Pattern Recognition 120 (2021) 108174

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Graph representation learning for road type classification

Zahra Gharaee[a,*], Shreyas Kowshik[b], Oliver Stromann[a,c], Michael Felsberg[a]

[a] Computer Vision Laboratory (CVL), Department of Electrical Engineering, University of Linköping Linköping, Sweden
[b] Department of Mathematics, Indian Institute of Technology Kharagpur, India
[c] Autonomous Transport Solutions Research, Scania CV AB, Sweden

ARTICLE INFO

Article history:
Received 17 February 2021
Revised 21 May 2021
Accepted 6 July 2021
Available online 15 July 2021

Keywords:
Road network graphs
Graph representation learning
Line graph transformation
Neighborhood aggregation
Topological neighborhood

ABSTRACT

We present a novel learning-based approach to graph representations of road networks employing state-of-the-art graph convolutional neural networks. Our approach is applied to realistic road networks of 17 cities from Open Street Map. While edge features are crucial to generate descriptive graph representations of road networks, graph convolutional networks usually rely on node features only. We show that the highly representative edge features can still be integrated into such networks by applying a line graph transformation. We also propose a method for neighborhood sampling based on a topological neighborhood composed of both local and global neighbors. We compare the performance of learning representations using different types of neighborhood aggregation functions in transductive and inductive tasks and in supervised and unsupervised learning. Furthermore, we propose a novel aggregation approach, Graph Attention Isomorphism Network, GAIN[1]. Our results show that GAIN outperforms state-of-the-art methods on the road type classification problem.

1. Introduction

Cities around the world are growing and increasingly more people are moving from rural to urban areas. Today, 55% of the world population lives in cities, and by 2050, the number is expected to become 68% [1]. Increased urbanization leads to a stronger need of urban planning and design, which deals with the infrastructure passing into and out of urban areas, such as transportation, communications, and distribution of road networks [1].[1]

In urban planning the physical layout of human settlements is considered as the main subject [2]. Urban design is the process of designing and shaping the physical features of the cities for the provision of municipal services to residents and visitors. In contrast to architecture, which focuses on the design of individual buildings, urban design deals with the larger scale of groups of buildings, infrastructure, streets, public spaces and the whole neighborhoods as well as districts of the entire cities. The main goal is to design urban environments, which are equitable, beautiful, performative, and sustainable [3].

One key element in designing an urban environment is the design of road networks, which ensures efficient traffic flows as well as connectivity. Road networks design also supports other domains such as autonomous vehicles and entertainment industry for instance gaming.

In this article, we design and implement a graph-based architecture capable of performing the following tasks: (1) Learning road networks of realistic cities and towns from open street map to accomplish road type classification. (2) Applying line graph transformation to use qualitative road segment features in learning the representations. (3) Proposing a neighborhood sampling based on the nodes in local and global topological neighborhoods. (4) Proposing a novel approach to aggregation, Graph Attention Isomorphism Network, GAIN, and comparing its performance with a variety of state-of-the-art approaches to representation learning of road network graphs.

2. Related work

One can find at least three main categories of approaches for modeling road networks in the literature: the earliest are procedural methods for modeling of road networks from a set of rules. Next are example-based approaches applying a preprocessing step to extract statistical information. Most recent are learning-based approaches including methods using deep learning techniques.

* Corresponding author.
E-mail addresses: zahra.gharaee@liu.se (Z. Gharaee), shreyaskowshik@iitkgp.ac.in (S. Kowshik), oliver.stromann@liu.se (O. Stromann), michael.felsberg@liu.se (M. Felsberg).

[1] Link to repository: https://github.com/zahrag/GAIN

https://doi.org/10.1016/j.patcog.2021.108174

The procedural modeling of road networks relies on a set of rules; Open L-System [4] gradually generates a road map from initial seed points to conform user guidance, the automatic procedural road generation [5] uses a weighted anisotropic shortest path algorithm and the procedural modeling in [6] applies a hierarchical generation of road networks from a geometric graph using a non-Euclidean metric combined with a path merging algorithm.

The next category of approaches for modeling road networks covers example-based approaches [7]. In contrast to procedural approaches, example-based methods [8] do not utilize a rule set for road generations. They rather analyze the data from road networks or city layouts in a preprocessing step to extract templates as well as statistical information.

The growth of machine learning techniques especially recent advancements in deep learning makes them a powerful tool for road networks learning, prediction and generation. Since road networks are growing extensively, it is required to apply methods capable of dealing with big data. Moreover, access to the satellite images makes it possible to even train deep learning algorithms end-to-end.

2.1. Learning-based methods

In recent years deep learning techniques have been used for procedural and data driven content generation. Among them is the method, which learns a low-dimensional generative model from a high-dimensional procedural model by using shape features [9]. To circumvent the large number of parameters involved in the rule sets and also their non-linear relationship to the resulting content, a sketch-based approach to procedural modeling trains a deep Convolutional Neural Network (CNN) to map sketches into the procedural model parameters [10].

An urban procedural model [11] also trains CNNs to recognize the procedural rule intended by a sketch and estimating its parameters. They use simple procedural grammars to turn sketches into realistic 3D models. A neurally-guided architecture [12] augments procedural models with deep neural networks to control the random selections of different models based on the output.

Using generative models based on deep neural networks have also been extensively studied for graph generations. One example is the method that applies deep learning to generate an initial segmentation of aerial images and feed them to an algorithm, which reasons about missing connections in the extracted road topology [13].

To automatically construct accurate road network maps from aerial images RoadTracker [14] uses an iterative search process guided by a CNN-based decision function to derive road network graph directly from the output of CNN. Graph neural networks is used to express probabilistic dependencies of a graph nodes and edges in order to learn the distributions over any arbitrary graph [15].

The GraphRNN [16] learns to generate graphs by training on a representative set of graphs and decomposes the graph generation process into a sequence of node and edge formations, conditioned on the graph structure generated so far. The Neural Turtle Graphics (NTG) [17] represents the road layout using a graph where nodes of the graph represent control points and edges of the graph represents road segments. NTG is a sequential generative model parameterized by a neural network, which iteratively generates a new node and an edge connecting to an existing node conditioned on the current graph.

There are a number of approaches for graph generation based on Generative Adversarial Networks (GAN). Among them is StreetGAN [18], in which a preprocessing layer is applied to convert a given representation of a road network into a binary image using pixel intensities to encode the presence or absence of streets. The model next trains a GAN to synthesize a multitude of arbitrary sized street networks. Finally the post-processing layer extracts a graph-based representation of the generated images.

The NetGAN [19] generates graphs from random walks and the model is trained using Wasserstein GAN objective function. Using GANs, GraphGAN [20] is proposed as a graph representation learning framework, which applies a new graph softmax to satisfy the properties of normalization, graph structure awareness, and computational efficiency.

2.2. Graph-based methods

Graphs are more commonly applied to describe information across many diverse fields where complex and unstructured data represents a particular conceptual network like social networks, molecular networks, biological protein-protein networks, telecommunication networks or brain connectomes [21].

The generic structures of such networks, especially when compared to the grid-like structure of the images, audio, and text, makes it difficult to analyze their representations. Since graphs are non-Euclidean and the number of vertices and edges can vary arbitrarily, they become a powerful tool for data representation in such irregular domains and therefore, it has been a surge in research on graph representation learning recently [22].

To incorporate high-dimensional non-Euclidean information of the graph structure, earlier approaches relied on hand-engineered features such as degrees or clustering coefficients [23], kernel functions [24] and engineered features to measure local neighborhood structures [25]. However, using hand-engineered features could be an inflexible and expensive task.

