Imperial College London

MEng Individual Project

Imperial College London

Department of Computing

Analytics for Urban Networks: Integrating Graph Neural Networks with Space Syntax

Author:
Wilson Chua Wei Cheng

Supervisor:
Dr. Naranker Dulay

Second Marker:
Dr. Yves-Alexandre de Montjoye

June 20, 2022

Abstract

Road networks play a critical role in cities by supporting the transportation of goods and people. It is important to understand road networks and their effects on society, economies, and the environment, to design and develop better future transportation systems. There is hence a need for scientific grounded methods to analyze urban space and predict how they evolve over time. One of these methods is *space syntax*: a collection of measures based on graph theory used to forecast movement patterns and link them to land use, density, and other spatial features of cities.

In the parallel field of road network representation learning, researchers aim to predict spatial patterns such as traffic by using graph neural networks. These techniques, when employed with space syntax, have the potential of offering new data insights and better performance.

As such, this paper proposes a machine learning pipeline that incorporates both space syntax measures and graph neural networks. We evaluate state-of-the-art architectures and contribute approaches for processing space syntax measures as input features. By demonstrating our pipeline on road link prediction and road classification tasks using a large road dataset of Great Britain, we show that space syntax enables our model to outperform previous feature set baselines under both transductive and inductive settings.

Acknowledgements

I would first like to give my warmest thanks to my supervisor, Dr. Naranker Dulay, for his insight and assistance from start to finish. His clear and detailed advice during our weekly meetings has guided me through twists and turns into fulfilling this project.

I am sincerely grateful to Ed Parham from Space Syntax for introducing me to his field. His expert advice and instrumental feedback has helped me to execute this project even as a newcomer. Special thanks also to Stephen Law and Eduardo Rico-Carranza for their invaluable suggestions that went a long way towards shaping my ideas.

Last, but certainly not the least, I would like to thank my friends and family who have supported me through my project, degree, and life. To my dear friends at Imperial and way back from high school: thanks for the continuous encouragement and for putting up with my unsolicited grievances. To my loving parents and sister: thanks for being an unwavering source of support and for always being there for me.

Contents

Chapter 1: Introduction

Road networks play significant roles in urban environments by enabling the transportation of goods and people. Researchers analyzing cities are interested in how the structure of road networks relates to vehicle and pedestrian movement patterns [1, 2], as well as socio-economic patterns such as social deprivation, land value, and crime [3: p. vii]. To explain and predict such spatial characteristics, urban scientists have developed *space syntax* to characterize road structures of varying scales by exploiting topological measures based on graph theory, which include integration, choice (betweenness centrality), and their angular variants. The insights gained have been used to inform decisions, particularly in urban planning and design [4].

With the emergence of machine learning, researchers are increasingly harnessing the power of data-driven pipelines to predict phenomena based on the urban configuration [5–9]. This gave rise to the field of *road network representation learning*, which aims to produce effective representations that capture intrinsic features of the road graph structure. These studies have proposed and applied various models based on **Graph Neural Networks (GNNs)**, deep neural networks designed for graph-structured data, for machine learning tasks on road networks. However, these tasks are complex: road attribute data is typically imbalanced [6], and road graphs are typically extremely heterogeneous and structurally inconsistent, making generalization difficult [5].

Space syntax measures, which incorporate spatial topology at different scales, may alleviate this issue by complementing the local graph structure information captured by GNNs. Yet, previous studies have not considered using space syntax metrics as input features. In this paper, we investigate a new road network representation that combines neighborhood road structure information derived by GNNs with larger-scale topological information from space syntax. We evaluate our approach on two kinds of machine learning tasks on roads: link prediction and classification, using a large spatial dataset consisting of over 2 million roads in Great Britain [10].

To optimize for model performance, we conducted extensive experiments to determine which layer types, data loading techniques, and aggregation functions work best in conjunction with space syntax measures as input features. We evaluate these under the *transductive* approach, which involves prediction within the same graph, and the *inductive* approach, which applies trained GNNs to produce representations of unseen graphs.

As the aim is to understand how space syntax can be applied in machine learning on road networks, we hope our contributions will be of interest to various stakeholders in urban road network analysis and their associated environments.

1. For **urban data analysts**, we contribute an optimized machine learning framework that can perform tasks such as road link prediction, clustering, or attribute

classification, while taking advantage of space syntax information.

2. For **researchers in space syntax**, we introduce novel techniques for applying space syntax with GNNs and evaluate their efficacy.

3. For **transport network planners** implementing road network modifications, our link prediction pipeline can extrapolate existing road structures or transfer them from another locality.

4. For **curators of spatial datasets**, our classification pipeline can assist in automatically assigning typical road attributes in areas lacking official or open data.

1.1 Objectives & Contributions

In summary, this project aims to explore the potential synergy between GNNs and space syntax, with the objectives highlighted below. In achieving these goals, we make several contributions to both fields of space syntax and road representation learning.

1. Determine the potential for space syntax features to enhance performance in road representation learning tasks.

We evaluate performance on two kinds of graph learning tasks: *road link prediction* (Chapter 4) and *road classification* (Chapter 5). Our results show that **space syntax features improve upon baseline coordinate feature sets**, by 39% and 26% in two different road classification tasks. For link prediction, we show that using **quantile-normalized** coordinate features achieves 17% higher average precision over similarly normalized space syntax features.

We also introduce different training regimes for the link prediction task and evaluate them under *transductive* and *inductive* settings. We propose that the achieved **average precision** on the link prediction task can be used to measure spatial homogeneity in the transductive setting, and network similarity in the inductive setting. We then analyze its relationships with socio-economic indicators for each city.

2. Devise optimal GNN-based architectures using space syntax measures as input.

We develop a **graph autoencoder** model for undirected link prediction on road networks and evaluate various state-of-the-art GNN layer types for the encoder. Our results show that attention-based mechanisms, such as the recently proposed GAIN operator [6], as well as symmetric DistMult decoding, yield better results when paired with space syntax for road link prediction.

For road classification, we show that the **GraphSAGE layer type with maximum aggregation** leads to the best performance in two different tasks: *road-type classification* and *accident count classification*. Our results also reveal that MLPs remain competitive with GNNs when space syntax measures are used as features. Lastly, we integrate a framework for **ordinal classification** with GNNs, and demonstrate its efficacy.

3. Implement a pipeline that can process space syntax measures from large road datasets, and evaluate the impact of pipeline parameters on model performance.

We implement a data processing pipeline that can process a dataset of over 40 million road geometries into mini-batches of subgraphs through state-of-the-art graph sampling modules: **Cluster-GCN** [11] and **GraphSAGE Neighbor Sampling** [12]. We evaluate aggregation functions for edge feature pooling and for simplifying road geometries, and show empirically that the minimum leads to better performance in the former case.

To support this pipeline, we developed `SSx-GNN`, a Python library containing utilities for processing spatial data into input features for graph learning, training GNNs, and visualizing predictions. In this library, we utilized PyTorch Geometric [13], a framework for graph representation learning. PyTorch Geometric is equipped with many state-of-the-art graph learning algorithms and is able to leverage dedicated CUDA kernels on GPUs for high performance. We provide details of `SSx-GNN` in Appendix C.

1.2 Ethical Considerations

The data used for this research include open data, for the most part only contains public information and poses no risk of identifying individuals. Several statistical metrics used are based on individual data, but have been aggregated and include no personal information. For example, census data stems from an individual's social status, but because statistics are published on the level of cities, they cannot be used for the re-identification of individuals. Produced model statistics are also not persisted in any storage medium and can only be retrieved via recomputation.

Beyond data, it is possible to use the tools we propose to identify patterns in real cities around the world. There is an unlikely possibility of bad actors making use of such tools to identify the crucial urban infrastructure that would severely cripple the city if compromised. The tools can also be used to identify community segregation in the city. This poses some political implications, such as justifying future electoral boundaries based on representation between segregated areas.

Planners should also be careful about making actual decisions that affect real people using insights gained from these black-box tools, because they produce only estimations of true data. Also, while we perform many correlation analyses in this paper, it is important to remember that these do not imply causation and cannot be directly used to justify decisions without context. The intention is that, instead, this study would contribute to urban planning to improve cities by making positive use of our methods to obtain data-driven insights.

Multiple open datasets are used in this project, all of which contain real data about the UK and its road networks. By making use of data from outside sources, we assume that the agencies that maintain these datasets have done their due diligence in assessing potential privacy risks. We list them and their potential ethical considerations in Table A.2 of Appendix A.

Chapter 2: Preliminaries

In this chapter, we introduce the space syntax measures used in this project, explain their mathematical derivation, and provide intuitions for how they are used (Section 2.1). Next, we explain deep neural networks and other concepts that form the basis for the models used in this project (Section 2.2). Finally, Section 2.3 covers graphs and the basics of graph neural networks used for machine learning on graph-structured data. Concepts specific to certain tasks are covered in their respective chapters.

2.1 Space Syntax

Space syntax comprises several techniques and theories that are used to study spatial configurations [3: p. vii]. They are built on the notion that space can be split into components and represented by graphs, which are then analyzed for connectivity and relationships. Space syntax measures, detailed in Section 2.1.2, have been found to correlate closely with human spatial behavior and traffic flows in urban networks [14, 15].

Since its inception, advancements in computing have expanded the capabilities of space syntax techniques, enabling it to be applied in larger contexts of entire cities and metropolitan regions. They have been used to develop new theories of how cities are built upon economic, social, and cognitive factors, and consequently how urban space is generative of such factors. Many cities around the world have been analyzed with space syntax, contributing to data stores used by researchers and planners alike [16].

2.1.1 Representations of Space

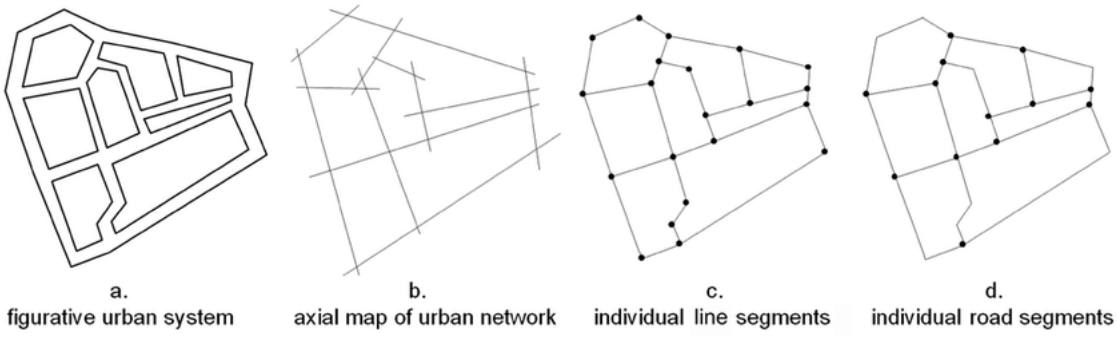


Figure 2.1: Axial Map, Line Segment, and Road Segment visualizations [17: p. 20]

> In space syntax research the term "street" tends to be used more than "road" unlike in road network representation learning. This paper uses the term "road" for consistency, but considers them generally synonymous.

The relationship between movement and road structure is well established, with pioneering studies relying on *axial maps* manually drawn by researchers [15, 18]. These maps depict axial lines indicating the *sightline* along possible movement edges, which have a large impact on human navigation [18]. However, in more recent studies [17, 19], newer measures have been developed on Geographic Information Science (GIS)-based representations of road networks, which use geo-referenced road segments as the primary unit of analysis. Metrics derived from these representations have been shown to differ only slightly from that of the axial line representation [14], and have become the standard for urban network analysis [20].

Definitions

Borrowing the definitions from Peponis et al. [19], we use the term "line segment" to refer to the actual straight lines stored by spatial databases, and the term "road segment" to refer to one or more line segments between two road nodes. The term "road node" is used to refer to an intersection indicating a choice of paths, or to an endpoint indicating a dead-end. The term "line node" then refers to the common point between two consecutive line segments. Figure 2.1 clarifies on the differences between axial maps, line segments, and road segments defined in this way.

Segment Maps and Distances

Segment maps were developed in response to the criticism [21] that previous space syntax measures do not account for metric distance. To create a segment map, the axial map is divided into road segments at each intersection. Segment maps are used to derive three notions of distance used in space syntax, each inducing different definitions of the "shortest distance" [16].

1. Topological distance: The lowest number of directional changes from each segment to all others.
2. Geometric distance: The lowest number of angular deviations from each segment to all others.
3. Metric distance: The shortest distance from a segment to all others.

In particular, *angular measures* derived from geometric distance are predominantly used in space syntax analyses [3: p. 83], and are detailed in Section 2.1.2.

Deriving graph representations of road networks

There are two ways to derive graphs (further defined in Section 2.3.1) from segment maps, as shown in Figure 2.2. The **primal** representation has intersections and dead-ends as

nodes, with the roads connecting nodes as edges. The **dual** representation takes the road segments to be the nodes and the intersections as edges. The dual, also known as a line graph, is the typical view taken by space syntax, since treating roads as nodes enables the use of techniques derived from graph theory to measure accessibility [18].

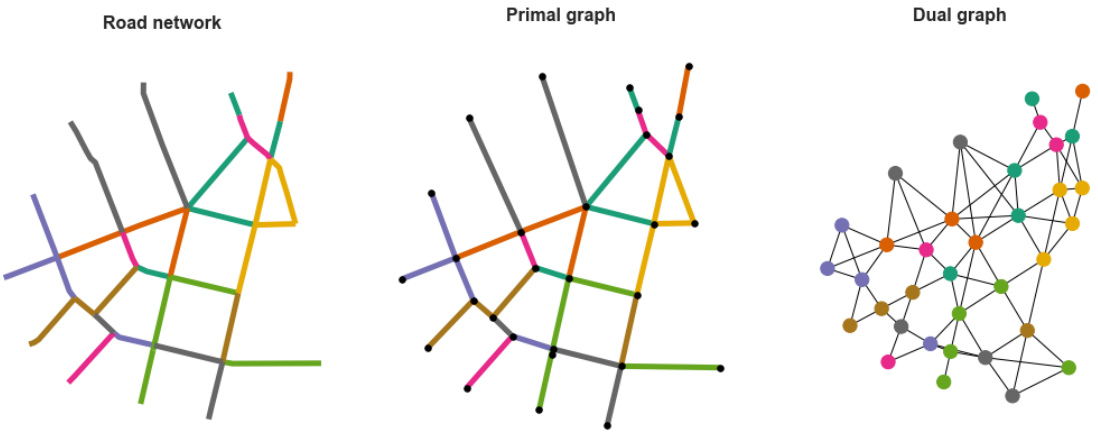


Figure 2.2: The primal and dual graph representations of a road network. (Line data from OpenStreetMap [22])

Automatic Generation of Representations

An algorithm to go from convex spatial representations such as (a) in Figure 2.1 to the graph representations depicted above has been achieved and incorporated into tools such as Depthmap [23]. These tools also compute the space syntax measures detailed later, and are thus integral to the space syntax analysis pipeline. In addition, most street networks in the world have been geo-referenced in open datasets such as OpenStreetMap [22]. Interfaces that seamlessly extract the networks as graphs exist and have been incorporated into open-source analysis tools such as `OSMnx` [24].

Nevertheless, an automatic and accurate algorithm to generate the above representations from raster images of urban plans or streets remains an open problem. Some studies have proposed using computer vision techniques [25], but a full pipeline has yet to be achieved.

2.1.2 Space Syntax Measures

From the above graph representations, many measures that involve concepts from graph theory have been developed and are used to quantify properties of road networks. This section covers the measures included in the **Space Syntax (SSx) OpenMapping** dataset used extensively in this project. Most of these measures are calculated from the perspective of each road as its corresponding node in a dual graph. Measures that are properties of the entire network exist, but are not covered in this paper.

Connectivity

The 1-step connectivity is the number of direct connections a road has with other roads, equivalent to the node degree in graph theory [26]. Connectivity can be measured over several steps, where a step (or "hop") is a change in direction going from one road or axial line to an adjacent one. The one-step or two-step analysis represents the degree of connectivity of roads to other roads in the neighboring vicinity. Increasing the number of steps measures the overall *catchment* of a road in increasingly larger contexts of the overall network. The N-step analysis takes the entire network into account, but involves the greatest computational cost. This measure can be useful for transport planners who wish to measure how well a public transport stop covers the city, or to quantify the ease of reaching amenities such as shopping streets from any location in a city [3: p. 44].

Mean Depth

The depth of a road node i refers to the shortest distance d_{ci} from a chosen root node of the graph c to i, where distance can take any one of the three definitions described in Section 2.1.1. The mean depth of the root node c is hence the mean shortest distance from c to all other nodes as follows, where k is the number of nodes in the graph [20].

$$\text{Mean Depth}_c = \frac{\sum_{i=1}^{k} d_{ci}}{k-1} \tag{2.1}$$

Integration (Closeness Centrality)

Integration, also known as the closeness centrality in graph theory, is a measure of a road's accessibility to all other roads [18]. It is computed by normalizing the reciprocal of the mean depth. As such, the fewer the steps required from a road to reach all other roads in the system, the lower its mean depth and the higher its integration value.

In contrast, roads that require many steps have low integration, and are considered "spatially segregated". *Global* integration, which considers all nodes in the network, is computed from the mean depth by including multiple normalizing constants to account for the effect of graph size on the output magnitude. This allows comparison across spatial systems of differing scale [3: p. 49]. As a measure of centrality, integration is commonly applied to analyze movement patterns in urban environments. Planners have applied integration to test different options of implementing transit systems [20].

Choice (Betweenness Centrality)

The choice at a segment is the measure of how often that segment lies on the shortest paths in the network [15]. It requires generating the geodesics (the shortest paths) between all segments in the network that go through that segment, a computationally expensive step for large networks. The higher the choice of a road segment, the more likely it is to be chosen to be a part of journeys between every possible origin and destination [27]. Choice is also known as the betweenness centrality in graph theory [28].

Both integration and choice were noted by Rashid [26] to be the two most useful space syntax measures in traffic flow studies. They differ in that integration indicates the *to-movement* potentials, or the ease of getting to a destination, while choice indicates the *through-movement* potentials, or the likelihood of being along journey routes [29].

Urban researchers often visualize these two metrics on maps with line color gradients like the ones in Figure 2.3. On such maps, choice tends to highlight the most important routes in the network, while integration highlights the neighborhood or city centers, depending on the scale of analysis [3: p. 77].

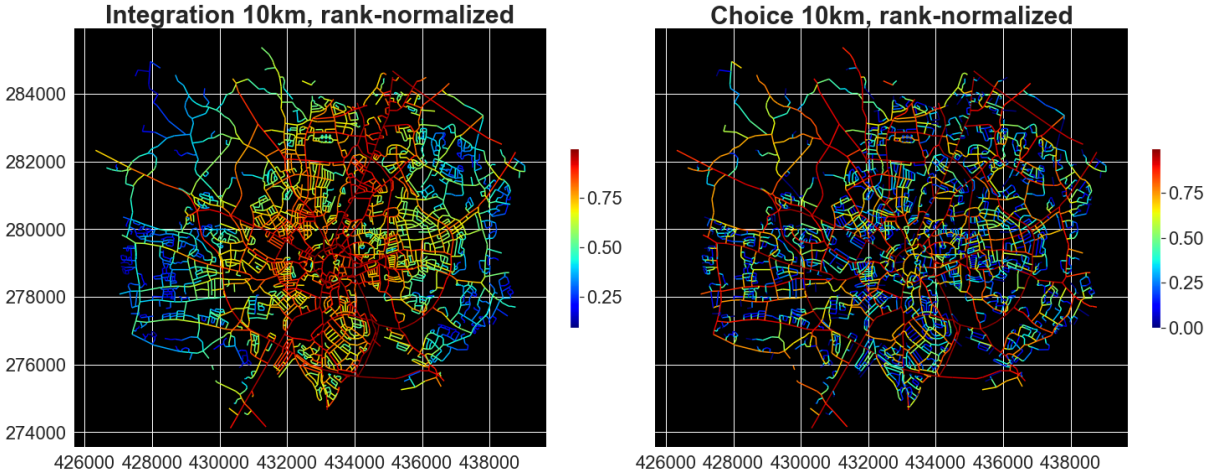


Figure 2.3: Rank-normalized angular integration and choice on the same road network in Coventry, England. (Values taken from the SSx OpenMapping dataset)

Angular Integration and Choice

The motivation for angular measures can be attributed to Dalton [30], who empirically observed that people often pick the simplest route with the least angle deviations in the path. Accordingly, both integration and choice have been found to be more predictive of actual human movement when computed using the geometric distance [15].

Angular integration is the reciprocal of the angular mean depth [14], which is the average shortest angular distance from the segment to all others. Angular integration at a segment r in a road network with total nodes k is hence computed as follows, where $d_\theta(r, i)$ refers to the shortest angular distance between two segments r and i.

$$\text{Angular Integration}_r = \frac{k}{\sum_{i=1}^{k} d_\theta(r, i)} \tag{2.2}$$

Angular choice requiring computing all shortest paths, similar to axial choice above, but using geometric distance [14]. The number of paths going through road r is then averaged over the total number of possible paths as follows.

$$\text{Angular Choice}_r = \frac{2\sum_{i=1}^{k}\sum_{j=1}^{k}\sigma(i, r, j)}{(k-1)(k-2)}, i \neq r \neq j \tag{2.3}$$

where $\sigma(i, r, j) = 1$ if r lies on the shortest path from i to j, and 0 otherwise.

Metric distance can be incorporated into these measures by considering only road segments within a certain radius [29]. This allows a finer *local* scale analysis, such as movement within a shopping district, by deliberately excluding the context of the entire road network. Several specific ranges of radii have been identified for regional (8,200 to 30,000 meters), city (1,200 to 8,200 meters), and neighborhood (400 to 1,200 meters) scale analysis [31].

Normalized Angular Measures in SSx OpenMapping

The Space Syntax OpenMapping (SSx OpenMapping) dataset comprises 40 million geometries representing the simplified Great Britain road network, taken from the OS Meridian 2, a dataset from the Ordnance Survey. It contains precomputed angular choice and angular integration at three selected scales: 2 km, 10 km and 100 km, for each road segment in the dataset. Also included are the road segment lengths, and the node count in the area for each scale, which gives a measure of road density. For a full list of the attributes that we use, refer to Appendix A.

In addition, SSx OpenMapping includes two types of measure normalization, denoted by the attribute name suffixes `log` and `rank`. These normalization techniques allow for smaller variations between values to be more easily seen in data analysis, with `rank` allowing for comparison across scales without being skewed by the different node counts. The effect of normalization can be seen in the distribution graphs in Figure A.4 of Appendix A.

Normalized angular measures are known to have higher correlation and predictive capability with human spatial behaviors such as pedestrian movement [1] and traffic [2]. Therefore, using such measures as input into machine learning pipelines on road graphs may enhance their predictive capability over the raw features.

2.1.3 Limitations of Space Syntax

Classic space syntax has been criticized for being sensitive to the defined network boundaries, potentially causing errors depending on the truncated elements. The usual solution for this is to establish a cut-off depth for analysis based on the maximum depth [26].

While the measures have been shown to positively correlate with human spatial behavior, using them to suggest improvements for the urban transport network still requires a domain expert with contextual knowledge. There are therefore some grounds for a system that incorporates space syntax measures with other domain-specific information, to assist urban planners in making such decisions.

Computational Complexity

The choice algorithm implemented in Depthmap involves traversing the graph between sampled node pairs and accumulating angular deviations to compute the geometric geodesics [29]. This traversal makes computing choice prohibitively expensive for large radii in regional or nationwide analyses.

Varoudis et al. [32] proposed an approximate algorithm that weights edges in the dual graph (representing intersections) by angular change and then computes the standard

weighted graph betweenness centrality. Even though this speeds up the computation time considerably over previous methods, it still involves computing the shortest paths between all pairs of nodes, a $O(n^3)$ problem for n total nodes. Taking several hours or even days to compute choice for road networks on large scales remains a significant detriment for planners wishing to quickly iterate through multiple planned networks.

2.2 Deep Neural Networks & Related Concepts

Due to the increasing availability of large datasets and the growth of computing power, deep neural networks have become some of the most successful machine learning techniques in recent years [33]. They provide state-of-the-art solutions to many problems, such as convolutional neural networks (CNN) in image recognition, recurrent neural networks (RNN) and Long Short Term Memory (LSTM) in natural language processing. Their major advantage over traditional linear models lies in their capacity to scale up to and infer patterns in big datasets. Big *spatial* datasets in particular have become common and are being used for street network analyses across large regions [34].

2.2.1 Multi-Layer Perceptron

The simplest case of a neural network is the single neuron. A neuron is a computational unit that takes one or several inputs, applies a linear transformation followed by a nonlinear activation, and outputs the result. Commonly used nonlinearities include the logistic sigmoid function σ and the piecewise linear function ReLU.

A neural network can be built from multiple neurons by connecting the output of a neuron to the input of another neuron. These neurons can then be stacked into a layer to enable multivariate input. Deep neural networks hence consist of multiple layers of neurons that take an input vector, propagate them through the layers, and then output a new vector, as shown in Figure 2.4.

A multi-layer perceptron (MLP) is a fully connected neural network. In theory, an MLP can be used to approximate any function from the input space into the output space. One limitation, however, is that they scale poorly to input data with many dimensions due to the number of parameters they must store. The amount of training data required also scales exponentially with the number of dimensions, a problem typically referred to as the "curse of dimensionality" [35].

2.2.2 Autoencoders

Autoencoders are another class of neural networks used to learn embeddings of unlabeled data in unsupervised learning [36], with its typical structure being illustrated in Figure 2.5. They consist of two parts: an encoder and a decoder. The encoder maps an input to a typically lower-dimensional latent vector, while the decoder then maps the latent vector back into the input space. The reconstruction loss, which measures the error between the reconstructed output and the original input, is used for end-to-end training of both the encoder and the decoder.

