INTER-UNIVERSITY MASTER'S DEGREE PROGRAMME

Master's Degree Programme
Artificial Intelligence for Science and Technology

Università degli Studi di Milano

Università degli Studi di Milano - Bicocca

Università degli Studi di Pavia

Towards interpretability of local and global attention in urban spatial graphs

Supervisor:
Prof. Pietro Faccioli
Università degli Studi di Milano - Bicocca

Co-Supervisor:
Dr. Riccardo Gallotti
Dr. Sebastiano Bontorin
Dr. Thomas Louf
Fondazione Bruno Kessler

Candidate:
Stefano Carotti
Matriculation number: 911255

Academic year: 2024/2025

Contents

List of Figures

List of Tables

Abstract

Urban street systems are complex, structured and embedded in space. This thesis aims to build a complete pipeline for applying attention-based graph neural networks to urban spatial networks.

Two supervised tasks are designed to test the approach. The first task constructs a speed proxy for road segments in Italian cities.
The second task aggregates street networks into hexagonal H3 cells and predicts a Green View index using population, points of interest, building count and street view features.
The modelling framework compares three types of architectures: a local Graph Attention Network, a global Performer model and hybrid networks that mix the two within a single residual stream.
All models share a common structure that embeds features with positional information, passes updates through attentional blocks and accumulates them in a residual stream before a final readout.
To understand how these networks operate, the thesis develops two interpretability tools. Attention graphs aggregate attention weights across heads and layers and are extended to hybrid models with a correlation-aware rule. A residual stream lens decomposes each layer updates into local, global and feed-forward contributions and tracks how node representations evolve.

Together, the data pipeline, the suite of models and the interpretability tools form a general template for graph-based learning on city networks. The work shows how to capture both neighborhood structure and long-range context, and how to inspect the resulting models to gain insight into their decision-making processes.

A Damiana,
alle parole che nella tua tesi scrivesti a macchina, alle formule che scrivesti a mano,
all'ispirazione che il tuo percorso ha creato attraverso le generazioni

A Rossella,
alle tesi che scrivesti con me a fianco
all'impegno e alla costanza che hai tramandato

Che questo lavoro mi porti a vivere anche un po' delle vite a cui avete rinunciato,
grazie alle quali sono nato.

Acknowledgements

I am deeply grateful to my supervisors for their guidance and support throughout this academic journey. In particular, I would like to thank Professor Pietro Faccioli and Dr. Riccardo Gallotti, for giving me the opportunity to go to Trento and carry out this project at Fondazione Bruno Kessler.
My sincere thanks go to Sebastiano Bontorin and Thomas Louf, for the continuous support, for the freedom to explore my ideas, and for their constructive feedback that kept me on track.
I also thank all the members in CHUB for the wonderful and stimulating environment you created, and for making me feel immediately included.

I would like to thank Stefano and Damiana for introducing me to the scientific world, thank you for building my first computer together so many years ago, I will always keep this beautiful memory.
I am deeply grateful to my father and mother for their unconditional, continuous, and boundless support. None of this would have been possible without you.
Thank you, Sofia, for the perspective you give me on the world, for your fraternal love, for the hurricane you are. You are the real fire of this family.

To Chiara, I owe not only my gratitude but also my deepest love. You have changed everything, improved everything, until you became everything. Thank you for your support and patience during this past year and a half, thank you for introducing me to Trento, and thank you for making me look forward to discovering what the future holds for us.

I want to thank Checco and Gaia. You have always been and will always be my family. Even from afar, you remain my anchor. You already know better than I could ever tell you, thank you.
To Rebe, Michi, and Marco, you are my constant, my certainty. Thank you for having been there for me, for being here now, and I know I will have you with me always.
To all my friends in Cremona, thank you for everything you do for me, for making our little city feel like the most beautiful place that exists.

To Annalisa, Ernesto, Viola, Pivaz, Michele, Benedetta, Caterina, Ulysse, Diego, Diego, Claudio, and Luca. I remember meeting each of you, but I cannot say when exactly you became among my favorite people in the world. Companions in lectures and exams, in trips and adventures. You did not just make this Master's better, you enriched my life, each of you in your own way, each of you many times over. Thank you for every exam studied together, for every class shared, thank you!
I write these words hoping I will never lose you.

Acronyms

GDL	geometric deep learning
GNN	graph neural network
GAT	graph attention network
CNN	convolutional neural network
RNN	recurrent neural network
RWPE	random-walk positional encoding
PE	positional encoding
H3	Uber hexagonal indexing system
GVI	green view index
GUN	global urban network
MSE	mean squared error
RMSE	root mean square error
MAE	mean absolute error
ITS	intelligent transportation systems
FAVOR+	fast attention via positive orthogonal random features
PCA	principal component analysis
POI	points of interest
MLP	multi-layer perceptron
GLGAT	global–local graph attention network
GraphGPS	hybrid local/global “GraphGPS” architecture
FFN	feed-forward network
GELU	Gaussian error linear unit

Chapter 1
Introduction

Urban spaces form complex systems shaped by many interactions between people, infrastructures and the built environments. From a network science perspective, an urban street system can be represented as a spatial graph: nodes correspond to intersections and edges represent street segments. The geometry of these networks is not incidental. Because traveling along a road comes with a distance dependent cost, the probability of an edge connecting two nodes decays with their spatial distance. Classical planar graph results ensure that such networks remain sparse. Sparsity is beneficial because it can influence the computation complexity of certain models applied. Yet, sparsity alone does not capture the richness of urban phenomena: land use patterns and environmental features often involve interactions between distant urban zones.

Analyzing these systems means searching insight on both mobility and structure: predicting flows, accessibility to services. Assessing the quality of several indicators, like air or greenery, simulating the impact of planning interventions.
These questions are inherently multi scale: local street context matters, but so do long-range dependencies across neighborhoods. This motivates learning methods that respect graph topology while modeling interactions that are able to go beyond strict adjacency.
Traditional urban mobility models relate flows to population masses and distance. They are interpretable and efficient, but lack flexibility and struggle with high dimensional data. Non graph machine learning methods treat each street segment or cell as an independent observation. They can achieve good accuracy yet fail to capture spatial interdependencies. Turning the city into an image and applying convolutional neural networks (CNNs) can help, but discards the network structural information.
These limitations have spurred interest in geometric deep learning (GDL) [1], which generalizes neural networks to non Euclidean domains while respecting symmetries. GDL provides a natural framework for urban street systems because it allows signals defined on nodes or edges to be processed in ways that respect topology and structural invariances.

Two complementary modelling paradigms are considered in this thesis. Local message passing propagate information over the graph via neighborhoods and excel at capturing short range structure. Global attention models learn relations from data that can connect distant nodes and capture long range effects.

These views are two ends of a continuum, and they can also be equivalent when considering complete graphs [2], this connection has profound implications: techniques developed for one an often be transferred to the other. This work objective is to leverage this complementarity by studying architectures that combine local and global components, and by developing interpretability tools that apply uniformly across them.

Purely local models may miss interactions between distant parts of a city, whereas purely global attention can learn spurious connections and is computationally costly. Combining these two is an attempt of taking the strengths of both. This thesis therefore contributes by implementing and evaluating two different hybrids models on city scale graphs and compares them against local only and global only baselines.

As neural networks grow more complex, interpretability becomes indispensable. The Attention graph [3] purpose is to create a directed graph summarizing how information flows among nodes. Crucially, the pipeline applies to both local and global attention and so to hybrids as well. A second tool, Residual stream visualization, inspired by recent transformer-circuit analyses [4], reconstructs the "story" of nodes through the network. Together, the Attention Graph and Residual-stream visualization are meant to form a complementary interpretability framework: the former reveals which relations the model emphasizes, the latter shows how those relations change representations. This unified framework is central to the thesis study and enables comparisons across local, global, and hybrid models for urban networks.

The thesis is guided by three research questions:

- *Can graph based neural networks applied to urban networks improve predictive accuracy compared to non graph models?*
  Addressing this requires benchmarks. The thesis considers two tasks: edge level travel speed prediction on primal street graphs and node level green view index (GVI) prediction on H3 aggregated graphs.
- *Can graph models generalize better to cities and conditions outside their training data?*
  Urban networks vary widely across cities, a model trained on one city probably does not directly transfer to another. To test inductive generalization, the edge level dataset includes seven Italian cities, with two held out entirely for testing. The node-level dataset comprises five European capitals, with one city left out for evaluation. The thesis examines how each model performs when predicting traffic speeds or GVI for unseen cities.
- *Can we obtain more understandable explanations of model predictions by analyzing information flows and the embeddings dynamics?* Attention Graphs and Residual stream visualization are applied to all models. The research explores whether the Attention Graph reveals meaningful long range links, whether global attention tends to focus on hubs or peripheral areas, and how local versus global contributions drive predictions. It also tests whether residual trajectories show clear separation between

nodes associated with high and low target values and whether certain layers or components are primarily responsible.

The thesis makes two main contributions:

1. Hybrid architectures for urban street networks:
   Building on GAT and Performer models, the thesis proposes two hybrid architectures that integrate local message passing with global attention through a shared residual stream. These models are, to the best of our knowledge, among the first to apply hybrid local–global attention to urban networks. They incorporate positional encodings and are designed for compatibility with interpretability tools. Ablations compare hybrid variants to pure GAT and pure Performer, highlighting cases where the combination bring benefits.
2. A unified interpretability framework:
   Adapting the Attention graph framework from [3], the thesis develops a correlation aware aggregation procedure to handle cases where attention heads exhibit negative or zero correlation. The thesis also proposes a hybrid Attention Graph by fusing local and global attention within each layer and composing them across layers. In parallel, the thesis introduces an original residual-stream visualization tool.

