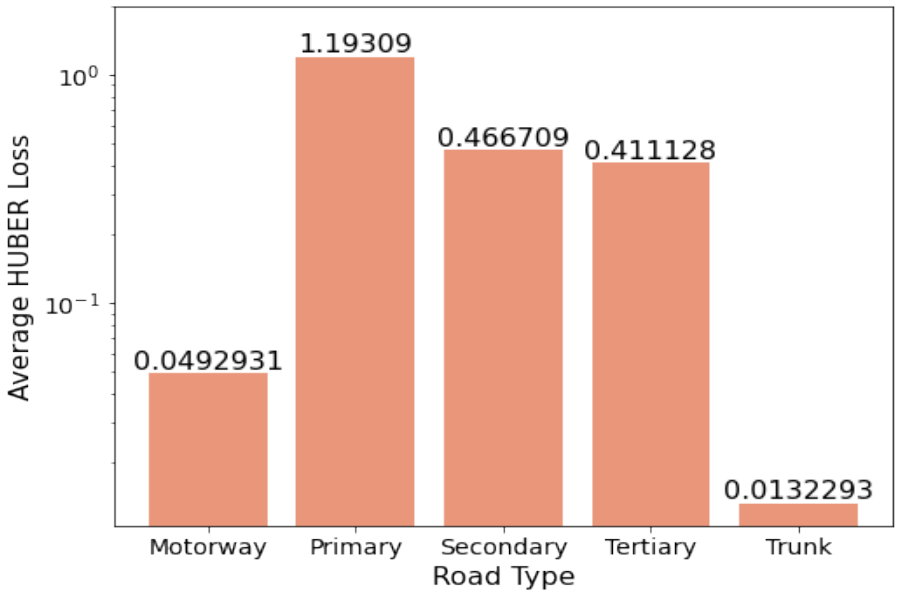

Figure 4.8: Model accuracy with different road types

roads connectivity, we did an experiment to observe how the model behaves with that. In the testing dataset, 7.33% samples are motorways, 17.6% samples are primary roads, 32.5% are secondary, 40.24% are tertiary roads, and 2.28% are trunk roads. The prediction performance is shown in figure 4.9. As the number of test samples in each group are uneven, the inference might not be fully accurate. However, regarding the definition of road types, the testing accuracy for each group is found highly relevant. Tertiary and trunk type roads are most common and connect with so many neighbor roads whereas motorways are limited to certain location and use..

4.4.3 Boundary Effect

Bounday effect is one of the common problem in the graph structured based analysis. As the BC calculation is highly dependent on the road connectivity, BC calculation does have this effect if a graph is formed by cutting off a region from a large network. We have used geospatial features to overcome this situation. We perform a variogram analysis to evaluate RoadCaps's capability to address boundary effects and the result is shown in figure 4.10. It is observed that our proposed model is not sensitive to boundary effect and performs almost equally for all geospatial regions of California state.