Recently learning representations is becoming more popular, which is based on learning a mapping that embeds nodes, or the entire (sub)graphs, as points in a low-dimensional vector space to summarize every node's position and the structure of its local neighborhood [22]. The learning representations facilitates downstream machine learning tasks like link prediction [26] or node classification [27].

Graph structure and methods have also been used to enhance the ability to learn a variety of tasks like image clustering and recognition [28], object identification [29], action recognition [30], and few-shot learning (FSL) [31]. The approach to image clustering and recognition [28] investigates graph characteristics from the heat kernel trace by exploring three different methods to characterizing it as a function of time. To identify objects, spectral graph analysis of a hierarchical description of an image is applied to construct a feature vector of fixed dimension [29].

Among the graph-based methods developing FSL are the EGNN [32], which learns to predict edge-labels rather than node-labels by iteratively updating the edge-labels using both intra-cluster similarity and the inter-cluster dissimilarity, and the DPGN [33], which incorporates distribution propagation in graph neural network to facilitate FSL tasks. In [34], however, a concept graph is used for weakly supervised FSL by applying a meta concept inference network, which quickly adapts to a novel task using the joint inference of the abstract concepts and a few annotated samples.

2.3. Graph convolutional networks

Graph Convolutional Networks (GCN) are a recent generation of approaches, which represent every node of a graph as a function of its neighborhood [26,35,36]. Learning node representations has three important advantages. First is its computational efficiency as a result of sharing network parameters, second is the integration of nodes attributes to generate information about their positions and roles in the graph, and third is the generalization of learned knowledge to the unseen nodes and graphs.

Aggregating the information of a local node neighborhood instead of the entire graph will therefore help to address the limitations of approaches relying on shallow embedding [37,38] and the ones using deep auto-encoders [39,40].

Similar to GCN, GraphSAGE [27] applies convolutional mechanism to train a set of aggregator functions in order to learn aggregated information of a local neighborhood. GraphSAGE can leverage node features in order to learn an embedding function, which generalizes to unseen nodes or graphs. However, GraphSAGE and GCN differ in the aggregation as well as vector combination methods they apply.

An attention based graph convolutional approach, Graph Attention Network (GAT) [21] operates on graph-structured data using masked self-attention layers to improve the performance of prior graph convolution based methods. The gated attention network (GaAN) [41], on the other hand, applies a convolutional subnetwork to control the importance of attention head based on a number of gates.

The more recent, Graph Isomorphism Network (GIN) [42] models injective multiset functions for neighborhood aggregation by developing a theory of deep multi-sets, which parametrizes universal multi-set functions using neural networks such as multi-layer perceptrons, MLP. GIN applies SUM aggregators to implement injective and universal functions over the multi-sets.

Using recent graph representation learning approaches such as GraphSAGE [27], GAT [21], GaAN [41], and GIN [42], a node learns representation by aggregating the information of nodes sampled from its local neighborhood through a certain number of hops, where one hop is moving one layer forward from a node.

Learning graph structure using local neighborhood aggregation, however, is not capable of integrating edge features in graph representation learning as algorithms rely on node features only. However, in road network graphs, edge features are more descriptive and could play a significant role in learning representations. To address this issue, representation fusion of nodes, edges, and between-edges, the Relational Fusion Network (RFN) [43] for road network graphs is proposed. RFN uses both primal and dual graphs where dual graph nodes and edges represent primal graph edges and between-edges, respectively. Applying relational fusion, RFN addresses speed limit classification and speed limit estimation problems in an inductive supervised setting using binary classification.

Similar to RFN [43], we apply dual graph in our experiments, however, we use exclusively dual graph generated by line graph transformation of the original graph in order to make use of informative road segment attributes in learning representations and, therefore, we do not make use of original graph in our analysis. To the best of our knowledge, the basic approach to generate line graph in this article is similar to the one used in dual graph presented in [43].

Moreover, in contrast to RFN, which only addresses speed limit classification and estimation in an inductive supervised setting using binary classes, our experiments explore the four major tasks of multi-class road type classification in unsupervised and supervised transductive as well as inductive settings. A more detailed description of experimental setup used by RFN in comparison to ours is presented in Section 4.2.2.

The unsupervised classification of road networks is an important problem since completing missing labels is logistically demanding and expensive task. In OSMnx [44], labels are frequently left out or road types are miss-labeled. Especially the inductive setting allows training on high-quality densely labeled road network and transferring this knowledge onto more sparsely labeled road networks.

Furthermore, this article proposes a novel approach, GAIN to aggregating knowledge for learning representations used to address road type classification problem. However, GAIN could be applied to learning representations of any types of graphs and not just road type graphs and, therefore, any classification task could be addressed using the same approach.

To enhance learning representations, we also propose sampling from a topological neighborhood composed of both local (GraphSAGE [27]) and global neighbor nodes by applying a new search mechanism presented in Section 3.3. This facilitates using relevant information of the neighbor nodes in further distances to a node.

3. Method

In a standard graph learning problem, a graph is represented by a set of nodes or the vertices connected by a set of links or the edges. This formulation is usually denoted by $G = (V, E)$, where G is the graph, V and E are vertices and edges respectively. In this article, the learning paradigm starts by setting a number of hops, where one hop is counted by moving forward one layer from a node. The number of hops is set to 2 and the graph node attributes are aggregated by the nodes in the deepest hop.

Sampling among nodes of the training set, the algorithm selects two sets of nodes, one set per hop. The second set contains nodes in the neighborhood of the first set. For every node of the second set, it aggregates information of direct neighbors to its own and, therefore, it generates a vector representing itself. Similar to the second set, each node of the first set generates its representation vector by aggregating information from direct neighbors to its own.

3.1. Line graph transformation

Having a first view to the road network graphs, it sounds reasonable to consider road segments as graph edges and crossroads, junctions, and intersections as graph vertices as shown in Fig. 1 (a). However, this approach suffers from a limited feature representation of vertices since there are not sufficient features describing crossroads and intersections that are essential for road network representation.

Furthermore, among many approaches addressing the graph representation learning problem [21,27,41,42], the features used in the learning process describe merely the vertices and not the edges.

To address feature representation problem, we instead replaced graph vertices as roads segments and, therefore, the connectivity between road segments will generate graph edges as shown by Fig. 1 (b). Using this strategy, the informative road features can be utilized in the learning representation process. A more detailed description about implementation of line graph transformation is available in Section 4.1.

3.2. Supervision modes

Supervised learning For supervised learning, ground-truth road type labels of graph vertices are used for calculating a cross-entropy loss function for our multi-class road type classification problem. The representation vectors received from the aggregation function are normalized by l_2 normalization and input to a one-layer fully connected neural network to predict class labels, which are then used to calculate the supervised loss value.