The remainder of the thesis is organized as follows:

- Chapter 2: State of the Art reviews the main topics of the thesis and surveys existing related models.
- Chapter 3: Tools, Framework and Implementation formalizes the two prediction problems and details data collection and preprocessing pipelines. The chapter then presents models architectures and outlines the interpretability framework.
- Chapter 4: Results. Presents quantitative comparisons on both tasks, including ablations and generalization tests, then interpretability results on the second task.
- Chapter 5: Conclusions summarizes the main findings, discusses limitations and outlines future directions.

Chapter 2
State of the art

This chapter provides the theoretical background and critical literature review for the thesis, in particular it covers the main concepts and existing related models, also regarding urban street networks. After a review on spatial networks and typical tasks in spatial network analysis (Section 2.1), the chapter introduces geometric deep learning (Section 2.2), then examinate the development of graph neural networks, graph-based attentional neural networks and hybrid models that combine local and global attentions. The chapter concludes with a brief overview of recent efforts to interpret how these models work (Section 2.2.8).

2.1 Spatial Networks

Since this thesis models urban street systems as spatial networks, the section begins by clarifying what a (spatial) network is and by going through key works that precede this analysis.

2.1.1 Why networks

Complex systems can be informally defined as a large number of components which interact with each other.
The components can be seen as nodes which can be anything: people, organizations, biological cells.
Two nodes interacting means two linked nodes that could represent, for example, two people that know each other, two organizations that exchange goods, two neurons that share a synapses passing signals.
What makes these systems complex is that it is impossible to understand or predict their overall behavior by looking into the behavior of individual nodes or links [5]. Once these systems are defined through nodes and links then it is clear why they are often described in the from of networks (or graphs) adopting nodes and edges terminology from graph theory.
It turns out we can often model real world situations in terms of networks, and it often happens in these situations that nodes and edges are embedded in space [6].

2.1.2 Key definitions

In general, *spatial networks* are defined as networks for which the nodes are located in a space equipped with a metric.

An immediately important consequence of space is that there is a cost associated to the length of edges. This definition implies that the probability of finding a link between two nodes will decrease with the distance, but this associated cost has other major effects on the topological structure [6, Sec I.A].
Often, in practical applications, the space is the two-dimensional space and the metric is the euclidean distance.
Transportation and mobility networks, Internet, power grids, social networks, neural networks are all cases where space is relevant and purely mathematical topology alone does not contain all the information. With this definition of a spatial network links are not necessarily embedded in space: social networks for example connect individuals through friendship relations [6, Sec II.A.1].

Another important definition is the one of planarity: a *planar graph* is one that can be drawn in the plane so that its edges do not intersect. This has important implications, but spatial networks are not assured to be planar, for instance airline networks are not, while road, rail and other transportation network are both spatial and planar networks. As more extensively shown by [6, Sec II.A.2], the Euler's formula is a fundamental result for planar graphs.
From Euler's results it is possible to obtain:

$$E \leq 3N - 6 \tag{1}$$

where E is the number of edges and N the number of nodes.

This means that planar (therefore also street) networks are always sparse with:

$$\langle k \rangle \leq 6 \tag{2}$$

where k is the *degree* of a node which is its number of connections.
These simple results are important because the sparsity of planar graphs will let graph neural networks operate in near linear time in the used framework (see Equation 11).

In order to describe a graph with N nodes it is possible to use its $N \times N$ *adjacency matrix* $\mathbf{A}$, defined as:

$$\mathbf{A}_{ij} = \begin{cases} 1 \text{ if } i \text{ and } j \text{ are connected} \\ 0 \text{ otherwise} \end{cases} \tag{3}$$

If the graph is undirected $\mathbf{A}$ is symmetric, which is not the case for street networks, while it is for a grid representation of them, a similar representation will also be implemented in this thesis .
The adjacency matrix captures the connectivity of the graph but not the space informations for spatial graphs.
An important metric, also used for traffic modelling in this thesis is *betweenness centrality* [7]:

$$g(i) = \sum_{s \neq t} \frac{\sigma_{st}(i)}{\sigma_{st}} \tag{4}$$

σ_{st} is the number of shortest paths from s to t, $\sigma_{st}(i)$ is the number of shortest paths from s to t through the node i. $g(i)$ represents the importance of the node i in the flow of information in the network. This definition can immediately be extended to edges by substituting the node i with an edge e.

2.1.3 Empirical findings and models

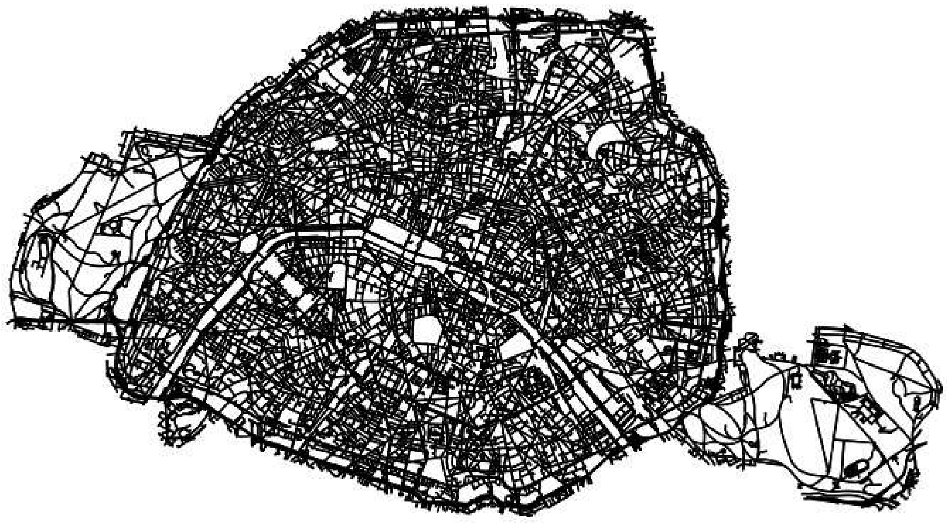


Figure 1: Urban street network.

As stated by [6, Sec III.B.1], urban areas were shaped in many different ways due to peculiar geographical, historical, social economical mechanisms. But in the last twenty years many empirical studies like [8], [9], [10] have shown that, at a coarse grained level, similarities exist between road networks of different cities.
The simplest description of the street network consists of a graph whose links represent roads, and nodes represent roads intersections and end points as shown in Figure 1, this is called primal street network. But, in most of this thesis work, a street network derived grid was used for the upon mentioned generalization possibilities [11].
In urban street network, the importance of a road can be characterized by the traffic and, with the assumption of equal traffic between all pairs of nodes, betweenness centrality as defined in Equation 4, is a natural proxy for traffic [6, Sec III.B.1.b]. This concept will return in the first part of the analysis, where edge prediction will be performed (Section 3.1.1).
A wide range of urban network predictive tasks are possible, supervised learning problems assign values or labels to node or edges with examples including travel speed/ time prediction, traffic volume/flow estimation, environmental indicators, and urban land usage classification.
Traditional approaches for modelling mobility used physical models, among those the timeless Gravity model [12] captures traffic in a simple form: the traffic volume from

one region to another is proportional to the product of population sizes of origin and destination regions and inversely proportional to the distance between those regions. Although simple, Gravity and subsequent Radiation models [13], still are an important reference in Urban mobility tasks, with integrated models that bridge deep learning and physical models achieving notable results [14].
Physical model present the clear advantage of being fast and interpretable but suffer in accuracy and flexibility.
Over the past decades, many statistical methods were applied to these tasks: logistic regression [15], support vector regression [16] and ensemble learning like random forests and gradient boosting [17]. These models learn relationships between features and target variables without explicitly modelling the network. As [17] note random forests even compete and show certain advantages in some prediction scenarios.
At the same time, deep non graph models, like Multi-layer perceptrons (MLPs), CNNs and Recurrent neural networks (RNNs), have strong predictive capabilities and generally outperform on a wide range of related tasks the previously cited methods in terms of mean absolute errors, with the drawback of having a higher data requirements and slower computations speed [17], [18], [19].

All the cited models treat each road segment as an independent observation, failing to capture spatial interdependencies or the influence of network topology. Some previous studies have attempted to address this limitation by transforming the network into a two-dimensional image and processing it with a CNN [20]. However, this approach still fails to incorporate the network structural information into the computation.
This recurrent limitation of both physical models and non graphical machine learning motivate the adoption of geometric deep learning techniques that natively model the structure of spatial networks.

2.2 Geometric deep learning

As a continuously evolving subject, in the recent years the term Geometric deep learning has been used in multiple ways, in the following thesis, the definition comes from [1], different network architectures that can be derived from principles of symmetry and invariance.

2.2.1 Basics

Learning arbitrary functions in high dimensions is a notoriously cursed estimation problem [21], but most tasks of interest are not completely generic, they have essential constraints and pre defined regularities arising from the underlying low dimensionality and structure of the physical world [1, Sec 1].
A classic way to make problems easier and contrast the curse of dimensionality is to exploit symmetries, invariances under certain transformations (translation, rotation, etc.), in order to reduce complexity.

Deep learning systems follow the same principles, researchers have adapted neural networks to also exploit domain symmetries [1, Sec 3.5]:
- CNNs exploit translation invariance in grids.
- RNNs exploit temporal structure in one dimensional grids (sequences).
- Graph Neural Networks (GNNs) exploit permutation invariance in nodes of graphs.
- Transformer exploit permutation invariance in nodes of complete graphs.