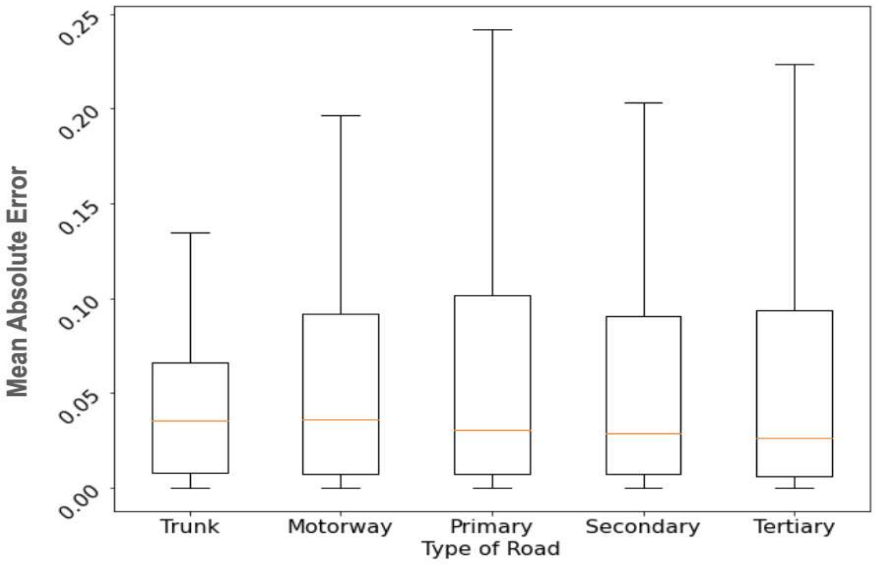

Figure 4.9: Model accuracy with different road types

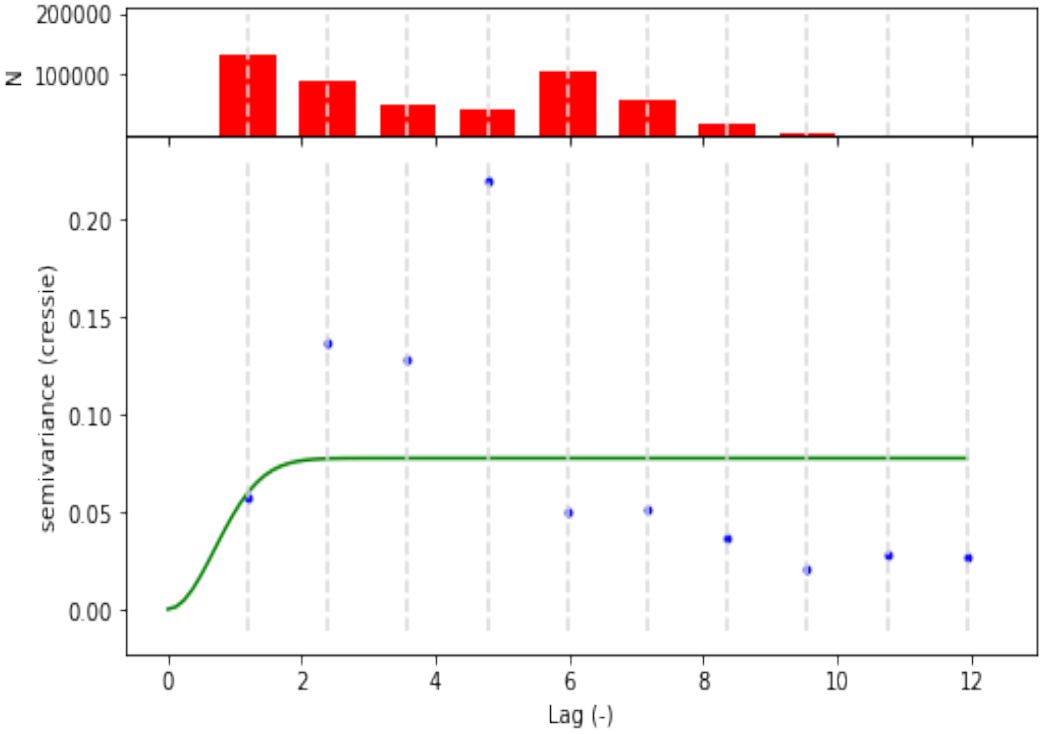

Figure 4.10: Variogram of Model accuracy

Chapter 5

Related Work

Data management challenges arise as the data volumes increase [40]; often this may include a mix of cloud and local resources that may need to be bridged [41]. Such issues have also been explored in grid based architectures [42, 43]. As the size and scale of graph networks with spatial underpinnings increase spatiotemporal datastores [44] and management schemes [45] including sketching [46] provide avenues for coping with data volumes.