Unsupervised learning A fully unsupervised setting uses a graph-based loss function shown in (1) to the positive case representation vectors sampled from a set of topological neighbors and negative case sampling distribution, P_n:

$$J_G(z_v) = -\log(\sigma(z_v^T z_u)) - \mathop{\mathbb{E}}_{u_n \sim P_n(u)} \log(-\sigma(z_v^T z_{u_n})), \tag{1}$$

where z_v and z_u are the output representation vectors of sampled node v and topological neighbor node $u \in N_t(v)$. The loss function

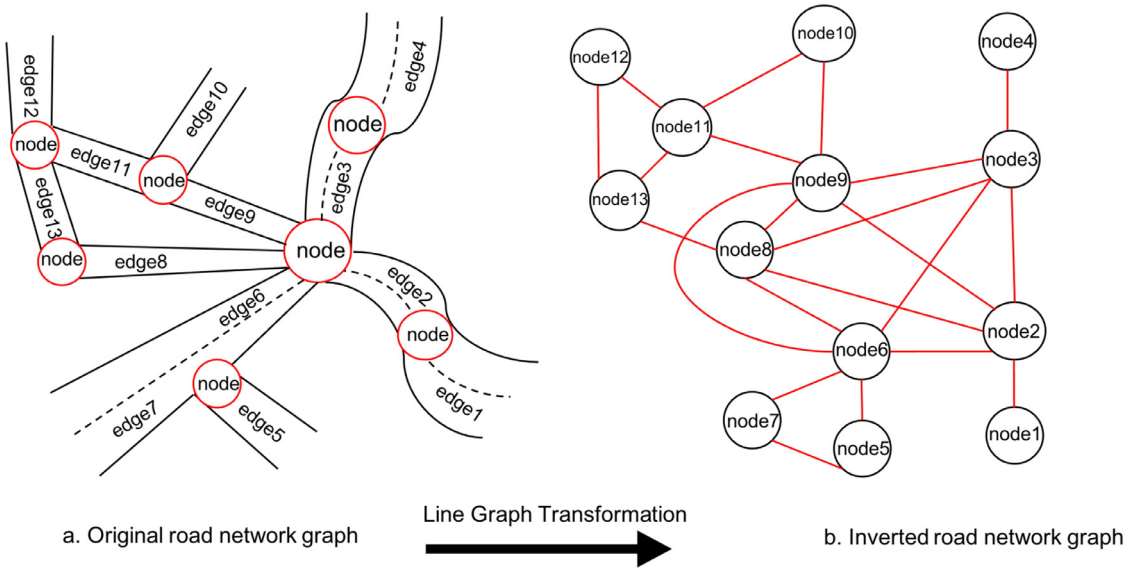

Fig. 1. A sketch of applying the line graph transformation on a road network graph data (a), which results in a new graphical network (b).

uses a sigmoid σ and negative case sampling distribution P_n. Thus, z_{u_n} is a negatively sampled neighbor of node v.

3.3. Building topological neighborhoods

We propose a set of topological neighborhoods, $N_t(v)$ for every sampled node v of the graph, which encompasses both local, $N_l(v)$, and global, $N_g(v)$, neighborhoods. To this end, a search mechanism is implemented, which selects local and a set of global neighbors for a node v based on an unbiased random walk as presented by Algorithm 1.

Algorithm 1 Topological Neighborhood: $N_t(v)$.

Input: $G = (V, E)$ & $N(v \in V) = \{u|(v, u) \in E\}$.
Output: Topological neighborhood $N_t(v)$ of local $N_l(v)$ & global $N_g(v)$ neighbors.

Initialize number of walks N_w, local walk length L_l & global walk length L_g.
for $v \in V$ **do**
  $N_l(v) = N_g(v) = N_t(v) = \{\}$.
  for $n = 1 : N_w$ **do**
    for $l = 1 : L_l$ **do**
      Sample u from $N(v)$.
      if $u \neq v$ **then**
        $N_l(v) \leftarrow u$.
      end if
    end for
    $u_1 = v$.
    Sample u_2 from $N(v)$.
    for $l = 1 : L_g$ **do**
      $u_0 = u_1$.
      $u_1 = u_2$.
      Sample u_2 from $N(u_1)$.
    end for
    if $u_1 \neq v$ **then**
      $N_g(v) \leftarrow u_1$.
    end if
  end for
  $N_t(v) = N_l(v) \cup N_g(v)$
end for

Applying both local and global neighborhood provides a node with two views. Having the first view, a node captures the closer vicinity and it extracts the information of its neighbors in a fixed length local area [27]. We propose adding a second view based on the node global neighborhood to facilitate the extraction of information from related nodes in further distances. This way, we extend the node representation to improve the performance of learning representations.

To generate the second view as mentioned above, we developed a global random walk with fixed length, L_g two times the size of local random walk, L_l as $L_g = 2 \times L_l$. All topological neighbors visible to a node through both views are then mixed and shuffled to be used for training the system.

3.4. Previous approaches proposed for aggregation

As mentioned earlier, a sampled node v aggregates the information of its direct neighbors $u \in N(v)$ in order to generate an output representation vector h_v^k for each hop layer k. There are a number of different approaches used to build the aggregation function, which are mentioned in the following.

GCN A graph convolutional network, GCN [35] aggregates the information as:

$$h_v^k \leftarrow \sigma\left(W \cdot \text{MEAN}\left(h_v^{k-1} \parallel h_{u \in N(v)}^{k-1}\right)\right), \tag{2}$$

where W is the set of weights, associated to the sampled and the neighbor nodes represented by v and u. k iterates over hop layers and σ is the sigmoid function. Concatenation shown by $\parallel$ is applied to the sampled and neighbor nodes representation vectors before applying the MEAN operation.

GraphSAGE GraphSAGE architecture [27] applies a different formulation to aggregation:

$$h_v^k \leftarrow \sigma\left(W \cdot \left(h_v^{k-1} \parallel \text{AGG}\left(h_{u \in \mathcal{N}(v)}^{k-1}\right)\right)\right), \tag{3}$$

where W shows the weights of aggregator functions and σ is the sigmoid function. For each hop k, the representation vectors of direct neighbors $h_{u \in N(v)}^{k-1}$ of the sampled node v are aggregated using an aggregation function AGG and then the aggregated vector is concatenated to the representation vector of the sampled node h_v^{k-1}.

To make a thorough analysis, we made our investigations based on a number aggregation functions [27]: GraphSAGE-MEAN, which applies mean operation to aggregate neighborhood information, GraphSAGE-MEANPOOL, which aggregates information by calculating mean pooling over MLP functions, GraphSAGE-MAXPOOL, which aggregates information using max pooling of neighborhood information over MLP functions and, finally, GraphSAGE-LSTM, which applies a standard LSTM to aggregate the neighborgood information.

GAT The graph attention network, GAT [21], calculates average of the weighted representation vectors of the first order neighbor nodes $u \in N(v)$ (including v) over multiple heads by applying attention weights, α^m_{vu}, to the corresponding neighbors:

$$\mathrm{h}_v^k = \sigma\left(\frac{1}{\mathrm{M}}\sum_{m=1}^{M}\sum_{u\in N(v)} \alpha^{k-1}_{m_{vu}} \mathrm{W}'_m \mathrm{h}_u^{k-1}\right), \tag{4}$$

where M is the total number of attention heads used for regularization and W′ is a set of corresponding input linear transformation weights matrix used to project nodes input features into a higher level feature space.

GIN The graph isomorphism network, GIN [42] models injective multi-set functions for aggregating information by parameterizing multi-layer perceptrons, MLP. GIN employs the aggregation formulation as:

$$\mathrm{h}_v^k \leftarrow \mathrm{MLP}^k\left((1+\epsilon^k)\cdot \mathrm{h}_v^{k-1} + \sum_{u\in N(v)} \mathrm{h}_u^{k-1}\right), \tag{5}$$

where MLP is the multi-layer perceptron and ϵ could be either learned by gradient descent as one variable of the network or be fixed to zero.

3.4.1. Novel formulation of the aggregation function

As an alternative to the explained aggregation methods, we developed a novel formulation of aggregation, the Graph Attention Isomorphism Network (GAIN)[2] Using this approach, a node aggregates information of its neighbors based on an importance value given to each neighbor node and applies the SUM for aggregation:

$$\mathrm{h}_v^k = \mathrm{MLP}^k\left((1+\epsilon^k)\cdot \mathrm{h}_v'^{k-1} + \sigma \sum_{u\in N(v)} a^{k-1}_{v,u}\cdot \mathrm{h}_u'^{k-1}\right), \tag{6}$$

where $\mathrm{h}'_u = \mathrm{W}'_m \cdot \mathrm{h}_u$ shows the linear transformation of node u into a higher level feature space using weight matrix W'_m. The attention weight $a_{v,u}$ is given to the neighbor node $u \in N(v)$.