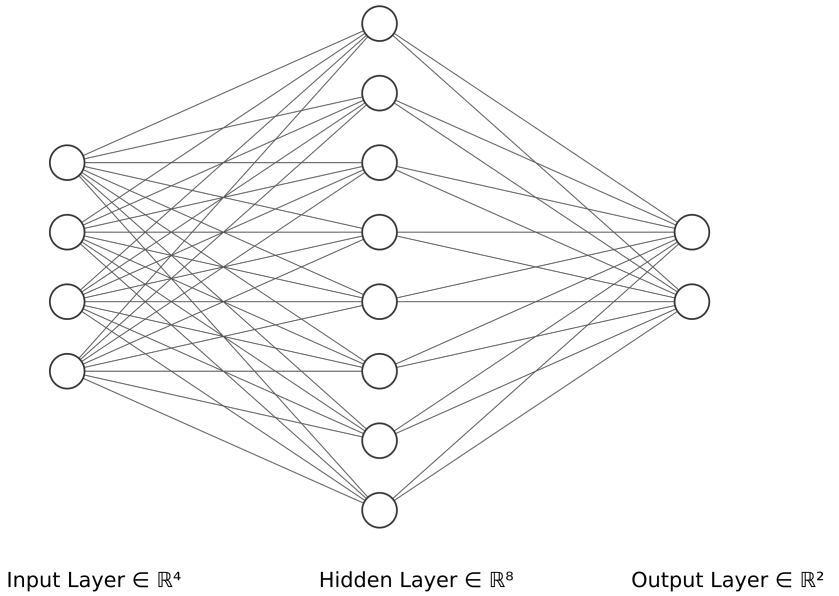


Figure 2.4: Structure of a 2-layer MLP, which will be used as a baseline for several tasks in this project

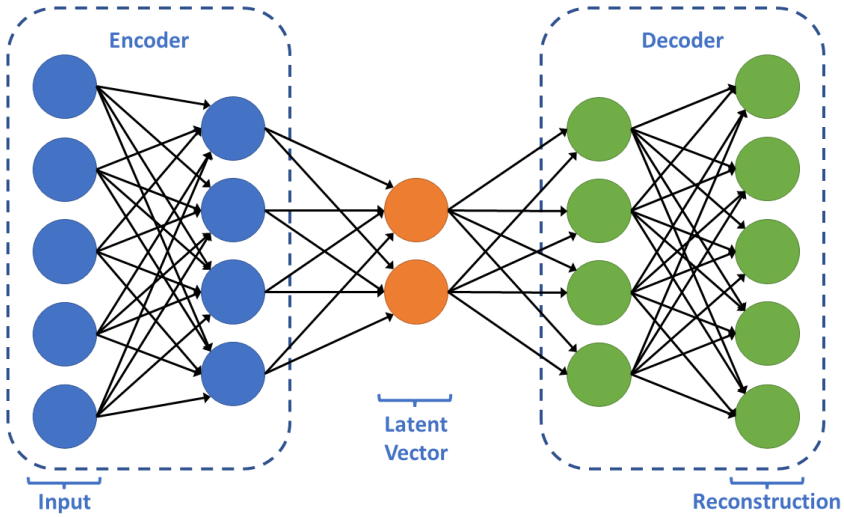


Figure 2.5: An overview of a typical autoencoder architecture.

The **variational autoencoder** (VAE) repurposes the autoencoder architecture to solve problems in approximate inference [37]. Practically, two vectors representing a mean and a standard deviation are generated at the output of the encoder. These vectors parameterize a Gaussian distribution, from which samples are drawn for input into the decoder. This opens the door for generative tasks, with VAE-based architectures being competitive with the state of the art on generated image fidelity [38].

2.2.3 Attention

Attention mechanisms, initially developed for encoder-decoder models [39], are used in state-of-the-art solutions to sequence-based tasks such as machine translation [40]. Similar to the psychological process of selectively concentrating on certain things while ignoring

others, attention in machine learning involves weighting features in different positions by similarity or relevance. This enables variable sized inputs by focusing on the most relevant parts to make decisions [41].

Attention weights are typically computed using the softmax function, which attributes a weight between 0 and 1 to each position such that the higher the weight, the greater the deemed relevance of that position. Generally, attention can be applied to two different sequences h and h'. Given a measure of similarity s (such as the dot product), the weights between positions h_u and h'_v are computed as follows:

$$\alpha_{uv} = \frac{\exp s(h_u, h'_v)}{\sum_k \exp s(h_u, h'_v)}, \tag{2.4}$$

Self-attention is a special case of attention where $h = h'$, which can be used to compute a new representation of that sequence. [40].

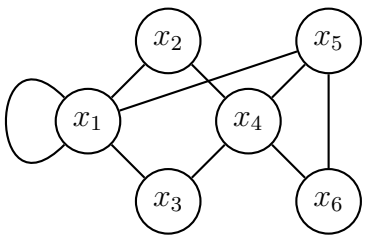
2.3 Machine Learning on Graphs

2.3.1 Graphs

Graphs are collections of links (edges) between entities (nodes). Specialized links can be represented by adding directionality or weights to edges. Graphs may have multiple edge types (heterogeneous graphs), but we only focus on the *homogeneous* case in this project, where every edge is the same. Figure 2.6 illustrates how edges can be represented using an *adjacency matrix* that indicates which pairs of nodes are linked.

Road networks, the focus of this project, can be represented by the primal and dual graphs described in Section 2.1.1. For simplicity, we assume the primal graph to be *planar*, i.e. no edges cross each other. Even though such abstractions are common in urban network analysis, they can be misrepresentative of the real world, as actual road networks are three-dimensional. [42].

Graph structures are used to represent constructs across many domains, even in indirect contexts such as texts or images [43]. This motivated the development of machine learning architectures that incorporate and exploit the graph properties of such structures.



	x_1	x_2	x_3	x_4	x_5	x_6
x_1	1	1	1	0	1	0
x_2	1	0	0	1	0	0
x_3	1	0	0	1	0	0
x_4	0	1	1	0	1	1
x_5	1	0	0	1	0	1
x_6	0	0	0	1	1	0

Figure 2.6: An undirected graph and its symmetric adjacency matrix.

2.3.2 Graph Neural Networks

Graph neural networks (GNNs) are neural models that capture the dependence of graphs via message passing between nodes. The problems GNNs can solve can be divided into the following categories [44], with the ones we tackle in this project in bold:

- Node-level tasks that include **node classification** and regression, which aim to predict a class or a value respectively for each node. Node clustering is another task that aims to partition the nodes into similar groups.
- Edge-level tasks that include edge classification and **link prediction**, which respectively use the model to classify edges or to predict the existence of an edge between two nodes.
- Graph-level tasks that include classification, regression, and matching, which require the model to understand the full graph structure.

The simplest GNN applies a separate MLP to each component of the graph. The MLP is applied to each input node feature vector, and outputs a learned vector (*node embedding*). If required, the same can be done for edges, obtaining edge embeddings, and for the global context, generating an embedding for the entire graph. This process is encapsulated in a single GNN layer [43].

Typically, GNNs combine the node feature vector with that of its neighbors before generating an embedding. There are many ways to combine these vectors (also described as *propagation* or *aggregation*), and they can be broadly categorized into convolutional, attentional, and message-passing approaches [45]. Convolutional approaches aggregate the features of neighbors with fixed weights. This includes the popular Graph Convolutional Network (GCN) [46] which is used for its simplicity and scalability.

Attentional approaches aggregate the features of neighbors with learnable weights, via self-attention as described in Section 2.2.3. Message-passing is the most general approach and involves computing and sending arbitrary vectors ("messages") across edges. This approach scales poorly to large graphs due to the cost of computing messages. In such cases, a sampling layer can be incorporated to reduce the input dimensions during propagation.

To move from node to edge representations, or vice-versa, a *pooling layer* can be used [44]. This involves a permutation-invariant function, such as the average or the sum, which produces the same output no matter the order of its inputs. There is no function that is optimal for every task, though the sum is often used [43].

While there has not been a study done using GNNs and space syntax measures, they share a common purpose in detecting and inferring patterns from graph-structured data. This might suggest that they are redundant as input features into GNNs. Still, measures like angular choice and integration (Section 2.1.2) capture geometric information that is orthogonal to the graph structure. As such, GNNs may be able to interpret more aspects of a road network by using space syntax measures as node or edge-level features.

Over-smoothing

Despite their success, GNNs commonly face the issue of feature over-smoothing [44]. Over-smoothing arises when node representations of different classes become less distinct while

stacking multiple GNN layers, to the point of negatively impacting model performance. This occurs due to the nature of graph convolution, which causes adjacent nodes to have similar output representations through an operation known as Laplacian smoothing [47].

As a result, GNN models in the literature typically stick to two or three layers, as increasing the number of layers past that usually degrades performance [46, 47]. However, as the number of layers indicates the number of hops over which feature propagation takes place, this restricts structural information captured by GNNs to the local neighborhood. This contrasts with space syntax measures that can capture information at a much bigger scale when measured at large radii.

2.4 Summary

In this chapter, we covered the major techniques used in this project: space syntax and GNNs. The space syntax measures of integration and choice are highly predictive of movement, and form the majority of features in the SSx OpenMapping dataset. We then covered several high-level concepts in deep neural networks such as MLPs, autoencoders and attention, which are used in our proposed architectures as well as the baseline comparisons for evaluation. We also defined and described graphs, including the primal and dual graph representations of road networks. Lastly, we explained the feature propagation mechanism behind GNNs, which are machine learning architectures designed for graph-structured data.

Chapter 3: Related Work

This chapter surveys and reviews the state of the art in techniques that aim to leverage the integration of GNNs with space syntax to analyze urban road networks. We first address previous works in road network representation learning (Section 3.1), which proposed methods and models that form the basis of our work in the following chapters. We then address works that incorporated machine learning in space syntax (Section 3.2), and explain how our methods will differ from theirs. Finally, we summarize and go over the open issues in this interdisciplinary area (Section 3.3).

3.1 Modeling Road Networks

Various state-of-the-art GNN architectures have been developed and applied in the following recent works to model road networks.

3.1.1 Road Link Prediction

Xue et al. [8] applied link prediction using Relational GCNs [48] to study urban growth patterns in 30 cities worldwide. They removed several roads from existing road network graphs and then trained the model to predict the missing roads. The authors then use the F1 score achieved by their model as a metric of spatial homogeneity. They inductively applied their model to unseen road networks of different cities, and used the F1 score to quantify the similarity between the two networks. The authors argue that such analyses can be used to understand and apply urban planning insights between cities.

As our project focuses on homogeneous graphs, we explore approaches other than the Relational GCN as it is designed for heterogeneous graphs. Furthermore, while they used a grid-based approach to divide road networks, we use officially-defined labels to derive subgraphs of different cities. We describe, evaluate and compare our approach in Chapter 4.

Wang et al. [49] proposed a novel pipeline for link prediction, including partitioning subgraphs and using heuristics like the Katz index to compute the similarity score between nodes. However, they performed link prediction on the dual graph, which represents predicting intersections between roads instead of the roads themselves. For our link prediction task in Chapter 4, we instead adopt Xue et al.'s approach [8] of using the primal graph.

3.1.2 Road Attribute Prediction

Jepsen et al. [5] used GCNs for two road prediction tasks: driving speed prediction, and speed limit classification. They noted that standard GCNs do not support edge attributes and between-edge attributes (such as the angle between road segments). In contrast to graphs typically modelled by GCNs, road network graphs exhibit *volatile homophily*, i.e. abrupt differences in features between adjacent roads, such as a sharp drop in the speed limit at motorway exits [5]. They proposed a modification of GCNs that attempts to resolve these issues: the Relational Fusion Network (RFN), which incorporates both primal and dual graphs with parallel GNNs, and uses attentional GNN layers to selectively exclude noise from neighboring nodes.

Although the RFN outperformed other approaches, it uses many attribute types. When limited to a node-only or edge-only feature set, a model with lower complexity can be deployed. In our project, space syntax constitutes only edge features, and can be simply represented by node features in the dual graph. Node features of the primal graph can also be obtained through edge-to-node feature aggregation, which is proposed and evaluated in Chapter 4.

Gharaee et al. [6] employed various GNN architectures for the task of inferring missing data in open road datasets via unsupervised classification of road labels. They proposed an attention-based aggregation, GAIN, and a method of sampling adjacent and distant node using random walks of different lengths. They evaluated their approach under the transductive (same city) and inductive (unseen city) settings with road networks from 17 different Swedish cities, and showed that the GAIN outperformed competing models like the RFN. We adopt the GAIN for our model for link prediction in Chapter 4, and also for road classification tasks in Chapter 5.

Zheng et al. [7] applied the GCN in the field of cartography to the problem of road selection for map display, which involves choosing the appropriate roads for a given map resolution while optimizing for an uncluttered visual impression. The architecture that produced the best results was the Graph Attentional Network (GAT) from JK-Nets [50], which flexibly changes the size of the node neighborhood used for aggregation. Out of these studies, they present a unique problem involving the prediction of subjective human (expert cartographer) inputs rather than data-defined labels. However, their experiments only involved relatively small-scale networks of around 5000 roads, whereas we examine much larger networks in this project.

The above works suggest that there is no optimal GNN for to prediction tasks involving road networks. Making accurate high granularity (node and edge) predictions on unseen road networks remains a significant challenge, since the urban configuration can change greatly between networks. Nevertheless, there is an opportunity for space syntax to overcome these issues with computed measures describing centrality and movement potential. In this project, we formulate and evaluate a GNN and space syntax approach in two road attribute classification tasks in Chapter 5: road-type classification and accident count classification.

3.1.3 Baseline Features for Road Learning

Below are some feature sets used by the papers above, and other studies in graph learning, as input node features to GNNs. We adopt these as baselines for evaluating space syntax

features in the following chapters.

- **Coordinate Features**: The coordinates of road nodes can be directly used as features of the primal graph, as was done in Xue et al. for link prediction [8]. For road classification on the dual graph, Gharee et al. proposed using a set of normalized interpolated points between both ends of the road segment, the segment length, and the absolute coordinate of the midpoint [6].
- **Data-derived labels**: Road geometries from open datasets often come with labeled data, such as the road type (highway, residential, etc.), speed limit and number of lanes. As these are categorical data, they are usually one-hot encoded before being used as features [5, 6]. However, such attributes are typically incomplete in open datasets, especially for rural areas [51]. We therefore propose using space syntax measures as a viable alternative to such features when they are not available.
- **Local Degree Profile**: The Local Degree Profile (LDP) was proposed by Cai et al. [52] for featureless graph classification, and involves appending the degree of each node and its neighbors to the node feature vector. As mentioned in Section 2.1.2, the degree is known in space syntax as the single step connectivity; however in contrast to the other space syntax measures, it only considers the 2-hop neighborhood (topological distance) compared to all roads within a fixed radius (metric distance). As such, we use the LDP as a baseline feature set to contrast with space syntax measures.

3.1.4 Predicting Betweenness Centrality

Several studies have developed and applied GNNs to predict the standard betweenness centrality based on the shortest paths in a weighted graph. Fan et al. [53] applied a convolutional GNN to predict the top-N nodes of a graph with the highest centrality. Maurya et al. [54] proposed another convolutional architecture that constrains GNN aggregation along edges which lie on the shortest paths. Both studies were able to give accurate predictions that were much faster than previous exact algorithms, and were also able to inductively predict the centrality rankings in unseen graphs.

Both of these studies proposed the alternative problem of predicting relative ranks of nodes instead of the actual betweenness values, recasting the problem as classification instead of regression. The exact values for choice are also in practice not as critical as their relative rankings. This motivation is similar to that of sampling algorithms approximating choice in modern space syntax tools such as Depthmap [23].

Predicting Choice

Despite its usefulness, computing space syntax's angular choice incurs a high computational cost, as described in Section 2.1.3. For example, it took 15 days to compute all choice values for a road network graph containing 1.5 million nodes [32]. Even though a decade has passed since that study, algorithmic efficiency has not improved much, especially for choice measured at large radii. Hence, it is worth investigating whether a reasonable proxy for choice could be predicted using graph neural networks, and if it can accurately generalize to unseen road networks.

We initially considered developing models for this problem, and tested models including the GNNBet described in Fan et al. [53]. This proved to be an intractable task, making it unlikely that it would take any faster than existing approximate methods. Due to the narrow local neighborhood perspective of GNNs, it is difficult for information about one side of the graph to propagate to the other side. Hence, GNNs are to some extent incompatible with predicting large-scale attributes such as choice, especially when vital routes, such as motorways and major roads, may be located far from the local roads that they serve. Many layers would be required to propagate information at that scale, which may induce over-smoothing (Section 2.3.2).

3.2 Space Syntax & Machine Learning

There has been limited previous work in the interdisciplinary area of space syntax and machine learning. The unavailability of large labeled spatial datasets incorporating space syntax-derived metrics is a reason for this shortcoming [25]. Many of the previous studies in this area have tended to view the two approaches comparatively rather than integrating them, such as by inputting measures into a model. Moreover, GNNs have not been considered for predictive tasks in space syntax. As such, this project investigates whether employing graph-based architectures improves results over linear models.

3.2.1 Pedestrian Route Simulation

Wang et al. [55] compared integration and other space syntax measures with simulated pedestrian movement from a trained Generative Adversarial Network (GAN). The study's objective was to identify pedestrian hot spots and simulate movement tracks within a fixed commercial area, using empirically collected images of pedestrian route trajectories. They interpreted the most common trajectories under the context of a global integration analysis.

As they focused on a relatively small commercial district, their proposed methods may not scale well to a larger network, mainly as increasing the image resolution would lead to unfeasible computational cost. The study also only applied space syntax measures to analyze their simulated trajectories. Using the measures in conjunction with the movement data to predict pedestrian hot spots might be a better approach. We use this "space syntax approach" to investigate a similar problem of predicting traffic accident counts (Section 5.2 of Chapter 5).

3.2.2 Unsupervised Classification

Chang et al. [56] used a K-means clustering approach to classify points of interest (POIs) data in Zurich, in order to determine features of *urban identity* in public spaces. The POIs were combined with weighted choice and integration statistics to measure the quality of the surrounding space. They proposed using classification as a method for planners to understand the relationship between urban identity and spatial layout.

As their POI clustering approach was based directly on the area's features, adjacent areas were not considered. A GNN-based approach would on the other hand produce a

neighborhood-aware classifier that incorporates features of the surrounding environment through aggregation. While this project does not tackle this problem of classifying urban identity, the GNN models we present for classification could be applied similarly to such tasks.

Varoudis et al. [57] trained a *convolutional VAE* using images of randomly generated subgraphs of the London road network. In doing so, they derived clusters as well as a generative model for urban networks. The clusters identified morphologically similar road networks, such as "perpendicular" and "organic tree" structures. Since their model outputs probability distributions of subgraph structures, it could also be used to generative new networks via random sampling.

However, they did not utilize space syntax measures, as the study was primarily investigating how a neural network could model intrinsic patterns of road networks. The study did not go into details of how they determined their architecture, nor was the architecture evaluated. It is possible that a graph-based VAE, such as the one that we discuss in the following chapter, could serve as an alternative for generating spatial configurations.

3.3 Summary, Insights & Open Issues

Space syntax measures have been shown to be predictive of human movement patterns. Yet, predicting movement and traffic, among other spatial features, can also be done with deep neural networks. Previous approaches, which include convolutional VAEs, GANs and GNNs, have used image and graph representations of road networks for prediction but did not consider incorporating space syntax. This motivates our investigation into their potential as input features for neural network-based approaches.

In road network representation learning, researchers have used GNNs to predict various urban road network features based on road data and graph structure. Since most space syntax measures are based in graph theory, the information they represent may overlap with the spatial characteristics GNNs model, making them potentially irrelevant as input. However, as angular measures are derived from topological road properties orthogonal to the graph structure, we hypothesize that these measures may enable standard GNNs to model road networks better when used as input node or edge features.

Still, space syntax measures such as choice have high computational costs, especially at large scales. This incentivizes a neural network approximation trained using precomputed measures from large street datasets. However, through initial work, we found that GNNs are not a good fit for predicting space syntax measures. Due to their emphasis on local neighborhoods, GNNs face difficulty in consolidating the network-wide information required for such a task.

An important point to keep in mind is that deep neural networks do not necessarily produce better results than traditional classification or regression methods [33]. Therefore, they should not be applied needlessly, especially if the dataset is small, such as when analyzing local neighborhoods of less than 1000 roads. Regardless, the SSx OpenMapping dataset used in this project contains over 40 million road geometries with 17 space syntax measures each, making it a suitable candidate for use with deep models.

Chapter 4: Road Link Prediction

This chapter focuses on link prediction to evaluate the relevance of space syntax measures as useful features for training GNNs on road networks. We also investigate and demonstrate the usefulness of the link prediction task for quantifying spatial homogeneity, comparing network similarity, and deriving spatial clustering.

Firstly, we explain potential applications for the link prediction task in Section 4.1. Next, we describe the link prediction task on road networks, the spatial dataset used, and our chosen evaluation metrics (Section 4.2). We then discuss our results in Section 4.3, which reveal that out of all the space syntax measures, integration leads to the best performance. However, we will also show that quantile-normalized coordinate features enables our model to outperform previous approaches (Section 4.3.3).

We discuss and evaluate our data processing and model parameter choices in Sections 4.4 and 4.5, where results show that our graph autoencoder model that uses the GAIN operator and symmetric DistMult outperforms other variations. We conclude with a discussion of the autoencoder's output latent representations, the limitations of this task, as well as other considered approaches (Section 4.6).

4.1 Applications

Link prediction is an established task in graph representation learning, most notably in applications such as social networks and recommender systems [44]. For road networks, predicting links has use cases in designing and planning transport infrastructure [49]. Reasonable estimates of how road networks evolve can assist transport planners in deciding which new roads to build in order to maximize traffic efficiency, connectivity, and pedestrian movement.

Link prediction can also be used *inductively* by applying a learned model trained on one city to another. For example, if one city is known to have good walkability and cycling routes, another city wishing to benefit from similar infrastructure can use link prediction to "transfer" the network structure over. Yet, there is still not much research investigating architectures for link prediction in road networks, as well as the potential enhancements space syntax can bring.

4.1.1 Spatial Homogeneity & Network Similarity

The F1 score, which can be used to evaluate link prediction performance, has been applied as a measure of spatial homogeneity in road networks [8]. Here, spatial homogeneity refers

to how topologically similar the road node connections are between separate areas in the same road network. The intuition for using the F1 score is that a road network with high homogeneity should correspond to a primal graph with edges that are more easily recoverable given the rest of the graph. High homogeneity can denote that the road structure in a city was developed in a unified manner, while a low homogeneity may indicate more organic growth.

When link prediction is applied inductively, the resultant metric gives an indicator of the similarity between different road networks [8]. This can be leveraged to assess intercity similarity based on transport infrastructure, or to evaluate the feasibility of transferring knowledge between cities. For example, one can plan new store locations in one city based on existing locations in a "similar" city [58].

In contrast to Xue et al.'s work [8], we use Average Precision (AP) instead of the F1 score, as it is a less biased measure. Our results show that the AP achieves a similar distribution to that of the F1 score when taken at optimal thresholds, and can thus function as a suitable measure for spatial homogeneity.

4.2 Link Prediction Task Formulation

We frame the link prediction task on a single road network as follows: Given the undirected primal graph $G = (V, E)$, where V (vertices) is the set of line nodes and E (edges) is the set of line segments, a set percentage (20%) of E is sampled and removed. The objective is to predict the existence of a link between pairs of nodes $u, v \in V$ by computing a similarity score $s(u, v)$. This score can be interpreted as the probability of a road between the two nodes, and is thresholded at a chosen breakpoint, e.g. $s(u, v) > 0.6$ to make a prediction.

All node pairs that demarcate the removed edges are tested as positive samples. Following that, in a process known as *negative sampling*, the model is tested on a random sample of node pairs that do not have links between them. The ratio of positive to negative samples is set to 1:5, which allows for more balanced prediction as non-existent links vastly outnumber existing links[1]. Testing every single negative edge would otherwise incur significant computation time.

Link prediction can be interpreted as a binary classification task, where false positives refer to newly predicted links, and the removed links that are not recovered successfully are false negatives. As such, classification metrics are often used for evaluating link prediction models [59].

4.2.1 Datasets for Link Prediction

To construct graphs for link prediction, the SSx OpenMapping dataset (see Appendix A) is split into subnetworks based on the 348 UK local authorities in 2011. Next, the geo-referenced line segments are converted into *primal graphs* for each local authority. Figure A.4 in Appendix A shows a sample of the graphs generated through this process, together with several compiled statistics.

[1] This approach is adopted from Xue et al. [8]. However, we deal with much bigger road networks in this project, which makes testing all negative edges prohibitively expensive.

Two other datasets are used in subsequent regression analyses. The English indices of deprivation 2011 is a dataset containing statistics on relative deprivation for small areas in England, summarized for local authorities [60]. We also use a dataset detailing the Gross Value Added (GVA), a measure of the economic output of local authorities, similar to the Gross Domestic Product for countries [61]. Such statistics have been applied in urban network analyses [62] to quantify the socio-economic status of cities.

4.2.2 Cross-validation Setup on Multiple Graphs

To optimize model parameters for performance across the dataset, a method to perform cross-validation over multiple graphs is required. Crucially, this involves extending the previous task definition so that the model trains to predict links of multiple graphs at once. We thus formulate the transductive-inductive split in two different ways:

- **Batched Graph**: Train on 80% of all graphs with a fifth of their edges removed, test it on the same graphs in full to obtain the *transductive* metrics, and test on the remaining 20% to derive the *inductive* metrics. As the model is forced to generalize to different graph structures, running such a task allows us to obtain generally optimal model hyperparameters. Accordingly, this task is used for cross-validation.

- **Single Graph**: Train on each graph and compute the average over the test metrics obtained on every other graph, using the model parameters optimized via the Batched Graph task. The metrics for transductive performance, which measures the intra-city spatial homogeneity, are obtained directly by testing the model on the same graph it was trained on. Applying the same model to other graphs reveals the inductive performance, which then determines network similarity.

Details of our model and how the space syntax measures are processed to form input features are provided in Sections 4.4 and 4.5.

4.2.3 Metrics for Link Prediction

The following binary classification metrics are used to quantify the performance of our model. Some metrics are also used for subsequent regression analyses.

1. **Precision, Recall, F1**: Precision (P) describes the classifier's ability to avoid labeling negative samples (sampled nonexistent roads) as positive, and recall (R) describes its ability to find all positive samples (existing roads). F1 balances the two measures via the harmonic mean: $\text{F1} = \frac{2(P+R)}{P \times R}$. These metrics require thresholding model outputs to make predictions, as described in Section 4.2.

2. **Area Under the Receiver-Operating Characteristic (ROC) Curve (AUC)**: This metric captures the extent to which a positive sample is more highly ranked by the classifier than a negative sample across varying thresholds. This metric is used to compare models, as it is less sensitive to the negative sampling ratio. A random classifier achieves a baseline AUC of 0.5.

3. **Average Precision (AP)**: This metric summarizes the relationship between precision and recall as the threshold is varied between values n, and is formulated by $\text{AP} = \sum_n (R_n - R_{n-1})P_n$. The AP achieved by a random classifier is the ratio of positive samples to total samples.

4. **Pearson's r**: The metric expresses the linear correlation between two variables, with a two-tailed p-value that determines its statistical significance. This metric is used to investigate relationships between the above metrics and potential explanatory variables. It ranges from 1 to -1, with 1 indicating perfect correlation, 0 indicating no linear relationship, and -1 perfect anti-correlation.

The AUC and AP are standard metrics used in link prediction to quantify performance [59], both ranging between 0 and 1. A random classifier for our tasks would achieve an AUC of 0.5 and an AP of 0.167 given our negative sampling ratio of 1:5. This ratio directly affects the range the AP can take and can be used as a control to manipulate the distribution of output metrics.