GDL emerges as an umbrella term for techniques that generalize deep models to non Euclidean domains such as graphs and manifolds.
Urban street networks present a rich collection of non Euclidean data: roads form graphs with complex topology and irregular geometry.
For the models that will be used in this thesis the idea is that signals which are defined on a graph, feature vectors associated with its nodes and/or edges, should be processed in a way that preserves the graph's symmetries, especially permutation invariance of the nodes and local interactions.

2.2.2 Graph Neural Networks

The Graph Neural Network is the canonical GDL architecture, introduced in 2009 by [22]. In its simplest form, each node maintains a feature vector and iteratively updates it by receiving messages from its neighbors. GNNs are currently among the most general classes of deep learning, as a variety of other architectures can be seen as a special case of them [1, Sec 5.3].

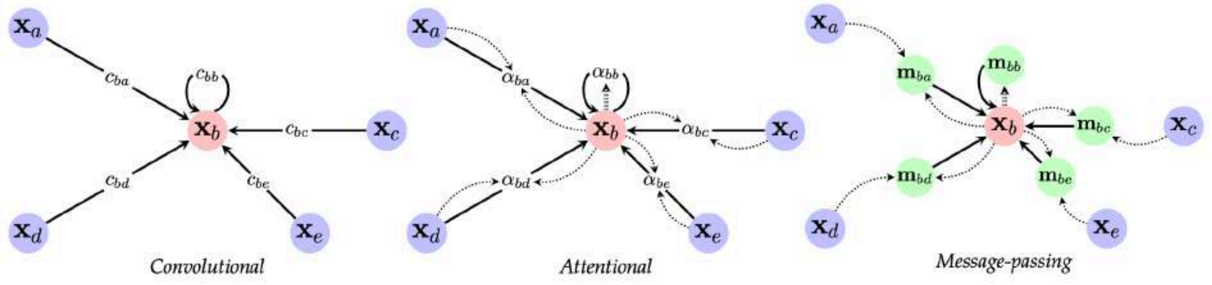


Figure 2: Visualization of the three flavours of GNN layers, reproduced from [1, Sec 5.3, Figure 17]

All considered GNNs are constructed by applying shared permutation invariant functions $\phi\left(\mathbf{x_u}, \mathbf{X}_{\mathcal{N}_u}\right)$ over local neighborhoods, where $\mathbf{x_u}$ are the node u features and $\mathbf{X}_{\mathcal{N}_u}$ are the neighbors' features . The function ϕ is called message passing, then $\boldsymbol{F}(\mathbf{X}, A)$ is the full permutational invariant function that applies ϕ to the whole graph using adjacency matrix $\mathbf{A}$ (Equation 3) and is referred to as the "GNN layer".

New designs of GNNs are created every day at the time of writing, [1] finds three macro-categories, or *flavours* that comprehend the vast majority of the existing literature. These flavours define how ϕ transforms the neighbors features.
All the flavours share a common structure:
1. features from $\mathbf{X}_{\mathcal{N}_u}$ are transformed by a function ψ

2. transformed features are aggregated with a permutation invariant function $\bigoplus$
3. features of node u are updated by a function ϕ

ψ and ϕ are learnable functions[1] while $\bigoplus$ is usually a non parametric operation (sum, mean or maximum).

The *convolutional* flavour aggregates neighbors with fixed weights and can be seen as a generalization to graphs of convolution.

$$\boldsymbol{h}_u = \phi\left(\mathbf{x}_u, \bigoplus_{v \in \mathcal{N}_u} c_{uv} \psi(\mathbf{x}_v)\right) \tag{5}$$

c_{uv} specifies the importance of node v with respect to u representation (embedding). With [23], [24] being the first popular implementation where the fixed c_{uv} were used to compute weighted averages.

The *attentional* flavour makes the interaction importance, in Equation 5 represented by c_{uv} implicit:

$$\boldsymbol{h}_u = \phi\left(\mathbf{x}_u, \bigoplus_{v \in \mathcal{N}_u} a(\mathbf{x}_u, \mathbf{x}_v) \psi(\mathbf{x}_v)\right) \tag{6}$$

a is a learnable *self-attention*, which is a mechanism that computes the importance coefficient $\alpha_{uv} = a(\mathbf{x}_u, \mathbf{x}_v)$ implicitly and usually are softmax normalized across the neighbors.
Most popular example of these models is [25], which created the Graph Attention Network (GAT), model that is extensively used in this thesis, both as the main component of a network and in hybrid models (Section 3).

At last [1] report also the more general *message passing* flavour: that computes arbitrary "messages" across edges:

$$\boldsymbol{h}_u = \phi\left(\mathbf{x}_u, \bigoplus_{v \in \mathcal{N}_u} \psi(\mathbf{x}_u, \mathbf{x}_v)\right) \tag{7}$$

here ψ is a learnable *message* function.

In general there is a hierarchy: *convolution* $\subseteq$ *attention* $\subseteq$ *message passing*. But due to the focus of this thesis on attention, only attentional networks will be considered going forward. Still keep in mind that various types of GNN have been applied to urban networks, [26] indicates that traffic forecasting, especially in node/edge level tasks, is the most dominant use case for GNNs in intelligent transportation systems (ITS), with high accuracy on real data. But there are many different application in the liter-

[1] typically they are learnable affine transformation, with ϕ including also an activation function σ to introduce non linearity.

ature. As an example [27] proposed a model (heterogeneous graph attention network) with general prediction capability on a wide range of application like crimes, average personal income or traffic flow while finding application also in spatial clustering tasks.

2.2.3 GAT and local attention

Graph Attention Network [25] introduced learnable, data dependent weights to focus on more relevant neighbors. In a GAT layer, each node learns attention coefficients for its neighbors based on their features and structural context.
A shared learnable linear projection ($\mathbf{W}$) transforms every node's features ($\mathbf{h}$) then an attention mechanism computes unnormalised scores for each neighbor node:

$$e_{ij} = a\left(\mathbf{W}\vec{h}_i, \mathbf{W}\vec{h}_j\right) \tag{8}$$

that indicates the importance of node j's feature to node i[2]. And is computed only for neighbors, this way the graph structure is injected into the model. Then a softmax normalizes these scores:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}_i} \exp(e_{ik})} \tag{9}$$

obtaining the normalized attention coefficients which are used to compute a linear combination of the features corresponding to them, to serve as the final output features for every node:

$$\vec{h}_i' = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{W} \vec{h}_j\right) \tag{10}$$

With σ being an activation function applying nonlinearity.

[2]the attention mechanism a is a feed forward neural network. Then a *LeakyReLU* activation function is applied to the output of a to obtain e_{ij}

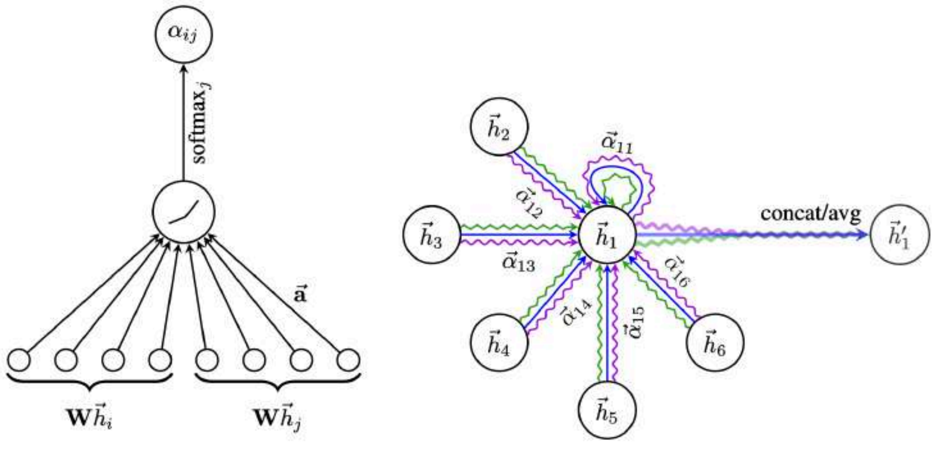


Figure 3: **Left:** attention mechanism, **Right:** illustration of multi-head attention with 3 heads (different colors), by node 1 w.r.t. its neighbors.
Reproduced from [25, Sec. 2.1, Figure 1]

Multiple independent attention mechanisms, each executing Equation 10, called *heads*, are found to be beneficial to the model [25, Sec. 2.1], allowing different heads to capture diverse patterns.

Their output can then be concatenated or averaged as shown in Figure 3.

By stacking multiple GAT layers, each layer processes features that have already been aggregated from the previous layer's neighbors. This way, later layers capture information from nodes several hops away, gradually producing the network's final *embedding*.

This local attention mechanism allows the network to focus on important elements of the network (e.g. in this thesis setting they can be roads with high traffic flow or adjacent zones with similar land use) and has been shown to obtain state of the art performances on various graph tasks, especially in *inductive* learning [25, Sec. 3,4, Table 3] while keeping the model efficient: a single GAT attention head presents a time complexity of

$$O(|V|FF' + |E|F') \tag{11}$$

where F and F' are the number of input features and computed features respectively, and $|V|$ and $|E|$ are the number of nodes and edges in the graph, respectively [25, Sec. 2.2].

However, GAT, while excelling in capturing local dependencies, relies on neighborhoods: a stack of K layers aggregates information at most K hops away, which is not sufficient for modelling long-range dependencies in big graphs like street networks.