In our implementations, we applied one MLP function with one hidden layer since an MLP can represent the composition of functions [45]. We implemented σ using both non-linearity ELU function and Identity function, and we observed the performance was slightly superior applying Identity function.

Inspired by GAT [21], a feed-forward neural network with weights W_a is applied to the concatenation of sampled and neighbor nodes. This concatenated vector is then given to a non-linear leaky RELU function:

$$\hat{a}_{w_{v,u}} = \mathrm{RELU}\left(\mathrm{W}_a \cdot \left(\mathrm{h}_v'^{k-1} \,\|_{u\in N(v)}\, \mathrm{h}_u'^{k-1}\right)\right). \tag{7}$$

Finally, a soft-max function is used to create attention weights:

$$a_{w_{v,u}} = \frac{\exp(\hat{a}_{w_{v,u}})}{\sum_{u\in N(v)} \exp(\hat{a}_{w_{v,u}})}. \tag{8}$$

[2] Isomorphism term in GAIN is used as a reference to GIN representing the bases of our approach, however our proposed approach to aggregation does not fully hold injective features due to the application of non-linearity in σ and attention weights.

Compared to GAT [21], GAIN applies attention weights only to the neighboring nodes of v (excluding v) and it uses SUM rather than MEAN over attention heads. On the other hand, GAIN applies attention weights to aggregate neighborhood representations rather than just aggregating neighborhood representations without weighting them as proposed by GIN [42].

GAIN formulation presented in (6) considers one attention head, however, the mathematical formulation of GAIN using multiple attention heads is presented in Appendix A.1. Our experimental investigation shows no improvement in performance applying multiple attention heads.

4. Experiments

To evaluate the ability to generalize to unseen graphs, we designed experiments using two different data sets, inductive vs. transductive setting, which denotes inductive reasoning as inferring knowledge from specific to general and transductive reasoning as inferring knowledge from specific to specific [46].

Hence, for the inductive experiments we report the results of learning representations on test sets containing graphs of unseen cities and for the transductive experiments we report the results of learning representations on test set of unseen nodes of the same graph. Another split of our experiments is designed to compare the performance of unsupervised vs. supervised learning.

4.1. Input dataset

To address the main problems of this project, road network datasets are represented as graphs composed of vertices and edges. To test transductive and inductive capabilities of the assessed methods, we generate two datasets of road networks. Using Open Street Map (OSMnx [44]), we extract the crowd-sourced geographic information of road networks in Swedish Cities from OSMnx.

Both datasets are preprocessed in the following way. The OSMnx data of driving roads is extracted from a 14 km × 14 km tile centered at the city centroid. The resulting graph is simplified such that intersections are consolidated within a 10 m distance and interstitial nodes are reduced. Directions of edges are removed and parallel edges are consolidated, a limitation necessary to apply the graph representation learning methods.

We convert graph G into a line graph $L(G)$, as described in Section 3.1. Each edge of G becomes a node in $L(G)$ and two edges that share a common node in G become an edge in $L(G)$. Fig. 2 illustrates edges of the original graph G colored by their different ground truth road type labels. Overlaid is the line graph representation with black edges and colored nodes corresponding to road type label.

The transductive line graph consists of the road network of Linköping (population of 106,502) with 500 nodes held out for validation and 1000 nodes for testing. The inductive line graph consists of disjoint road networks of 17 Swedish Cities with populations between 50.000 and 150.000 inhabitants. We excluded the three major cities Stockholm, Göteborg and Malmö as well as any suburbs of them, as they have different magnitudes of populations and exhibit different road network characteristics. Two cities each were held-out for validation and testing respectively and, therefore, 13 graphs are allocated to training the network. Table1 provides a summary of the transductive and inductive datasets.

4.1.1. Raw features

To address the road-type classification problem, we extracted descriptive attributes of road-segments represented by the edges of the original graph, G as well as the nodes of the transformed graph, $L(G)$. These attributes are used to create raw feature vectors

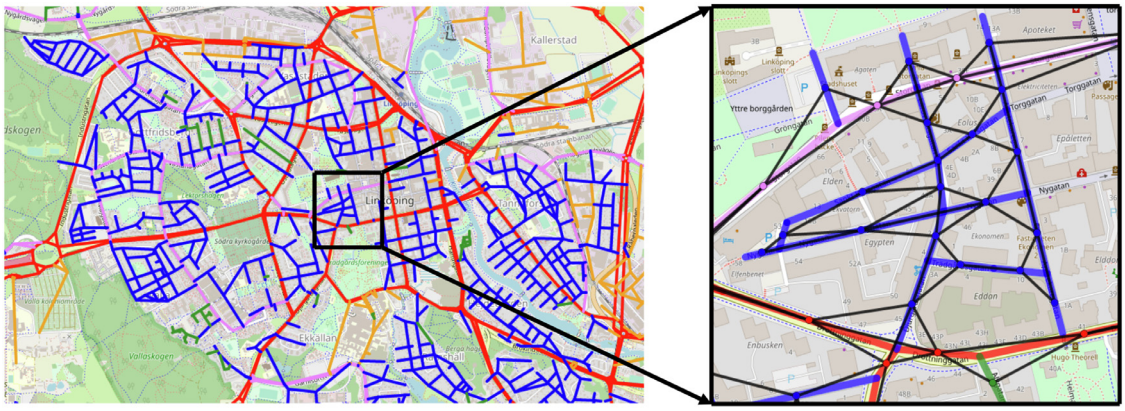

Fig. 2. Input data generation as road network graphs. (a) Linköping road networks represented as a graph. Colors represent different ground truth labels of road types. (b) A closeup of Linköping road networks represented as a graph with a line graph representation overlaid in black. Colors represent different ground truth labels of road types.

Table 1
Summary of data sets created for our experiments.

Road Network Datasets		
Task	Transductive	Inductive
# Graphs	1	17
# Nodes	6903	66580
# Edges	13199	128632
# Raw Features	56	58
# Road-Type Classes	5	5
# Training Nodes	5403	53494
# Validation Nodes	500	6575
# Test Nodes	1000	6511
Avg. Node Degree	3.8241	3.8640

applied in the graph representation learning task. In general the feature vector is composed of 4 main component described as:

- The length of road segments with 1 dimension.
- The midpoint coordinates of adjacent start and end nodes in longitude and latitude with 2 dimensions.
- The geometry sampled to a fixed-length vector of 20 equally distanced points along the length of road segments, which is subtracted by the midpoint coordinate (i.e. 20 longitudinal and latitudinal distances from the midpoint) and is composed of 40 dimensions.
- The one-hot-encoding of the speed limits with 13 and 15 standard values for transductive and inductive tasks, respectively.

As a consequence, the raw feature vectors of roads segments is composed of 56 and 58 values for the transductive and inductive tasks, respectively.

4.1.2. Ground-truth labels

To accomplish supervised experiments, it is required to have graph vertices annotated using ground-truth labels. In OSMnx, roads are tagged with road type labels applicable for the classification tasks. However, due to extreme class imbalances shown by Fig. 3, some classes rarely occurring in our data set. Therefore, we chose to merge and relabel classes according to the following scheme:

- Class (1): Highway, yes, primary, secondary, motorway-link, trunk-link, primary-link, secondary-link.
- Class (2): Tertiary, tertiary-link.
- Class (3): Road, planned, unclassified (minor roads of lower classification than tertiary).
- Class (4): Residential.
- Class (5): Living-street.