4.3 Results

4.3.1 Best Performing SSx Feature: Integration

	Transductive		Inductive	
Feature set	AUC	AP	AUC	AP
Ranked Choice 2km & 10km	0.705	0.366	0.702	0.364
Ranked Integration 2km & 10km	**0.880**	**0.569**	**0.878**	**0.564**
Ranked Choice & Integration	0.844	0.533	0.842	0.528
Node Counts 2km & 10km	0.857	0.480	0.855	0.476
All SSx features	0.653	0.305	0.651	0.302
Coordinate Features	0.540	0.240	0.531	0.233
Local Degree Profile	0.573	0.252	0.573	0.252

Table 4.1: **[Batched Graph Task]** Feature set experiment results under transductive and inductive testing. Out of the space syntax measures, the GNN achieves the best performance with ranked integration. The "SSx (space syntax) features" are described in Appendix A, while "Coordinate Features" refers to the point coordinates in the EPSG:27700 coordinate reference system. Each experiment was run for 5 iterations, taking an average of 15 minutes per iteration (500 epochs). The average margin of error is 0.015 ($p = 0.05$).

The results in Table 4.1 show that the best performing set of space syntax measures performs better than baseline coordinate feature data, with a 62.9% and 65.5% improvement on transductive and inductive AUC respectively[2]. Out of all the space syntax measures, ranked angular integration provides the most relevant information for the link

[2]In these earlier experiments, sum-to-one normalization was used, which scales every input feature vector such that all of its components sum to one. In Section 4.3.3, we however show that quantile-normalization leads to a better performing feature set.

prediction task. A possible rationale is that areas with high integration tend to be denser in road connections, and as such, the model may be learning to predict more links if the nodes have higher integration values.

However, when other measures are used, our model returns poorer performance, and combining those metrics with choice or integration causes a decline in performance. This suggests that using too many redundant features can confuse the model while it attempts to produce discriminating representations from the high dimensional input space. In other words, this leads to the curse of dimensionality [35].

4.3.2 Average Precision as a Spatial Homogeneity Measure

Xue et al. [8] used the F1 score of the link prediction task as a measure of spatial homogeneity. However, their choice of 0.61 for the binary prediction threshold may not generalize well to arbitrary graphs, as this measure is computed based on the average best performance on their dataset. In contrast, our alternative metric of average precision forgoes the need for thresholding while still being able to measure performance.

For comparison, Figure 4.1 shows the distribution of AP and F1 scores with a threshold of 0.6[3] achieved per local authority in the Single Graph task, and we report a Pearson's r value of **0.821**, indicating a strong linear relationship between the two metrics. A map visualization is provided in Figure 4.2, further showing a similar spatial distribution. Since the metrics are highly correlated, we choose the unbiased AP instead to measure spatial homogeneity.

Avoiding the filtering procedure saves on computation

In the same paper by Xue et al. [8], predicted road links were filtered to eliminate impossible new roads, such as ones that cross or form a tight acute angle with existing roads. When filtering is applied in addition to our pipeline, performance significantly improves since many false positives are eliminated, as Figure 4.3 shows. In spite of the benefits, such an operation takes up considerable computation time and is not crucial when measuring spatial homogeneity. With space syntax, our pipeline is able to achieve comparable performance without filtering, while taking less time to compute.

Correlation analysis with space syntax features and network measures

To interpret our results, we perform correlation analyses between our model's achieved performance (measured using the AP), and various statistics of each graph.

Results in Table 4.2 reveal that spatial homogeneity has a significant inverse relationship with the mean space syntax measures, as well as other several graph measures. For integration, a large mean value indicates high road density. Dense graphs may induce poorer performance, as there are more links that the model has to predict successfully. The significant relationship with choice also corroborates the correlation between spatial homogeneity and average betweenness shown by Xue et al. [8].

[3]This is the threshold that gives the highest average F1 on our graphs.

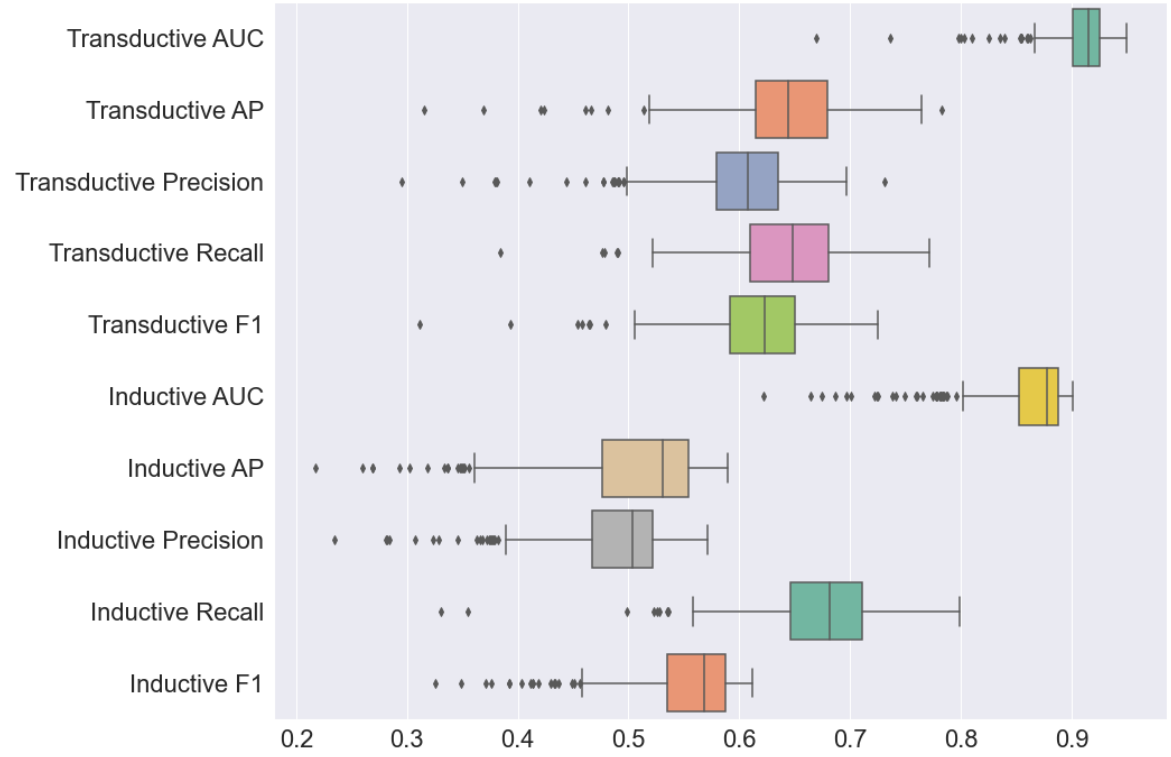


Figure 4.1: **[Single Graph Task]** Box plots showing the distribution of transductive and inductive metrics. As expected, the inductive setting induces worse performance, mostly due to false positives as precision drops while recall stays about the same. The distributions of the AP and F1 are also comparable, showing that potentially biased thresholding required for the F1 may not be necessary for a homogeneity measure based on link prediction.

Measures like the number of nodes and edges, the mean shortest path length, and network diameter all indicate graph size in some aspect. Their small positive correlation with performance may be due to larger networks contributing more data for link prediction, allowing the model to make more informed predictions. However, measures like the average degree and network density indicate more complex graphs with many edges. This potentially leads to a more challenging classification task for the model, causing a negative correlation between those metrics and the AP.

Correlation analysis with socio-economic indicators

Spatial homogeneity has been shown to share nuanced relations with societal factors that may influence, or be influenced by, urban development patterns [8]. To exemplify such studies, we compute the Pearson's r between the transductive AP and several indicators of development across all local authorities for which the statistics were available. These include economic indicators such as the GVA (348 local authorities) and social indicators such as the Index of Multiple Deprivation (IMD, 326 local authorities). These statistics come from the datasets described earlier in Section 4.2.1, and are filtered to the year 2011, which corresponds with our road data.

The plots in Figure 4.5 show several interesting inverse correlations, including a trend where an area with high homogeneity tends to experience a smaller IMD, albeit also with

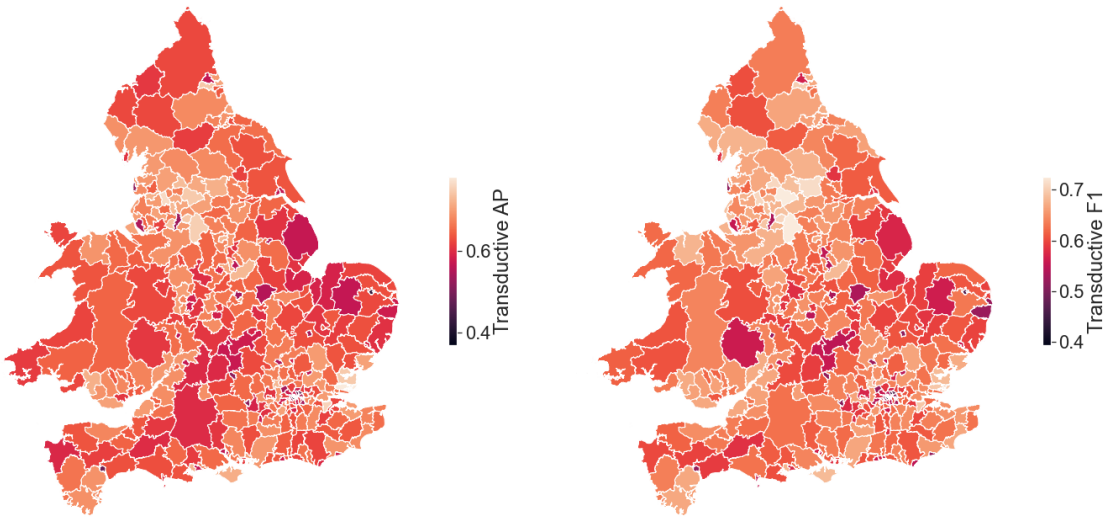


Figure 4.2: **[Single Graph Task]** Choropleth map of UK Local Authorities based on the achieved transductive AP and F1 (lighter regions see better link prediction performance). The spatial distributions are similar for both metrics, which share a Pearson's r of 0.821 ($p = 2.43 \times 10^{-86}$).

a lower GVA. A possible reason is that larger cities generate more income, but due to greater diversity within their communities, their road networks are less homogeneous.

These analyses are meant to exemplify possible space syntax studies that use link prediction performance as a metric for analyzing urban development and its effects on people. A more in-depth analysis would incorporate more contextual features and methods to diminish the impact of confounding variables, such as road network size, on the results.

Measuring Network Similarity

When link prediction is applied inductively, the resultant metric gives an indicator of the topological similarity between road networks. This represents the potential of transferring knowledge from one area to another, for example, to develop a new town based on an existing city. To illustrate how well the inductive AP can reveal network similarity, Figure 4.4 shows several pairs of local authority graphs and their corresponding metrics.

4.3.3 Quantile Normalization Substantially Improves Performance

Late in the project, we discovered that changing the normalization function used for the experiments in Table 4.1 results in substantially improved performance, especially for the baseline coordinate features. Expanding on this discovery, we investigate the following normalization techniques, which are applied to the features of each graph just before input.

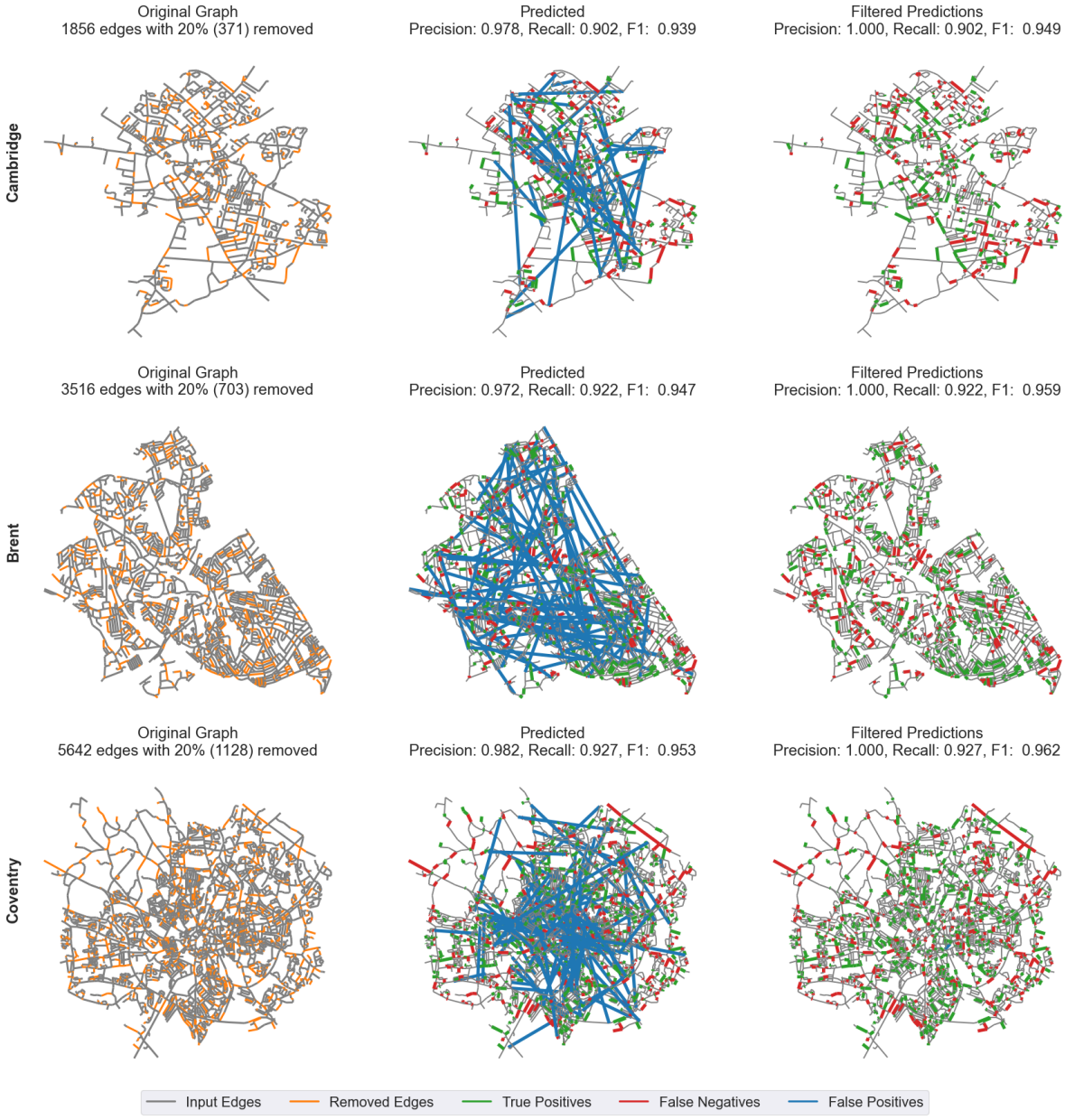


Figure 4.3: Map visualization of link prediction on the Cambridge, Brent, and Coventry road networks, with 20% of links in the initial graph randomly sampled and removed. The model tries to recover those edges, while also avoiding positively predicting edges that do not exist in the original graph. The ratio of negatively sampled edges is lowered to 1:1 for less map clutter. The figures on the right show the effect of filtering invalid new edges, which greatly boosts the Precision score and in turn the F1. The model used for all figures is the same GAIN-DistMult GAE trained using the Batched Graph Task.

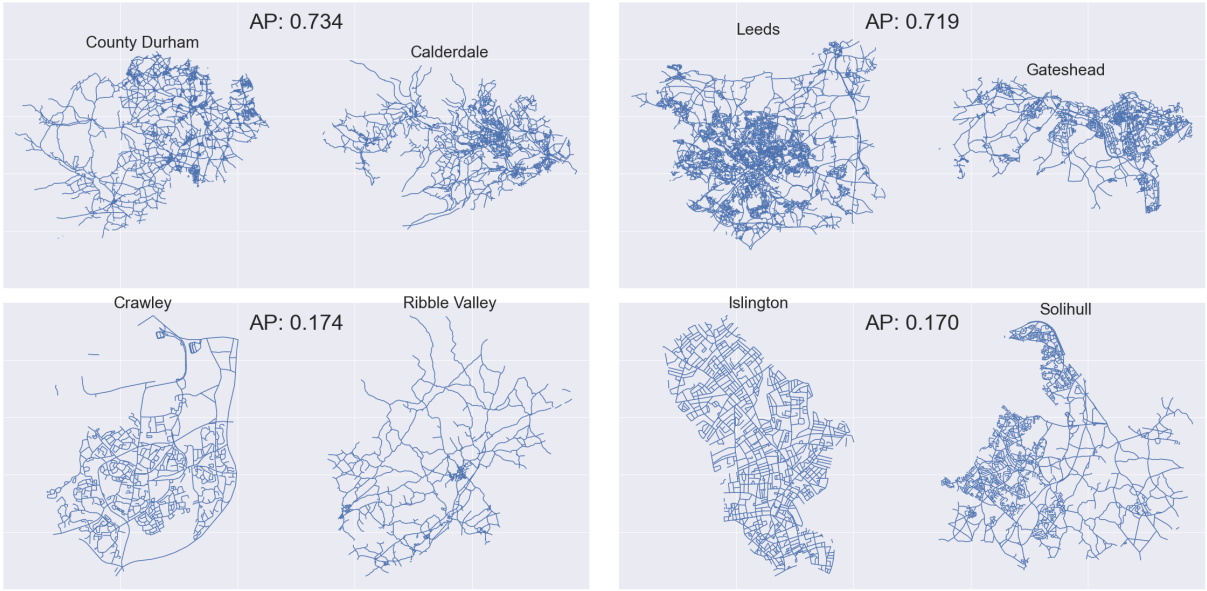


Figure 4.4: **[Single Graph Task]** Pairs of local authority graphs that give rise to the highest (row above) and lowest (row below) inductive AP scores. Here, the model is trained on the left graph and applied to the right graph. This visual comparison clearly demonstrates how the inductive AP can measure network similarity.

- **Min-Max Normalization**: Every feature is adjusted into the range $[0, 1]$ based on their minimum and maximum value. This technique scales the feature range while keeping the probability distribution the same. Consequently, it is prone to outliers.
- **Quantile Normalization**: This technique transforms the input feature's distribution to match that of a new probability distribution, for which the Standard Normal is used. This transformation consists of applying the estimated cumulative distribution function of the input features and then the quantile function of the Standard Normal. In doing so, the relationships between individual values are smoothed out and mapped into the new distribution. The most frequent values also get spread out, with outliers becoming less impactful.

As normalization is done *after* the feature split into local authorities, the techniques only consider the feature distribution within each graph. Figure A.2 in Appendix A displays the effect of both approaches on coordinates and integration at 10 km (the tested features).

Our results show that with quantile-normalized coordinate features, our model is able to reach an AUC of 0.992 and an AP of 0.918 in the inductive setting. This improves upon the AP achieved by our previous approach of ranked integration by a very substantial 43.4%, as well as the filtering architecture discussed in Section 4.3.2. We also investigate normalizing the space syntax measures in the same way, but the AP with coordinate features remains 17% higher than that of quantile-normalized integration. It is likely that with quantile-normalized coordinate features, our model is able to derive a distance representation that allows it to innately filter out impossible links.

As also shown in Figure 4.6, appending quantile-normalized integration does not improve upon the achieved AP. While space syntax does not enhance performance, these

	Category	Target variable	Pearson's r	Sign	p-value
(a)	Space Syntax	choice2km	0.453	-	5.18e-19
		choice10km	0.444	-	2.79e-18
		choice100km	0.020	+	0.708
		integration2km	0.431	-	3.44e-17
		integration10km	0.392	-	3.13e-14
		integration100km	0.040	-	0.458
(b)	Graph Statistics	Number of nodes	0.201	+	0.000163
		Number of edges	0.200	+	0.000173
		Average degree	0.273	-	2.41e-07
		Density	0.224	-	2.55e-05
		Mean shortest path length	0.159	+	0.00301
		Network diameter	0.170	+	0.00147
(c)	Indices of Deprivation (rank) & GVA	IMD	0.191	-	0.000539
		Crime	0.186	-	0.000728
		Income	0.106	-	0.0548
		Health Deprivation & Disability	0.024	-	0.672
		Barriers to Housing	0.442	-	5.45e-17
		Living Environment	0.358	-	2.80e-11
		2011 GVA	0.418	-	3.09e-15

Table 4.2: **[Single Graph Task]** Pearson's r with corresponding p-values between the transductive AP and (a) the average unnormalized space syntax measures at various radii, (b) graph metrics, and (c) socio-economic indicators. The greatest correlation is achieved with choice and integration at smaller radii, from which it can be inferred that highly integrated networks are less topologically homogeneous. There is no statistically significant correlation for measures taken at the large radius of 100 km, which is expected as these measures are computed on a scale much larger than individual local authorities. For the indices of deprivation, a larger number indicates worse conditions, whereas for GVA, a larger number indicates high regional income.

results show that the choice of data normalization is critical for the link prediction task on road networks. For the full results of our experiments, refer to Appendix B.

4.4 Data Processing

4.4.1 Graph Processing

To obtain the graphs corresponding to each local authority, the line geometry in SSx OpenMapping is split based on local authority designations, and then transformed into `networkx` [63] graphs using `gdf_to_nx` from `momepy` [64], a Python library for urban form analysis. The graphs are then truncated to their largest connected component, as the division into local authorities occasionally produces several disconnected subgraphs for one local authority graph.

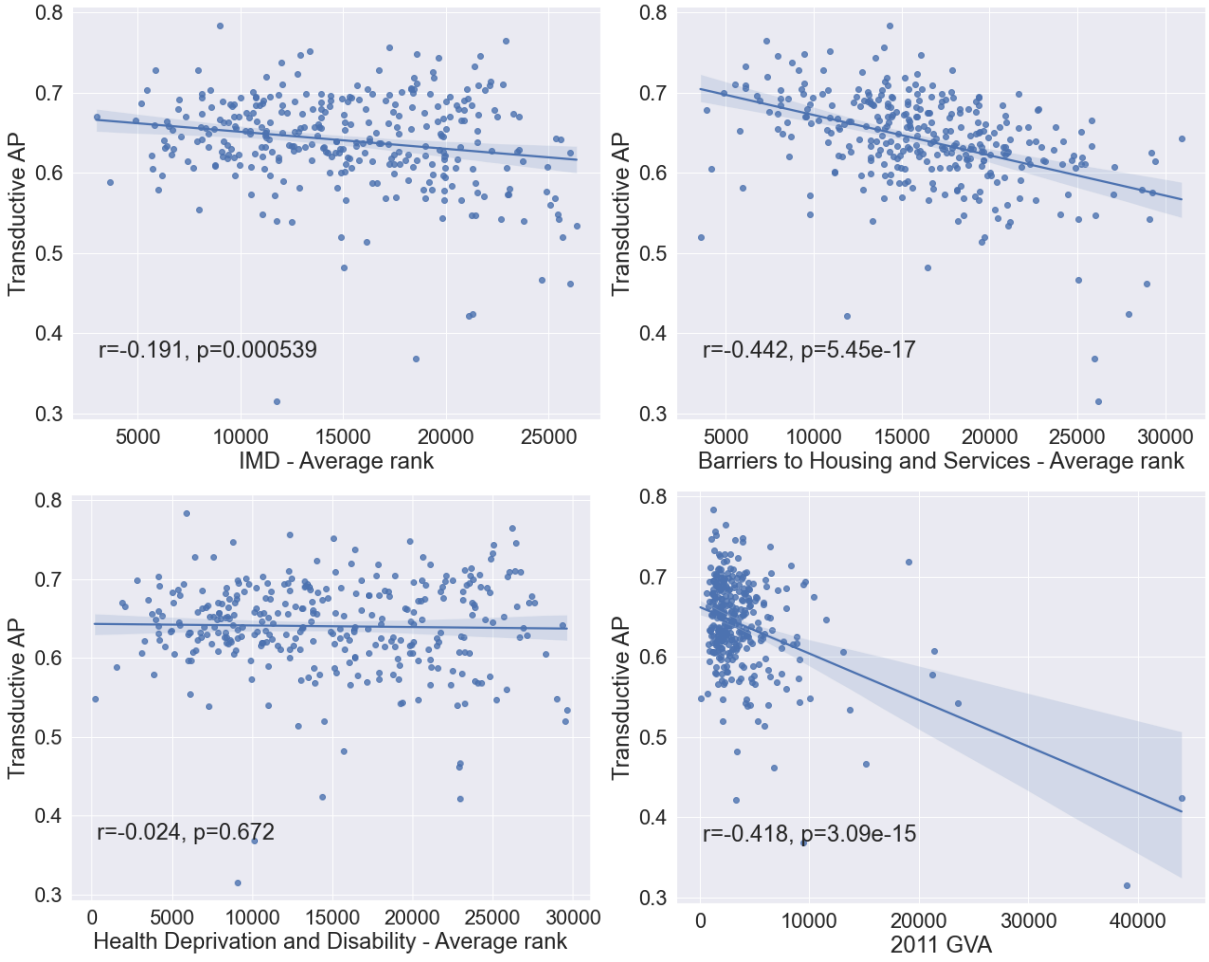


Figure 4.5: **[Single Graph Task]** Correlation analyses between various local authority statistics and the AP achieved on its road network. Not all have a significant relationship: Health Deprivation and Disability, among other statistics, do not exhibit any correlation with the AP.

4.4.2 Feature Processing

Although the `log` and `rank` features in SSx OpenMapping are already normalized (see Section 2.1.2), other features are of dominantly large scales, such as choice at 100 km as shown in Figure A.4 of Appendix A. For each feature to be considered evenly, they must first be normalized into a similar range of values. However, since we only use ranked integration for our experiments in this section, any further normalization approaches are not considered.

> As Section 4.3.3 revealed, quantile normalization appears to be a superior approach. The results in this section pertaining to relative performance remain valid nevertheless, as we did not evaluate feature sets for these earlier experiments.

