Graph Convolutional Networks for Road Networks

Skovgaard Jepsen, Tobias; Jensen, Christian S.; Nielsen, Thomas Dyhre

Published in:
Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems

DOI (link to publication from Publisher):
10.1145/3347146.3359094

Publication date:
2019

Document Version
Accepted author manuscript, peer reviewed version

Graph Convolutional Networks for Road Networks

Tobias Skovgaard Jepsen
Department of Computer Science
Aalborg University
tsj@cs.aau.dk

Christian S. Jensen
Department of Computer Science
Aalborg University
csj@cs.aau.dk

Thomas Dyhre Nielsen
Department of Computer Science
Aalborg University
tdn@cs.aau.dk

ABSTRACT

The application of machine learning techniques in the setting of road networks holds the potential to facilitate many important transportation applications. Graph Convolutional Networks (GCNs) are neural networks that are capable of leveraging the structure of a network. However, many implicit assumptions of GCNs do not apply to road networks.

We introduce the *Relational Fusion Network (RFN)*, a novel type of GCN designed specifically for road networks. In particular, we propose methods that substantially outperform state-of-the-art GCNs on two machine learning tasks in road networks. Furthermore, we show that state-of-the-art GCNs fail to effectively leverage road network structure on these tasks.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning algorithms**;
• **Applied computing** → **Transportation**.

KEYWORDS

Road Network, Machine Learning, Graph Representation Learning, Graph Convolutional Networks

ACM Reference Format:
Tobias Skovgaard Jepsen, Christian S. Jensen, and Thomas Dyhre Nielsen. 2019. Graph Convolutional Networks for Road Networks. In *27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '19), November 5–8, 2019, Chicago, IL, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3347146.3359094

1 INTRODUCTION

Machine learning on road networks can facilitate important transportation applications such as traffic forecasting [12], speed limit annotation [7], and travel-time estimation. However, machine learning on road networks is difficult due to the low number of attributes, often with missing values, that typically are available [7]. This lack of attribute information can be alleviated by exploiting the network structure into the learning process [7]. To this end, we propose the *Relational Fusion Network (RFN)*, a type of Graph Convolutional Network (GCN) designed specifically for road networks.

GCNs are neural networks that operate directly on graph representations of networks. GCNs can in theory leverage road network structure by aggregating over a road segment's neighborhood when computing the segment's representation, e.g., computing the mean representations of its adjacent road segments. However, state-of-the-art GCNs are designed for node classification tasks in social, citation, and biological networks. Although GCNs have been highly successful at such tasks, machine learning tasks in road networks differ substantially.

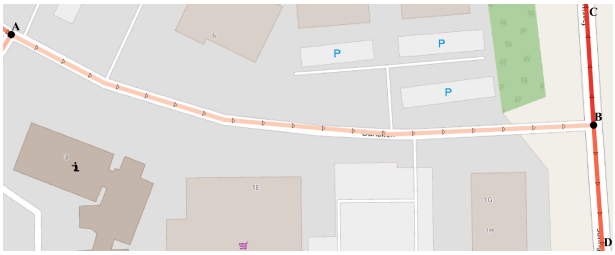

Figure 1: Volatile homophily in a three-way intersection.

First, many implicit assumptions in GCN proposals do not hold in the context of road networks. First, road networks are *edge-relational* and contain not only node and edge attributes, but also *between-edge attributes* that characterize the relationships between road segments (edges). For instance, the angle between two road segments is informative for travel time estimation since it influences the time it takes to move from one segment to the other.

Second, GCNs implicitly assume that the underlying network is homophilic meaning that adjacent road segments tend to be similar, and that changes in network characteristics, e.g., driving speeds, occur gradually. Although road networks exhibit homophily, the homophily is volatile in the sense that homophilic regions have sharp boundaries characterized by abrupt changes in, e.g., driving speeds. In the most extreme case, a region may consist of a single road segment, *in which case there is no homophily.* As an example, the three-way intersection to the right in Fig. 1 exhibits volatile homophily. The two vertical road segments to the right and the road segments connected to the intersection to the form two regions that each is internally homophilic: the road segments within each region have similar driving speeds. The two regions are adjacent, but, a driver moving from one region to the other experiences an abrupt change in driving speed.