5.1 Graph Neural Networks

Recently, graph neural network models (GNNs), which work on graph structured data and apply deep neural network models , have drawn significant interests of the researchers. These GNN models have been used to different applications in many important areas. GNNs are models that mainly capture the dependence of graphs by passing message between the graph nodes [47]. Based on graph embedding and convolutional neural network, the new variants of graph neural networks were mainly proposed to aggregate information collectively from graph structure. Thus the input and/or output, consisting of elements and their dependency can be modeled. Several comprehensive reviews on graph neural networks are available. Bronstein et al. [48] does a detailed review of geometric deep learning, which describes its difficulties, problems, solutions, various applications and future scopes. Zhang et al. [49] offer another detailed overview of graph convolutional networks. However, they specifically focus on convolution operators applied on graph structures. The significant part of our proposed work is also comprised of graph convolution operation on graph like road networks.

In recent years, different type of variants of GNNs have been proposed and analyzed by the researchers. Most recent survey papers mainly focusing on GNN models were published by Zhang et al. [50], Wu et al. [24] , Chami et al. [51]. [52] categorize GNNs into four groups: recurrent graph neural networks, convolutional graph neural networks, graph autoencoders, and spatial-temporal

graph neural networks. Zhang et al. [50] proposed a organized overview of various graph deep learning methods and Chami et al. [51] offered a Graph Encoder Decoder Model to unify graph neural network models and network embedding.

The most fundamental operation of CNNs is the convolution. However, the same convolution operation used for images or text is not directly applicable to graphs because of its lack of grid structure [53]. At first, convolution for the graph data was introduced by Bruna et al. [54] from the spectral domain using the graph Laplacian matrix L [55], which does a similar role as the Fourier basis in signal processing [53]. Among other variants of GNN, Graph convolutional networks (GCNs) are certainly the most popular topic in graph-based deep learning. imitating CNNs, modern GCNs learn the frequent local and global patterns of graph structure through planned convolution and readout functions. Because task specific loss can be imposed to train the most GCNs via backpropagation [56]. Here, to analyze the importance of road network, the structure of road plays the vital role to aggregate information and passes through from one node to it's neighbour node alike CNN on image data.

5.2 Graph Classification and Node Classification

Among all the applications of GNNs, in last few years, Two graph learning problems have got a lot of attention i.e., graph classification and node classification. Graph classification predicts the class label of graphs. Historically, to solve the problem of graph classification, the most dominant techniques are graph kernels [57–61] and in last few years, another popular approach, deep learning approaches [62, 63] have been designed.

GNNs, on the other hand, directly classify graphs depending on the extracted graph representations and that is why GNNs are much more efficient than graph kernel methods [24]. Recently, there are so many works shows the efficiency of GNN for Graph classification [64–66]. GNNs have also gained popularity for node level analysis by aggregating the graph structure information which is node classification [67] [68]. But here, instead of node classification our work is on regression, predicting node importance of road network which is much more challenging compare to

node classification. Nowadays, GNNs are gaining attention in different other fields as well. Few of them are recommendation system [52, 69] , Computer Vision [70, 71], NLP [72, 73] and so on.

5.3 Road network analysis using GNN

As road network forms the structure of graph, to analyze road network, GNNs show the significant improvement of efficiency in different aspects. Most common road network analysis using GNN are traffic flow estimation or traffic forecasting [74–76]. The existing model to forecast ride hailing demand is complex because region-level demand forecasting is challenging. For ride hailing service, it is an essential task because accurate ride-hailing demand forecasting can improve vehicle utilization, can guide vehicle dispatching, reduce the wait-time, and solve traffic congestion. This task can be done more efficiently by GNN [77, 78]. GNN is also used in road network analysis for city-wide parking availability prediction [79], the future trajectories prediction of multiple interacting pedestrians [80]. Our purpose is analysing road network to predict the road importance which then can be used to model any emergency evacuation plan during any kind of natural disaster like wildfire.

5.4 Distributed graph computing frameworks

One of the biggest challenge of applying GNN in road network analysis is to work on large graph structure of having million to billion of nodes and edges. Researchers propose different ways of using GNN for large scale training. In the paper [81], a method was proposed named DistGNN that optimizes the Deep Graph Library (DGL) to use an efficient shared memory implementation while training on CPU clusters. A minimum vertex-cut graph partitioning algorithm with delayed update has been introduced to reduce the communication latency. On the other hand, instead of whole-graph based training Marco et al. [82] did a case review and provide importance on sample-based training for large scale network. For this case, graph sampling algorithms become important. Here, we have also followed sample based training using GNN [82].