Edge Feature Pooling

Node-level representations are obtained by pooling edge features, which involves combining the space syntax measures from the roads incident on that node. This procedure is

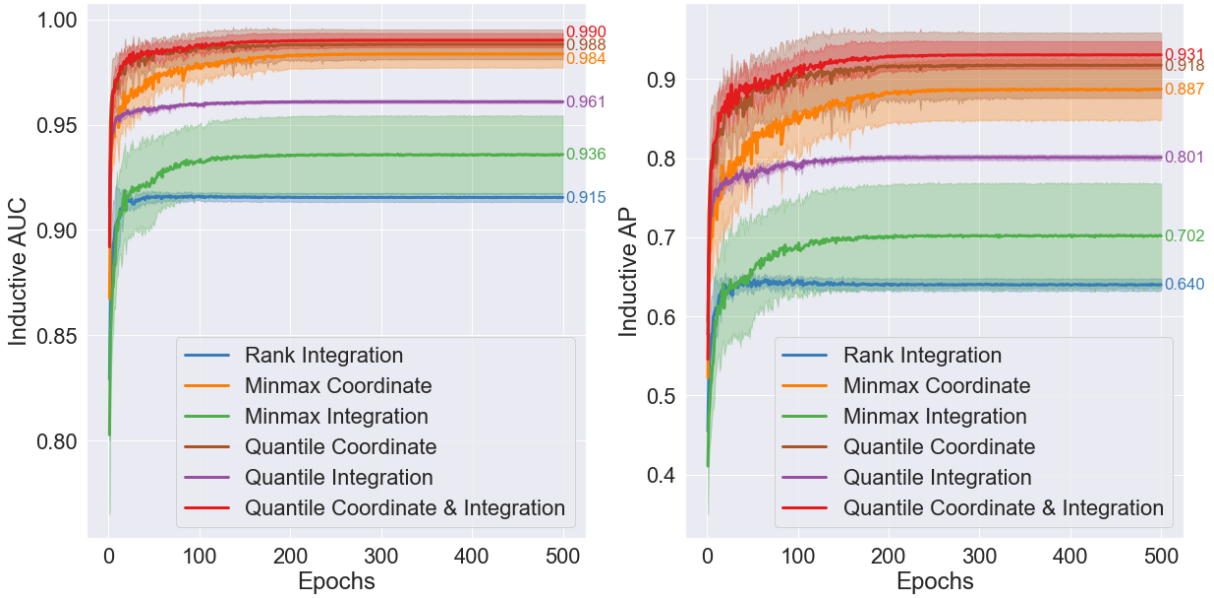


Figure 4.6: **[Batched Graph Task]** Inductive performance over the course of training, using the GAIN-DistMult GAE and various quantile-normalized features. Normalized coordinates give a large increase in performance compared to our previous approach. For the full results, which include other space syntax feature sets, refer to Table B.1 of Appendix A.

Algorithm 1: Edge feature pooling: Creating road node features from edge features in the primal graph representation

Data: Graph $\mathcal{G}(V, E)$; input edge features $\mathbf{x_e}$ for $e \in E$; aggregation function AGGREGATE; edge lookup table $edges : V \to E$
Result: Node features $\mathbf{x_v}$ for $v \in V$
for $v \in V$ **do**
  $x_v \leftarrow \text{AGGREGATE}(\{\mathbf{x_e}, \forall e \in edges(v)\})$
end

illustrated by Algorithm 1. Although pooling does smooth out the distribution of the measures, it is necessary as all the space syntax measures are edge features while our model requires node representations.

Instead of just taking the average, other aggregation functions such as the sum or maximum can be used, analogous to neighborhood propagation in convolutional GNNs. We hence evaluate different aggregation functions and report the achieved metrics in Table 4.3. Our results show that minimum aggregation leads to the best performance, although not by a large margin.

4.5 Modified Graph Autoencoder

Our model is based on the graph autoencoder (GAE) [59], which is capable of generating node embeddings in an unsupervised fashion using a GNN encoder and a decoder. Figure 4.7 illustrates our model.

	Transductive		Inductive	
Aggregation	AUC	AP	AUC	AP
Max	0.892 ± 0.001	0.570 ± 0.005	0.890 ± 0.002	0.564 ± 0.006
Min	$\mathbf{0.908 \pm 0.003}$	$\mathbf{0.614 \pm 0.012}$	$\mathbf{0.907 \pm 0.003}$	$\mathbf{0.608 \pm 0.014}$
Sum	0.846 ± 0.004	0.477 ± 0.009	0.843 ± 0.004	0.470 ± 0.008
Mean	0.904 ± 0.003	0.606 ± 0.009	0.903 ± 0.003	0.600 ± 0.008
Median	0.898 ± 0.002	0.594 ± 0.010	0.897 ± 0.002	0.590 ± 0.009
Std	0.639 ± 0.012	0.264 ± 0.006	0.633 ± 0.011	0.258 ± 0.007

Table 4.3: **[Batched Graph Task]** Performance achieved by different aggregating operations for edge feature pooling ($p = 0.05$). The minimum results in the best performance, although the differences are fairly small overall (save for standard deviation, which performs significantly worse). The features used are `integration2kmrank` and `integration10kmrank`, with the GAIN-DistMult GAE, as these were shown to derive the best prediction results. Each experiment was run for 5 iterations over 500 epochs, with the reported metrics being the ones achieved at the end of each iteration.

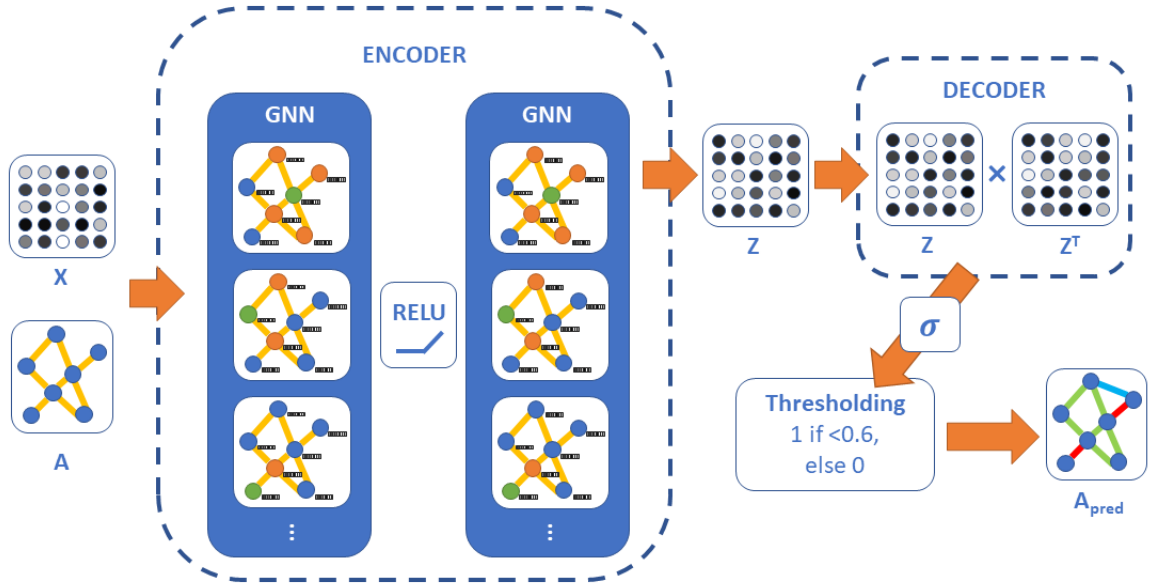


Figure 4.7: An overview of the graph autoencoder architecture based on Kipf et al. [59] used for the self-supervised link prediction task. X represents the input node features, A the adjacency matrix, and Z the latent features obtained at the output of the encoder. The decoder reflects the standard inner product version, and the two GNN layers of the encoder are flexible in their message-passing scheme.

4.5.1 Encoder Layer Types

Given a primal graph representation of a road network with features attached to each node, the encoder maps each node feature vector to a latent vector z of set length 10 (derived from cross-validation). As the GNN layers within the encoder are flexible in their message-passing strategies, several layer types are tested as follows.

The **Graph Convolution Network (GCN)** [46] generates the node representations $\mathbf{H}$ of the next layer $l + 1$ by convolving neighboring node representations from layer l as

follows:

$$\mathbf{H^{(l+1)}} = \sigma(\tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}\mathbf{H^{(l)}}\mathbf{W}) \tag{4.1}$$

where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ is the sum of the adjacency and identity matrix for numerical stability, $\tilde{\mathbf{D}}$ is the degree matrix for magnitude normalization, $\mathbf{W}$ is the learnable weight matrix, and $\sigma(.)$ is the activation function.

An alternative layer type that has seen success when applied to road network graphs is the **Graph Attention Network (GAT)** [41]. It applies attention to the convolution operation, and is hence more expressive while incurring greater computational cost. For the GAT, a single node representation, h_v, at layer $l+1$ is computed as follows:

$$h_v^{(l+1)} = \sigma\left(\alpha_{vv}\mathbf{W}h_v^{(l)} + \sum_{u\in\mathcal{N}(v)} \alpha_{uv}\mathbf{W}h_u^{(l)}\right) \tag{4.2}$$

where $\mathcal{N}(v)$ is the set of neighborhood nodes for node v, and α_{uv} refers to attention weights, computed via self-attention (Section 2.2.3). The similarity score for attention is parameterized with a weight vector $\mathbf{a}$ as follows:

$$s(h_u, h_v) = \text{LeakyReLU}(\mathbf{a}^T(\mathbf{W}h_u \| \mathbf{W}h_v)) \tag{4.3}$$

where LeakyReLU is a variant of the ReLU function and $\|$ denotes concatenation.

The operator **Graph Sample and Aggregate (GraphSAGE)** [12] is designed for inductive performance by uniformly sampling a fixed-sized set of neighbors $\tilde{\mathcal{N}}(v) \subseteq \mathcal{N}(v)$ per node and combining their features with an aggregation function AGG. For AGG, the original paper recommends taking the mean and maximum. This paper uses the format SAGE-AGG to refer to the selected aggregation function (e.g. SAGE-MEAN, SAGE-MAX). The GraphSAGE operator also has separate projection matrices W_1 and W_2 for self loops and edges respectively as follows:

$$h_v^{(l+1)} = \sigma(\mathbf{W_1}h_v^{(l)} + AGG_{u\in\tilde{\mathcal{N}}(v)}\mathbf{W_2}h_u^{(l)}) \tag{4.4}$$

The **Graph Isomorphism Network (GIN)** [65] uses the following message passing scheme:

$$h_v^{(l+1)} = \text{MLP}\left((1+\epsilon)h_v^{(l)} + \sum_{u\in\mathcal{N}(v)} h_u^{(l)}\right) \tag{4.5}$$

where MLP is an input neural network that models an injective function, and ϵ can be learned as a parameter or set to zero. In the same paper, the GIN was shown to be able to distinguish graph structures that the GraphSAGE and GCN could not. For our evaluation, the MLP was fixed to 2 layers following the original paper, with $\epsilon = 0$ as it was shown to have similar performance.

Gharaee et al. [6] enhanced the GIN with GAT-like attention using the following **Graph Attention Isomorphism Network (GAIN)** operator:

$$h_v^{(l+1)} = \text{MLP}\left((1+\epsilon)\mathbf{W}h_v^{(l)} + \sum_{u\in\mathcal{N}(v)} \alpha_{uv}\mathbf{W}h_u^{(l)}\right) \tag{4.6}$$

In their paper, this GAIN model performed better than GAT or GIN for road network classification tasks, although it also incurs the greatest computation complexity.

4.5.2 Decoder and the Symmetric DistMult

We evaluate two types of decoders:

1. Taking the inner product between nodes as the similarity score, which was used in the original GAE paper [59]
2. A weighted inner product decoder based on DistMult from the RGCN architecture [48]. Given latent vectors z_u and z_v, the similarity score for nodes u, v is computed as $s(u, v) = z_u^T W z_v$

The weights in DistMult allow for a more refined extraction of score components from the pair of latent vectors, in a manner similar to attention. DistMult was designed for directed heterogeneous graphs, and can thus produce an asymmetric adjacency matrix, i.e. given two nodes u and v, $s(u, v) \neq s(v, u)$ in general. However, road networks are represented using *undirected* primal and dual graphs in this project. To enforce symmetry of the output adjacency matrix, we modify DistMult by adding the transposed weights prior to taking the product. This symmetric DistMult is therefore computed as $s(u, v) = z_u^T (W + W^T) z_v$.

4.5.3 Evaluation

Training Setup

The training methodology employed in our evaluation is adapted from Kipf et al. [59]. The various parameters that follow were again obtained by 5-fold cross-validation with a fixed model on the Batched Graph task.

Our model is trained by minimizing the reconstruction loss (binary cross-entropy) on training graphs that have 20% of edges removed, with negative edges that are resampled every epoch. For the Batched Graph task, we utilize `Batch` from PyTorch Geometric, an abstraction that allows multiple graphs to be treated like a single graph by stacking adjacency matrices diagonally. To speed up convergence, each `Batch` is set to contain 8 graphs, enabling multiple parameter updates per epoch. We use a learning rate of 0.01, but reduce it over time as performance on the inductive set plateaus.

In the Single Graph task, only a single graph is used for training, which requires a lower learning rate of 0.0005 as the limited amount of data causes more training instability. To *inductively* test the model in the Single Graph task, the model is evaluated on the graphs of every other local authority, and we report the average metric achieved over them.

Hyperparameters

The 2 layers and 10 output dimensions are decided by 5-fold cross-validation on the Batched Graph task, using the inductive AUC as the evaluating metric. The best performance is achieved with just 2 layers; having more layers fails to achieve better performance while taking longer to train. This result is likely due to the over-smoothing issue mentioned in Section 2.3.2.

One solution, known as Jumping Knowledge (JK), borrows the idea of residual connections by concatenating the outputs of each layer before a final output linear layer [50]. In this way, the final layer still sees the raw, non-smoothed features. However, initial testing showed that enhancing the encoder with JK and more layers still results in similar or even worse AUC, and hence this line of investigation was not pursued.

Results

Table 4.7 contains link prediction metrics for varying encoders and decoders on the Batched Graph task, with the number of model parameters and training time per epoch for comparison.

				Transductive		Inductive	
Decoder	Encoder	P. count	Time(s)/epoch	AUC	AP	AUC	AP
Inner	GCN	140	1.524	0.811	0.434	0.809	0.431
	GAT	180	1.527	0.825	0.518	0.822	0.514
	SAGE-MEAN	260	1.307	0.732	0.381	0.729	0.377
	GIN	400	1.349	0.745	0.316	0.743	0.310
	GAIN	660	1.683	0.862	0.589	0.863	**0.586**
DistMult	GCN	240	1.551	0.851	0.441	0.849	0.438
	GAT	280	1.557	0.882	0.570	0.880	0.565
	SAGE-MEAN	360	1.339	0.734	0.404	0.729	0.399
	GIN	500	1.385	0.786	0.349	0.782	0.343
	GAIN	760	1.703	**0.901**	**0.591**	**0.899**	0.585

Table 4.4: **[Batched Graph Task]** Link prediction metrics obtained by varying encoder (2 layers, 10 output dimensions) and decoders, with the total learnable parameter count of the model and the average time taken in seconds per training epoch. Best performances are in bold. We report the mean results over 5 iterations, with an average margin of error of ± 0.0015 ($p = 0.05$). The feature set used is ranked integration (the best performing set). The training run times are obtained by single-threaded execution on a system with an Intel i7-7700K CPU and a NVIDIA GeForce GTX 1080 GPU.

These results showcase the importance of attention in link prediction for road networks, as the GAT and GAIN models are the only models that achieved an AP of over 0.5. From the metrics, we can also determine that the GAIN encoder with the DistMult decoder obtains the best transductive and inductive performance overall through the AUC.

Generally, additional parameters are beneficial: the DistMult decoder nearly always improves performance, while the GAIN model maintains a slight edge over both the GAT and the GIN models. The GAIN also maintains its performance on the Single Graph task, for which the achieved ROC and PR curves for differing encoders are depicted in Figure 4.8. The PR curves in particular suggest that the expressive power of attention greatly enhances link prediction performance, with both the GAT and GAIN having the largest area under the curve.

It is worth mentioning that high link prediction performance is not necessary when measuring spatial homogeneity or network similarity. High performance may even be

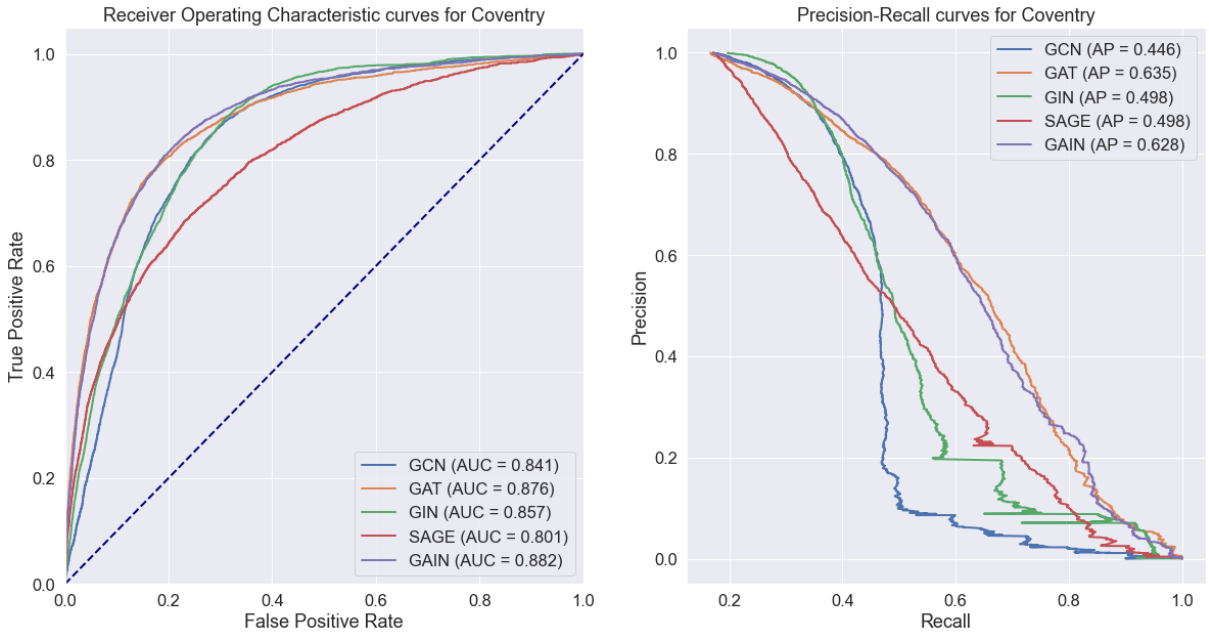


Figure 4.8: **[Single Graph Task]** ROC and PR curves (average of 5 iterations) achieved by the various encoders with the DistMult decoder on the graph of Coventry's road network. The GAIN encoder achieves the highest AUC, while the GAT encoder achieves the highest AP, which is related to the area under the PR curve.

detrimental if the output distribution becomes so narrow that differences become insignificantly small. Hence, even though improving a model to achieve F1 scores above 0.9 is possible, it may be undesirable for comparative analysis.

4.6 Discussion

4.6.1 Spatial Clustering of the Latent Variable

The GAE passes intermediate node representations (*latent vectors*) from the encoder to the decoder, which then computes a link probability between nodes. As roads typically connect nearby nodes, these representations encode spatial information intrinsically while optimizing for link prediction. To investigate this, we conducted a **spatial autocorrelation** analysis in Figure 4.9 using Moran's I [66], which quantifies the extent of the relationship, if any, between location and attribute values. Since the analysis is restricted to a single variable, the 10-dimensional latent vectors are reduced to 2-dimensional vectors via TSNE [67], a dimensionality reduction algorithm, and we analyze each dimension.

Interestingly, the latent representations capture some spatial clustering even though the model does not use spatial features like coordinate data. The nature of neighborhood propagation in GNNs could be contributing to this result, as nearby connected nodes are optimized to have similar latent representations. Yet, as the model does not achieve perfect accuracy on the link prediction task, the clustering is not perfect.

It may be the case that a GAE that achieves high accuracy can produce latent vectors with better clustering metrics. However, that model would have to achieve this without manual filtering techniques (Section 4.3.2).

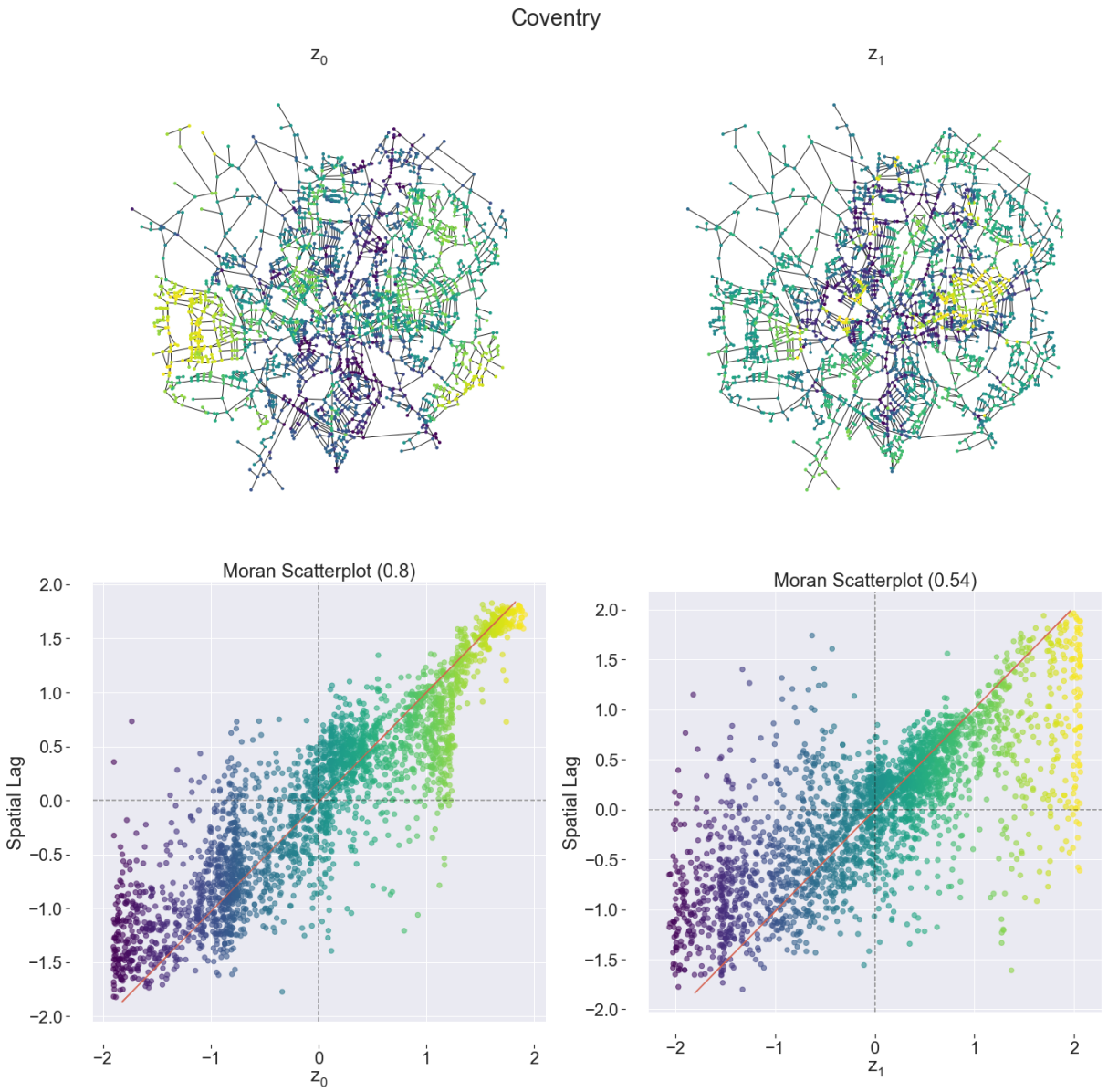


Figure 4.9: Above - Map visualization of output node latent vectors, reduced to 2 dimensions, for Coventry. Below - A Moran's I plot showing the spatial autocorrelation of the two dimensions, with the value bracketed in the title. A value of 1 indicates a completely clustered distribution, while a random distribution would result in 0. A uniform spatial distribution would be indicated by -1. This analysis reveals significant spatial clustering captured by the latent vector.

4.6.2 Limitations & Other Approaches

Other than graph-based approaches, link prediction can be done through heuristics that directly produce similarity scores between nodes. An example is the Jaccard coefficient, which computes similarity based on how many neighbors nodes share [68]. Such heuristics have been integrated with GNNs in the SEAL framework [69], which is the state of the art for general link prediction problems.

As we aimed to demonstrate the synergy between space syntax and GNNs, a comprehensive look at all possible strategies in link prediction is beyond this project's scope. Furthermore, these measures were developed for link prediction in social networks, and

may not be as compatible on road networks, which are more heterogeneous and sparse [5]. Still, it may be worth investigating the integration of space syntax with such techniques.

For simplicity, we assumed the use of undirected planar graphs in this project, but road networks are directed and 3-dimensional in reality. However, our model can be adapted for directed road graphs by reverting the DistMult modification, while no adjustments are required for non-planar graphs.

While we expressed the potential of link prediction in transport network design and planning, more work in realistic settings is required to demonstrate its practical applicability. Yet as the evolution of actual road infrastructure is affected by factors such as topological limitations, politics, and culture [49], such predictive models may be constrained to a limited part of the overall decision process.

Subgraph Partitioning by Local Authority

Partitioning based on the local authorities was done to allow correlation analyses with city statistics, and to reduce the scale of the link prediction task. This approach does have limitations: the boundaries of local authorities are defined by data and do not necessarily preserve graph connectivity after partitioning. There is also a variety of sizes across the split graphs, which may have influenced the trends in our regression analyses. Modularity-based partitioning algorithms such as METIS [70] may hence be a better approach to split a large network for link prediction analysis.

Negative Sampling

With quantile-normalization, our model predicted fewer roads between distant road nodes randomly chosen by our negative sampling approach. To exceed this level of performance, a better approach for negative sampling may be to only consider the k-nearest unconnected road nodes. In this way, the model can be evaluated on more ambiguous links that cannot be easily filtered by distance.

With this modified objective, space syntax measures may contribute better as input features. A robust link prediction model can then comprise this model, as well as a simple filter that eliminates any predicted roads longer than the longest road segment in the dataset.

Variational GAE

The Variational GAE (VGAE) [59] generates mean and standard deviation vectors which can alleviate overfitting, analogous to the VAE described in Section 2.2.2. Yet, initial experiments revealed that VGAE performs worse than our standard autoencoder in link prediction for both transductive and inductive settings. Training a VGAE also introduces another sensitive parameter, β, in the loss function, and the optimal value for β may differ between graphs.

Nonetheless, the VGAE can be used as a generator of edges through random sampling, which can be useful for road network design. The VGAE is still unable to form new road nodes that can allow, for example, the creation of a new intersection to form a new road.

Models for generative tasks that go beyond a fixed set of nodes include the GraphRNN [71]. These deep graph generative models, which were designed for molecular graphs, may have the potential to generate small-scale road network designs.

4.7 Summary

Our initial results in road network link prediction showed that rank-normalized angular integration contributes to better performance compared to other space syntax measures. However, we later discovered that using quantile-normalized coordinate features outperforms this approach under both transductive and inductive settings, showing the importance of data normalization techniques for this task. To enable space syntax measures to form node representations, we proposed edge feature pooling via minimum aggregation.

We also proposed using the average precision of link prediction to measure spatial homogeneity, and demonstrated statistically significant correlations with various socioeconomic factors. We argued that the metric can be used as a measure of network similarity in the inductive setting.