4.2. Results

The input dataset of road networks graphs described in Section 4.1 are then used to train the graph representation learning algorithm using 8 different aggregation functions. The experiments are designed to investigate performances of different aggregation functions first when learning supervised vs. unsupervised and then when performing an inductive vs. a transductive task.

As shown in Table 2a, the experiments are conducted by 8 different graph representation learning approaches using ADAM optimiser through an exhaustive grid search over a set of different learning rates and output dimensions. The best performing model for each approach on the validation set in terms of micro-averaged F1-Score is selected and tested on the test set.

4.2.1. Hyperparameter settings

The settings of all hyperparameters required to run experiments are presented in Table 3. Learning rate and output dimensions are used to conduct exhaustive grid search to find the best performing model based on the validation set.

Unsupervised For all unsupervised experiments, an exhaustive grid search is conducted over the learning rates of $\{2e^{-8}, 2e^{-7}, 2e^{-6}, 2e^{-5}\}$ and the output dimensions of $\{64, 128, 256\}$ for the representation vectors at every depth k of the recursion. There are 9 and 3 neighbors sampled for aggregation in the first and second hop layers, respectively. There are 12 negative neighbors sampled and selected for the unsupervised graph-based loss function shown by (1) and a dropout rate of 0.1 is used. We have also made an exhaustive grid search to study the role of ϵ in GAIN (6) by first fixing it to zero and then as a variable of the network learned by gradient descent initialized by 0.001 and 0.5. Further investigations about the role of ϵ could be conducted in future studies.

For each unsupervised graph representation learning approach, the combination of learning rates and output dimensions result in 12 models where each model is trained for 1000 epochs (Figs. 4(a) and Fig. 4c) with a batch size of 1024 for the transductive and 2048 for the inductive tasks.

Mean values of micro-averaged F1-Scores for 1000 runs of the classifier on the representation vectors generated by each model are then calculated on the validation set and used to select our best performing model. Finally, micro-averaged F1-Score of 1000

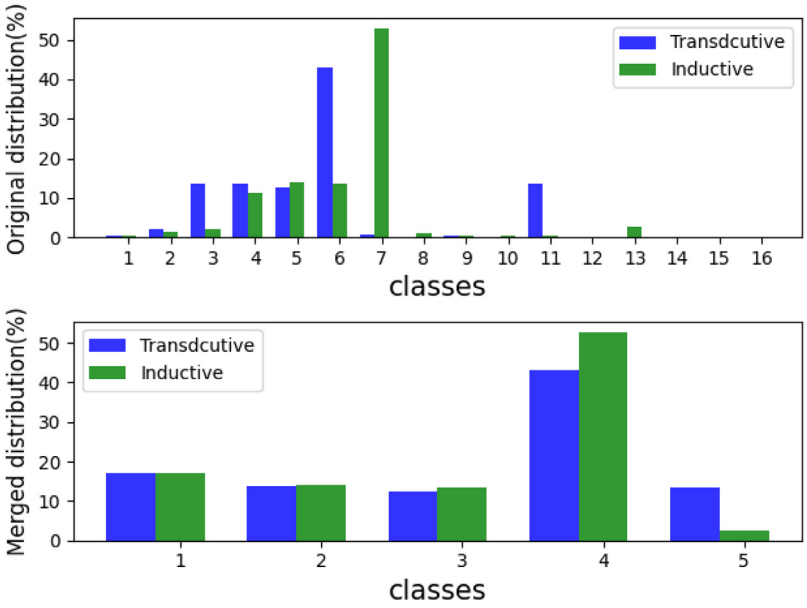

Fig. 3. Percentage of road network class distribution for transductive and inductive data sets generated by OSMnx [44]. As shown by the upper image the class distribution is extremely unbalanced and, therefore, we merged classes to get a more balanced distribution of class labels shown by the bottom image.

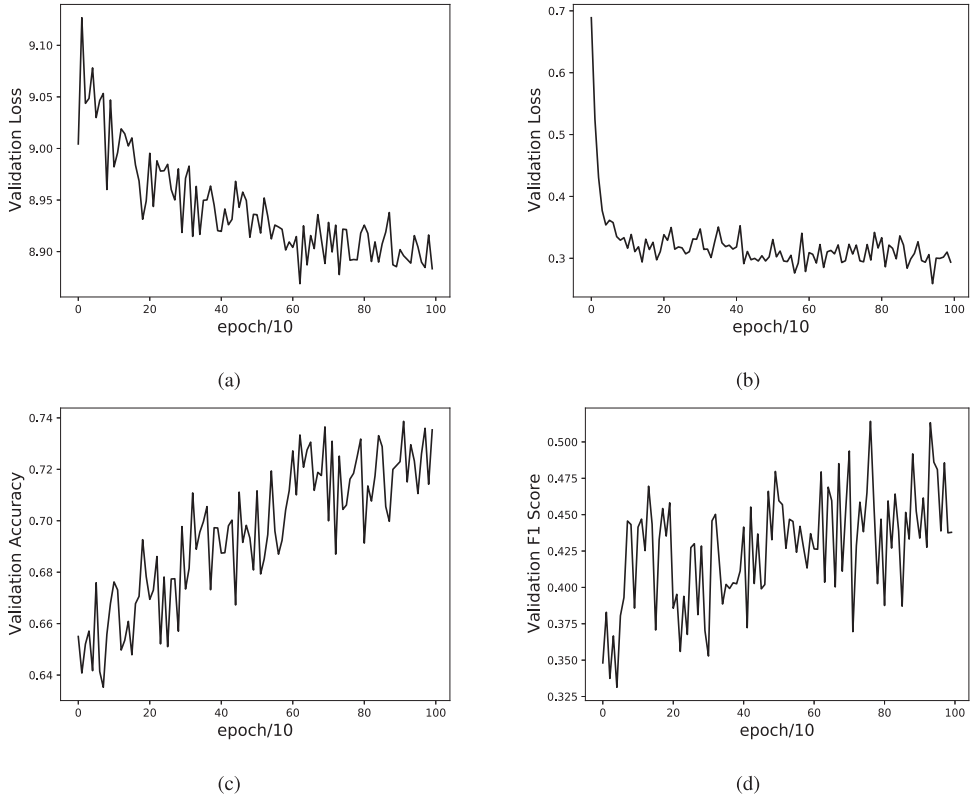

Fig. 4. Performance of GraphSAGE-MEAN: validation loss (a) and accuracy (c) of the unsupervised model and validation loss (b) and F1 score of the supervised model (d) calculated every 10 training epochs for a total number of 1000 epochs. Based on the performance development we set 1000 and 500 epochs to train unsupervised and supervised models, respectively.

runs of the classifier on the representation vectors generated by the best performing model using the test set is reported in Table 2.

The standard deviations of the F1-Scores tend to zero applying 1000 runs of the classifier on the representation vectors generated by each model.

Supervised In all supervised experiments, the exhaustive grid search is conducted over the learning rates of $\{1e^{-4}, 1e^{-3}, 1e^{-2}\}$ while the output dimensions of the representation vectors at every depth k of the recursion are set to $\{64, 128, 256\}$. There are 9 and 3 neighbors sampled for aggregation in the first and second hop layers, respectively. There is a dropout rate of 0.1 used in this setting. Similar to our unsupervised settings, an exhaustive grid search is conducted to study the role of ϵ in GAIN (6).