Chapter 6

Conclusions and Future Work

We present our model, RoadCaps, that estimates BC accurately and rapidly over an extensive road network. RoadCaps addresses model performance challenges emanating from topological complexity and geospatial variability with a custom deep network architecture that incorporates GCN and Deep Capsule Network . GCN captures topological graph structure and features of regional road networks, and the Capsule network maps different levels of information and extracts patterns from high-level information effectively. Appropriately extracted topological and geospatial features improve the model accuracy and convergence rate. RoadCaps outperforms our base models such as GCN and GCNFCL in terms of accuracy and robustness. Our analysis shows that RoadCaps demonstrates stable performance regardless of the location of target intersection in the sample. RoadCaps trains and makes inferences with compact sized samples that allows computational effectiveness.

Our model performance was evaluated only for the road network in one of the largest states of USA. The performance of RoadCaps could be evaluated for other geospatial locations. The case studies of RoadCaps could also be done over road networks with extensive attributes and weights (e.g., charging stations, energy consumption, or wild fire debris). A lot of real life graph based applications could be experimented by our model. By changing the attribute and features, this model can be used to solve those problems having road structure like use pattern. There are different applications in traffic system where RoadCaps can be useful to generate real-time decision. The correlation analysis with real-time traffic data could also be done to measure the effectiveness of our model inference.

Bibliography

[1] Hossein Tabari. Climate change impact on flood and extreme precipitation increases with water availability. *Scientific reports*, 10(1):1–10, 2020.

[2] LAPJ Gonçalves and PJG Ribeiro. Resilience of urban transportation systems. concept, characteristics, and methods. *Journal of Transport Geography*, 85:102727, 2020.

[3] Akvan Gajanayake, Guomin Zhang, Tehmina Khan, and Hessam Mohseni. Postdisaster impact assessment of road infrastructure: State-of-the-art review. *Natural hazards review*, 21(1):03119002, 2020.

[4] Lars-Göran Mattsson and Erik Jenelius. Vulnerability and resilience of transport systems–a discussion of recent research. *Transportation research part A: policy and practice*, 81:16–34, 2015.

[5] Shouzheng Pan, Hai Yan, Jia He, and Zhengbing He. Vulnerability and resilience of transportation systems: A recent literature review. *Physica A: Statistical Mechanics and its Applications*, 581:126235, 2021.

[6] Nazli Yonca Aydin, H Sebnem Duzgun, Hans Rudolf Heinimann, Friedemann Wenzel, and Kaushal Raj Gnyawali. Framework for improving the resilience and recovery of transportation networks under geohazard risks. *International journal of disaster risk reduction*, 31:832–843, 2018.

[7] Shangjia Dong, Haizhong Wang, Ali Mostafavi, and Jianxi Gao. Robust component: a robustness measure that incorporates access to critical facilities under disruptions. *Journal of the Royal Society Interface*, 16(157):20190149, 2019.

[8] Yaoming Zhou, Junwei Wang, and Jiuh-Biing Sheu. On connectivity of post-earthquake road networks. *Transportation Research Part E: Logistics and Transportation Review*, 123:1–16, 2019.

[9] Bahrulla Abdulla. Characterization of the vulnerability of road networks to fluvial flooding using network percolation approach. *ASCE Computing in Civil Engineering 2019*, 2019.

[10] Bahrulla Abdulla, Amin Kiaghadi, Hanadi S Rifai, and Bjorn Birgisson. Characterization of vulnerability of road networks to fluvial flooding using sis network diffusion model. *Journal of Infrastructure Preservation and Resilience*, 1(1):1–13, 2020.

[11] Chao Fan, Xiangqi Jiang, and Ali Mostafavi. A network percolation-based contagion model of flood propagation and recession in urban road networks. *Scientific Reports*, 10(1):1–12, 2020.