Contributions. We introduce the *Relational Fusion Network (RFN)*, a type of GCN designed specifically to address the shortcomings of state-of-the-art GCNs in the road network setting.[1] A novel relational fusion operator is at the core of a Relational Fusion Network (RFN). This graph convolutional operator aggregates over representations of relations instead of over representations of neighbors. To learn a representation of a relation (u, v), an RFN uses a fusion

[1] Due to page limitation, we give only an introduction of our method in this paper. See [6] for a detailed description.

SIGSPATIAL '19, November 5–8, 2019, Chicago, IL, USA

ACM ISBN 978-1-4503-6909-1/19/11.
https://doi.org/10.1145/3347146.3359094

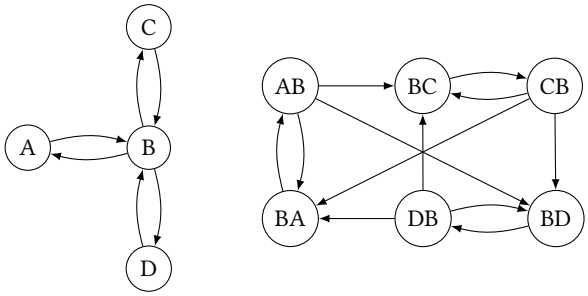

Figure 2: The (left) primal and (right) dual graph representations of the three-way intersection to the right in Fig. 1.

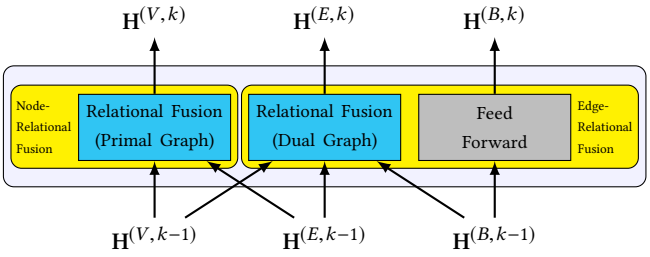

Figure 3: Relational Fusion Layer.

function that represents a relation (u, v) by fusing the representations, e.g., attributes, of road segments u and v and the attributes of their relation (u, v) that describe the nature of the relationship between u and v. This fusion mechanism allows an RFN to capture volatile homophily and makes it robust to aberrant neighbors in small neighborhoods.

RFNs are capable of leveraging node attributes, edge attributes, and *between-edge attributes* jointly during the learning process by considering both a *node view* and an *edge view*: two perspectives that capture the relationships between intersections and road segments, respectively. In comparison, state-of-the-art GCNs consider at most one of these perspectives and can leverage only one source of attributes. We evaluate the proposed RFN architecture on two road segment prediction tasks and find that the RFNs outperform state-of-the-art GCNs significantly on both tasks. Interestingly, our results suggest that an RFN can leverage neighborhood information in cases where state-of-the-art GCNs cannot.

The remainder of the paper is structured as follows. In Section 2, we give the necessary background on graph modeling of road networks and GCNs. In Section 3, we describe RFNs in detail. In Section 4, we report on empirical studies. Finally, we conclude in Section 5.

2 PRELIMINARIES

Road Network Modeling. We model a road network as an attributed, directed graph $G = (V, E, A^V, A^E, A^B)$, where V is the set of nodes and E is the set of edges. Each node $v \in V$ represents an intersection (or the end of a road), and each edge $(u, v) \in E$ represents a road segment that enables traversal from u to v. Next, A^V and A^E maps intersections and road segments, respectively, to their attributes. In addition, A^B maps a pair of road segments $(u, v), (v, w) \in E$ to their between-segment attributes such as the angle between (u, v) and (v, w) based on their spatial representation. An example of a graph representation of the three-way intersection to the right in Fig. 1 is shown to the left in Fig. 2. Attribute information not shown.