Finally, we described our proposed graph autoencoder, which consists of a 2-layer GAIN encoder and a symmetric DistMult decoder. Our results showed that this combination results in higher AUC than other variants. To determine model hyperparameters, we developed and used the Batched Graph task for cross-validation. We also verified that the latent node embeddings produced by the encoder achieves high spatial clustering after training for link prediction.

Chapter 5: Road Classification

In machine learning involving road networks, open datasets are prone to issues of missing or inaccurate values [51]. Space syntax measures can be more reliable than using attributes from these datasets, as they can be computed precisely with just the road network's topological information.

This chapter investigates the potential of space syntax measures with GNNs in two particular classification tasks on roads: **road-type classification** in Section 5.1 and (traffic) **accident count classification** in Section 5.2. We show that when space syntax measures are added to baseline feature sets, the models improve in both tasks, achieving in the transductive setting a 39% higher weighted F1 score in road-type classification, and a 26% lower error in accident count classification. However, we also found that GNNs do not significantly outperform MLPs when space syntax measures are used. These findings and other insights are discussed in Section 5.3.

We optimize and evaluate various data processing methods and models, some of which have yet to be applied in road network representation learning. These include data loaders for dividing large road network graphs into mini-batches (Section 5.1.4) and aggregation functions for simplifying road geometries (Section 5.2.5). Furthermore, we integrate CORAL [72], a framework for ordinal classification, with our model for accident count classification (Section 5.2.3).

> While the project focuses on two specific tasks, the proposed training methodology, data processing and architecture are applicable, with minimal modifications, for predicting other target values. For example, in places where there is not much data beyond open datasets, typical **speed limits** can be automatically allocated based on existing road attributes.

5.1 Road-type Classification

5.1.1 Motivation

Labeling data in spatial datasets is often time-consuming and labor-intensive. This often leads to missing, incorrect, or outdated labels, especially in open datasets which rely on contributions from the public [51]. Instead, researchers exploit the road graph structure by applying GNNs to perform prediction of road attributes [5, 6]. With the predicted labels, dataset maintainers can automatically assign labels based on trends from existing data.

To exemplify downstream tasks in road representation learning, this chapter analyses a task to classify roads in the UK by their *road-type*, which is defined by the attribute `meridian_class` in the SSx OpenMapping dataset. By comparing multiple feature sets against previous baselines, we evaluate space syntax measures as input features for GNN classifiers. This section will also detail the data processing, models and training setup steps used for these tasks.

Predictions on the Dual Graph

For all experiments in the following section, the roads in the SSx OpenMapping dataset are transformed into the dual representation, with the attributes directly forming input node features. Unlike in the link prediction task, the road network is typically represented by the dual graph (see Section 2.3.1) in road classification tasks [6].

In the dual graph, space syntax measures directly become input node features while providing information about how each road relates to the larger graph network. This may be preferable to adding many message passing layers to propagate information to and from distant nodes, which leads to over-smoothing (see Section 2.3.2).

5.1.2 Methods: Data Processing & Models

`meridian_class`	Description	Count
`motorway`	Multi-carriageway public roads connecting important cities.	9592
`aroad`	A road: Major roads intended to provide large-scale transport links within or between areas.	218558
`broad`	B road: Roads intended to connect different areas, and to feed traffic between A roads and smaller roads on the network.	138544
`minor`	Smaller roads intended to connect together unclassified roads with A and B roads, often linking a housing estate or a village to the rest of the network.	1696628

Table 5.1: **[Road-type Classification]** Counts of road-type classes in the SSx OpenMapping dataset, and their descriptions as defined by the UK Department of Transport's guidance on road classification [73]. Counts are taken after data processing. The dataset is simplified: lower-ranked and unclassified roads are not included.

Dataset and Data Preprocessing

In road-type classification, the objective is to predict the road's category out of the four `meridian_class` shown in Table 5.1. Several combinations of input features were considered here, including the baseline Local Degree Profile and Coordinate Features detailed in Section 3.1.3. We test the space syntax measures taken at 2 km, 10 km, and 100 km, the different categories of features (choice, integration, node counts), as well as all of them combined. The initially unnormalized features (non-`rank` measures) are **min-max normalized** to reduce the variance that results from input features of large magnitude.

Figure 5.1 shows the distribution of the categorical road-type variable over the full dataset, as well as a map to help visualize the typical spatial distribution of the classes in a road network. Since the road-types are labelled at the granularity of line segments, there can be multiple classes within a road segment. Therefore, the line segments are not combined into road segments in this task.

The road network and the `meridian_class` labels originate from the OS Meridian 2, which is a simplified dataset of the full UK road network [10]. Motorways and dual carriageways have been collapsed into a single geometry, and numerous lower order and unclassified roads are not included. As the space syntax features had been computed using this simplification, some bias is incurred in our results.

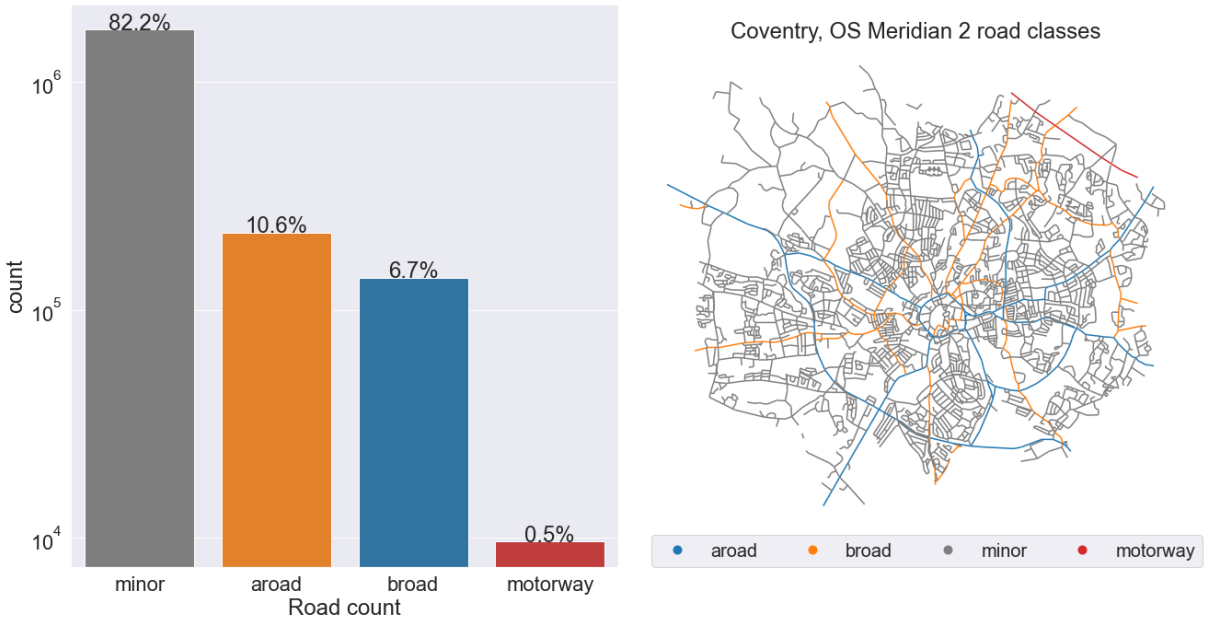


Figure 5.1: OS Meridian 2 class counts (log scale), with a map of Coventry to showcase the typical road types in a city. The vast majority of roads are `minor`, with only a few `motorways`.

Mini-batch Training using Neighborhood Sampler

The SSx OpenMapping dataset comprises over 40 million line segments, which are processed into a dual graph consisting of **2,040,269** nodes and **3,189,373** edges. In order to facilitate machine learning on such a large graph, the Neighborhood Sampling algorithm from GraphSAGE [12] is used, which involves sampling nodes and obtaining their 2-hop computational graphs. Multiple sampled nodes and their computational graphs are then grouped into mini-batches for training the model via stochastic gradient descent (SGD).

We also consider several alternative methods of mini-batching, and the results of our comparisons are detailed in Section 5.1.4.

GNN Architecture

For a multi-class node classification task, a GNN can be applied directly to aggregate neighborhood information and output embeddings describing class probabilities. We use

a GNN with 2 layers to limit over-smoothing, with 20 hidden dimensions and 4 output dimensions (one for each possible class). Similar to link prediction, initial testing via cross-validation showed that increasing model depth past 2 layers does not improve performance. The dimensionality of 20 is also selected in this way.

Our model for classification can be seen as just the encoder part of the autoencoder shown in Figure 4.7 of the previous chapter. We compare varying layer types with baseline MLPs, as well as a naïve classifier that always predicts the mode of the distribution.

Transductive-Inductive Split

To assess the transfer learning potential of our models, a section of the road network corresponding to the county of Kent was removed for use as the inductive test set, leaving about 97.5% of the dataset for the primary training graph. Kent was chosen as it lies on the boundary of the network; truncating it results in few lost road links overall.

Our cross-validation setup for node classification is based on Errica et al.'s comparison of GNNs [74], which divides nodes in the training graph with a 90%-10% training to validation ratio. While the entire adjacency matrix of the graph is input into the GNN to enable message passing over all nodes, the gradient update only considers the loss from training nodes. We report the metrics achieved on the validation nodes, which are indicative of the model's *transductive* performance. The model is then tested on all nodes within the truncated graph of Kent to obtain the *inductive* metrics.

Training Setup with Weighted Loss

Training is performed for 100 epochs, by which the model typically converges with a learning rate that decays over time from 0.01. This is repeated over 5 iterations, each time with a new training-validation split. The model is trained to minimize the **cross-entropy loss**, which is weighted by the reciprocal of the class representation, i.e. the percentages reported in Figure 5.1. This acts as an alternative to upsampling the minority class, as the model weights the gradient signal from minority classes more than that of the majority class (`minor`).

Metrics for Imbalanced Classification

The following metrics are used to evaluate models for the road-type classification task.

- **Weighted F1** weights the F1 score achieved on each class by its representation, and ranges from 0 to 1. As such, this metric is used to assess the overall performance the classifier would have on the actual data distribution.
- **Matthews Correlation Coefficient (MCC)** ranges from -1 to 1, where 1 indicates perfect prediction, 0 random prediction and -1 inverse prediction. This metric considers true negatives and thus captures more of the confusion matrix than the F1 score. As it is more robust to imbalanced data [75], the MCC is used to assess the balanced class performance.

5.1.3 Results & Insights

	Transductive		Inductive	
Feature Set	Weighted F1	MCC	Weighted F1	MCC
SSx features @ 2km	0.570 ± 0.003	0.215 ± 0.002	0.586 ± 0.012	0.195 ± 0.003
SSx features @ 10km	0.755 ± 0.017	0.360 ± 0.033	0.755 ± 0.002	0.347 ± 0.002
SSx features @ 100km	0.776 ± 0.020	0.418 ± 0.030	0.788 ± 0.004	0.400 ± 0.004
Choice: 2, 10, 100km	0.679 ± 0.010	0.255 ± 0.010	0.676 ± 0.011	0.227 ± 0.010
Integration: 2, 10, 100km	0.163 ± 0.085	0.042 ± 0.005	0.328 ± 0.015	0.071 ± 0.003
Node Count: 2, 10, 100km	0.118 ± 0.071	0.022 ± 0.001	0.171 ± 0.031	0.021 ± 0.003
All SSx features	$\mathbf{0.792} \pm 0.003$	$\mathbf{0.453} \pm 0.014$	0.787 ± 0.007	0.410 ± 0.008
Coordinate Features	0.570 ± 0.051	0.082 ± 0.014	0.496 ± 0.373	0.017 ± 0.026
with SSx features	0.788 ± 0.008	0.450 ± 0.010	0.788 ± 0.010	0.410 ± 0.011
Local Degree Profile	0.429 ± 0.020	0.071 ± 0.005	0.426 ± 0.037	0.066 ± 0.009
with SSx features	0.789 ± 0.015	$\mathbf{0.454} \pm 0.014$	$\mathbf{0.792} \pm 0.005$	$\mathbf{0.414} \pm 0.006$
Naïve baseline	0.744	0	0.774	0

Table 5.2: **[Road-type Classification]** Feature set experiments with a 2-layer GAIN model ($p = 0.05$). "All SSx features" refers to the first 17 features in Table A.1. Baseline feature sets fail to outperform the naïve baseline, which predicts all roads to be `minor`. Using the space syntax measures instead allows the GNN to fit to the imbalanced data with a MCC of 0.453, while outperforming the naïve classifier by a margin of 6.5% on the Weighted F1. Performance remains similar even in the inductive setting.

Space syntax measures enhance prediction accuracy

Based on the results in Table 5.2, with space syntax, our model outperforms the naïve classifier by 6.5% on the distribution-skewed metric of Weighted F1, while having a significantly higher MCC and thus balanced class performance. By adding the space syntax measures to the baseline coordinate features, our model improves by a margin of **39.0%** (transductive) and **58.7%** (inductive) on the weighted F1 score.

The best performance is achieved with *all* space syntax features, with no significant difference between the transductive and inductive settings. This is different from the link prediction task, where adding more features did not improve performance. Although choice and integration *alone* do not achieve the same prediction accuracy as the full set of features, choice appears to be significantly more predictive of road-types than integration.

But performance was expected as choice measures through-movement potential, and major roads typically have higher choice compared to minor roads. Moreover, space syntax features computed at the largest radius (100 km) are the most predictive features, with an inductive MCC of 0.4 when considered alone. Since these features measure large-scale accessibility, they are likely to be more predictive of road class hierarchies that are defined at the same scale according to the definitions in Table 5.1.

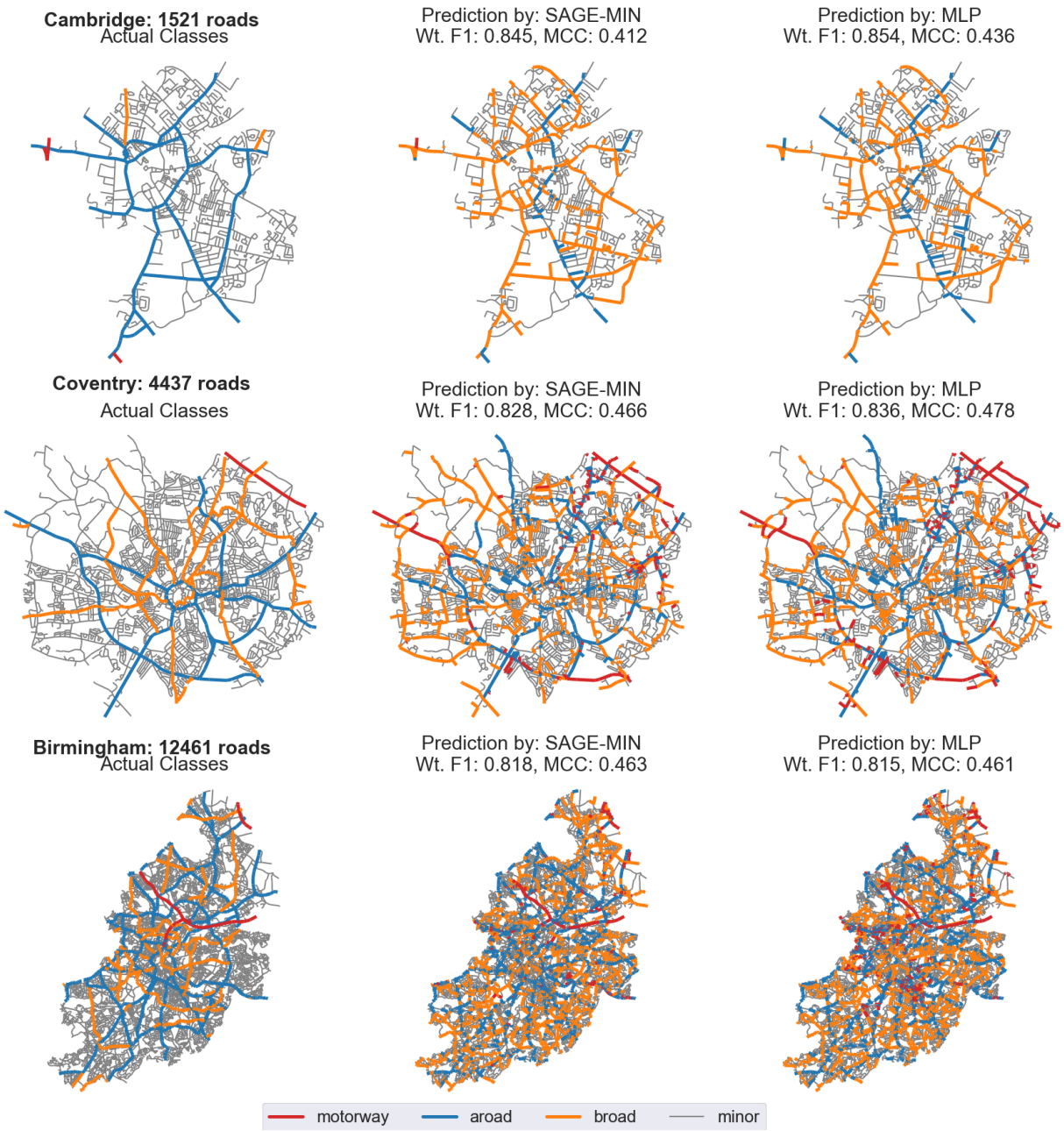


Figure 5.2: **[Road-type Classification]** Map visualization of road-type labels and their predicted values by a 3-layer MLP and a 2-layer GraphSAGE-MIN, on three local authority graphs of increasing scale. In general, the models predict fairly similar patterns, both making errors such as overestimating the number of B roads. The predictions are rather sporadic in places, with adjacent roads along the same sightline having different predicted classes.

			Transductive		Inductive	
Layer Type	P. count	Time(s)/epoch	Weighted F1	MCC	Weighted F1	MCC
2-Layer MLP	344	4.503	0.806	0.478	0.776	0.416
3-Layer MLP	804	4.424	**0.807**	0.484	0.771	0.411
4-Layer MLP	1264	4.465	**0.807**	**0.488**	0.774	0.412
GCN	304	4.428	0.790	0.438	0.759	0.385
GAT	352	5.800	0.785	0.449	0.768	0.401
SAGE-MIN	584	5.418	0.806	0.479	0.779	0.419
SAGE-MEAN	584	6.346	0.805	0.478	0.779	0.421
SAGE-MAX	584	5.932	**0.808**	0.481	**0.782**	**0.424**
SAGE-SUM	584	4.359	0.805	0.477	0.778	0.418
GIN	792	6.075	0.788	0.445	0.764	0.396
GAIN	1280	8.704	0.805	0.480	0.776	0.413

Table 5.3: **[Road-type Classification]** Performance achieved by various GNN layer types taking all space syntax features as input, with a margin of error less than 0.005 ($p = 0.05$). The GraphSAGE-MAX performs slightly better than MLPs in the inductive setting. Models are trained for 100 epochs with a scheduled learning rate that decreased from 0.01. The dimensionality is fixed to 20 for all models, with only two layers for GNNs as increasing the number of layers results in similar or worse performance. The Neighbor Sampler is used for these experiments.

GNNs do not outperform MLPs

As shown in Table 5.3, the performance achieved with MLPs is similar to that of the GNNs when space syntax measures are provided as input. This may be due to the volatile homophily shown in the road-types, where roads that are adjacent, yet perpendicular, may belong to different classes. For such data distributions, the feature smoothing effect caused by GNN aggregation is less compatible with such drastic differences within the neighborhood. MLPs, on the other hand, avoid such issues.

Since choice, integration, and node counts are already predictive of the road type, it is likely that additional information about neighbor node features gained through 2-hop propagation does not aid in prediction. In addition, the GAT and GIN performs worse than the GCN even though they have more expressive power, which may indicate some undesirable bias introduced by more complex propagation.

Out of all the layer types, the GraphSAGE models and especially the MAX variant, manage to be on par with the MLPs, even slightly outperforming them in the inductive setting. The pattern of predictions also differs slightly, as shown by the maps in Figure 5.2. Furthermore, the maps reveal that the predicted road classes tend to be discontinuous, with different classes along the same axial sight-line. Since the road segment representation divides roads at intersections, the models do not take the continuity of roads into account, leading to such sporadic predictions.

5.1.4 Data Loaders: Scaling GNNs

One challenge in applying GNNs directly to larger graphs is scale, as the memory cost of the feature matrix is $\mathcal{O}(n)$ while that of the adjacency matrix is $\mathcal{O}(n^2)$ with n total nodes.

However, space syntax analyses of nation-wide road networks may involve large graphs of over a million nodes [32]. For prediction tasks where there may be multiple floating point features per road segment, the feature matrix can become too large to fit into GPUs with limited memory.

To resolve such space constraints, methods have been developed to sample mini-batches of graph data, which can then be input separately to perform SGD. Figure 5.3 illustrates two of these methods: Neighbor Sampler and Cluster-GCN.

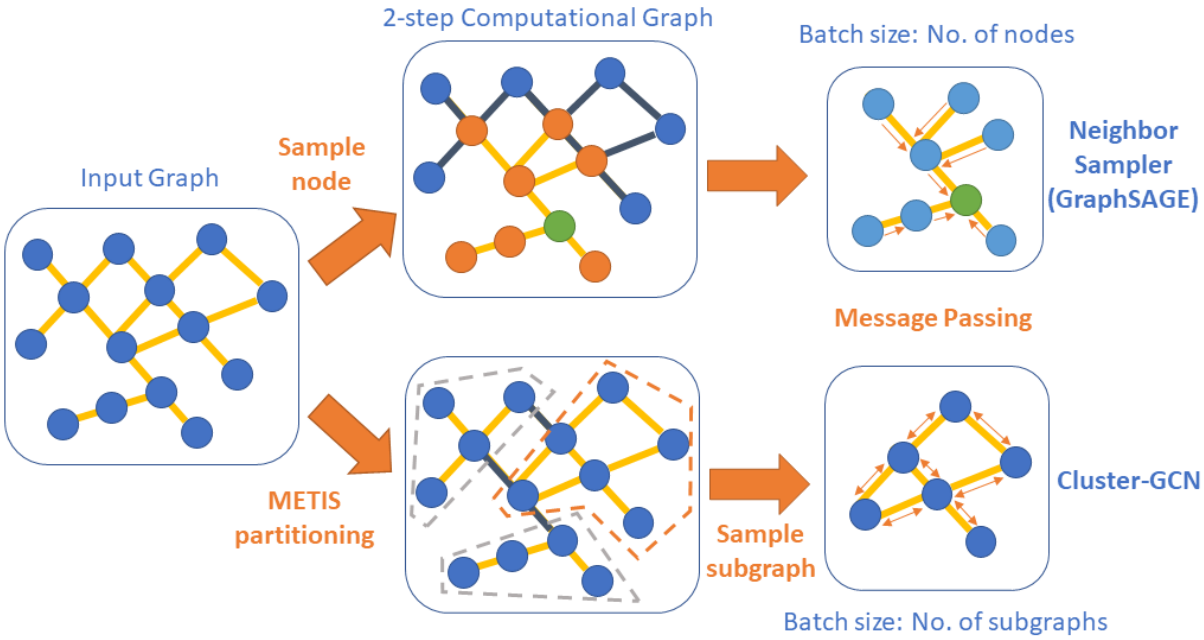


Figure 5.3: Diagram comparing how the GraphSAGE Neighbor Sampler and the Cluster-GCN derive mini-batches. Note that Neighbor Sampler produces computational graphs for the sampled node, and not induced subgraphs like in Cluster-GCN. As such, it only generates a single node embedding per sample, while Cluster-GCN generates embeddings for every node in the sampled subgraph. This does not mean that the Neighbor Sampler is less efficient, as multiple nodes can be sampled per batch.

In this section, we evaluate the following approaches[1], implemented as `DataLoader` classes in PyTorch Geometric [13].

- **Cluster-GCN** [11]: This method involves partitioning nodes, and inputting their induced subgraphs in mini-batches for SGD. Since GNNs perform node embedding by passing information through edges, ideal partitions should retain as much edge connectivity from the original graph as possible. Such partitions can be computed efficiently with modularity-based algorithms such as METIS [70]. Moreover, several subgraphs can then be used in a single mini-batch to reduce variance between steps of SGD. The number of partitions and the number of subgraphs per batch are hyperparameters that are varied in our experiments.
- **Neighbor Sampler (GraphSAGE)** [12]: In addition to the layer type, Hamilton et al. also proposed a method of sampling nodes and their k-hop neighborhoods,

[1]To the best of our knowledge, there has yet to be work evaluating the effect of these sampling algorithms for road network representation learning.

where k is picked to match the number of GNN layers. The neighborhoods correspond to the computational graph for each node, and several can batched together for SGD. For GNNs with many layers, the number of sampled neighbors per hop should be limited to avoid an exponentially growing graph (neighborhood explosion). However, since our model only has 2 layers, all possible neighbors are sampled. We evaluate the Neighbor Sampler with varying batch sizes, i.e. nodes per batch.

- **GraphSAINT Random Walk** [76]: Instead of considering computational graphs like GraphSAGE, this method samples nodes and constructs induced subgraphs based on a fixed-length random walk from the node. Similar to the Cluster-GCN, these subgraphs are directly input as batches, avoiding potential neighborhood explosion issues. A walk length of 2 is used as it corresponds to our 2-layer GNN, while the batch size is varied.

- **Full Batch**: As a baseline, the entire graph is passed through the GNN without partitioning, akin to full batch gradient descent. This method is only feasible with a GPU that has sufficient memory.

Loader	Batch Size	Weighted F1	MCC	Time(s)/epoch
Cluster-GCN, 512 parts	4	0.809 ± 0.001	0.472 ± 0.002	10.459
Cluster-GCN, 512 parts	8	0.808 ± 0.001	0.470 ± 0.002	4.934
Cluster-GCN, 512 parts	16	0.807 ± 0.002	0.468 ± 0.003	2.550
Cluster-GCN, 1024 parts	4	$\mathbf{0.811} \pm 0.002$	$\mathbf{0.476} \pm 0.002$	21.021
Cluster-GCN, 1024 parts	8	0.809 ± 0.001	0.472 ± 0.001	9.808
Cluster-GCN, 1024 parts	16	0.808 ± 0.002	0.471 ± 0.002	4.804
Neighbor Sampler	1024	0.805 ± 0.001	$\mathbf{0.477} \pm 0.001$	12.363
Neighbor Sampler	4096	0.806 ± 0.001	0.476 ± 0.001	6.195
Neighbor Sampler	16384	0.805 ± 0.003	0.472 ± 0.004	4.685
SAINT, 2-hop RW	5000	0.736 ± 0.017	0.364 ± 0.013	8.217
SAINT, 2-hop RW	10000	0.750 ± 0.019	0.373 ± 0.030	8.224
SAINT, 2-hop RW	20000	0.769 ± 0.006	0.406 ± 0.010	8.197
Full Batch	-	0.800 ± 0.002	0.452 ± 0.004	0.543

Table 5.4: **[Road-type Classification]** Transductive performance ($p = 0.05$) and execution times for different data loaders and batch sizes. The model used is a 2-layer GAIN with 20 dimensions, trained for 100 epochs with an initial learning rate of 0.01. Training times are achieved on a system with an Intel i7-7700K CPU and a NVIDIA GeForce GTX 1080 GPU.