[12] Shangjia Dong, Alireza Mostafizi, Haizhong Wang, Jianxi Gao, and Xiaopeng Li. Measuring the topological robustness of transportation networks to disaster-induced failures: A percolation approach. *Journal of Infrastructure Systems*, 26(2):04020009, 2020.

[13] Alec Kirkley, Hugo Barbosa, Marc Barthelemy, and Gourab Ghoshal. From the betweenness centrality in street networks to structural invariants in random planar graphs. *Nature communications*, 9(1):1–12, 2018.

[14] David Aldous and Karthik Ganesan. True scale-invariant random spatial networks. *Proceedings of the National Academy of Sciences*, 110(22):8782–8785, 2013.

[15] David J Aldous. Routed planar networks. *Electronic Journal of Graph Theory and Applications (EJGTA)*, 4(1):42–59, 2016.

[16] Marc Barthélemy. Crossover from scale-free to spatial networks. *EPL (Europhysics Letters)*, 63(6):915, 2003.

[17] Paolo Crucitti, Vito Latora, and Sergio Porta. Centrality measures in spatial networks of urban streets. *Physical Review E*, 73(3):036125, 2006.

[18] Marc Barthelemy, Patricia Bordin, Henri Berestycki, and Maurizio Gribaudi. Self-organization versus top-down planning in the evolution of a city. *Scientific reports*, 3(1):1–8, 2013.

[19] Timothy C Jarrett, Douglas J Ashton, Mark Fricker, and Neil F Johnson. Interplay between function and structure in complex networks. *Physical Review E*, 74(2):026116, 2006.

[20] Linton C Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.

[21] Ulrik Brandes. A faster algorithm for betweenness centrality. *Journal of mathematical sociology*, 25(2):163–177, 2001.

[22] Shivam Bhardwaj, Rajdeep Niyogi, and Alfredo Milani. Performance analysis of an algorithm for computation of betweenness centrality. In *International Conference on Computational Science and Its Applications*, pages 537–546. Springer, 2011.

[23] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016.

[24] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[25] Murray A Beauchamp. An improved index of centrality. *Behavioral science*, 10(2):161–163, 1965.

[26] Edward Batschelet. Circular statistics in biology. *ACADEMIC PRESS, 111 FIFTH AVE., NEW YORK, NY 10003, 1981, 388*, 1981.

[27] Bernard Grofman and Guillermo Owen. A game theoretic approach to measuring degree of centrality in social networks. *Social Networks*, 4(3):213–224, 1982.

[28] Elise Henry, Loïc Bonnetain, Angelo Furno, Nour-Eddin El Faouzi, and Eugenio Zimeo. Spatio-temporal correlations of betweenness centrality and traffic metrics. In *2019 6th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, pages 1–10, 2019.

[29] Pauline Gauthier, Angelo Furno, and Nour-Eddin El Faouzi. Road network resilience: how to identify critical links subject to day-to-day disruptions. *Transportation research record*, 2672(1):54–65, 2018.

[30] Jonathan Bennett. *OpenStreetMap*. Packt Publishing Ltd, 2010.

[31] Zoran Balkić, Damir Šoštarić, and Goran Horvat. Geohash and uuid identifier for multi-agent systems. In *KES International Symposium on Agent and Multi-Agent Systems: Technologies and Applications*, pages 290–298. Springer, 2012.

[32] Rianne van den Berg, Thomas N Kipf, and Max Welling. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263*, 2017.

[33] Zhang Xinyi and Lihui Chen. Capsule graph neural network. In *International Conference on Learning Representations*, 2019.

[34] Rui Yang, Wenrui Dai, Chenglin Li, Junni Zou, and Hongkai Xiong. NCGNN: node-level capsule graph neural network. *CoRR*, abs/2012.03476, 2020.

[35] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. 2017. *ArXiv abs/1609.02907*, 2017.