2.2.4 Transformers

Until now, considered sets (therefore their respective graphs), always had a priori structure among the application covered so far. In [1, Sec. 5.4] is shown how starting from a matrix of node features $\mathbf{X}$ with no assumed ordering and no given adjacency, it is possible to recover two important models by choosing how much interaction is

assumed between set elements (nodes).
If all nodes were assumed independent, $\mathbf{A} = \mathbf{I}$, deep sets [28] would arise as a special case of a convolutional GNN.
If instead, which is a more interesting case for urban networks, some kind of relational structure is expected between the nodes, a possibility would be to assume that no possible link between nodes should be excluded a priori.
Following this approach a *complete* graph is considered: $\mathbf{A} = \mathbf{1}\mathbf{1}^\top$ or $\mathcal{N}_u = \mathcal{V}$, where $\mathcal{V}$ is the whole set of nodes.
Without assuming any kind of coefficient of interaction, applying a convolutional GNN would add the same contribution from neighbors to all nodes' embeddings. Making it pointless and equivalent to the previous case of $\mathbf{A} = \mathbf{I}$.
Using attentional GNN instead results in:

$$\boldsymbol{h}_u = \phi\left(\mathbf{x}_u, \bigoplus_{v \in \mathcal{V}} a(\mathbf{x}_u, \mathbf{x}_v)\psi(\mathbf{x}_v)\right) \tag{12}$$

this is exactly Equation 6 applied on a complete graph and brings to the *self-attention*, the core of Transformer architecture introduced in the landmark paper [29].

Since all attentional coefficient a are normalized between $[0, 1]$, self-attention can be seen as a soft-adjacency matrix inferred by the optimization of a given task.
In this perspective Transformers are attentional GNNs in a complete graph [2].
Transformers were proposed to model sequences, with positional encoding injected into the output: node features $\mathbf{x}_u$ are augmented to encode u's position in the sequence, typical way to do it is to use sine functions whose frequency is dependent on u [29, Sec. 3.5].

2.2.5 Graph Transformer and global attention

If on sequences the order of each node is clear, on the contrary, on graphs, no natural ordering of nodes exists. Instead, one injects structure through graph positional/structural encodings, such as eigenvectors of the graph Laplacian, shortest-path distances or random-walks [30].
With these encodings, graph transformers extend complete self-attention to graphs, enabling global (all pairs) interactions while retaining permutation invariance at the node set level.
In this thesis, two positional encodings (PEs) that are consistent with graph symmetries were tested. In particular, Random-Walk Positional Encodings (RWPE) specifically the one introduced by [31], which only considered the landing probability of a node i to itself. RWPE provide a unique node representation under the condition that each node has a unique k-hop topological neighborhood for a sufficient large k. This encoding is defined as *relative*.
Laplacian eigenvector encodings provide *global* spectral coordinates computed from

the whole-graph Laplacian [32]. Both PEs are defined from graph structure and can be used in a permutation invariant way.
In contrast, supplying raw (x, y) coordinates introduces an external reference frame, since they are absolute values they are not invariant to Euclidean transforms (e.g., translation/rotation), so the model can learn to overfit the absolute positions rather than the structure. This can damage generalization in spatial settings.
The positional encoding is concatenated inside the input and was meant to not introduce any constraints linked to the existing network structure so that the transformer would be free to create any global link across the whole graph in the attention computation [30].

This global view introduced by applying attentional models to complete graphs trades expressivity for cost, time complexity becomes $O(|V|^2)$ [29, Sec. 3.4], in return it changes the inductive bias: attention can rewire the information flow beyond the given edges by $\mathbf{A}$. This is precisely why graph transformers pair well with GNNs in hybrid designs (See Section 2.2.7): self-attention supplies long-range and adaptive connectivity. Message passing preserves local, topology aware inductive bias. A "best of both worlds" that later sections will explore.

2.2.6 Linear transformer: Performer

As stated in the previous chapter, transformers on complete graphs come with computational cost disadvantage, complexity quadratic in the number of nodes can quickly become a problem when working with very long sequences or with big graphs like street networks.
Linear transformers started to emerge in the beginning of this decade [33], approximating self-attention with linear dot products. This new technology is an ideal candidate for graph transformers, and it was quickly adopted for this application since it provides a way to achieve a scalable model [34]. Like [34], in this thesis a specific linear transformer has been used: the Performer model from [35].
This transformer variant replaces quadratic softmax attention with a "kernel based", linear in time and space approximation called FAVOR+ (Fast Attention Via positive Orthogonal Random features). The key idea is to rewrite softmax attention as a kernel: $K(q, k) = \exp(q^\top k)$, where q is the attention query and k is the key and approximate it with a positive random feature map ϕ such that $K(q, k) \approx \mathbb{E}[\phi(q)^\top \phi(k)]$.
This way attention computation takes the form: $Q'(K'^\top V)$, with $Q' = \phi(Q)$ and $K' = \phi(K)$ and V is the attention value. The described steps avoid materializing the full $|V| \times |V|$ matrix and reduce the complexity to $\approx O(|V|)$. All the theoretical guarantees[3] are furtherly described in [35, Sec. 3].

[3]such as unbiased or nearly-unbiased estimation of the soft-max attention matrix, uniform convergence and low estimation variance.

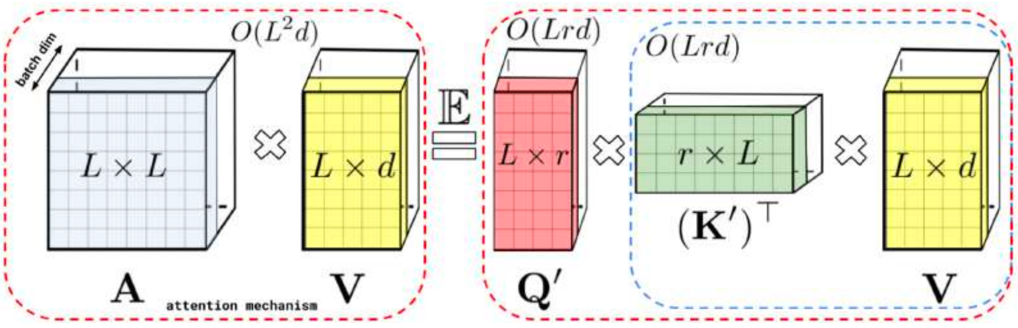


Figure 4: Approximation of the regular attention mechanism before normalization. Dashed-blocks indicate order of computation.
Reproduced from [35, Sec. 1, Figure 1]

In this thesis, global attention operates over the nodes set of a city-scale street (or grid cells) graph, where the number of nodes can reach tens of thousands. A vanilla Transformer would require storing a $|V| \times |V|$ attention matrix for each head and layer, making the approach impractical on typical hardware and often demands high end systems with an intensive energy use.

Performer keeps all pairs, providing global attention on the complete graph with the budget of a linear operator. Still, what is provided is and approximation that comes with errors, which can be mitigated by increasing the random feature maps dimension, and with an intrinsic interpretability problem: not materializing the full attention matrix, as shown in Figure 4, provides a computational advantage but no attention maps are natively available for interpretability purposes (see Section 2.2.8), this required a dedicated attention extraction which will be treated in the respective section.

2.2.7 Hybrid models: combining local and global operations

Neither purely local nor purely global models are fully able to capture the multi scale nature of urban systems. Local message passing respects the street network's topology and is computationally efficient but cannot capture interactions between distant parts of the city. Global attention can model long-range correlations but are expensive and may learn spurious connections. Hybrid models aim to combine the strengths of both. Existing examples like the GraphGPS [34] architecture combine local and global attention: each layer performs a local message-passing step (any GNN), computes global attention (Transformer) and both outputs are integrated by summing these contributions with the residual stream, which passes each layer's input untouched and adds it to its output, to update the node representations.

A similar model, the Global–Local GAT (GLGAT) model, has been recently used for origin–destination flow inference using a single global Transformer layer alongside several local GAT layers and combining their outputs showing impressive accuracy improvement in real world dataset predictions [36]. Once again the global component captures correlations between distant zones, while the local layers model neighborhood effects.

Hybrid architectures, already explored various domains like recommender systems [37] or even pollution prediction [38], are promising to be relevant for urban applications, where, for example, traffic congestion on a road may be influenced both by its neighbors and by global bottlenecks. However, within urban and infrastructure literature, this approach remains exceptionally scarce.
In 2025, some works appeared, such as [39], focusing primarily on prediction accuracy and showing that combining local and global attention can achieve new state of the art results. Despite the limited number of examples in this domain, the trend in closely related application areas shows a clear growing interest and adoption of this technology.

2.2.8 Mechanistic interpretability

As graph models grow more complex, it becomes essential to understand how they make predictions.
Attention mechanisms provide an intuitive window into a model's operation, since attention weights indicate which nodes influence a target. However, recent research shows that raw attention not always corresponds to a faithful explanation: different attention patterns can yield the same output, and high attention weight does not guarantee high importance [40].
Moreover, self-attention weights alone capture only part of the information flow. Feed-forward layers and residual connections also mix information, making attribution challenging.
To address these issues, new interpretability tools have been developed:
Attention graph aggregates multi-head, multi-layer attention matrices into a single weighted graph, enabling the comparison of the learned information flow with the original network [3].
Head correlation analysis studies correlations between attention heads to identify redundant or complementary heads, informing head pruning and clustering. Inspecting the residual stream hidden representations has become more and more popular for transformer since the first transformer circuit article came out [4].
Via dimension-reduction, it could be possible have insights how node embeddings evolve across layers. Together, these techniques provide informations into how GNNs and Graph Transformers process data.

Chapter 3
Tools, framework and implementation

This chapter formalizes the learning problems addressed in this thesis, discusses the data collection and preprocessing pipeline, describes the graph based architectures used, the training and evaluation protocols and presents the interpretability framework.