Bias and variance from samplers affect performance

From our results in Table 5.4, we see that the Neighbor Sampler and the Cluster-GCN are comparable in performance, with similar training speeds and achieved metrics. Given the low depth of our model and the sparsity of road network graphs, neighborhood sampling does not incur a large computational overhead, and can be preferred to Cluster-GCN.

Since Cluster-GCN eliminates edges that go between partitions, it is known to produce systematically biased gradient signals that can adversely impact performance [11]. How-

ever, our results do not show a significant drop in performance when using Cluster-GCN. This may be due to the large size of our dataset, which offsets the loss of a few edges.

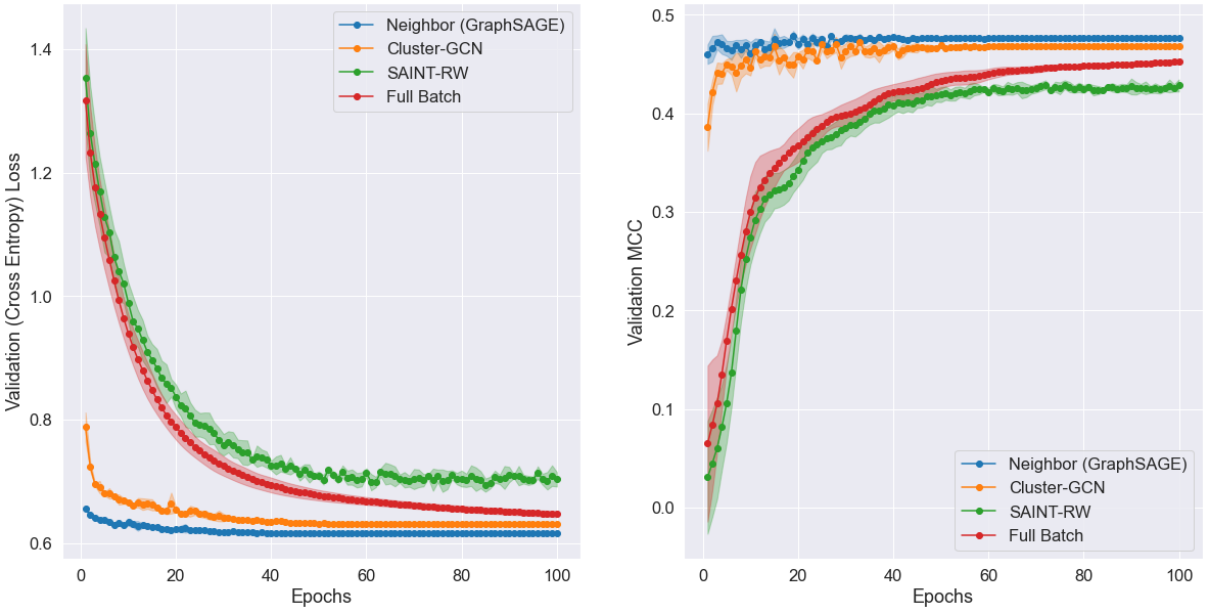


Figure 5.4: **[Road-type Classification]** Average validation cross entropy loss and MCC curves corresponding to different data loaders over 5 iterations, with the margin of error ($p = 0.05$) indicated by the shaded areas. To aid convergence, the learning rate is reduced whenever the validation loss rises.

Full batch training led to slightly poorer performance while also taking longer to converge, as shown by its training curve in Figure 5.4. On the other hand, the methods involving stochastic updates converged faster as the model parameters are updated multiple times per epoch, and achieve slightly better MCC. It may be possible that randomness through sampling and mini-batching leads to better performance. Yet, GraphSAINT's random walk sampler did not see the same level of performance as the other stochastic methods. This may have resulted from inconsistent gradient signals caused by the incomplete random walk neighborhoods.

Cluster-GCN provides faster training with competitive performance

The results also show that Cluster-GCN takes more advantage of larger batch sizes than other loaders, as the time taken sees a directly inverse relationship with batch size. There is, however, a slight trade-off in performance as batch size is increased. Still, if GPU memory is able to accommodate large batches, then using Cluster-GCN can save on the overhead incurred by sampling methods. We hence apply the Cluster-GCN for the next task: classifying traffic accident counts.

5.2 Accident Count Classification

5.2.1 Motivation

Research in space syntax has shown that choice and integration are predictive of vehicular and pedestrian movement [15]. Under the presumption that traffic accidents happen more often on roads with high vehicular movement, this task uses accident counts as a proxy to assess whether GNNs can boost performance when used with space syntax measures in such prediction tasks. In addition, this task serves as an example of *ordinal classification* where the classes are intrinsically ordered. We will see that these distinctions lead to slightly different results compared to the previous task of road-type classification.

5.2.2 Generating Classes From Traffic Accident Data

Accident counts are generated for every road using the Road Safety traffic accident dataset from the Department of Transport [77]. The dataset contains spatial coordinate points where traffic accidents have occurred across Great Britain. Since the SSx OpenMapping dataset uses road data from 2011, accident data prior to that year may be biased towards older road infrastructure. As such, the point data is filtered to the year range of 2011 to 2020.

Next, the points are assigned to their nearest line segment via the spatial join methods available from `geopandas`, a Python library for manipulating spatial datasets [78]. Since the SSx OpenMapping dataset is simplified and does not include unclassified roads, several accident points fall onto non-existent segments. These points are filtered out using a maximum threshold join distance of **15 meters**. This value was chosen by quantile analysis on the join distances between the datasets, which revealed that the large majority (90%) are under the 15-meter threshold.

After this filter, the accident counts for every road are tallied. This results in a highly right-skewed distribution as shown in Figure 5.5, which is expected as accidents are rare events that do not occur on the majority of roads.

Graph Simplification and Feature Aggregation

In order to reduce the imbalance in the target count data, we perform graph simplification, combining line segments into road segments with straight line geometries. To accomplish this, interstitial nodes only used for geometry are removed, i.e. all nodes that are neither intersections nor dead-ends. For a visualization of graphs produced by this process, refer to Figure A.4 in Appendix A.

The accident count of the output segment is set to the sum of the counts of its constituent line segments. Attributes (the space syntax measures) of combined geometries are also aggregated from the components, and we evaluate the various aggregation operations for numerical features in Section 5.2.5. For categorical attributes, we simply take the mode of the constituents.

Following this, the target is categorized into the $K = 5$ quantiles shown in Figure 5.5. This reduces the impact of the right tail outliers, and transforms the task from regression

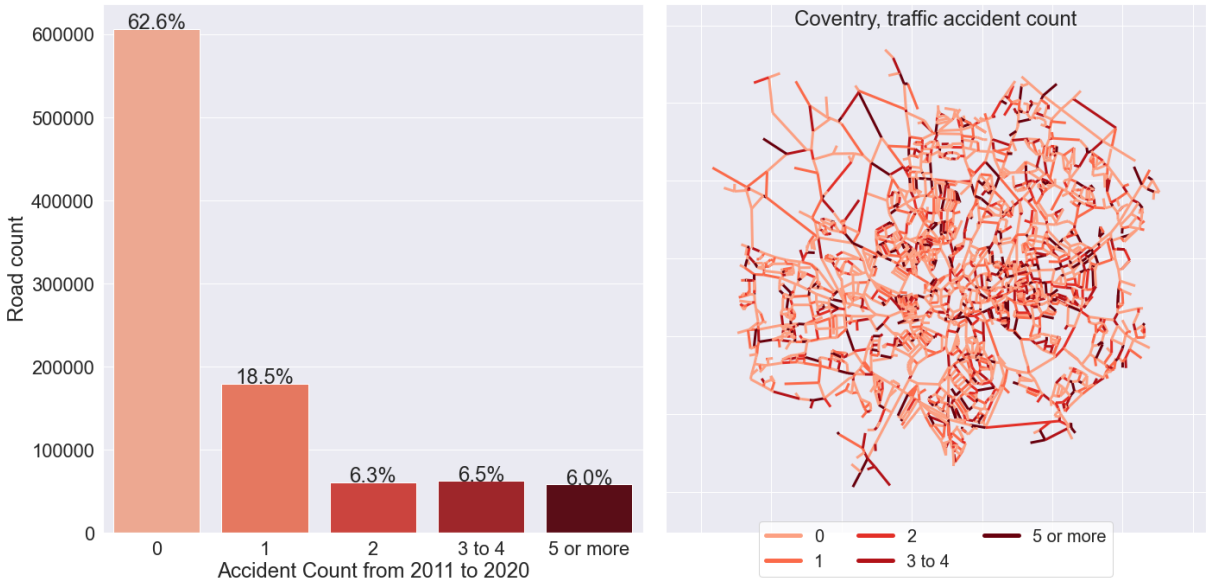


Figure 5.5: **[Accident Count Classification]** As traffic accidents are rare events, the global distribution of counts shown on the left is right-skewed. These quantiles are used to simplify the task from regression into ordinal classification. The map on the right portrays the accident count classes in the road network of Coventry. Unlike the graphs used in the road-type classification task, these graphs are simplified, i.e. line segments are merged into a single road segment with aggregated attributes.

into ordinal classification. The resultant quantiles remain imbalanced, but less so than the road-types in the previous task. The number of quantiles $K = 5$ was chosen for balanced counts between classes, and constitutes the number of classes for this classification task.

5.2.3 Ordinal Classification Using CORAL

We then use CORAL [72], an architecture-agnostic framework for training classifiers of ordinal targets. While the authors demonstrated its effectiveness in convolutional neural network architectures, to the best of our knowledge, this framework has yet to be applied with GNNs. CORAL involves transforming every road i's quantile, y_i, into a series of $K-1$ binary labels that indicate whether the road ranks above the corresponding quantile. Given the ordered quantiles $r_k, k = 1, \ldots, K-1$ (where the 0 accident count class corresponds to r_1), the labels y_i^k are computed as follows:

$$y_i^k = \begin{cases} 1 & \text{if } y_i > r_k \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

For example, a road with 0 accidents (rank r_1) will take the vector $[0, 0, 0, 0]$, while a road with 3 accidents (rank r_4) turns into $[1, 1, 1, 0]$. In our classifier, the CORAL layer (illustrated in Figure 5.6) is added just after the final layer of the GNN[2]. It adds shared weight parameters $\mathbf{w}$, and $K-1$ *independent* bias units b_k that are initialized

[2]Initially, we applied binary level labelling without the CORAL layer, but it led to inconsistent performance, likely due to the model's inability to ensure rank-consistency between predicted probabilities of different quantiles.

to descending values in the range $[0, 1]$ for faster convergence. With sigmoid activation, the predicted probability that a road ranks above quantile r_k can be described as $\hat{P}(y_i > r_k) = \sigma(\mathbf{z_i} \cdot \mathbf{w} + b_k)$, where $\mathbf{z_i}$ is the output embedding from the GNN for road i.

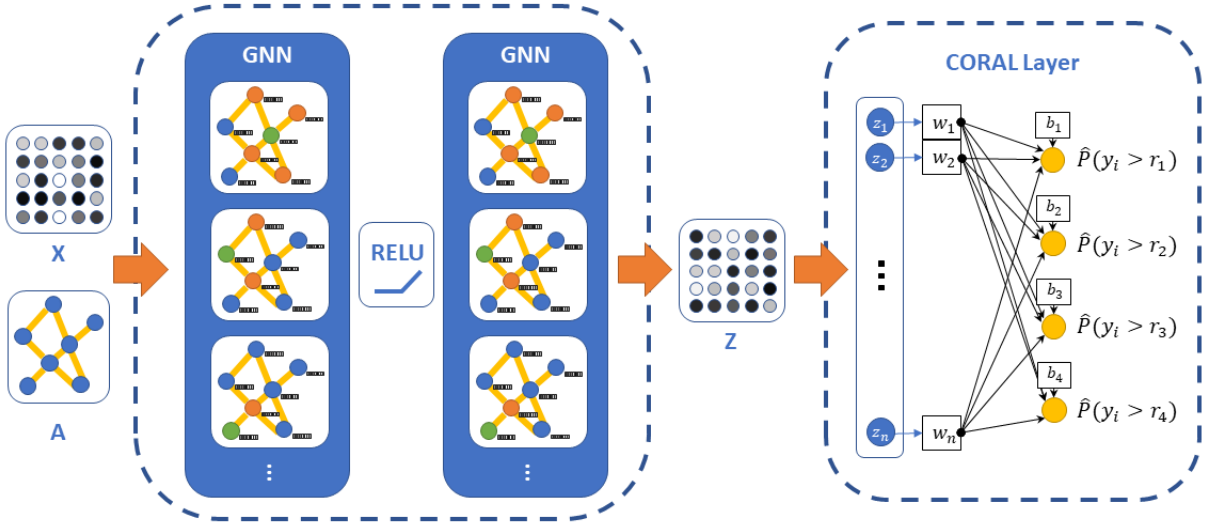


Figure 5.6: **[Accident Count Classification]** GNN model with an added CORAL layer, comprising shared weights $\mathbf{w}$ and $K - 1 = 4$ independent bias terms $\mathbf{b}$. n refers to the number of hidden dimensions. The diagram of the CORAL layer is based on Cao et al. [72: p. 328].

To predict a quantile rank, the output probabilities are thresholded at 0.5 and summed. For more details about this procedure and other aspects of CORAL, including the specialized loss function, please refer to Cao et al. [72].

5.2.4 Evaluation Setup & Metrics

A similar setup to the road-type classification task is employed, with the same training-validation split, transductive-inductive split, and training hyperparameters. However, this task involves different optimization targets and metrics for ordinal classification: the model is trained on the CORAL loss function and evaluated using the **Mean Average Error (MAE)** and **Root Mean Squared Error (RMSE)** metrics. These metrics are used as they produce a larger error if, for example, the model outputs r_1 compared to r_3 for a road with quantile r_4. To turn the labels back into continuous values for these metrics, the quantile's rank is taken as its integer value, i.e. r_1 corresponds to 1, r_2 to 2 and so on.

The CORAL layer adds some overhead to the training process, as the conversion from predicted probabilities to labels is done sequentially per batch. We hence utilize the Cluster-GCN for this task, since neighborhood sampling would incur a higher overhead with its large batch sizes.

	Transductive		Inductive	
Layer Type	MAE	RMSE	MAE	RMSE
SSx features @ 2km	0.623 ± 0.001	1.089 ± 0.002	0.769 ± 0.002	1.248 ± 0.004
SSx features @ 10km	0.584 ± 0.001	1.034 ± 0.002	0.731 ± 0.002	1.184 ± 0.005
SSx features @ 100km	0.617 ± 0.001	1.088 ± 0.002	0.760 ± 0.002	1.259 ± 0.006
Choice: 2, 10, 100km	0.580 ± 0.001	1.029 ± 0.001	0.692 ± 0.002	1.161 ± 0.004
Integration: 2, 10, 100km	0.652 ± 0.003	1.142 ± 0.004	0.826 ± 0.004	1.324 ± 0.007
Node Count: 2, 10, 100km	0.708 ± 0.004	1.253 ± 0.006	0.890 ± 0.004	1.438 ± 0.011
All SSx features	0.522 ± 0.002	0.947 ± 0.003	0.628 ± 0.002	1.070 ± 0.004
Coordinate Features	0.695 ± 0.001	1.221 ± 0.002	0.874 ± 0.002	1.435 ± 0.002
with SSx features	$\mathbf{0.510} \pm 0.002$	0.932 ± 0.002	0.639 ± 0.002	1.084 ± 0.004
Local Degree Profile	0.719 ± 0.001	1.273 ± 0.007	0.896 ± 0.003	1.496 ± 0.005
with SSx features	0.529 ± 0.003	0.957 ± 0.004	0.630 ± 0.005	1.072 ± 0.006
Meridian Class	0.625 ± 0.000	1.095 ± 0.001	0.741 ± 0.003	1.238 ± 0.008
with SSx features	$\mathbf{0.508} \pm 0.002$	$\mathbf{0.926} \pm 0.004$	$\mathbf{0.603} \pm 0.001$	$\mathbf{1.036} \pm 0.002$
Naïve baseline	0.776	1.446	0.978	1.655

Table 5.5: **[Accident Count Classification]** Feature set experiments with a 2-layer GraphSAGE-MAX model ($p = 0.05$). Other parameters remain the same as in the previous task (see Table 5.2), except for the use of Cluster-GCN (512 parts, 16 per batch). For all tested baselines, appending space syntax features lowers the prediction error.

5.2.5 Evaluation Results

As shown by Table 5.5, the best performance is again achieved by adding the full set of space syntax features. This lowers the RMSE achieved by the coordinate feature set by 26.6% in the transductive setting and 28.3% in the inductive setting. The inductive setting incurs a greater performance loss compared to the road-type task, with an 11.9% to 17.6% increase in the RMSE across all the feature sets.

Next, we use the one-hot encoded `meridian_class` from the previous task as an example of *data-derived* input features, and found that it generalizes better than the other baselines. Nevertheless, adding space syntax measures to it still improves performance, lowering the RMSE by 15.4% and 19.5% under the transductive and inductive settings respectively. The combination of space syntax features and the `meridian_class` proves to be the most effective feature combination in the inductive setting.

Choice and 10 km measures are more predictive than other features

In contrast to the 100 km measures for road-type classification, the 10 km measures have a more positive impact on predictive performance. This suggests that the intrinsic spatial pattern of traffic accidents is smaller in scale compared to that of road-types. Choice remains more predictive than integration, a result that corroborates with the role of choice in highlighting the most vital routes in the network [20: p. 77]. These routes presumably experience the highest traffic and thus accident rates in the road network.

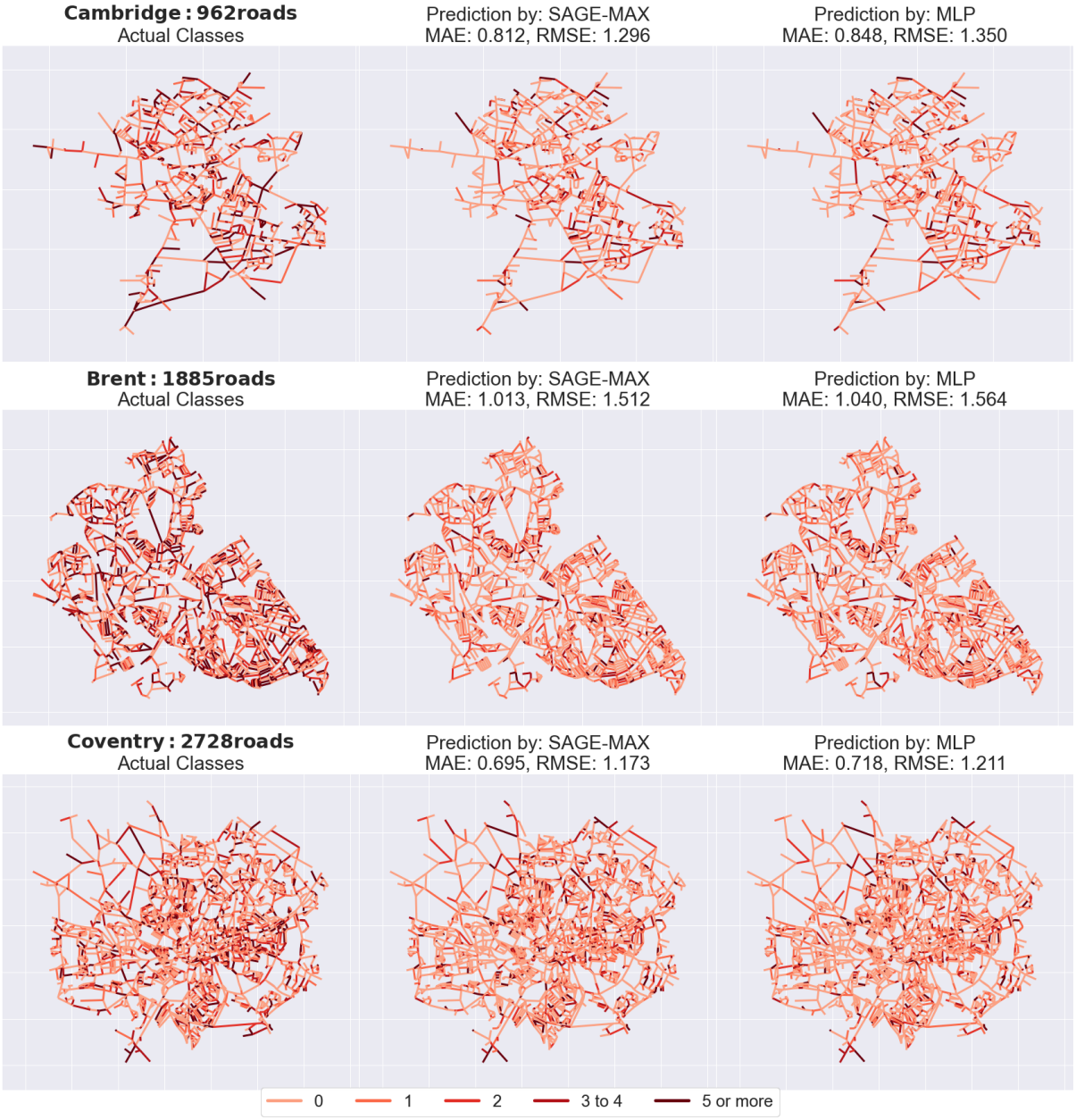


Figure 5.7: **[Accident Count Classification]** Map visualization of accident counts and their predicted values by a 4-layer MLP and a 2-layer GraphSAGE-MAX GNN, on three local authority graphs of increasing scale. The models are generally able to identify the "0" accident roads, but underestimate roads with "5 or more" accidents. Again, the pattern of predictions is similar between the GNN and the MLP.

			Transductive		Inductive	
Layer Type	P. count	Time(s)/epoch	MAE	RMSE	MAE	RMSE
2-Layer MLP	924	1.529	0.521	0.952	0.613	1.057
3-Layer MLP	1384	1.562	0.519	0.947	0.613	1.056
4-Layer MLP	1844	1.591	0.518	0.946	0.613	1.057
GCN	884	1.652	0.558	0.992	0.693	1.164
GAT	964	1.731	0.552	0.984	0.680	1.144
SAGE-MIN	1704	1.582	**0.510**	**0.929**	0.608	1.048
SAGE-MEAN	1704	1.578	**0.506**	**0.924**	0.609	1.050
SAGE-MAX	1704	1.568	**0.508**	**0.926**	**0.603**	**1.037**
SAGE-SUM	1704	1.566	**0.507**	**0.926**	0.613	1.054
GIN	1804	1.616	0.549	0.982	0.712	1.200
GAIN	2724	1.852	0.512	0.935	0.611	1.051
Naïve baseline	—	—	0.776	1.446	0.978	1.655

Table 5.6: **[Accident Count Classification]** Mean performance and parameter counts for various architectures, taking all space syntax features and the one-hot encoded `meridian_class` as input. The average margin of error is below 0.005 ($p = 0.05$). All other parameters remain the same as in the previous task (see Table 5.3), except for the use of Cluster-GCN (512 parts, 16 per batch). All GNN models contain only two layers, as increasing that results in similar or worse performance. Note that the CORAL layer is included in the parameter count.

GraphSAGE models remain competitive with MLPs

Again, the GraphSAGE layer types outperform the others, with the MAX variant showing the best performance overall. It may be possible that the separate weight matrix for self loops in the GraphSAGE operator enables the model to weight the road's own features more than that of its neighbors during aggregation. This would allow the local features to be more represented in the output node embedding and avoid over-smoothing.

However, the GNN models do not significantly outperform a standard MLP when using space syntax features. In theory, MLPs should generalize to the inductive setting better than the GNNs, as they do not perceive the graph structure and as such avoid biasing towards the training graph. Yet, GraphSAGE remains competitive with MLPs, which aligns with its intended purpose of inductive performance [12].

CORAL integrates well with GNNs

Results in Table 5.6 show that with CORAL, the GNNs perform better than the naïve baseline. Given the difficulty of this task with its imbalanced class distribution, this indicates that the CORAL framework can positively integrate with GNNs. This result is useful for road network representation learning: most classification tasks on roads, such as assigning speed limits, involve predicting ordered variables.

Sum aggregation leads to slightly better performance

Table 5.7 reports the metrics achieved by the model with features processed by varying aggregation functions. We saw a similar analysis for edge feature pooling (see Table 4.3 of

	Transductive		Inductive	
Aggregation	MAE	RMSE	MAE	RMSE
Max	0.509 ± 0.002	0.929 ± 0.003	0.606 ± 0.002	1.041 ± 0.004
Min	0.507 ± 0.003	0.925 ± 0.005	0.604 ± 0.001	1.037 ± 0.003
Sum	0.508 ± 0.001	0.927 ± 0.002	$\mathbf{0.600} \pm 0.002$	$\mathbf{1.028} \pm 0.003$
Mean	0.508 ± 0.003	0.926 ± 0.004	0.604 ± 0.001	1.038 ± 0.002
Median	0.507 ± 0.003	0.926 ± 0.004	0.604 ± 0.002	1.038 ± 0.004
Std	0.519 ± 0.002	0.944 ± 0.004	0.614 ± 0.002	1.044 ± 0.005

Table 5.7: **[Accident Count Classification]** Varying aggregation function for simplifying line segments into road segments, using the SAGE-MAX model and the best performing feature set (space syntax & `meridian_class`). These results show that taking the sum results in about 1% better inductive performance. This choice of aggregation is only applied to the space syntax measures.

Chapter 4); however, here we have a separate context of combining line segment features into road segment features. In this case, no significant difference was found between aggregation functions, although the sum is very slightly better in the inductive setting. In comparison to other functions, the sum results in a more diverse set of features, which may help the model to distinguish roads comprising many line segments.

5.3 Discussion & Future Directions

In optimizing our model and pipeline, the multiple hyperparameters involved made covering all possibilities difficult. Several data-dependent assumptions were made during processing, such as the quantile boundaries, the choice of min-max normalization, and the maximum distance for the spatial join between accident points and road geometries. This was to prioritize optimizing model parameters, but given more time we would investigate how these assumptions impact performance.

Since space syntax features by themselves can be already predictive of road-based attributes, we found that GNNs do not consistently outperform the MLP baseline that avoids smoothing those precisely measured angular features. This could be due to some degree of volatile homophily in the target variable, as shown in Figure 5.5 where a road with 5 or more accidents can be directly adjacent to a road with no accidents. As such, these features may be better predicted directly without neighborhood aggregation.

Predicting Traffic Data

Instead of class labels, traffic data that is more contiguous over space may be better suited for prediction by GNNs. Unfortunately, such datasets are typically incomplete and can be challenging to obtain. We were unable to find a complete dataset that accounts for the large area covered by our dataset, and thus used accident counts as a proxy.