[36] Geoffrey E Hinton, Alex Krizhevsky, and Sida D Wang. Transforming auto-encoders. In *International conference on artificial neural networks*, pages 44–51. Springer, 2011.

[37] Geoffrey E Hinton, Sara Sabour, and Nicholas Frosst. Matrix capsules with EM routing. In *International Conference on Learning Representations*, 2018.

[38] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. *Advances in neural information processing systems*, 30, 2017.

[39] Ashkan Esmaeili and Farokh Marvasti. A novel approach to quantized matrix completion using huber loss measure. *IEEE Signal Processing Letters*, 26(2):337–341, 2019.

[40] Sangmi Lee Pallickara, Shrideep Pallickara, and Marlon Pierce. Scientific data management in the cloud: A survey of technologies, approaches and challenges. *Handbook of Cloud Computing*, pages 517–533, 2010.

[41] Matthew Malensek, Sangmi Lee Pallickara, and Shrideep Pallickara. Hermes: Federating fog and cloud domains to support query evaluations in continuous sensing environments. *IEEE Cloud Computing*, 4(2):54–62, 2017.

[42] Geoffrey Fox, Shrideep Pallickara, and Xi Rao. Towards enabling peer-to-peer grids. *Concurrency and Computation: Practice and Experience*, 17(7-8):1109–1131, 2005.

[43] GC Fox, Hasan Bulut, Kangseok Kim, Sung-Hoon Ko, Sangmi Lee, Sangyoon Oh, Shrideep Pallickara, Xiaohong Qiu, Ahmet Uyar, Minjun Wang, et al. Collaborative web services and peer-to-peer grids. *SIMULATION SERIES*, 35(1):3–12, 2003.

[44] Matthew Malensek, Sangmi Pallickara, and Shrideep Pallickara. Fast, ad hoc query evaluations over multidimensional geospatial datasets. *IEEE Transactions on Cloud Computing*, 5(1):28–42, 2015.

[45] Kevin Bruhwiler, Paahuni Khandelwal, Daniel Rammer, Samuel Armstrong, Sangmi Lee Pallickara, and Shrideep Pallickara. Lightweight, embeddings based storage and model construction over satellite data collections. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 246–255. IEEE, 2020.

[46] Thilina Buddhika, Matthew Malensek, Sangmi Lee Pallickara, and Shrideep Pallickara. Synopsis: A distributed sketch over voluminous spatiotemporal observational streams. *IEEE Transactions on Knowledge and Data Engineering*, 29(11):2552–2566, 2017.

[47] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, 2020.

[48] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[49] Si Zhang, Hanghang Tong, Jiejun Xu, and Ross Maciejewski. Graph convolutional networks: a comprehensive review. *Computational Social Networks*, 6(1):1–23, 2019.

[50] Ziwei Zhang, Peng Cui, and Wenwu Zhu. Deep learning on graphs: A survey. corr abs/1812.04202 (2018). *arXiv preprint arXiv:1812.04202*, 2018.

[51] Ines Chami, Sami Abu-El-Haija, Bryan Perozzi, Christopher Ré, and Kevin Murphy. Machine learning on graphs: A model and comprehensive taxonomy. *arXiv preprint arXiv:2005.03675*, 2020.

[52] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 346–353, 2019.

[53] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE signal processing magazine*, 30(3):83–98, 2013.

[54] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[55] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Advances in neural information processing systems*, 14, 2001.

[56] Ziwei Zhang, Peng Cui, and Wenwu Zhu. Deep learning on graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[57] Nicolò Navarin and Alessandro Sperduti. Approximated neighbours minhash graph node kernel. In *ESANN*, pages 281–286, 2017.

[58] S Vichy N Vishwanathan, Nicol N Schraudolph, Risi Kondor, and Karsten M Borgwardt. Graph kernels. *Journal of Machine Learning Research*, 11:1201–1242, 2010.

[59] Karsten M Borgwardt and Hans-Peter Kriegel. Shortest-path kernels on graphs. In *Fifth IEEE international conference on data mining (ICDM'05)*, pages 8–pp. IEEE, 2005.