Two distinct prediction tasks are considered:

1. Edge-level traffic speed prediction: given a primal street graph, predict the average travel speed on each edge. Inputs include the graph structure and node/edge attributes (e.g., degree, betweenness centrality, presence of traffic lights). This task was used to design and validate the architectures and to gain familiarity with geometric deep learning; edge-level predictions naturally align with network-flow phenomena and often benefit from considering the graph structure.
2. Node-level green view index prediction: Given an H3 aggregated (Section 3.1.2) graph derived from a street network, predict the Green View Index (a continuous measure of visible vegetation from street-level imagery) for each node. This task supported the development of the interpretability framework and a comparison between graph based and non graph baselines under progressively richer inputs, using a feature rich dataset with information about demographics, land use, and other visual indicators.

Throughout the chapter we use $G = (V, E)$ to denote a graph with nodes V and edges E.
The definition of V and E differs depending on the task therefore the two problems settings are described separately.

3.1 Problem formulation

3.1.1 Edge-level traffic-speed prediction

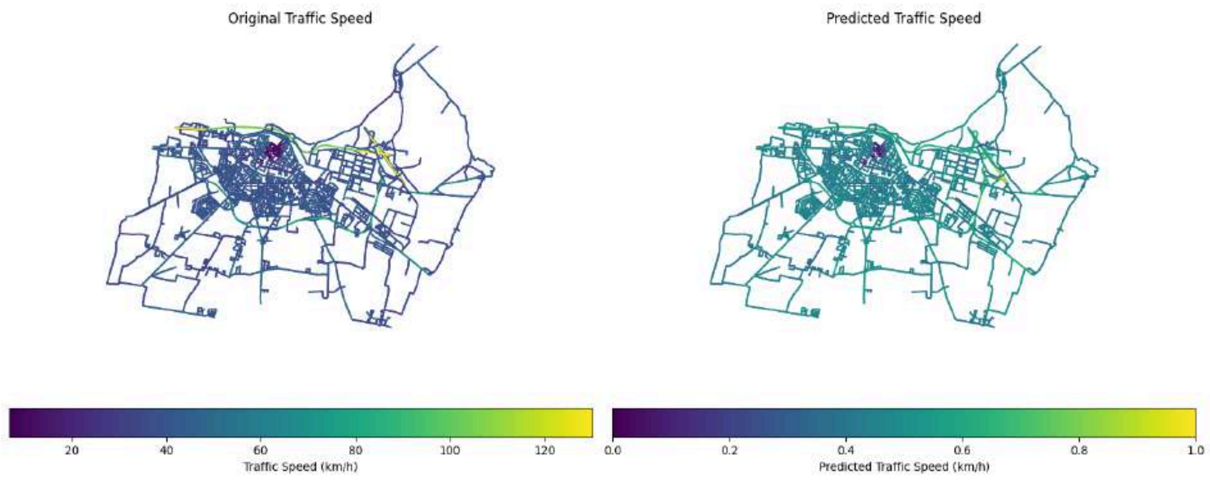


Figure 5: First task prediction example: ground truth (**LEFT**) and task prediction (**RIGHT**)

The first task tackled was an edge-level one operated on (directed) primal street networks $\boldsymbol{G}_c = (\boldsymbol{V}_c, \boldsymbol{E}_c)$ (one per each city) extracted from OpenStreetMap [41]. Each node $v \in \boldsymbol{V}_c$ has features such as its coordinates, degree, betweenness centrality and a binary indicator of whether a traffic light is present. The vector of features for the node v is denoted with $\mathbf{X}_v \in \mathbb{R}^d$

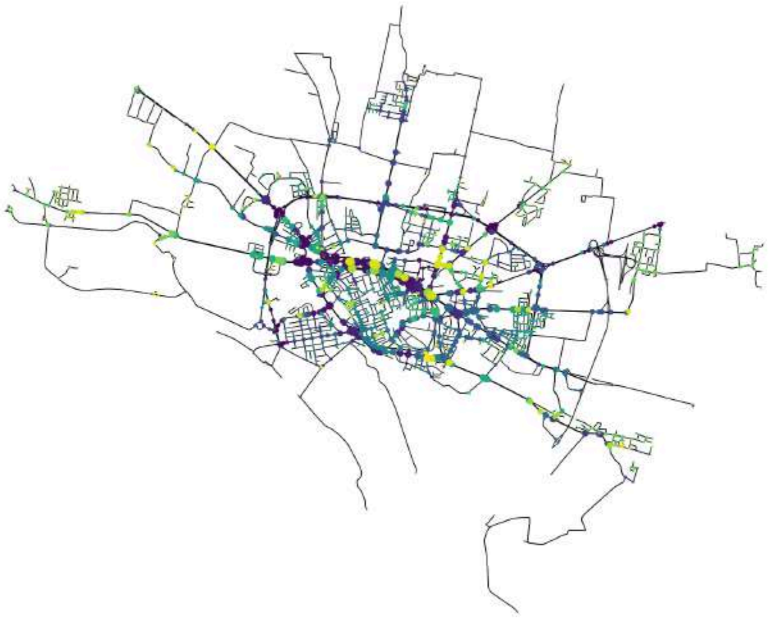


Figure 6: Betweenness centrality visualized on city Cremona

The target $e = (u, v)$ for each edge is the travel speed on that road segment. This speed is estimated by combining the road limit speed with penalties derived from betweenness centrality (as shown in Figure 6), traffic lights and proximity to the city center. It is important to note that this speed is not meant to be a real, proper, modelling of the traffic, just a way to test the network on an edge level tasks:

$$\text{traffic\_speed} = \text{limit\_speed} \times (1 - 0.7 \times \text{bc}) \times (1 - 0.2 \times \text{traffic\_light}) \times (1 - 0.5 \times \text{city\_center}) \tag{13}$$

where bc is the betweenness centrality of the upstream node, traffic light is boolean true if the street segment has a traffic light and city_center is a boolean indicating whether the road is inside a 300 meters radius from a chosen "city center".
The resulting dataset provides synthetic travel speeds for several Italian cities (Section 3.2.1).
The objective is to learn a function that predicts the travel speed for each edge:

$$\boldsymbol{f}_{\text{edge}} : ((u, v) \mid \boldsymbol{G}_c, \boldsymbol{X}_c, \boldsymbol{P}_{uv}; \theta) \to \hat{y}_{uv} \tag{14}$$

where $\boldsymbol{P}$ is the positional encoding and θ are the model parameters.
This is achieved through the usual method when training neural networks: gradient descent, where an optimizer updates the model parameters, so that the predictions better match the targets, by following the gradient of the loss function: after each step of the training, parameters are changed in the direction that reduces error, with the step size controlled by a learning rate [42].
In this work Adam was used: a popular optimizer that improves standard gradient descent by adaptively adjusting the learning rate of each parameter using estimates of the gradients mean and variance, bringing in more stable and efficient training [43].
Since this problem has been formulated as an edge-level regression, the chosen loss function to minimize was the mean squared error:

$$\text{MSE} = \frac{1}{N} \sum_{e=1}^{N} (\hat{y}_e - y_e)^2 \tag{15}$$

Where y_e is the real target value for the e-th edge and $\hat{y}_e$ is the respective prediction from the model; N here is the total number of edges.

3.1.2 Node-level greenness prediction

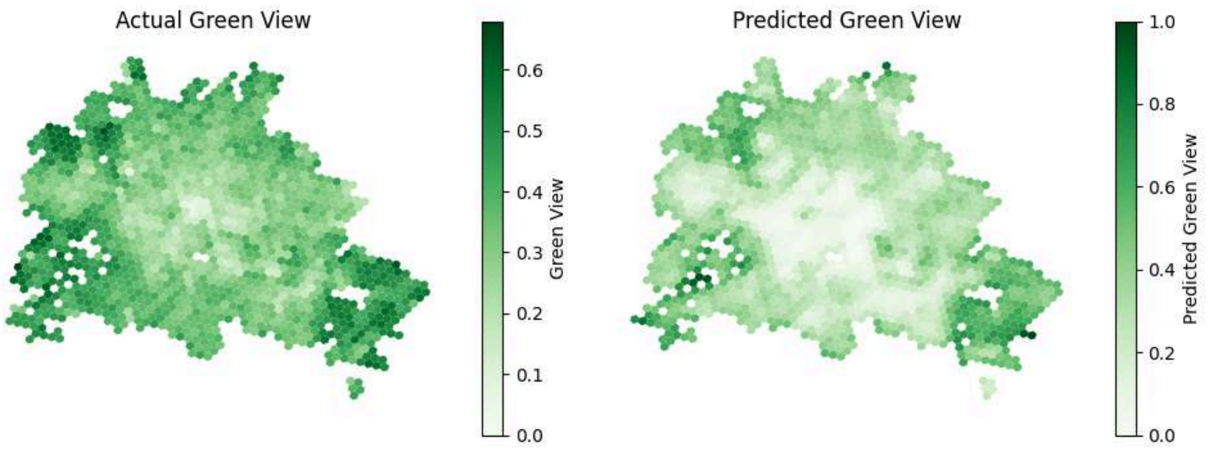


Figure 7: Second task prediction example, ground truth (**LEFT**) and task prediction (**RIGHT**)

Each city is discretized into hexagonal cells using the H3 indexing system. H3 is a discrete global grid system for indexing geographies into a hexagonal grid, developed at Uber [11]. It has a hierarchical structure which allows to aggregate geographically while preserving adjacency relationships.
The goal of this task is to predict the Green View Index (GVI) of each cell, which is an objective measurement of urban green at street level: unlike satellite derived indexes, GVI uses street-level imagery (mainly Google Street View), to quantify the presence of vegetation that can be seen walking in a given street [44].