Table 2
Performance of different graph representation learning neural networks trained unsupervised and supervised to accomplish transductive (a) and inductive (b) tasks in order to address road type classification problem. The results represented by micro-averaged F1 scores are compared with both random baseline performance and when only the raw features are given to the classifier. Gain over baseline computes the percentage of improvement to when using raw features only. The results show a challenging classification problem mainly due to the imbalances apparent in the class distribution presented in Section 4.2.1.

(a) Transductive-Task

Approach	Unsup.	Sup.
Random Baseline	0.20	0.20
Raw Features	0.59	0.59
GCN	0.60	0.58
GSAGE-MEAN	0.67	0.62
GSAGE-MEANPOOL	0.69	**0.81**
GSAGE-MAXPOOL	0.68	0.80
GSAGE-LSTM	0.69	**0.81**
GAT	0.69	0.75
GIN	0.69	0.78
GAIN (ours)	**0.71**	**0.81**
%gain over Baseline	20%	37%

(b) Inductive-Task

Approach	Unsup.	Sup.
Random Baseline	0.20	0.20
Raw Features	0.49	0.49
GCN	**0.61**	0.43
GSAGE-MEAN	0.60	**0.60**
GSAGE-MEANPOOL	**0.61**	0.45
GSAGE-MAXPOOL	0.55	0.44
GSAGE-LSTM	0.59	0.45
GAT	0.51	0.43
GIN	0.47	0.46
GAIN (ours)	0.56	**0.59**
%gain over Baseline	24%	22%

Table 3
The table shows parameter settings for the experiments. *Dimension* denotes the number of output dimensions of the representation vector at every depth k of the recursion, which is set the same for both first and second layers. *Task1* and *Task2* represent transductive and inductive tasks, respectively.

Parameter Settings		
Supervision	Unsupervised	Supervised
Learning rate	$\{2e^{-8}, 2e^{-7}, 2e^{-6}, 2e^{-5}\}$	$\{1e^{-4}, 1e^{-3}, 1e^{-2}\}$
Dimension	{64, 128, 256}	{64, 128, 256}
Epochs	1000	500
Sample Nodes	Layer1(9), Layer2(3)	Layer1(9), Layer2(3)
Dropout	0.1	0.1
Batch size	Task1(1024), Task2(2048)	Task1(1024), Task2(2048)

For each supervised graph representation learning approach, all combinations of learning rates and output dimensions result in 9 models where each model is trained for 500 epochs (Figs. 4b and Fig. 4d) with a batch size of 1024 for the transductive and 2048 for the inductive tasks.

Mean values of micro-averaged F1-Scores of the class labels predicted by each model are then calculated for the validation set and used to select the best performing model when learning the task supervised. Finally, micro-averaged F1-Score of our best model on the test set is calculated and reported in Table 2.

4.2.2. Comparison to RFN

An RFN[43] consists of relational fusion layers where each layer is generated by a single layer perceptron and takes as input the node, edge, and between-edge representations from previous layer.

To make a thorough comparison, the major differences of the method proposed in this study with RFN are mentioned in the following. In both cases road network graphs are extracted from OSMnx [44] and used to generate input dataset, however RFN uses directed road network graphs of Danish cities while we make use of un-directed road network graphs of the Swedish cities.

RFN makes use of 3 raw node features represented by one-hot-encoded three dimensional vectors of city, rural, and summer cottage zones categories together with 10 raw edge features represented by one-hot encoded nine-dimensional vectors of road segment categories and one-dimensional length. Finally, they used 5 raw between-edge features represented by one-hot encoded four-dimensional vector of turn-direction together with one-dimensional turn angle.

There are two sets of experiments conducted by RFN to address speed limit classification and speed limit estimation problems. The experiments are performed for the inductive task using a supervised binary classification on 4 versions of RFN based on the fusion and aggregation functions. We, on the other hand, address the road-type classification problem of multi-class input space using both supervised and unsupervised learning performed for the transductive as well as the inductive tasks.

To run our experiments with RFN, we utilized our graphs node, edge and between-edge features similar to other experiments performed in this study. The node features are represented by 2D node coordinates on the world map and the edge features are composed of 4 main components as described in Section 4.1.1. Since between-edges represent common-nodes of the original graph, we, therefore, used the common-nodes attributes to generate between-edge features.

For comparison, we use the best performing model of [43], RFN-A-I, which applies GAT [21] aggregation and interactional fusion. Our supervised experiments with RFN-A-I are conducted using the soft-max cross-entropy loss function and ground-truth road type labels of multi-class input in our inductive task for 500 epochs. Similar to our other experiments, an exhaustive grid search is conducted over the learning rates of $\{1e^{-4}, 1e^{-3}, 1e^{-2}\}$ and the output dimensions of $\{64, 128, 256\}$ for the representation vectors at every depth k of the recursion. The best performing model on the validation set is selected and tested on the test set. The results of 0.43% accuracy for the transductive and 0.54% accuracy for the inductive tasks are achieved, respectively.

5. Discussion

In this article, we propose a novel graph representation learning approach to address a road type classification problem. We also investigate state-of-the-art graph convolutional neural networks and their applications on road networks extracted from crowd-sourced open data. To this end, we designed 4 sets of experiments with a combination of transductive and inductive tasks when applying supervised and unsupervised learning.

To generate road segments representation vectors using GCN, GraphSAGE, GAT and GIN, we propose a transformation of the original graph into its line graph, where its nodes are composed of the original graph edges. Using this approach, we can perform downstream machine learning tasks on the road network as we demonstrate with the road type classification in our experiments.

To address the classification problem especially on large graphs, we proposed using a topological neighborhood composed of both local and global neighbors to train the graph-based loss function. This expands the node visibility of the graph structure and eventually improves performance. However, the extension of the topological neighborhood increases the amount of time and resources required to train the network. To address this limitation, we created the topological neighborhood of the node using one set of global neighbors only.

We developed a novel approach to aggregation, Graph Attention Isomorphism Network (GAIN), which based on our experiment re-

sults GAIN outperforms the state-of-the-art methods on unsupervised transductive task, and its performance is competitive in supervised transductive and inductive experiments.

For comparison, we conducted experiments with a number of state-of-the-art graph convolutional neural network methods proposed to aggregate the information of a node with its neighboring nodes. These are 7 state-of-the-art methods, including GCN [35], GraphSAGE (MEAN, MEANPOOL, MAXPOOL and LSTM) [27], GAT [21] and GIN [42], trained using our road network data sets.

To make evaluations, we based our comparisons of the results with baseline performance achieved by random baseline and by applying only the raw features to the classifier. Baseline results are then compared with the performance of using representation vectors generated by different graph representation learning approaches shown in Table 2.

As shown in Table 2a, all graph representation learning networks outperform the baseline on both supervised and unsupervised transductive task. However, GAIN outperforms the rest of networks in unsupervised experiments with 20% improvement of performance compared to using raw features only. In supervised experiments, GAIN, GraphSAGE-LSTM and GraphSAGE-MEANPOOL perform the best with 37% improvement over the classification results when using raw features only.

According to the results shown by Table 2b, unsupervised inductive experiments are more successful than supervised ones. We hypothesize that using our proposed topological neighborhood composed of both local and global neighboring nodes, improves the classification of unsupervised experiments. However, the inductive experiment results shown in Table 2b, also confirm that applying representation vectors trained by graph representation learning networks is superior to the baseline performance of road type classification problem.

In the unsupervised inductive task, GraphSAGE-MEANPOOL and GCN achieve the best results, which is 24% above the baseline performance using raw features only. In supervised inductive experiments only GraphSAGE-MEAN and GAIN outperform baseline performance, with a maximum 22% improvement to the baseline results when using raw features.