Two intersections u and v in V are adjacent if $(u, v) \in E$ or $(v, u) \in E$. Similarly, two road segments (u_1, v_1) and (u_2, v_2) in E are adjacent if $v_1 = u_2$ or $v_2 = u_1$. The function $N\colon V \cup E \rightarrow 2^V \cup 2^E$ returns the neighborhood, i.e., the set of all adjacent intersections or road segments, of a road network element $g \in V \cup E$. The dual graph representation of G given by $G^D = (E, B)$, where $B = \{((u, v), (v, w)) \mid (u, v), (v, w) \in E\}$ is the set of *between-edges*. Thus, E and B are the node and edge sets, respectively, in the dual graph. An example of a dual graph can be seen to the right in Fig. 2. For disambiguation, we refer to G as the primal graph representation.

Graph Convolutional Networks. A GCN is a neural network that operates on graphs and consists of one or more graph convolutional layers. A graph convolutional network takes as input a graph $G = (V, E)$ and a numeric node feature matrix $\mathbf{X}^V \in \mathbb{R}^{|V| \times d_{in}}$, where each row corresponds to a d_{in}-dimensional vector representation of a node. Given these inputs, a GCN computes an output at a layer k s.t.

$$\mathbf{H}_v^{(V,k)} = \sigma(\textsc{Aggregate}^k(\{\mathbf{H}_n^{(V,k)} \mid n \in N(v)\})\mathbf{W}), \qquad (1)$$

where σ is an activation function, and $\textsc{Aggregate}\colon 2^V \rightarrow \mathbb{R}^{d_{in}}$ is an aggregate function, e.g., a mean. As in $\mathbf{X}^V$, each row in $\mathbf{H}^{(V,k)}$ is a vector representation of a node. In some cases, $\mathbf{X}^V$ is linearly transformed using matrix multiplication with a weight matrix $\mathbf{W}$ before aggregation [11], while in other cases, weight multiplication is done after aggregation [5, 9], as in Eq. 1.

3 RELATIONAL FUSION NETWORKS

Relational Fusion Networks (RFNs) aim to address the shortcomings of state-of-the-art GCNs in the context of machine learning on road networks. We now proceed to give a brief introduction of our method. A more detailed description may be found in the full paper [6].

3.1 Overview

The basic premise of the RFN is to learn representations based on two distinct, but interdependent, views: the node-relational and edge-relational views. An RFN consists of K relational fusion layers, where $K \geq 1$. We illustrate a single relational fusion layer in Fig. 3.

Each layer k takes as input the learned node, edge, and between-edge representations from layer $k-1$, denoted by $\mathbf{H}^{(V,k-1)}$, $\mathbf{H}^{(E,k-1)}$, and $\mathbf{H}^{(B,k-1)}$, respectively. The first layer takes as input the feature matrices $\mathbf{X}^V \in \mathbb{R}^{|V| \times d^V}$, $\mathbf{X}^E \in \mathbb{R}^{|E| \times d^E}$, and $\mathbf{X}^B \in \mathbb{R}^{|B| \times d^B}$ that numerically encode the node, edge, and between-edge attributes, respectively. Then *node-relational fusion* and *edge-relational fusion* are performed to learn new node and edge representations $\mathbf{H}^{(V,k)}$ and $\mathbf{H}^{(E,k)}$ from the node- and edge-relational views, respectively.

Using node-relational fusion, we seek to learn representations of nodes, i.e., intersections, based on their node attributes and the relationships between nodes indicated by the edges E in the primal

graph $G^P = (V, E)$ and described by their edge attributes. Similarly, we seek to learn representations of edges, i.e., road segments, using edge-relational fusion, based on their edge attributes and the relationships between edges indicated by the between-edges B in the dual graph $G^D = (E, B)$. The relationship between two adjacent roads (u, v) and (v, w) is described by the attributes of the between-edge connecting them in the dual graph, including the angle between them, but also the attributes of the node v that connects them. These node and edge views are interdependent and can be exploited by RFNs to leverage node, edge, and between-edge attributes simultaneously.