For each city $c \in \boldsymbol{C}$ a set of H3 cells (nodes) $\boldsymbol{V}_c$ is obtained, different resolutions correspond to different size and number of the cells, and an edge set $\boldsymbol{E}_c$ is built by connecting each cell to its first order neighbors, using the H3 *k-ring* operator (Figure 8), obtaining a graph $\boldsymbol{G}_c = (\boldsymbol{V}_c, \boldsymbol{E}_c)$

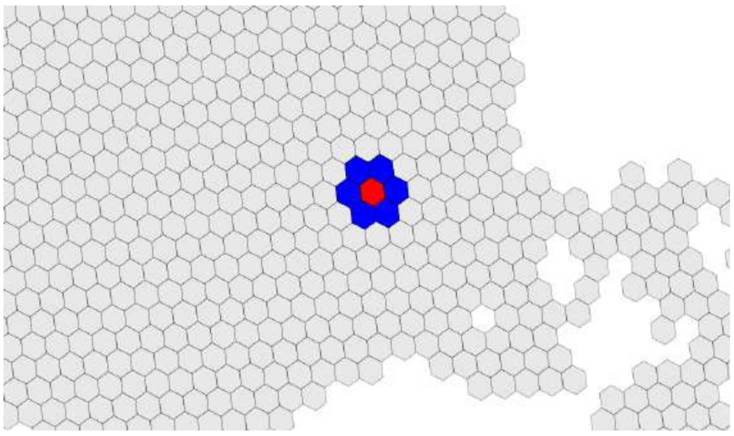


Figure 8: H3 representation of Paris (zoomed in): selected cell (**RED**) and its first order neighbors cells computed by the k-ring operator (**BLUE**)

In this second task, many H3 graphs created from European capital were used (Section 3.2.2) and each node $v \in \boldsymbol{V}_c$ presents various associated features that will be later discussed. Once again by denoting with $\mathbf{X}_v \in \mathbb{R}^d$ the vector of selected features for node v, the goal of this task is to learn a function which for a specific node v:

$$\boldsymbol{f}_{\text{node}} : (v \mid \boldsymbol{G}_c, \boldsymbol{X}_c, \boldsymbol{P}_v; \theta) \to \hat{y} \in \mathbb{R} \tag{16}$$

which maps each node's selected features (d) and positional encoding (m) into a scalar value: the node's GVI. Even if this time the problem has been formulated as node-level, it still is a regression, therefore once again the loss to minimize is MSE.

3.2 Data sources and preprocessing

The two tasks use completely different data, although they are all urban datasets:

3.2.1 Edge-level task:

the data was collected from OpenStreetMap (OSM) [41], with the features available on the open source platform. As it is possible to see from the function `city_speed_preprocessing` available in the GitHub repository of this project [45],

the primal street network for each city was obtained directly using OSMnx's `graph_from_place` function (`network_type='drive'`) [46].
The graph is then projected to ensure that edge lengths are expressed in meters, enriched with imputed edge speeds and travel times via OSMnx routines that infer street speeds from max speed tags or road classes.
The projected graph was then converted to GeoDataFrame, so that edges and nodes can be managed using GeoPandas [47]. To incorporate the traffic proxy previously mentioned (see Equation 13), node and edge betweenness centrality was computed. Both centrality vectors were min–max normalized to [0,1] and added to the corresponding node/edge data frames.
From OSM also the location of traffic lights has been downloaded and a boolean value (`traffic_lights`) was added to mark edges incident to a traffic light.
In order to make the proxy more complex, each city center has been defined by a buffer around a central reference point. Then an edge-level boolean flagged edges within the buffer to model an area of increased congestion (`traffic_center`).
This way all the variables needed for the target computation of Equation 13 are created for each city and supervised learning is possible with the traffic proxy from Equation 13 as target. It was also computed a derived edge travel time which was used for sanity checks.
This whole procedure was applied for seven different cities, to create the training set composed of: Cremona, Piacenza, Pesaro, Trento, Rimini and two cities to not use during the training and test the generalization capabilities: Verona and Ferrara.

Finally, each city graph is converted into a PyTorch dataset with `preprocess_data`. First, edges are split into 70/15/15 train/validation/test sets. Next, there is the selection of node features, the scaling of the nodes incident to the training edges (and on the training edges for edge features such as length), and apply the same scaling to all nodes/edges. A random walk positional encoding is added per each node, and the result is ready for GNN and Transformer models.

3.2.2 Node-level task:

For this task the Global Urban Network (GUN) dataset from Urbanity has been used as the source [48].
GUN dataset provides pre-computed node and edge attribute features for various big cities, nodes and edges are already distinct geo data frame and, as usual, they compose primal street networks. The attributes span network structure (e.g. degree, centralities), footprint metrics, land-use/POI counts, demographics and visibility indicators (e.g. Green View, Sky View).
For each city, the H3 network had to be constructed, the function `create_h3_aggregated_graph` in the repository [45] shows the exact procedure applied: the street network was first indexed to H3 using H3PANDAS, in particular to resolution 9, by assigning an h3 index to each node in the network (`geo_to_h3`), then

creating the geometries of the cells (`h3_to_geo_boundary`).
All the attributes of the individual nodes had to be aggregated into the respective cells, the following table shows the criterion of aggregation for all the categories of features (using `groupby.agg(agg_dict)`) :

Category	**Aggregation**
coordinates	replaced by centroids
street-network structure	deleted
building-footprint morphology	averaged
demography	summed
POI and land-use	summed
street-level vision	averaged

Green View Index, once averaged is the target of the supervised learning of this node level task.
After aggregation, a geo data frame of H3 nodes is built with the cell polygons as geometry. In order to create the edges, and the consequent adjacency structure, one-ring neighbors (Figure 8) are computed (`h3.k_ring(1, explode=True)`). Repeated pairs are removed, symmetry is checked, so the final set has undirected adjacency.

This procedure was repeated for five different cities: Amsterdam, Paris, Berlin, Edinburgh and Milan. Four of these are used to train the models and one is left out for generalization testing, different permutations of the left out city have been tried.

Each H3 graph is, once again, converted into a PyTorch dataset with the respective `preprocess_data`. The function first selects node features when x,y are present, they are centered per city (subtracting their means) to remove translation bias, then the function adds the random walk positional encoding.
To mitigate spatial leakage, splits are done on the node H3 resolution. The nodes are first grouped by their parent H3 cells at an inferior resolution (instead of `resolution = 9`, `resolution = 8`) and the parents are randomly partitioned 75/15/15 into train/val/test; children nodes then inherit the masks by the parents.
Feature standardization is once again performed with a fit on the training nodes only and then applied to all nodes. Finally, the dataset is packed as a data object ready for the models.

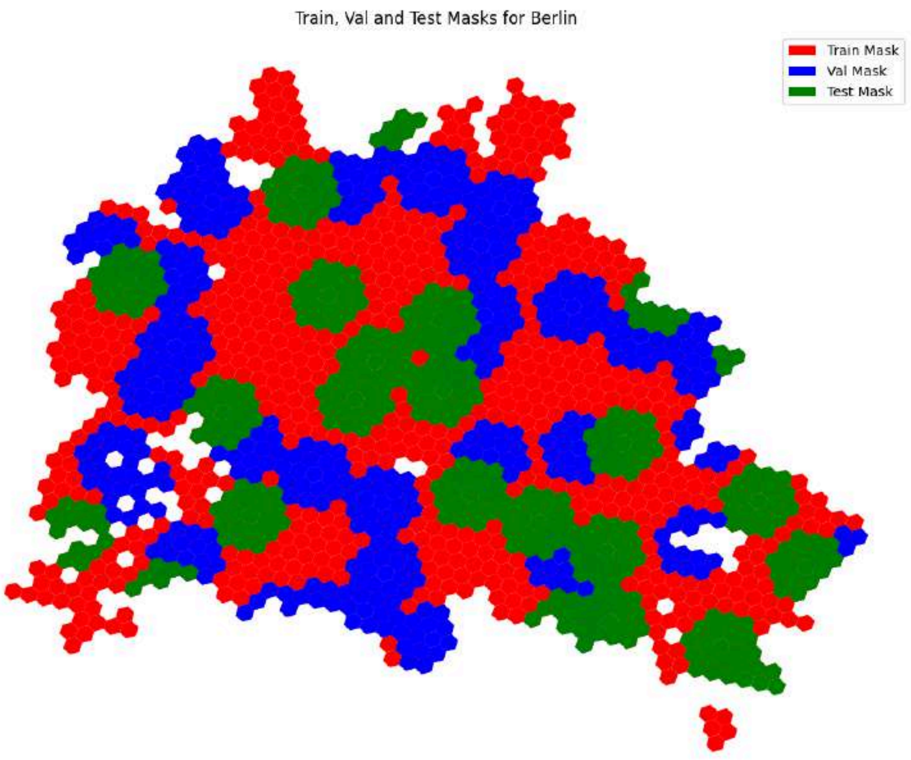


Figure 9: Example of train/validation/test split done on parent resolution for the city of Berlin

3.3 Models

This subsection presents the model architectures used in this thesis: a pure GAT and a pure graph Performer model, and the proposed hybrid models, describing their design, key components, and training configuration.

In order to be compatible with the whole interpretability framework (Section 3.4), all the proposed models are structured in a similar way:

- Positional encoding and initial embedding:
  Each model begins by fusing input features with a positional signal into an hidden state h
- Attentional layers adding to the residual stream:
  Attentional blocks (GAT or Performer) produce updates that are added to a running residual stream, preserving information flow and enabling residual-stream analysis.
- Feed-forward layers after global attention:
  For Performer and hybrid variants, a feed-forward layer follows each global attention step. The respective output also is summed to the residual stream.
- Final linear layer predicting the output:
  A single linear readout maps hidden states to prediction.