As presented by Table, road type classification task is quite challenging, which is mainly due to the input data characteristics such as imbalances of class distributions presented in Section 4.1.2, and Fig. 3, upper image. Our motivation to merge some of the classes tries to address this problem, however, the problem still remains (see Fig. 3, bottom image). Moreover, it generates identical roads with different features, which are labeled the same.

We also evaluate the performance of best RFN model, RFN-A-I [43] to address road type classification problem when learning supervised. Our results with RFN shows that it outperforms baseline performance on the inductive task but it fails on the transductive task. This could relate to the types of features used to describe nodes, edges and between-edges. Our results show that GAIN outperforms RFN [43] in supervised representation learning of road network graphs applying informative road segments attributes only.

6. Conclusion

In this paper, a novel approach to graph representation learning, GAIN, is proposed, which outperforms state-of-the-art methods shown in a wide set of experiments. GAIN is motivated through exploring the applications of different graph representation learning approaches in how to aggregate information available in the graph nodes vicinity using a designed topological neighborhood.

Highly representative attributes of the original graph edges, road segments, are applied to train graph vertices representation vectors using a line graph transformation. This addresses the problem of limited feature representation of original graph vertices (crossroads, junctions and intersections) since there are not sufficient features describing crossroads and intersections that are essential for road network representation.

To expand the graph nodes visual space, neighbor nodes are sampled from a topological neighborhood composed of both local and a set of associated global nodes through application of a random walk search mechanism proposed by algorithm 1.

A main objective of this study is to present the application of our approach to unsupervised classification of real world road network graphs, since the complete annotation of the world is extremely expensive and it is also too cumbersome to re-adjust labeling of the roads, which are wrongly labeled. However, any classification task could be addressed using the same approach proposed in this paper.

Shown by our results the current work might be not ready for practical applications of road network graphs. It is still required to apply further tricks and regularization to improve its performance. As a future step, the challenges of road type classification could be addressed in a more realistic setting having more information of before and after segments as well as more informative road attributes.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation and the Swedens innovation agency, Vinnova, through project iQDeep (number 2018–02700).

Appendix A

Appendix with supplementary information regarding training and evaluation of the system.

A1. Multi-head GAIN

GAIN proposed by (6) can be easily extended using multiple heads for regularization, which might be interesting for other applications than the one presented in this article:

$$\mathrm{h}_v^k = \mathrm{MLP}^k(\mathrm{W}\cdot((1+\epsilon^k)\cdot \mathrm{h}_v^{k-1} + \sigma \sum_{m=1}^{M}\sum_{u\in N(v)} a_{m_{v,u}}^{k-1}\cdot \mathrm{h}_u'^{k-1})), \quad (9)$$

where $\mathrm{h}_u' = \mathrm{W}_m'\cdot \mathrm{h}_u$ shows the linear transformation of node u into a higher level feature space using weight matrix W_m'. The attention weight $a_{m_{v,u}}$ is given to the neighbor node $u \in N(v)$ where m iterates over a number of attention heads used for regularization. W represents the shared weights matrices applied to optimize representation vectors of the sampled and the neighbor nodes. We also made experiments using (9) applying $M = 1$ and we gained the results of unsupervised and supervised transductive task as 0.69 and 0.80, while for the unsupervised and supervised inductive task, we achieved the results as 0.60 and 0.57. This indicates that applying single head attention with different settings, (6) performs slightly better to when applying (9) in most cases of our experiments. Increasing the number of heads might improve the results for a more regularization, however it will be much more time consuming.