As illustrated in Fig. 3, an RFN captures the interdependence between the node and edge views by using the node and edge representations from the previous layer $k-1$ as input to the node-relational and edge-relational fusion in layer k In addition, each layer applies a regular feed-forward neural network to the between-edge presentations $\mathbf{H}^{(B,k-1)}$ to learn new between-edge representations $\mathbf{H}^{(B,k)}$.

3.2 Relational Fusion

We present the pseudocode for relational fusion at the kth layer in Algorithm 1. The operator takes as input a graph $G' = (V', E')$, that is either the primal or dual graph representation of a road network, along with appropriate feature matrices $\mathbf{H}^{(V',k-1)}$ and $\mathbf{H}^{(E',k-1)}$ that describe nodes and edges in G'. Then, a new representation is computed for each element $v' \in V'$ by first computing relational representations. Given an element v', each relation $(v', n') \in N(v')$ that v' participates in, is converted to a *relational representation*. To be explicit, $G' = G^P = (V, E)$, $\mathbf{H}^{(V',k-1)} = \mathbf{H}^{(V,k-1)}$, and $\mathbf{H}^{(E',k-1)} = \mathbf{H}^{(E,k-1)}$ in the case of node-relational fusion. In the case of edge-relational fusion, $G' = G^D = (E, B)$, $\mathbf{H}^{(V',k-1)} = \mathbf{H}^{(E,k-1)}$, and $\mathbf{H}^{(E',k-1)}$ combines node and between-edge features, e.g., s.t. the representation of a between-edge $((u, v), (v, w)) \in B$ is $\mathbf{H}^{(E',k-1)}_{((u,v),(v,w))} = \mathbf{H}^{(B,k-1)}_{((u,v),(v,w))} \oplus \mathbf{H}^{(V,k-1)}_{v}$, where $\oplus$ denotes vector concatenation.

Algorithm 1 The Relational Fusion Operator

1: **function** RelationalFusion$^k(G' = (V', E'), \mathbf{H}^{(V',k-1)}, \mathbf{H}^{(E',k-1)})$
2: **let** $\mathbf{H}^{(V',k)}$ be an arbitrary $|V'| \times d^{F^k}$ real feature matrix.
3: **for all** $v' \in V'$ **do**
4: $F_{v'} \leftarrow \{$ Fuse$^k(\mathbf{H}^{(V',k-1)}_{v'}, \mathbf{H}^{(E',k-1)}_{(v',n')}, \mathbf{H}^{(V',k-1)}_{n'}) \mid n' \in N(v')\}$
5: $\mathbf{H}^{(V',k)}_{v'} \leftarrow$ Aggregate$^k(F_{v'})$
6: $\mathbf{H}^{(V',k)}_{v'} \leftarrow$ Normalize$^k(\mathbf{H}^{(V',k)}_{v'})$
7: **return** $\mathbf{H}^{(V',k)}$

In Algorithm 1, the relational representations at layer k are computed by a fusion function Fuse^k. For each relation, Fuse^k takes as input representations of the source v' and target n' of the relation, $\mathbf{H}^{(V',k-1)}_{v'}$ and $\mathbf{H}^{(V',k-1)}_{n'}$, respectively, along with a representation $\mathbf{H}^{(E',k-1)}_{(v',n')}$ describing their relation, and then it fuses them. The resulting relational representations are subsequently fed to an Aggregate^k function, that aggregates them into a single representation of v'. Finally, the representation of v' may optionally be normalized by invoking the Normalize^k function., e.g., using L_2 normalization [5]. This latter step is particularly important if the relational aggregate has different scales across elements with different neighborhood sizes.