Still, as a popular research topic, many dedicated architectures have been developed for traffic prediction. An example is the Spatio-Temporal Graph Convolutional Network [79], which incorporates recurrent units from sequence-to-sequence models into GCNs to

model the temporal dimension. It would be interesting to evaluate how well space syntax features could improve such models.

Training Variance and Data Loaders

Our models see high volatility during training as shown in Figure 5.4, which required the use of a learning rate scheduler. In our experiments, the GAIN in particular incurred the greatest variance, likely because it has many parameters that inflate model capacity. In addition, the batch size parameter of the `DataLoader` incurs a trade-off, where increasing batch size lowers variance while increasing bias. The parameter hence requires careful adjustment depending on the chosen sampling method and the model.

Computing SSx Measures for Transfer Learning

Our results revealed that space syntax measures at large radii (100 km) can be highly predictive of road-type classes. However, these measures incur high computation time, making it tricky to transfer the learned model to other road network graphs that lack precomputed space syntax measures. The context of the full road network is required to compute these large-scale measures, which may make computation infeasible if the network is located in sprawling metropolitan regions.

Instead, limiting the metric radii can mitigate these issues. For our case, just using the 10 km measures can be advantageous as they are sufficiently predictive of our chosen target variables, while being more computationally feasible than measures at 100 km.

5.4 Summary

In this chapter, we demonstrated that space syntax measures improve classification of road-types and accident counts. Compared to link prediction, choice appeared to be the most predictive measure, although the combination of all space syntax measures resulted in the best metrics. As space syntax measures are computed with just topological data, they can be used for featureless classification tasks in locations lacking reliable sources of other data.

We showed that the type of aggregation for combining line segment features into road segments does not have a significant impact on performance. To enable training on large road datasets, neighborhood sampling from GraphSAGE works well for road graphs, while Cluster-GCN is an alternative for faster training. The optimal model can vary based on the task, with the MLP being superior in the road-type classification task, but not in accident count classification. Our results also showed that the GraphSAGE-MAX layer type leads to the best performance in both tasks. Finally, we saw that CORAL integrates well with GNNs for ordinal classification.

Since MLPs remain competitive and can perform better than GNNs with space syntax features, not all space syntax-based models may benefit from GNNs. Nevertheless, employing space syntax with MLPs or GNNs can serve as effective baselines for future work in road network representation learning, be it for classification or other tasks.

Chapter 6: Conclusion

In this project, we investigated the potential of improving urban road network analysis by incorporating space syntax into machine learning with GNNs on road networks. We implemented a pipeline for doing so using PyTorch Geometric, and evaluated it on two task types: road link prediction in Chapter 4 and road classification in Chapter 5. The pipeline is supported by `SSx-GNN` (Appendix C), a library containing utilities for data processing, model training and visualization that we developed.

When used as input features, **space syntax measures substantially improved on the prediction accuracy on the road classification task**, under both the transductive and inductive task settings. For link prediction, however, space syntax measures did not improve performance over **quantile-normalized** coordinate features. Our results also showed that exactly which space syntax measures work best is task-dependent: for link prediction, integration provided the best performance, but for the classification tasks, choice was more predictive of the target attributes.

We proposed a **graph autoencoder model** with a 2-layer **GAIN** encoder and a **symmetric DistMult** decoder to solve the task of road link prediction. Our data processing pipeline used **minimum edge pooling** of space syntax features to produce node representations, giving the best link prediction performance. We also introduced the Batched Graph task as a method of optimizing model parameters over multiple graphs.

The unbiased metric of **average precision** was proposed as a measure of spatial homogeneity and network similarity. Our approach of partitioning the nationwide road dataset into local authority graphs enabled correlation analyses of the achieved metrics with city statistics. In exemplifying such analyses, we reported several graph features that correlated with link prediction performance, such as graph size, density, and several socio-economic indicators.

In road classification, our results showed that the **GraphSAGE-MAX** layer type was optimal for both our chosen tasks of road-type classification and accident count classification. We found that using the **Neighborhood Sampler** and **Cluster-GCN** to partition large road datasets for input into GNNs is generally preferable to other methods of mini-batching. We also integrated **CORAL** for ordinal classification with GNNs, and showed that it works effectively for training classifiers. However, depending on the dataset and task, we observed that **MLPs remain competitive with GNNs** when space syntax features as used as input.

Still, such models are useful as baselines for urban data analysts wishing to analyze spatially continuous data. Further work in realistic settings with more comprehensive datasets can go a long way in showing the practicality of our methods. Nevertheless, we believe that our extensive evaluation of the synergy between GNNs and space syntax will be useful for stakeholders in urban network analysis.

6.1 Future Directions

As a pioneering work in the interdisciplinary area between space syntax and road representation learning, there are many promising directions for extending our proposed ideas.

6.1.1 Road Network Generation

As mentioned in Chapter 4, the VGAE can be used to generate sets of possible links from a latent distribution of node embeddings. Beyond VGAEs, there are architectures that generate both nodes and edges [80], which may be able to create realistic designs of small-scale road networks for urban planners. Different from Varoudis et al.'s approach [57] that used convolutional VAEs, such a method would generate graphs instead of images. However, since topological information is required to convert graphs into actual road networks, the difficulty in such a task would lie in determining the precise coordinates of the generated nodes alongside their embeddings.

6.1.2 Graph Classification

In this project, we did not cover graph-level tasks in road network representation learning. *Graph pooling* layers can nonetheless be added to GNNs to enable such tasks. This could be a way for urban data analysts to link local network structure to region-level statistics at different scales. Still, as road networks are structurally very different from the networks that these algorithms were designed for, it would be useful to understand which pooling layers work well for both the primal and dual graph representations.

6.1.3 Analyzing Other Urban Datasets

While the primary dataset used in this paper was the Space Syntax OpenMapping Dataset, external datasets such as accident counts can be obtained from the UK Department of Transport at `data.gov.uk`. The site does directly include traffic count data; however, those datasets typically only feature a small subset of roads chosen for data collection. A study that considers reliable traffic data, perhaps at a smaller scale, would make for a feasible practical evaluation of our methods.

6.2 Final Remarks

As our project makes inroads into the largely unexplored interdisciplinary area of space syntax and machine learning, the main challenges were in developing our specific approaches and processing steps required to make the pipeline work, as well as evaluating the impact of the many parameters involved. While it certainly does not bring benefits in every case, this project provides evidence that space syntax can improve neural models that represent road networks. We hope that our contributions will be of use in advancing the state of the art in road network representation learning, in space syntax as a precedent for how it can be applied with machine learning, and also in the wider context of deriving data-driven insights to inform city planning and development.

Bibliography

1. Sharmin S and Kamruzzaman M. **Meta-analysis of the relationships between space syntax measures and pedestrian movement**. *Transport Reviews* 2018; 38(4):524–550. DOI: 10.1080/01441647.2017.1365101
2. Choubassi R, Dibble JL, and Bazzoni F. **Space syntax as a foundation for a transport development strategy**. In: *Proceedings of the 12th International Space Syntax Symposium*. Vol. 95. Beijing, China, 2019. https://research.systematica.net/wp-content/uploads/2020/11/095_paper-SSS12.pdf
3. Nes A van and Yamu C. **Introduction to Space Syntax in Urban Studies**. Springer, 2021. DOI: 10.1007/978-3-030-59140-3
4. Nes A van. **The Impact of the Ring Roads on the Location Pattern of Shops in Town and City Centres. A Space Syntax Approach**. *Sustainability* 2021; 13(7):3927. DOI: 10.3390/su13073927
5. Jepsen TS, Jensen CS, and Nielsen TD. **Graph Convolutional Networks for Road Networks**. In: *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. New York, NY, USA: ACM, 2019:460–463. DOI: 10.1145/3347146.3359094
6. Gharaee Z, Kowshik S, Stromann O, and Felsberg M. **Graph representation learning for road type classification**. *Pattern Recognition* 2021; 120:108174. DOI: 10.1016/J.PATCOG.2021.108174
7. Zheng J, Gao Z, Ma J, Shen J, and Zhang K. **Deep Graph Convolutional Networks for Accurate Automatic Road Network Selection**. *ISPRS International Journal of Geo-Information* 2021; 10(11):768. DOI: 10.3390/ijgi10110768
8. Xue J, Jiang N, Liang S, Pang Q, Yabe T, Ukkusuri SV, et al. **Quantifying the spatial homogeneity of urban road networks via graph neural networks**. *Nature Machine Intelligence* 2022; 4(3):246–257. DOI: 10.1038/s42256-022-00462-y
9. Chen Y, Li X, Cong G, Bao Z, Long C, Liu Y, et al. **Robust Road Network Representation Learning**. In: *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. Vol. 1. New York, NY, USA: ACM, 2021:211–220. DOI: 10.1145/3459637.3482293
10. Space Syntax Limited. **Space Syntax OpenMapping**. Version 1. Dataset. 2018. https://github.com/spacesyntax/OpenMapping
11. Chiang WL, Liu X, Si S, Li Y, Bengio S, and Hsieh CJ. **Cluster-GCN: An Efficient Algorithm for Training Deep and Large Graph Convolutional Networks**. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. New York, NY, USA: ACM, 2019:257–266. DOI: 10.1145/3292500.3330925

12. Hamilton WL, Ying R, and Leskovec J. **Inductive Representation Learning on Large Graphs**. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017:1025–1035. DOI: 10.5555/3294771.3294869

13. Fey M and Lenssen JE. **Fast Graph Representation Learning with PyTorch Geometric**. Version 3 2019. ArXiV [Preprint]. DOI: 10.48550/arxiv.1903.02428

14. Turner A. **From Axial to Road-Centre Lines: A New Representation for Space Syntax and a New Model of Route Choice for Transport Network Analysis**. *Environment and Planning B: Planning and Design* 2007; 34(3):539–555. DOI: 10.1068/b32067

15. Hillier B and Iida S. **Network and Psychological Effects in Urban Movement**. In: *Spatial Information Theory*. Ed. by Cohn AG and Mark DM. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005:475–490

16. Hillier B, Turner A, Yang T, and Park HT. **Metric and topo-geometric properties of urban street networks: some convergencies, divergencies and new results**. In: *Proceedings, 6th International Space Syntax Symposium*. Vol. 1. 2. Istanbul, Turkey, 2007. http://spacesyntaxistanbul.itu.edu.tr/papers/longpapers/001%20-%20Hillier%20Turner%20Yang%20Park.pdf

17. Özbil A. **Modeling Walking Behavior in Cities Based on Street Network and Land-Use Characteristics: The Case of İstanbul**. *METU Journal of the Faculty of Architecture* 2013; 30(2). DOI: 10.4305/METU.JFA.2013.2.2

18. Hillier B and Hanson J. **The Social Logic of Space**. Cambridge University Press, 1984. DOI: 10.1017/CBO9780511597237

19. Peponis J, Bafna S, and Zhang Z. **The Connectivity of Streets: Reach and Directional Distance**. *Environment and Planning B: Planning and Design* 2008; 35(5):881–901. DOI: 10.1068/b33088

20. Yamu C, Nes A van, and Garau C. **Bill Hillier's Legacy: Space Syntax—A Synopsis of Basic Concepts, Measures, and Empirical Application**. *Sustainability* 2021; 13(6):3394. DOI: 10.3390/su13063394

21. Ratti C. **Space Syntax: Some Inconsistencies**. *Environment and Planning B: Planning and Design* 2004; 31(4):487–499. DOI: 10.1068/b3019

22. Haklay M and Weber P. **OpenStreetMap: User-Generated Street Maps**. *IEEE Pervasive Computing* 2008; 7(4):12–18. DOI: 10.1109/MPRV.2008.80

23. Turner A. **UCL Depthmap 7: From Isovist Analysis to Generic Spatial Network Analysis**. *New developments in space syntax software* 2007 :43–51. http://www.vr.ucl.ac.uk/events/syntaxsoftware07/ndsss07-turner.html

24. Boeing G. **OSMnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks**. *Computers, Environment and Urban Systems* 2017; 65:126–139. DOI: 10.1016/J.COMPENVURBSYS.2017.05.004

25. Ferrando C. **Towards a Machine Learning Framework in Space Syntax**. PhD thesis. Carnegie Mellon University, 2018. DOI: 10.1184/R1/7178417.V1

26. Rashid M. **Space Syntax: A Network-Based Configurational Approach to Studying Urban Morphology**. In: *Modeling and Simulation in Science, Engineering and Technology*. 2019:199–251. DOI: 10.1007/978-3-030-12381-9_10

27. Turner A, Doxa M, O'Sullivan D, and Penn A. **From Isovists to Visibility Graphs: A Methodology for the Analysis of Architectural Space**. *Environment and Planning B: Planning and Design* 2001; 28(1):103–121. DOI: 10.1068/b2684

28. Freeman LC. **A Set of Measures of Centrality Based on Betweenness**. *Sociometry* 1977; 40(1):35. DOI: 10.2307/3033543

29. Hillier B, Yang T, and Turner A. **Normalising least angle choice in Depthmap-and how it opens up new perspectives on the global and local analysis of city space**. *Journal of Space Syntax* 2012; 3(2):155–193

30. Dalton N. **Fractional Configurational Analysis And a solution to the Manhattan problem**. In: *Proceedings of the 3rd International Space Syntax Symposium*. Atlanta, GA, USA, 2001

31. Serra M and Pinho P. **Tackling the structure of very large spatial systems - Space syntax and the analysis of metropolitan form**. *Journal of Space Syntax* 2013; 4(2). http://joss.bartlett.ucl.ac.uk/journal/index.php/joss/article/view/179/pdf

32. Varoudis T, Law S, Karimi K, Hillier B, and Penn A. **Space syntax angular betweenness centrality revisited**. In: *Proceedings of the Ninth International Space Syntax Symposium*. Seoul, Korea, 2013

33. Zhang A, Lipton ZC, Li M, and Smola AJ. **Dive into Deep Learning**. 2021. ArXiV [Preprint]. http://arxiv.org/abs/2106.11342

34. Boeing G. **Street Network Models and Indicators for Every Urban Area in the World**. *Geographical Analysis* 2020. DOI: 10.1111/gean.12281

35. Bellman R. **Dynamic Programming**. *Science* 1966; 153(3731):34–37. DOI: 10.1126/science.153.3731.34

36. Baldi P. **Autoencoders, Unsupervised Learning, and Deep Architectures**. *Journal of Machine Learning Research* 2012; 27:37–50. DOI: 10.5555/3045796.3045801

37. Kingma DP and Welling M. **Auto-Encoding Variational Bayes**. 2013. ArXiv [Preprint]. http://arxiv.org/abs/1312.6114

38. Larsen ABL, Sønderby SK, Larochelle H, and Winther O. **Autoencoding beyond pixels using a learned similarity metric**. *33rd International Conference on Machine Learning, ICML 2016* 2015; 4. http://arxiv.org/abs/1512.09300

39. Bahdanau D, Cho K, and Bengio Y. **Neural Machine Translation by Jointly Learning to Align and Translate**. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings* 2014. DOI: 10.48550/arxiv.1409.0473

40. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. **Attention is All you Need**. In: *Advances in Neural Information Processing Systems*. Ed. by Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, et al. Vol. 30. Curran Associates, Inc., 2017. https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

41. Veličković P, Casanova A, Liò P, Cucurull G, Romero A, and Bengio Y. **Graph Attention Networks**. In: *International Conference on Learning Representations*. ICLR, 2018. DOI: 10.48550/arxiv.1710.10903

42. Boeing G. **Planarity and street network representation in urban form analysis**. *Environment and Planning B: Urban Analytics and City Science* 2020; 47(5):855–869. DOI: 10.1177/2399808318802941

43. Sanchez-Lengeling B, Reif E, Pearce A, and Wiltschko AB. **A Gentle Introduction to Graph Neural Networks**. *Distill* 2021. DOI: 10.23915/distill.00033

44. Zhou J, Cui G, Hu S, Zhang Z, Yang C, Liu Z, et al. **Graph Neural Networks: A Review of Methods and Applications**. *AI Open* 2018; 1:57–81. DOI: 10.1016/j.aiopen.2021.01.001

45. Veličković P. **Theoretical Foundations of Graph Neural Networks**. 2021. https://petar-v.com/talks/GNN-Wednesday.pdf. [Accessed 10th January 2022]

46. Kipf TN and Welling M. **Semi-Supervised Classification with Graph Convolutional Networks**. In: *International Conference on Learning Representations (ICLR)*. Toulon, France, 2017. DOI: 10.48550/arXiv.1609.02907

47. Li Q, Han Z, and Wu XM. **Deeper Insights into Graph Convolutional Networks for Semi-Supervised Learning**. *32nd AAAI Conference on Artificial Intelligence, AAAI 2018* 2018 :3538–3545. DOI: 10.48550/arxiv.1801.07606

48. Schlichtkrull M, Kipf TN, Bloem P, Berg R van den, Titov I, and Welling M. **Modeling Relational Data with Graph Convolutional Networks**. In: *The Semantic Web*. Ed. by Gangemi A, Navigli R, Vidal ME, Hitzler P, Troncy R, Hollink L, et al. Cham: Springer International Publishing, 2018:593–607. DOI: 10.1007/978-3-319-93417-4_38

49. Wang B, Pan X, Li Y, Sheng J, Long J, Lu B, et al. **Road network link prediction model based on subgraph pattern**. *International Journal of Modern Physics C* 2020; 31(6):2050083. DOI: 10.1142/S0129183120500837

50. Xu K, Li C, Tian Y, Sonobe T, Kawarabayashi Ki, and Jegelka S. **Representation Learning on Graphs with Jumping Knowledge Networks**. *35th International Conference on Machine Learning, ICML 2018* 2018; 12:8676–8685. DOI: 10.48550/arxiv.1806.03536

51. Jepsen TS, Jensen CS, and Nielsen TD. **On Network Embedding for Machine Learning on Road Networks: A Case Study on the Danish Road Network**. *Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018* 2019 :3422–3431. DOI: 10.1109/BigData.2018.8622416

52. Cai C and Wang Y. **A simple yet effective baseline for non-attributed graph classification**. 2018. ArXiV [Preprint]. DOI: 10.48550/arxiv.1811.03508

53. Fan C, Zeng L, Ding Y, Chen M, Sun Y, and Liu Z. **Learning to Identify High Betweenness Centrality Nodes from Scratch**. In: *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. New York, NY, USA: ACM, 2019:559–568. DOI: 10.1145/3357384.3357979

54. Maurya SK, Liu X, and Murata T. **Graph Neural Networks for Fast Node Ranking Approximation**. *ACM Transactions on Knowledge Discovery from Data* 2021; 15(5):1–32. DOI: 10.1145/3446217

55. Wang SM and Huang CJ. **Using Space Syntax and Information Visualization for Spatial Behavior Analysis and Simulation**. *International Journal of Advanced Computer Science and Applications* 2019; 10(4):510–521. DOI: 10.14569/IJACSA.2019.0100463

56. Chang MC, Bus P, and Schmitt G. **Feature Extraction and K-means Clustering Approach to Explore Important Features of Urban Identity**. In: *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2017:1139–1144. DOI: 10.1109/ICMLA.2017.00015

57. Varoudis T and Penn A. **Variational Beauty of Space: Machine Intuition and Non-Linear Neural Aggregations**. In: *Proceedings of the 12th International Space Syntax Symposium*. Ed. by Duan J. Beijing, China: 12th International Space Syntax Symposium (12SSS), 2019. http://www.12sssbeijing.com/proceedings/

58. Yabe T, Tsubouchi K, Shimizu T, Sekimoto Y, and Ukkusuri SV. **Unsupervised Translation via Hierarchical Anchoring**. In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. Vol. 20. New York, NY, USA: ACM, 2020:2841–2851. DOI: 10.1145/3394486.3403335

59. Kipf TN and Welling M. **Variational Graph Auto-Encoders**. ArXiV [Preprint]. 2016. DOI: 10.48550/arXiv.1611.07308

60. Department for Communities and Local Government. **English Indices of Deprivation 2015 - Summaries at Local Authority Level**. Version 1. Dataset. 2015. https://data.gov.uk/dataset/1014339c-de8f-43c6-955e-d45d0a96afb1/english-indices-of-deprivation-2015-summaries-at-local-authority-level

61. Office of National Statistics. **Regional gross value added (balanced) by local authority in the UK**. Version 1. Dataset. 2017. https://www.ons.gov.uk/economy/grossvalueaddedgva/datasets/regionalgrossvalueaddedbalancedbylocalauthorityintheuk

62. Law S, Penn A, Karimi K, and Shen Y. **The Economic Value of Spatial Network Accessibility for UK Cities**. In: *Proceedings of the 11th Space Syntax Symposium*. Lisbon, Portugal, 2017. http://spacesyntax.com/wp-content/uploads/2017/09/The-Economic-Value-of-Spatial-Network-Accessibility-for-UK-Cities.pdf

63. Hagberg AA, Schult DA, and Swart PJ. **Exploring Network Structure, Dynamics, and Function using NetworkX**. In: *Proceedings of the 7th Python in Science Conference*. Ed. by Varoquaux G, Vaught T, and Millman J. Pasadena, CA, USA, 2008:11–15. https://conference.scipy.org/proceedings/SciPy2008/paper_2/full_text.pdf

64. Fleischmann M. **momepy: Urban Morphology Measuring Toolkit**. *Journal of Open Source Software* 2019; 4(43):1807. DOI: 10.21105/joss.01807

65. Xu K, Hu W, Leskovec J, and Jegelka S. **How Powerful are Graph Neural Networks?** In: *International Conference on Learning Representations (ICLR)*. 2019. DOI: 10.48550/arXiv.1810.00826

66. Moran PAP. **The interpretation of statistical maps**. *Journal of the Royal Statistical Society. Series B (Methodological)* 1948; 10(2):243–251

67. Van Der Maaten L and Hinton G. **Visualizing Data using t-SNE**. *Journal of Machine Learning Research* 2008; 9(11):2579–2605. https://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf

68. Liben-Nowell D and Kleinberg J. **The link prediction problem for social networks**. In: *Proceedings of the twelfth international conference on Information and knowledge management - CIKM '03*. New York, NY, USA: ACM Press, 2003:556. DOI: 10.1145/956863.956972

69. Zhang M and Chen Y. **Link Prediction Based on Graph Neural Networks**. In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. NIPS'18. Montréal, Canada: Curran Associates Inc., 2018:5171–5181. DOI: 10.48550/arxiv.1802.09691

70. Karypis G and Kumar V. **A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs**. *SIAM Journal on Scientific Computing* 1998; 20(1):359–392. DOI: 10.1137/S1064827595287997

71. You J, Ying R, Ren X, Hamilton WL, and Leskovec J. **GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models**. In: *35th International Conference on Machine Learning, ICML 2018*. Vol. 13. International Machine Learning Society (IMLS), 2018:9072–9081. DOI: 10.48550/arxiv.1802.08773

72. Cao W, Mirjalili V, and Raschka S. **Rank consistent ordinal regression for neural networks with application to age estimation**. *Pattern Recognition Letters* 2020; 140:325–331. DOI: 10.1016/j.patrec.2020.11.008

73. Department for Transport. **Guidance on road classification and the primary route network**. 2012. https://www.gov.uk/government/publications/guidance-on-road-classification-and-the-primary-route-network/guidance-on-road-classification-and-the-primary-route-network. [Accessed 31st May 2022]

74. Errica F, Podda M, Bacciu D, and Micheli A. **A Fair Comparison of Graph Neural Networks for Graph Classification**. In: *Proceedings of the 8th International Conference on Learning Representations (ICLR)*. 2020. DOI: 10.48550/arXiv.1912.09893

75. Jurman G, Riccadonna S, and Furlanello C. **A Comparison of MCC and CEN Error Measures in Multi-Class Prediction**. *PLoS ONE* 2012; 7(8). Ed. by Biondi-Zoccai G:e41882. DOI: 10.1371/journal.pone.0041882

76. Zeng H, Zhou H, Srivastava A, Kannan R, and Prasanna V. **GraphSAINT: Graph Sampling Based Inductive Learning Method**. In: *International Conference on Learning Representations*. 2020. DOI: 10.48550/arxiv.1907.04931

77. Department for Transport. **Road Safety Data - Accidents 1979 - 2020**. Version 1. Dataset. 2021. https://data.gov.uk/dataset/cb7ae6f0-4be6-4935-9277-47e5ce24a11f/road-safety-data

78. Jordahl K, Bossche JV den, Fleischmann M, Wasserman J, McBride J, Gerard J, et al. **geopandas/geopandas: v0.8.1**. Version v0.8.1. 2020. DOI: 10.5281/zenodo.3946761

79. Yu B, Yin H, and Zhu Z. **Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting**. In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*. California: International Joint Conferences on Artificial Intelligence Organization, 2018:3634–3640. DOI: 10.24963/ijcai.2018/505

80. Guo X and Zhao L. **A Systematic Survey on Deep Generative Models for Graph Generation**. 2020. TechRxiv [Preprint]. DOI: 10.36227/techrxiv.12733037.v1

Appendix A: Space Syntax OpenMapping

Feature	Type	Description
`metres`	float64	Length of the line segment

`choice2km`
`choice10km`
`choice100km`
	float64
	float64
	float64
		Angular choice computed by considering surrounding road geometries within the specified radius.

`integration2km`
`integration10km`
`integration100km`
	float64
	float64
	float64
		Angular integration computed in the same way as choice.

`nodecount2km`
`nodecount10km`
`nodecount100km`
	float64
	float64
	float64
		Number of line nodes within the specified radius, which measures density.

`choice2kmlog`
`choice10kmlog`
`choice100kmlog`
	float64
	float64
	float64
		Log-normalized (base 10) choice.

`choice2kmrank`
`choice10kmrank`
`integration2kmrank`
`integration10kmrank`
	float64
	float64
	float64
	float64
		Rank-normalized choice and integration, i.e. the percent rank of the road relative to others within the neighborhood.

`geometry`	LineString	Projected coordinates indicating the geometry of the line segment.
`meridian_class`	string	OS Meridian 2 road classes. Takes one of 4 values: `aroad`, `broad`, `minor`, `motorway`.
`lad11nm`	string	Name of the local authority to which the line segment is assigned.

Table A.1: A summary of the features from the Space Syntax OpenMapping dataset[1]. In this paper, "SSx features" refers to the 17 attributes from `metres` to `integration10kmrank`. An explanation of the measures is provided in Section 2.1.2 of Chapter 2.