In particular the initial input embedding part is in common for all models: each model begins concatenating a learnt linear transformation of the input features $X_v \in \mathbb{R}^d$ with a linear projection of the positional encoding $\mathrm{pe}_v \in \mathbb{R}^k$. Resulting in an hidden initial embedding h_v. Then an initial LayerNorm [49] is applied to h_v in order to normalize before feeding to any layer. All the models are inspectable in the repository [45], they present small variations in each task which can be seen in the respective folder inside the files [`models.py`].

3.3.1 GAT model

This is the pure GAT model and follows the common blueprint adopted for all architectures in this thesis (embedding $\longrightarrow$ attentional updates added to the residual stream $\longrightarrow$ linear readout). The formal definition of a GAT layer was introduced in Section 2.2.3. In the subsection follows the description of the architecture.

Each node v is embedded by the common stated procedure, producing the hidden state $h_v \in \mathbb{R}^H$, where H is the hidden dimension that is kept constant for the whole model in order to make possible the residual stream updates inspection.
After the embedding, each node's h_v is processed by L **GATv2** ([50]) stacked layers, with multi-head attention, each producing an output of dimension $\frac{H}{\#\text{heads}}$ that gets concatenated into $h_v^{\mathrm{GAT}_k}$. After the first GAT layer, an activation function (LeakyReLU REF) is applied, whose contribution is to apply element wise:

$$\mathrm{LeakyReLU}(x) = \begin{cases} x, \text{ if } x > 0 \\ \text{negative\_slope} \times x, \text{ otherwise} \end{cases} \tag{17}$$

where negative_slope was set equal to 0.1, and the result is added back to the residual stream updating the hidden embedding and obtaining $h_v^{\text{1st layer}}$. Eventual intermediate GAT layers are plain attention updates without the activation function which are directly added to the residual stream. The final GAT layer uses a single head of dimension H in order to preserve the dimensionality and allows transparent readouts. After the attentional part of the model the final readout is task specific:

- Edge-level task: endpoint embeddings of each edge $e = (u, v)$ gets concatenated $[h_u \mid\mid h_v]$, also with edge attributes (e.g. length of the edge) $[h_u \mid\mid h_v \mid\mid e_{\text{att}}]$ and result is fed in a linear layer: $\mathbb{R}^{2H+d_{e_{\text{att}}}} \rightarrow \mathbb{R}$ which produces one prediction per edge.
- Node-level task: final embedding of each node h_v is directly fed in a linear layer: $\mathbb{R}^H \rightarrow \mathbb{R}$ giving back the node prediction.

The idea behind this model is that since urban street graphs are sparse and locally structured, a simple GAT stack like this one, efficiently captures short to medium range dependencies while remaining scalable. The flow of information inside the model was also thought possible to interpret: both by inspecting the residual stream and with an implemented attention extraction method: `get_attention_weights`, that gives back an attention matrix per each layer given an input.

3.3.2 Performer model

The pure Performer model is a graph transformer where the Performer layer [35] replaces the canonical multi-head attention layer from [29].

Each node v is embedded by the common stated procedure, producing the hidden state $h_v \in \mathbb{R}^H$. After the embedding, each hidden state is processed by L Performer stacked blocks, each composed as follows:

1. Global attention: a multi-head Performer Attention layer computes a global attention over all nodes. Each head produces an output of dimension $\frac{H}{\text{heads}}$ that is concatenated back to size H. The result is added to the residual stream and the whole thing is normalized (LayerNorm), obtaining $h_v^{\text{attn}_k}$
2. Feed-forward: A 2 layer MLP $\mathbb{R}^H \to \mathbb{R}^{2H} \to \mathbb{R}^H$, with an activation function (GELU[4]) in between, adds nonlinearity to the representation and follows the typical transformer encoder architecture. Also this output is added to the residual stream and normalized, obtaining $h_v^{\text{ffn}_k}$.

After the embedding part of the model the final layer which outputs the prediction is task specific and is built in the exact same way as the one described in Section 3.3.1.

The idea behind this model is that urban graphs often require global context. The performer keeps global attention while reducing memory requirements and being linearly scalable with the dimension of the graph. The residual-stream layout is compatible with the interpretability tooling used in this thesis. An attention extraction method was implemented again, `get_performer_attention`, which returns an attention matrix per layer and head:
Performer layers do not form the $N \times N$ attention matrix during a forward pass. Instead, they use a kernel (random feature) factorization to compute the attention value product in linear time (Section 2.2.6).
For interpretability, however, the explicit, row-stochastic matrix that say "how much node i attends to node j" was needed. Therefore, only for analysis, the approximate attention matrix per head was reconstructed:

1. Run the forward until the selected layer ℓ
2. Extract q, k, v corresponding to the given input from the ℓ-th performer layer, and reshape them by heads
3. Apply the feature map used by that performer layer, `generalized_kernel(·)`, to get $q' = \Phi(q), k' = \Phi(k)$
4. Compute $q' k'^{\top}$ then divide each row by $q' \left(\sum_i k'_i\right)$, this produces an attention matrix for a given head, with rows summing to 1

[4]

$$\text{GELU}(x) = 0.5 \times x \times \left(1 + \tanh\left(\sqrt{\frac{2}{\pi}}(x + 0.044715 \times x^3)\right)\right) \tag{18}$$

5. Stack heads so that the result has shape `[heads,N,N]`

This reconstruction has quadratic time and memory in N (per head), unlike the linear time training or inference path, it was only run when producing attention visualizations or the Attention Graph. This algorithm reproduces the layer generalized attention that using ReLU, approximates the softmax attention [35, Sec. 2, Lemma 1], it is not the softmax attention. However, this is appropriate here because the rest of the analysis is done within the same approximation framework.
The matrix A reconstructed is exactly the object that would have been multiplied by V in the standard attention formulation, up to the Performer approximation. Making it explicit allows ti aggregate over heads and layers to form the Attention Graph and to compare local and global attentions (see Section 3.4), while staying faithful to the computations performed by the trained layer.

3.3.3 Hybrid models

Two hybrid architectures are proposed:
1. Late fusion hybrid: two independent branches, one local (GAT) and one global (Performer), process the shared embedded input in parallel. Their outputs are summed and passed to the final predictor.
2. Layer-wise fusion hybrid: at each layer, a global Performer update and a local GAT update are computed. Both updates are added to the residual stream, updating the embedding

Both variants follow the common blueprint (embedding $\longrightarrow$ attentional updates $\longrightarrow$ optional FFN $\longrightarrow$ linear readout) and have compatibility with the interpretability tools (attention extraction and residual-stream inspection).

3.3.3.1 Late fusion hybrid

Following the common procedure, the common initial embedding for each node is $h_v \in \mathbb{R}^H$, from here:

- Global branch (Performer). The hidden state is processed by L_T stacked Performer blocks. Each block is composed of the Performer Attention layer whose result is added to the residual stream and normalized (LayerNorm, REF), and of a 2 layer MLP whose output is added to the residual stream and normalized, just like in Section 3.3.2. The whole branch after processing through all the layers produces $h_v^{\text{Performer}}$ per each node.
- Local branch (GAT). The hidden state is processed by L_G stacked GATv2 layers (REF). On the first GAT layer, a LeakyReLU (REF) is applied and the update is added to the residual stream; subsequent GAT layers are plain attention updates added to the stream, like in Section 3.3.1. The branch produces h_v^{GAT}.

After the two branches complete, their outputs are summed (each node results in $h_v^{\text{final}} = h_v^{\text{Performer}} + h_v^{\text{GAT}}$ and a linear layer $\mathbb{R}^H \to \mathbb{R}$ produces the prediction. The late-fusion design keeps the two contributions distinct, allowing clean ablations (Transformer-only, GAT-only) and an immediate residual-stream interpretation.

For attention interpretability, the model has both `get_performer_attention` for Performer layers and `get_attention_weights` for GAT layers. Therefore also the attention interpretability can be done in the same way as for the pure models discussed upon.

3.3.3.2 Layer-wise hybrid

Here, global and local contributions are computed at each layer and combined within the residual stream before moving to the next block.
As usual, from initial embedding for each node $h_v \in \mathbb{R}^H$, for k = 1,...,L. Layer k is composed as follows:

- Global update: A Performer Attention layer is applied to the current hidden state h_v^{k}, the result for v is $h_v^{\mathrm{Performer}}$.
- Local update: A GATv2 layer is applied to the same hidden state h_v^k. On first layer a LeakyReLU is used, the result is h_v^{GAT}.
- Fusion: both results are added to the residual stream, $h_v^{\mathrm{attn}_k} = h_v^{\mathrm{Performer}} + h_v^{\mathrm{GAT}}$ and normalized using LayerNorm.
- The 2 layers MLP of Section 3.3.2 is applied to $h_v^{\mathrm{attn}_k}$, then another LayerNorm is applied to its output, resulting in $h_v^{\text{k-th layer}}$.

After L layers, the task specific usual linear layer outputs the prediction.
The layer-wise fusion lets global context and local structure interact at every depth while mirroring the transformer encoder pattern but with both global and local attention updates merged per layer.
Also this model presents the two functions for both global and local attention extraction at a given layer just like the other hybrid does.

3.3.4 Baselines

To benchmark the performance of the proposed attention-based models several baselines were implemented.
Two distinct versions of MLP are used:

- one which maps node features and positional encodings to hidden representations through two or three fully connected layers with batch normalization, activation function (ReLU) and dropout.
- one that works exactly like the first but without positional encoding, in order to verify how a basic deep model without any information regarding the graph performs compared to the other models.

A dummy baseline that predicts the mean of the training targets for all nodes, its mean squared error serves as a naive reference.