References

[1] L. Allison, What is urban planning for? Town Planning Review 57 (1) (1986) 5.
[2] N. Taylor, Urban planning theory since 1945, Sage, 1998.
[3] G. Boeing, D. Church, H. Hubbard, J. Mickens, L. Rudis, Leed-nd and livability revisited, Berkeley Planning Journal 27 (1) (2014) 31–55.
[4] Y.I.H. Parish, P. Müller, Procedural modeling of cities, in: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, 2001, pp. 301–308.
[5] E. Galin, A. Peytavie, N. Maréchal, E. Guérin, Procedural generation of roads, in: Computer Graphics Forum, volume 29, Wiley Online Library, 2010, pp. 429–438.
[6] E. Galin, A. Peytavie, E. Guérin, B. Beneš, Authoring hierarchical road networks, in: Computer Graphics Forum, volume 30, Wiley Online Library, 2011, pp. 2021–2030.
[7] A. Emilien, U. Vimont, M.-P. Cani, P. Poulin, B. Benes, Worldbrush: interactive example-based synthesis of procedural virtual worlds, ACM Transactions on Graphics (TOG) 34 (4) (2015) 1–11.
[8] G. Nishida, I. Garcia-Dorado, D.G. Aliaga, Example-driven procedural urban roads, in: Computer Graphics Forum, volume 35, Wiley Online Library, 2016, pp. 5–17.
[9] M.E. Yumer, P. Asente, R. Mech, L.B. Kara, Procedural modeling using autoencoder networks, in: Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology, 2015, pp. 109–118.
[10] H. Huang, E. Kalogerakis, E. Yumer, R. Mech, Shape synthesis from sketches via procedural models and convolutional networks, IEEE Trans Vis Comput Graph 23 (8) (2016) 2003–2013.
[11] G. Nishida, I. Garcia-Dorado, D.G. Aliaga, B. Benes, A. Bousseau, Interactive sketching of urban procedural models, ACM Transactions on Graphics (TOG) 35 (4) (2016) 1–11.
[12] D. Ritchie, A. Thomas, P. Hanrahan, N.D. Goodman, Neurally-guided procedural models: learning to guide procedural models with deep neural networks, in: 30th Conference on Neural Information Processing Systems (NIPS), 2016, pp. 1–9.
[13] G. Máttyus, W. Luo, R. Urtasun, Deeproadmapper: Extracting road topology from aerial images, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 3438–3446.
[14] F. Bastani, S. He, S. Abbar, M. Alizadeh, H. Balakrishnan, S. Chawla, S. Madden, D. DeWitt, Roadtracer: Automatic extraction of road networks from aerial images, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4720–4728.
[15] Y. Li, O. Vinyals, C. Dyer, R. Pascanu, P. Battaglia, Learning deep generative models of graphs, in: Proceedings of the International Conference on Learning Representation (ICLR), 2018, pp. 1–22. http://arxiv.org/abs/1803.03324
[16] J. You, R. Ying, X. Ren, W.L. Hamilton, J. Leskovec, GraphRNN: Generating realistic graphs with deep auto-regressive models, in: J.G. Dy, A. Krause (Eds.), Proceedings of the 35th International Conference on Machine Learning, ICML, volume 80, PMLR, 2018, pp. 5694–5703.
[17] H. Chu, D. Li, D. Acuna, A. Kar, M. Shugrina, X. Wei, M.-Y. Liu, A. Torralba, S. Fidler, Neural turtle graphics for modeling city road layouts, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 4522–4530.
[18] S. Hartmann, M. Weinmann, R. Wessel, R. Klein, Streetgan: Towards road network synthesis with generative adversarial networks, Václav Skala-UNION Agency, 2017.
[19] A. Bojchevski, O. Shchur, D. Zügner, S. Günnemann, NetGAN: Generating graphs via random walks, in: Proceedings of the 35th International Conference on Machine Learning, volume 80, PMLR, 2018, pp. 610–619.
[20] H. Wang, J. Wang, J. Wang, M. Zhao, W. Zhang, F. Zhang, X. Xie, M. Guo, GraphGAN: Graph representation learning with generative adversarial nets, in: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, AAAI Press, 2018, pp. 2508–2515.
[21] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph attention networks, in: 6th International Conference on Learning Representations, ICLR, OpenReview.net, 2018, pp. 1–12.
[22] W.L. Hamilton, R. Ying, J. Leskovec, Representation learning on graphs: methods and applications, IEEE Data Engineering Bulletin 40 (3) (2017) 52–74.
[23] S. Bhagat, G. Cormode, S. Muthukrishnan, Node Classification in Social Networks, in: Social network data analytics, Springer, 2011, pp. 115–148.
[24] S.V.N. Vishwanathan, N.N. Schraudolph, R. Kondor, K.M. Borgwardt, Graph kernels, The Journal of Machine Learning Research 11 (2010) 1201–1242.
[25] D. Liben-Nowell, J. Kleinberg, The link-prediction problem for social networks, Journal of the American society for information science and technology 58 (7) (2007) 1019–1031.
[26] B.R. van den, T.N. Kipf, M. Welling, Graph convolutional matrix completion, in: KDD18 Deep Learning Day, 2018, pp. 1–7.
[27] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: Advances in neural information processing systems, 2017, pp. 1024–1034.
[28] B. Xiao, R.H. Edwin, C.W. Richard, Graph characteristics from the heat kernel trace, Pattern Recognit 42 (2009) 2589–2606.
[29] B. Xiao, S. Yi-Zhe, P. Hall, Learning invariant structure for object identification by using graph methods, Comput. Vision Image Understanding 115 (2011) 1023–1031.
[30] Y. Chen, G. Ma, C. Yuan, B. Li, H. Zhang, F. Wang, W. Hu, Graph convolutional network with structure pooling and joint-wise channel attention for action recognition, Pattern Recognit 103 (2020) 107321, doi:10.1016/j.patcog.2020.107321.
[31] V.G. Satorras, J.B. Estrach, Few-shot learning with graph neural networks, in: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30, - May 3, 2018, Conference Track Proceedings, OpenReview.net, 2018, pp. 1–13.
[32] J. Kim, T. Kim, S. Kim, C.D. Yoo, Edge-labeling graph neural network for few-shot learning, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Computer Vision Foundation / IEEE, 2019, pp. 11–20, doi:10.1109/CVPR.2019.00010.
[33] L. Yang, L. Li, Z. Zhang, X. Zhou, E. Zhou, Y. Liu, DPGN: distribution propagation graph network for few-shot learning, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, IEEE, 2020, pp. 13387–13396, doi:10.1109/CVPR42600.2020.01340.
[34] B. Zhang, K.-C. Leung, X. Li, Y. Ye, Learn to abstract via concept graph for weakly-supervised few-shot learning, Pattern Recognit 117 (2021) 107946, doi:10.1016/j.patcog.2021.107946.
[35] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: Proceedings of the International Conference on Learning Representation (ICLR), 2017, pp. 1–14.
[36] T.N. Kipf, M. Welling, Variational graph auto-encoders, in: Proceedings of the NIPS Workshop on Bayesian Deep Learning, 2016, pp. 1–3.
[37] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: Online learning of social representations, in: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, 2014, pp. 701–710.
[38] A. Grover, J. Leskovec, node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 2016, pp. 855–864.
[39] S. Cao, W. Lu, Q. Xu, Deep neural networks for learning graph representations, in: AAAI, volume 16, 2016, pp. 1145–1152.
[40] D. Wang, P. Cui, W. Zhu, Structural deep network embedding, in: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 2016, pp. 1225–1234.
[41] J. Zhang, X. Shi, J. Xie, H. Ma, I. King, D.-Y. Yeung, GaAN: Gated attention networks for learning on large and spatiotemporal graphs, in: Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence, UAI, AUAI Press, 2018, pp. 339–349.
[42] K. Xu, W. Hu, J. Leskovec, S. Jegelka, How powerful are graph neural networks? in: 7th International Conference on Learning Representations, ICLR, OpenReview.net, 2019, pp. 1–17.
[43] T.S. Jepsen, C.S. Jensen, T.D. Nielsen, Relational fusion networks: graph convolutional networks for road networks, IEEE Trans. Intell. Transp. Syst. (2020) 1–12, doi:10.1109/TITS.2020.3011799.
[44] G. Boeing, Osmnx: new methods for acquiring, constructing, analyzing, and visualizing complex street networks, Comput Environ Urban Syst 65 (2017) 126–139.
[45] B. Weisfeiler, A.A. Lehman, A reduction of a graph to a canonical form and an algebra arising during this reduction, Nauchno-Technicheskaya Informatsia 2 (9) (1968) 12–16.
[46] W.L. Hamilton, Graph representation learning, Synthesis Lectures on Artificial Intelligence and Machine Learning 14 (3) (2020) 1–159.

Zahra Gharaee received the B.Sc. degree in electrical control engineering from the University of Khaje-Nasir Toosi, Tehran, Iran, in 2009, the M.Sc. degree in mechatronics from the University of Khaje-Nasir Toosi, Tehran, Iran, in 2012, and the Ph.D. degree in cognitive science from the Department of Philosophy and Cognitive Science, Lund University, Lund, Sweden, in 2018. She is currently Post.Doc. at the Computer Vision Laboratory (CVL), Department of Electrical Engineering, University of Linkping, Linkping, Sweden. She has authored and co-authored over 10 peer reviewed international publications. She has been involved in different research projects WASP, CCL and EU-FET WYSIWYD running research in action recognition and autonomous systems. Her professional interest topics are artificial intelligence, machine learning and computational cognitive science. Her current research interests include graph convolutional neurals networks, biologically inspired cognitive architectures and deep reinforcement learning.

Shreyas Kowshik is an Integrated Masters (B.Sc. & M.Sc.) student in Mathematics and Computer Science at The Indian Institute of Technology Kharagpur, India, from 2017. He has successfully completed a research internship with Computer Vision Laboratory (CVL), Department of Electrical Engineering, University of Linkoping through Spring 2020. He has been affiliated in a research collaboration with StarAI Laboratory, University of California Los Angeles through Autumn 2020. His research interests include machine learning and statistics with applications in various domains. Specifically, he is quite interested in applications in robotics.

Oliver Stromann received his B.Eng. In Electrical Engineering with a minor in Computer Engineering in 2015 from the University of Applied Sciences Emden/Leer during his jointpractical training program at Volkswagen AG. In 2018 he received his M.Sc. from KTH Royal Institute of Technology in Transport and Geoinformation Technology. Currently he is an industrial Ph.D. student in Computer Vision at Linkping University and the Autonomous Transport Solution Pre-Development and Research division at Scania CV AB. His current research interests are machine learning on graph-structured data as well as autonomous and intelligent transport systems.

Michael Felsberg is a full professor in Computer Vision at the Department of Electrical Engineering, Linkping University, Sweden. He holds a Ph.D. degree from Kiel University, Germany (2002) and a docent degree from Linkping University (2005). He received two paper awards (2000, 2004) and the Olympus award (2005) from the DAGM, best paper awards from SSBA (2007, 2010), Mobile Vision CVPR workshop (2014), ICPR (2016), and VISAPP (2021). His Google Scholar h-index is 43. M. Felsberg co-chaired among others DAGM 2011, ICPR 2014, 2016, CAIP 2017, and SCIA 2019. He has been serving as associate editor (among others JMIV and JIMAVIS) and area chair (among others ECCV2018, BMVC2019, CVPR2020). His research interests include, besides visual object tracking, video object and instance segmentation, point cloud processing, and efficient machine learning techniques.