The relational fusion operator is compatible with many existing aggregators from the GCN literature, e.g., a mean aggregator [5]. We use a single-layer perceptron as the fusion function, i.e.,

$$\text{Fuse}^k(\mathbf{H}^{(V',k-1)}_{v'}, \mathbf{H}^{(V',k-1)}_{n'}, \mathbf{H}^{(E',k-1)}_{(v',n')}) = \sigma\big((\mathbf{H}^{(V',k-1)}_{v'} \oplus \mathbf{H}^{(V',k-1)}_{n'} \oplus \mathbf{H}^{(E',k-1)}_{(v',n')})\mathbf{W}^R + \mathbf{b}\big),$$

where σ is an activation function, $\oplus$ denotes row-wise vector concatenation, $\mathbf{W}^R$ is a weight matrix, and $\mathbf{b}$ is a bias term. We explore aggregator and fusion function designs in the full paper [6].

4 EXPERIMENTAL EVALUATION

To investigate the generality of our method, we evaluate it on two tasks using the road network of the Danish municipality of Aalborg: driving speed estimation and speed limit classification. These tasks represent a regression task and a classification task, respectively.

Many details of the experiments have been omitted due to the page limitation. We refer to the full paper [6] for further information. Our RFN implementation is available online[2].

4.1 Data Set

We extract the spatial representation of the Danish municipality of Aalborg from OpenStreetMap (OSM) [10], and convert it to its primal and dual graph representations as described in Section 2. We combine the OSM data with a zone map from the Danish Business Authority[3], and we derive 3 node features, 16 edge features, and 2 between-edge features from this dataset.

For the driving speed estimation task, we use a dataset of 8 675 599 observed driving speeds, each matched to a road segment, that stem from a set of vehicle trajectories [1]. For the speed limit classification task, we use 19 510 speed limits collected from the OSM data and additional speed limits are collected from the municipality of Aalborg. This dataset is highly imbalanced. Finally, we split speed limits and driving speeds into training, validation, and test sets.

4.2 Experimental Setup

We compare four algorithms in our experiments:

- *MLP*: A regular multi-layer perceptron that performs predictions independent of adjacent road segments by using only the edge features as input.
- *GraphSAGE*: The Max-Pooling variant of GraphSAGE, which achieved the best results in the authors' experiments [5].
- *GAT*: The graph attention network by Veličković et al. [11].
- *RFN*: An RFN using a mean aggregator [5].

The GraphSAGE and GAT models are run on the dual graph representations of the road network s.t. they learn edge representations directly. All models are two-layer models and use the ELU [2] activation function, with the exception that the ReLU [4] activation function is used in the GraphSAGE pooling operation. We

[2] https://github.com/TobiasSkovgaardJepsen/relational-fusion-networks
[3] https://danishbusinessauthority.dk/plansystemdk

select layer sizes, learning rates, and GAT-specific hyperparameters by evaluating different hyperparameter configurations on the validation sets in a grid search and selecting the best-performing configuration.

All algorithms are implemented using the MXNet[4] deep learning library.

Model Training and Evaluation. We initialize the weights of all models using Xavier initialization [3] and train the models using the ADAM optimizer [8] in batches of 256 segments. In preliminary experiments, we observed that all models converged within 20 and 30 epochs for driving speed estimation and speed limit classification, respectively. We therefore use these values for training. For speed limit classification, we use random oversampling on the training set to handle the class imbalance in the dataset and use early stopping to regularize the model.

To train the models, we minimize a per-segment mean squared loss and the binary cross entropy loss for driving speed estimation and speed limit classification, respectively. To evaluate the models, we use a per-segment mean absolute error for driving speed estimation and the F_1 macro score for speed limit classification.

4.3 Results

We report the mean performance and standard deviations of each algorithm across ten runs in Table 1. Note that when reading Table 1, low values and high values are desirable for driving speed estimation and speed limit classification, respectively.

Table 1: Algorithm performance on Driving Speed Estimation (DSE) and Speed Limit Classification (SLC).