[60] Nino Shervashidze, Pascal Schweitzer, Erik Jan Van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(9), 2011.

[61] Nino Shervashidze, SVN Vishwanathan, Tobias Petri, Kurt Mehlhorn, and Karsten Borgwardt. Efficient graphlet kernels for large graph comparison. In *Artificial intelligence and statistics*, pages 488–495. PMLR, 2009.

[62] Annamalai Narayanan, Mahinthan Chandramohan, Rajasekar Venkatesan, Lihui Chen, Yang Liu, and Shantanu Jaiswal. graph2vec: Learning distributed representations of graphs. *arXiv preprint arXiv:1707.05005*, 2017.

[63] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *International conference on machine learning*, pages 2014–2023. PMLR, 2016.

[64] Davide Bacciu, Federico Errica, and Alessio Micheli. Contextual graph markov model: A deep and generative approach to graph processing. In *International Conference on Machine Learning*, pages 294–303. PMLR, 2018.

[65] Jia Li, Yu Rong, Hong Cheng, Helen Meng, Wenbing Huang, and Junzhou Huang. Semi-supervised graph classification: A hierarchical graph perspective. In *The World Wide Web Conference*, pages 972–982, 2019.

[66] Federico Errica, Marco Podda, Davide Bacciu, and Alessio Micheli. A fair comparison of graph neural networks for graph classification. *arXiv preprint arXiv:1912.09893*, 2019.

[67] Hogun Park and Jennifer Neville. Exploiting interaction links for node classification with deep graph neural networks. In *IJCAI*, pages 3223–3230, 2019.

[68] Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. *arXiv preprint arXiv:1905.10947*, 2019.

[69] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 974–983, 2018.

[70] Siyuan Qi, Wenguan Wang, Baoxiong Jia, Jianbing Shen, and Song-Chun Zhu. Learning human-object interactions by graph parsing neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 401–417, 2018.

[71] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[72] Victoria Zayats and Mari Ostendorf. Conversation modeling on reddit using a graph-structured lstm. *Transactions of the Association for Computational Linguistics*, 6:121–132, 2018.

[73] Xiao Liu, Zhunchen Luo, and Heyan Huang. Jointly multiple events extraction via attention-based graph information aggregation. *arXiv preprint arXiv:1809.09078*, 2018.

[74] Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. AAAI'19/IAAI'19/EAAI'19. AAAI Press, 2019.

[75] Zhiyong Cui, Kristian Henrickson, Ruimin Ke, and Yinhai Wang. High-order graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting. *CoRR*, abs/1802.07007, 2018.

[76] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*, 2017.

[77] Xu Geng, Yaguang Li, Leye Wang, Lingyu Zhang, Qiang Yang, Jieping Ye, and Yan Liu. Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3656–3663, 2019.

[78] Yuandong Wang, Hongzhi Yin, Hongxu Chen, Tianyu Wo, Jie Xu, and Kai Zheng. Origin-destination matrix prediction via graph convolution: a new perspective of passenger demand modeling. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1227–1235, 2019.

[79] Weijia Zhang, Hao Liu, Yanchi Liu, Jingbo Zhou, and Hui Xiong. Semi-supervised hierarchical recurrent graph neural network for city-wide parking availability prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1186–1193, 2020.

[80] Vineet Kosaraju, Amir Sadeghian, Roberto Martín-Martín, Ian Reid, Hamid Rezatofighi, and Silvio Savarese. Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[81] Vasimuddin Md, Sanchit Misra, Guixiang Ma, Ramanarayan Mohanty, Evangelos Georganas, Alexander Heinecke, Dhiraj Kalamkar, Nesreen K Ahmed, and Sasikanth Avancha. Distgnn: Scalable distributed training for large-scale graph neural networks. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–14, 2021.

[82] Marco Serafini and Hui Guan. Scalable graph neural network training: The case for sampling. *ACM SIGOPS Operating Systems Review*, 55(1):68–76, 2021.