[1] Available from https://github.com/spacesyntax/OpenMapping

Dataset	License	Description & Ethical Considerations
Space Syntax OpenMapping [10]	CC BY-SA 4.0 by Space Syntax Limited	A road dataset of Great Britain containing line segment geometries. It includes labels from authoritative sources and computed space syntax measures, and does not constitute personal data.
English indices of deprivation 2011 [60]	Open Government License by the Department for Communities and Local Government	Local authority statistics on deprivation, spanning multiple categories such as education, crime and health. The data from this dataset is derived from public census information about real people. However, as these statistics are aggregated on the level of local authorities, they are no longer personally identifying.
Regional gross value added (balanced) by local authority in the UK [61]	Open Government License by the Office of National Statistics	Gross Value Added statistics aggregated by local authority. This has similar considerations to the above dataset, just in the context of economic status.
Road Safety Data - Accidents 1979 - 2020 [77]	Open Government License by the Department for Transport	Coordinates and details of traffic accidents that have taken place since 1979 on roads in Great Britain. However, they have been duly anonymised of any directly identifying information.

Table A.2: Datasets used in this project, the licenses that permit their use, and any ethical considerations they may engender.

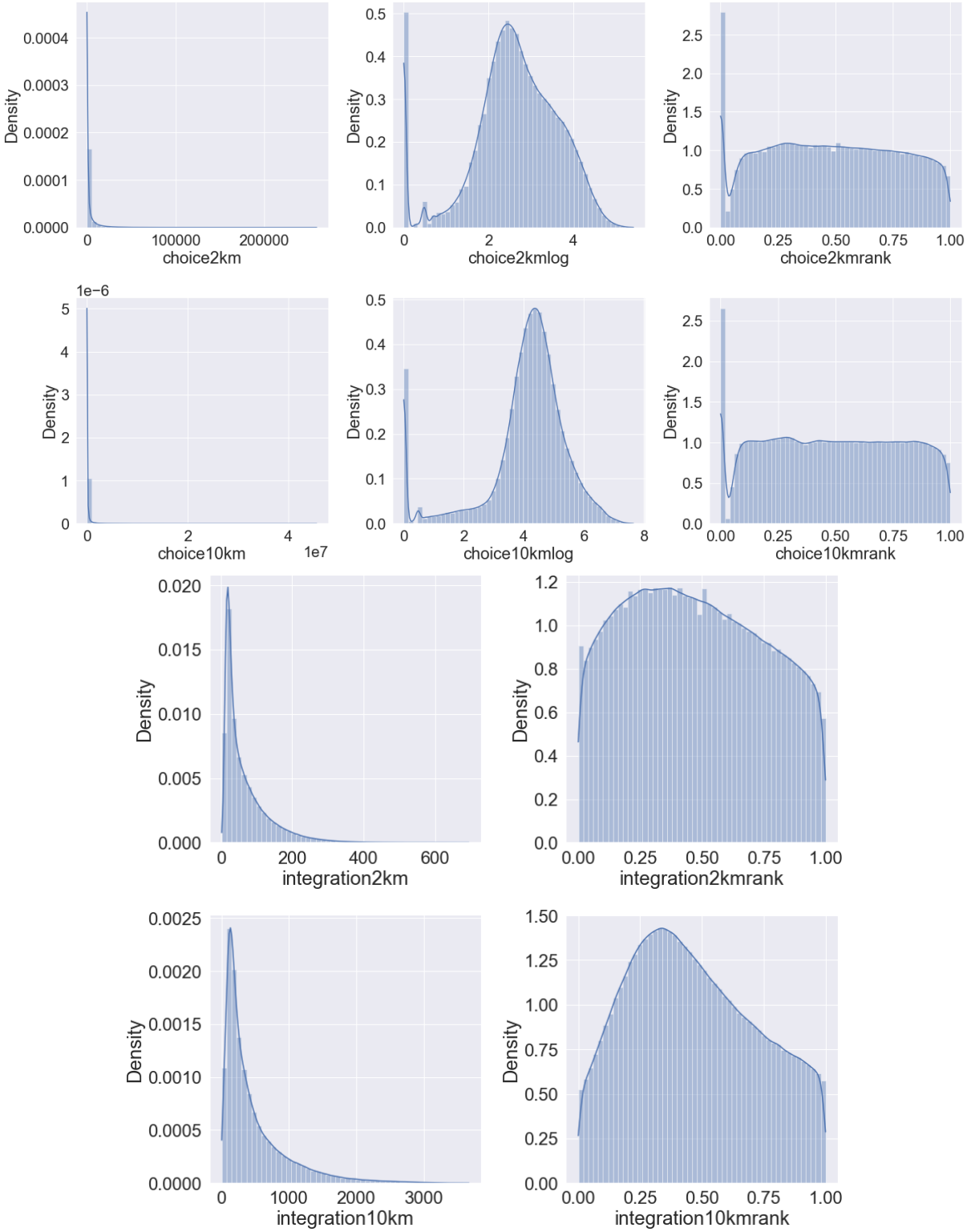


Figure A.1: Density plot showing the effect of the normalization on the full distribution of choice and integration measures before preprocessing. Kernel Density Estimate curves approximating the shape of the distributions are shown for reference.

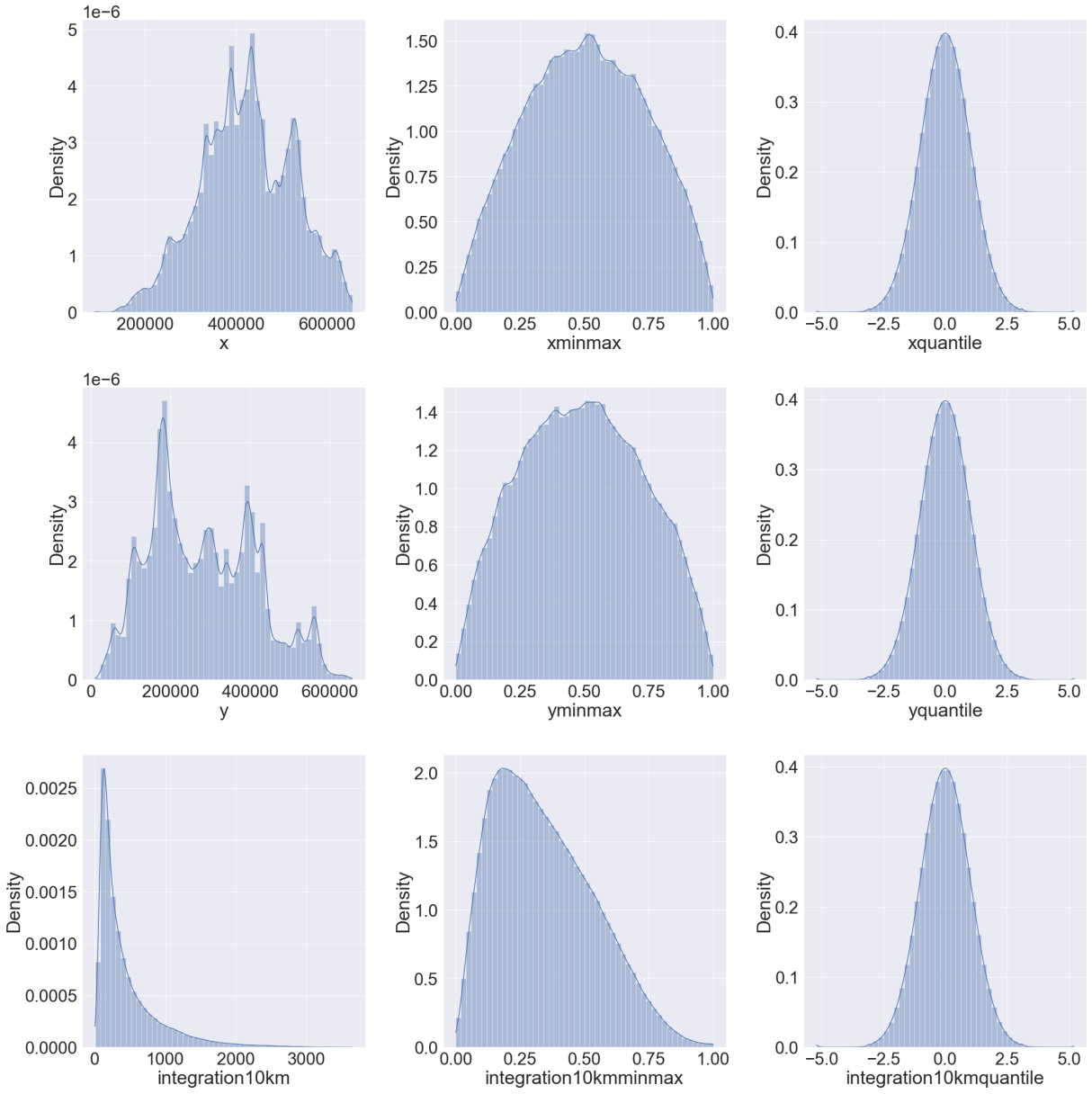


Figure A.2: Density plots comparing the effect of minmax and standard normal quantile normalization on coordinate and integration features.

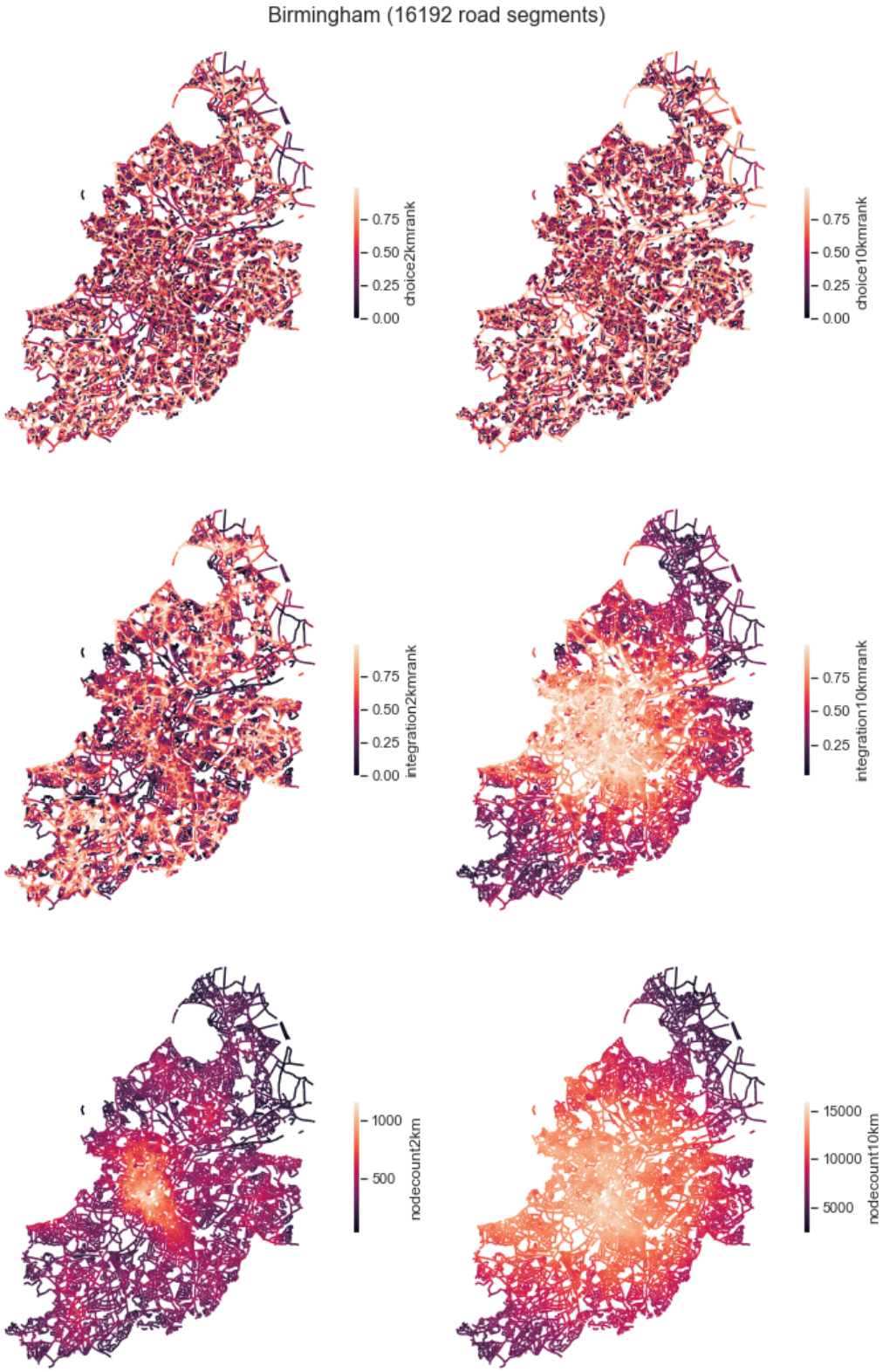


Figure A.3: Choropleth maps for key space syntax measures corresponding to the road network of Birmingham. These measures also include information about the wider network, especially near the graph boundary.

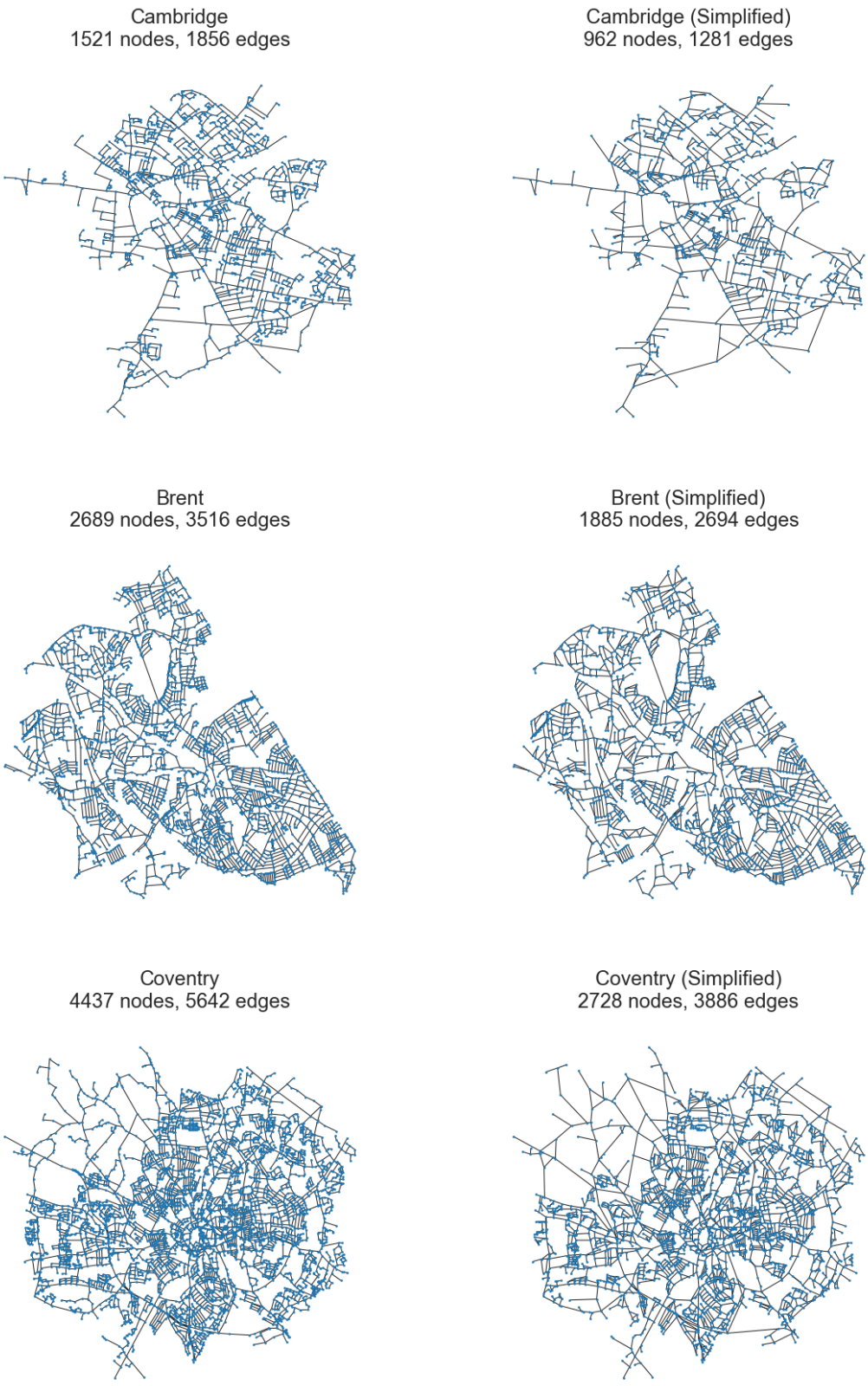


Figure A.4: A sample of primal graphs produced from the SSx OpenMapping dataset, and their corresponding simplified versions. We use the function `simplify_graphs` from `OSMnx` [24] to achieve this.

Appendix B: Additional Results

Feature set	Transductive		Inductive	
	AUC	AP	AUC	AP
SSx features @ 2km	0.928 ± 0.004	0.688 ± 0.013	0.928 ± 0.004	0.687 ± 0.013
SSx features @ 10km	0.953 ± 0.002	0.775 ± 0.008	0.953 ± 0.002	0.775 ± 0.008
SSx features @ 100km	0.952 ± 0.002	0.777 ± 0.005	0.953 ± 0.002	0.779 ± 0.006
Choice: 2, 10, 100km	0.769 ± 0.008	0.420 ± 0.024	0.768 ± 0.007	0.417 ± 0.023
Integration: 2, 10, 100km	0.959 ± 0.001	0.795 ± 0.006	0.961 ± 0.001	0.801 ± 0.005
Node Count: 2, 10, 100km	0.967 ± 0.001	0.835 ± 0.002	0.967 ± 0.001	0.834 ± 0.003
All SSx features	0.969 ± 0.001	0.832 ± 0.007	0.970 ± 0.001	0.833 ± 0.006
Coordinate Features	$\mathbf{0.988} \pm 0.009$	$\mathbf{0.918} \pm 0.053$	$\mathbf{0.988} \pm 0.010$	$\mathbf{0.918} \pm 0.057$
with SSx features	0.978 ± 0.013	0.872 ± 0.058	0.978 ± 0.013	0.870 ± 0.060

Table B.1: **[Batched Graph Task, Quantile Normalization]** Link prediction performance with varying quantile-normalized feature sets, using the GAIN-DistMult autoencoder. Using coordinate features results in the highest metrics, while adding space syntax features does not appear to improve performance. Minmax-normalized features generally follow these trends, but with lower achieved metrics.

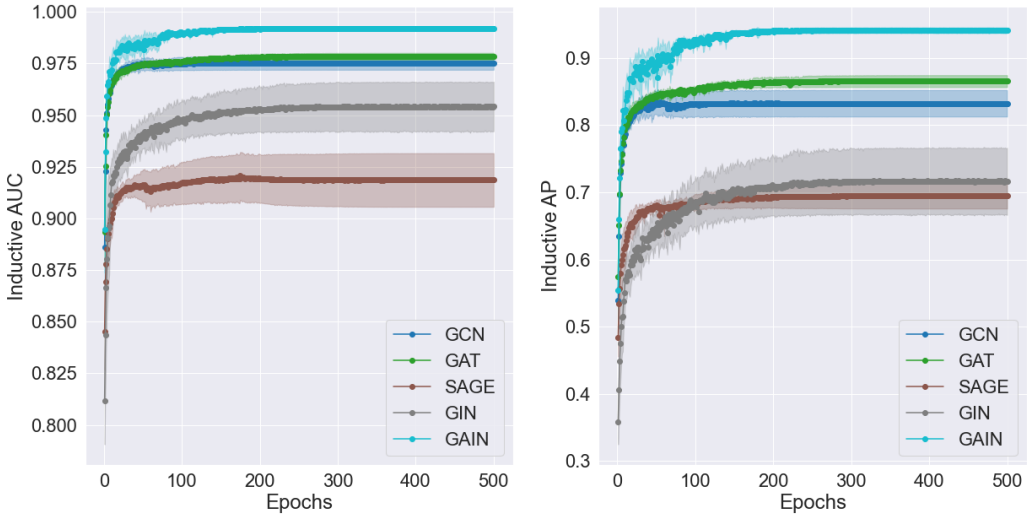


Figure B.1: **[Batched Graph Task, Quantile Normalization]** Link prediction performance with varying encoder layer types, using quantile-normalized coordinate features and the DistMult decoder. The shaded area represents the margin of error ($p = 0.05$). The relative performance of layer types is similar to the experiments that used rank-normalized integration (Table 4.4).

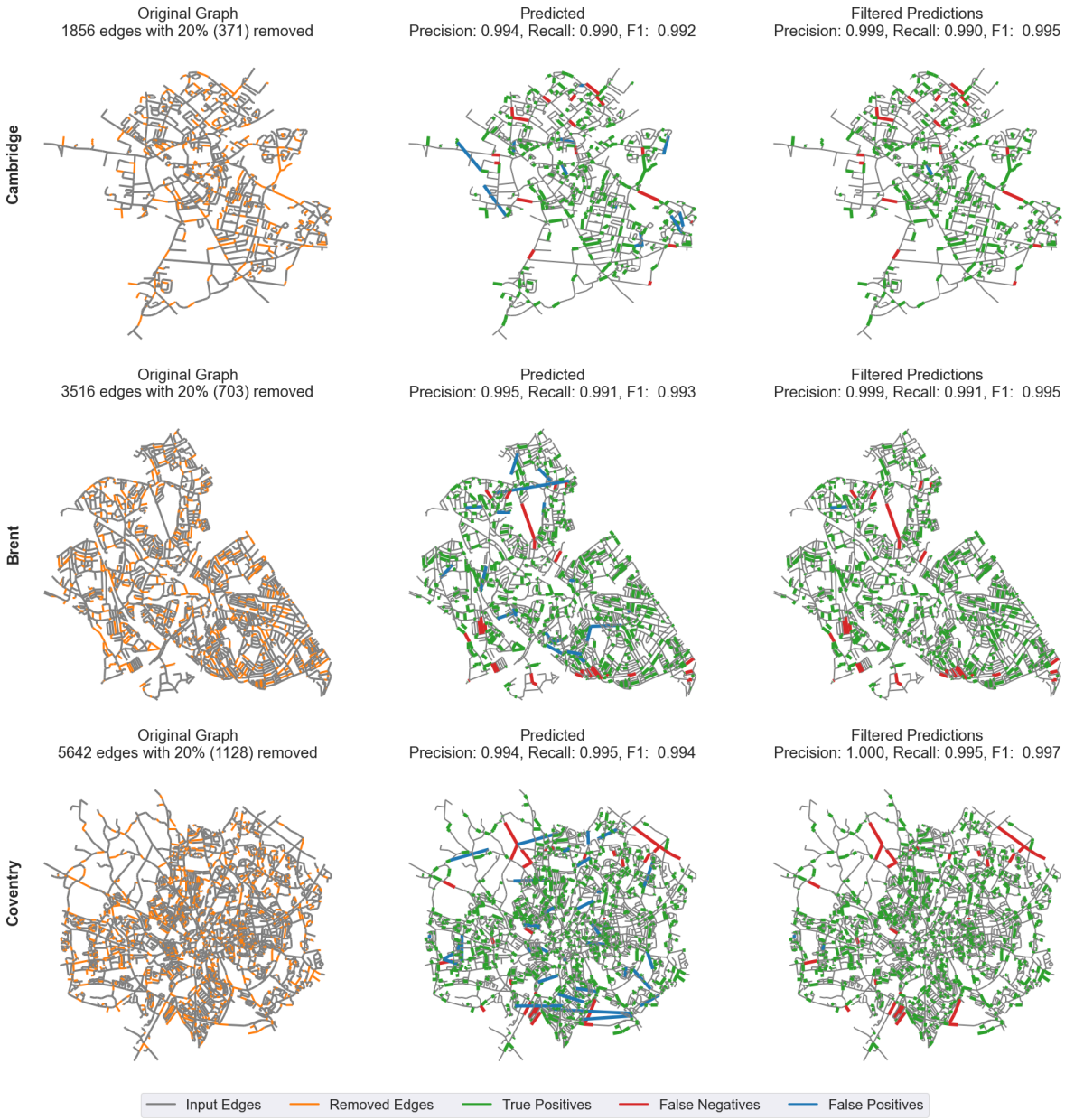


Figure B.2: **[Quantile Normalization]** Map visualization of links predicted by our GAIN-DistMult GAE trained using quantile-normalized coordinate features, on the Cambridge, Brent, and Coventry road networks. Compared to the predictions made by the model that was trained on ranked integration (see Figure 4.3), this model predicts fewer false positives. Negative sampling ratio was set to 1:1 for consistency.

Appendix C: SSx-GNN: Utilities for Data Processing, Model Training & Visualizations

To realize our machine learning pipeline, we developed `SSx-GNN`, a Python library containing utilities for reading and processing spatial data for graph learning. Although the pipeline is highly tailored to the tasks investigated in this project, it can be easily adapted for similar problems involving road network graphs. Concisely, it consists of utilities for reading the datasets described in Appendix A, for converting between various data structures for spatial data, including `GeoDataFrame` tables, `networkx` graphs and PyTorch Geometric `Data` graphs, as well as for training link prediction and road classification models.

The overall pipeline assumes that the input spatial dataset contains precomputed space syntax measures. The Jupyter notebooks in the root directory contain examples of how we call the utility and training functions to run experiments. In particular, `data_analysis_visualizations.ipynb` contains most of the code used to create the figures in this paper.

Table C.1 lists and provides a brief description of the various high level utilities in `SSx-GNN`.

File & Category	Function/Class	Description
`utils/load_geodata.py` Data Loading and Processing Functions	`load_gdf`	Loads a `.gpkg` containing the SSx OpenMapping dataset and returns a `GeoDataFrame` corresponding to a local authority or a bounding box of coordinates.
	`load_graph`	Converts a `GeoDataFrame` to a `networkx` graph, and calls the preprocessing utilities below, before finally converting it to a PyTorch Geometric `Data` object.
	`construct_accident_dataset`	Loads and performs a spatial join between an input road dataset and traffic accident point data, attributing a total accident count for each road.
	`clean_gdf`	Combines line segments from a `GeoDataFrame` into road segments, and returns them in a `networkx` graph while aggregating features.
`models/init_model.py` Initialization of GNN models	`ModGAE`	Graph autoencoder class with flexible encoder GNN layer types and decoder types.
	`CoralGNN`	GNN integrated with CORAL for use in ordinal classification.
	`GAINConv`	Implementation of the GAIN operator as a PyTorch Geometric `MessagePassing` class.
`train.py` Link prediction training code	`process_dataset`	Performs min-max feature normalization, feature selection, random link split with negative sampling, as well as the transductive-inductive split. Returns the split `DataLoader` objects.
	`run`	Executes the Batched Graph task for hyperparameter search.
	`run_single`	Executes the Single Graph Task, and returns the transductive and average inductive metrics.
`prediction.py` Road classification training code	`load_data`	Loads processed graphs, normalizes and selects features, and finally outputs the data in a `DataLoader` object.
	`run`	Trains classifiers and outputs evaluation metrics.

Table C.1: Brief description of utility functions and classes in `SSx-GNN`, which were written using the PyTorch Geometric framework.