Algorithm	*DSE*	*SLC*
MLP	10.160 ± 0.119	0.443 ± 0.027
GraphSAGE	8.960 ± 0.115	0.432 ± 0.014
GAT	9.548 ± 0.151	0.442 ± 0.018
RFN	$\mathbf{7.685 \pm 0.189}$	$\mathbf{0.500 \pm 0.011}$

As can be seen, our proposed RFN outperforms all baselines on both driving speed estimation and speed limit classification. RFN outperforms the state-of-the-art graph convolutional approaches, i.e., GraphSAGE and GAT, by 17% and 24%, respectively, on the driving speed estimation task. On the speed limit classification task, the best RFN outperforms GraphSAGE and GAT by 16% and 13%, respectively. The more sophisticated aggregation and fusion functions that we present in the full version of the paper substantially improve these results s.t. the best RFN variant outperforms GraphSAGE and GAT by 32–40% and 21–24%, respectively [6].

Interestingly, the MLP outperforms the GraphSAGE and GAT (but not the RFN) models on speed limit classification without using the network structure. This suggests that RFNs can leverage road network structure in cases where GraphSAGE and GAT cannot.

5 CONCLUSION

We report on a study of GCNs from the perspective of machine learning on road networks. We argue that many built-in assumptions of existing proposals do not apply in the road network setting, in particular the assumption of smooth homophily in the network. In addition, state-of-the-art GCNs can leverage only one source of attribute information, whereas we identify three sources of attribute information in road networks: node, edge, and between-edge attributes. To address these short-comings, we propose the *Relational Fusion Network (RFN)*, a novel type of GCN for road networks.

We compare the RFN against state-of-the-art GCN algorithms on two machine learning tasks in road networks. We find that the proposed RFN outperforms the GCN baselines significantly on these tasks. Although not presented here, we also investigate alternative aggregation and fusion functions that yield even higher predictive performance [6].

In future work, it is of interest to investigate to which extent RFNs are capable of transferring knowledge from, e.g., one Danish municipality to the rest of Denmark, given that the inductive nature of our algorithm allows RFNs trained on one road network to be used for prediction on another. If the results are positive, it would suggest that RFNs can learn traffic dynamics that generalize to unseen regions of the network. This may make it easier to train RFNs with less data, but also give more confidence in predictions in regions that are labeled sparsely with speed limits. In addition, RFNs do not incorporate temporal aspects, although many road networks tasks are time-dependent. For instance, this applies to driving speed estimation, for which reason we explicitly excluded driving speeds during peak-hours from our experiments. Extending RFNs to learn temporal road network dynamics, e.g., through time-dependent fusion functions that accept temporal inputs, is an important future direction.

ACKNOWLEDGMENTS

The research presented in this paper is supported in part by the DiCyPS project and by grants from the Obel Family Foundation and the Villum Fonden.

REFERENCES

[1] Ove Andersen, Benjamin B. Krogh, and Kristian Torp. 2013. An Open-source Based ITS Platform. In *Proc. of MDM*, Vol. 2. 27–32.
[2] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. 2016. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs) . In *Proc. of ICLR*. 14 pp.
[3] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proc. of AISTATS*. 249–256.
[4] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep Sparse Rectifier Neural Networks. In *Proc. of AISTATS*. 315–323.
[5] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Proc. of NIPS*. 1024–1034.
[6] Tobias Skovgaard Jepsen, Christian S. Jensen, and Thomas Dyhre Nielsen. 2019. Graph Convolutional Networks for Road Networks. *arXiv e-prints* (2019). arXiv:1908.11567
[7] Tobias Skovgaard Jepsen, Christian S. Jensen, Thomas Dyhre Nielsen, and Kristian Torp. 2018. On Network Embedding for Machine Learning on Road Networks: A Case Study on the Danish Road Network. In *Proc. of Big Data*. 3422–3431.
[8] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proc. of ICLR*. 15 pp.
[9] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *Proc. of ICLR*. 14 pp.
[10] OpenStreetMap contributors. 2014. Planet dump retrieved from https://planet.osm.org. (2014).
[11] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph Attention Networks. In *Proc. of ICLR*. 12 pp.
[12] Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2017. Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting. In *Proc. of IJCAI*. 3634–3640.

[4] https://mxnet.incubator.apache.org/