These baselines provide context for evaluating eventual benefits of GAT, Performer and hybrid architectures.

3.3.5 Evaluation metrics

During training, MSE defined in Equation 15, is minimized in both node and edge level tasks.
RMSE (square root of MSE) is also used to add interpretability since it brings back the error in the same units as the original data, but also the Mean Absolute Error (MAE[5] is reported since it is way less sensitive to outliers (due to not taking the square of the errors) For the node-level task, it was additionally computed the spatial Pearson correlation between predicted and observed GVI values using the adjacency matrix of the H3 graph. This metric evaluates how well the spatial pattern of predictions matches that of the ground truth.
Models with lower MSE and higher spatial correlation are considered superior.
In Section 4.2 there will be examples of why an auxiliary metric was implemented and MSE only results were not indicative enough in this application case.

3.4 Interpretability framework

This thesis work is intended to not only apply graph based neural networks to urban frameworks, but also to try to scrape the surface of interpreting these incredibly complicated models. In the long run, understanding how these models operates and produce predictions matters not only for validating them, it could provide new ways to study the urban structures, revealing hidden dependencies between spatial components, exposing whether the models rely on possible urban mechanisms or spurious correlations, and offering a basis for building trust in their application to real world planning and policy.

This subsection introduces the tools used to inspect and start to explain the models trained in the thesis. The goal is twofold:

1. provide a graph level interpretability of the models outputs, showing what graph objects the model utilizes to make predictions.
2. explain not only what is important for the model, but also what is happening inside it, and how it arrives to its conclusions and prediction.

Since the framework is based on a mechanistic reconstruction of the output, it is model internal: it relies only on signals produced by the network during a forward pass, avoiding proxy explanations.

Since all the models share the same input embedding and a constant hidden size H, signals are directly comparable making a unique analysis possible.

This chapter is organized into two parts:

[5] $\frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i|$

1. Attention Graph: an adaptation, with minimal additions, to the models of this thesis, especially to hybrids models of the work of [3]. Building a summary of each model's attention into a single graph that encodes information flow among nodes.
2. Residual stream visualization: a tracking of the hidden state h through the network and decompose it into layer-wise increments. This reveals when and where the model distinguish between different nodes and separates local vs global vs MLP contributions.

Together, Attention Graph and Residual Lens provide complementary views: the former explains which relations the model emphasizes; the latter explains how these selected relations change the representation and, at the end, the prediction.
The concrete algorithms and figures are implemented in the respective notebooks which are on [45] for exact reproducibility.

3.4.1 Attention Graph

The aim is to aggregate all the per-head, per-layer attention produced by the models into one directed graph that summarizes how information flows among nodes. This is implemented following the framework introduced for mechanistic interpretability of graph transformers and adapted here to the proposed architectures [3]. Concretely, according to [3, Sec. 3.2], for each layer $\ell = 1, \dots, L$ and head $h = 1, \dots, H$ the respective attention matrix $A_{\ell,h} \in \mathbb{R}^{N \times N}$ is taken, which will be sparse on existing graph edges for GAT heads, dense for Performer heads, then there is the aggregation of heads within a layer and composition of layers across depth to obtain a single "Attention Graph": A_{agg}.
Therefore attention matrices must be aggregated, first inside a single layer, afterwards across layers:

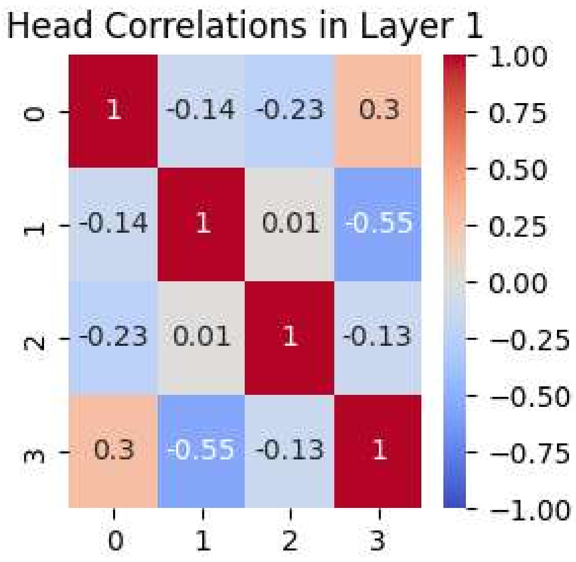


Figure 10: Example of non-positive correlation in Layer-wise fusion hybrid model, trained on node-level task

1. Aggregation across heads: multi-head attention can be viewed as parallel relation types existing between the same nodes. The authors of the reference paper note a strong positive correlation between heads in both local and global models [3, Sec. 3.2]. Claiming they had never observed negative correlation and so suggesting these models "learn complementary rather than competing patterns". From this claim they justify averaging the attention to capture the overall information without loosing important signals.
   In this thesis analysis, correlation between heads was computed and while being strongly positive for all GAT layers, it consistently showed examples of both negative and zero correlation in Performer layers both in the pure and in hybrids models (Figure 10).
   This prompted to a different intra layer aggregation, defining the aggregation between the attention from the head i and head j belonging to the layer ℓ ($A_{i,j}$):
   $$\begin{cases} \text{if } \mathrm{corr}_{i,j} \geq 0.5 \rightarrow A_{i,j} = \frac{1}{2}\left(A_{\ell,i} + A_{\ell,j}\right) \\ \text{if } \mathrm{corr}_{i,j} < 0.5 \rightarrow A_{i,j} = \max\left(A_{\ell,i}, A_{\ell,j}\right) \end{cases} \tag{19}$$
   Where max here is the element-wise maximum of the two matrices.

   This way the information across heads is preserved when the heads are operating in a "competing" way and averaged when the heads are operating in a "complementary" way.

2. Aggregation across layers: layers represent successive transformations of information across depth. In [3, Sec. 3.2], they propose using matrix multiplication of aggregated (across heads) attention matrices from successive layers to model the information flow from successive layers:
   $$A_{\text{agg}} = A_{\ell_2} A_{\ell_1} \tag{20}$$
   In this way the matrix multiplication captures indirect attention patterns. if node i attends to node j in layer 2, it indirectly attends to all nodes that j attended to in layer 1. Mathematically, row i of A_{agg} represents a linear combination of rows in A_{ℓ_1}, weighted by attention coefficients in row i of A_{ℓ_2} (as shown in Figure 11).

The combination of these two aggregations result in the final unique matrix: the Attention Graph. This is the pipeline of operation that turns a stack of attentions matrices into a single flow graph. This pipeline is the exact same for both local and global (sparse and dense) attentions, so there is no distinction in how the attention graph is built from GAT or from Performer. But, the attentions from GAT layers are sparse because they must respect the graph edges structure, this means that the resulting Attention graph will still be sparse and respect the graph structure, while the one resulting from Performer does not have any constraint, so the two graphs will be completely different in the result.

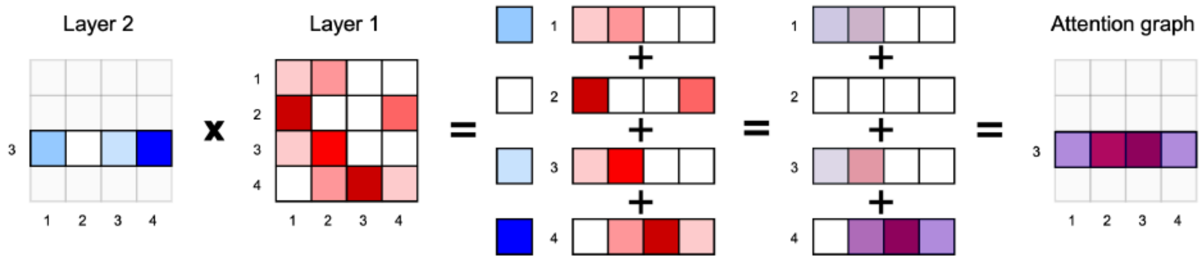


Figure 11: For node 3, row 3 in the attention matrix A_{ℓ_2} represents how much it attends to each intermediate node $1, 2, 3, 4$. Each row in A_{ℓ_1} captures how those intermediate nodes attend to other nodes. Matrix multiplication $A_{\ell_2}A_{\ell_1}$ combines these patterns, revealing how node 3 indirectly attends to the other nodes through intermediate nodes.

To reflect the actual computation of the layer-wise hybrid model, It was also built an hybrid attention graph that fuses, within each layer, the local and global attentions and then composes them across depth. Layer fusion is correlation-aware: if the two branch matrices are positively correlated (correlation ≥ 0.5) they are averaged, otherwise their element wise maximum is taken, like in Equation 19.
This creates an “effective” flow of information that matches what the network actually uses. For each layer $\ell = 1, ..., L$:

1. Aggregation of all GAT heads (as Equation 19) creating A_{GAT_ℓ}
2. Aggregation of all Performer heads (as Equation 19) creating $A_{\mathrm{Performer}_\ell}$
3. Fusing GAT and Performer creating A_ℓ
4. Fusing all the A_ℓ across layers (as in Equation 20)

The Attention Graph, both the hybrid version implemented and the original from [3], is not an explanation by itself, but a structured hypothesis about how information flows in the network. It provides a mechanistic interpretation of the way the information is captured by the network. It makes possible a quantitative assessment of how far information travels, where it concentrates, and which urban areas or attributes are considered more “important” by the network. In practice, locality can be summarized by the distribution of the shortest-path distance of the top outgoing links per node on the graph. The top outgoing links mean the nodes that are most important for the computation of a given node GVI. Shortest-path distances with a distribution peeked on one indicate predominantly local exchanges, while larger values signal non-local edges created by the global component.
Whether there exist a preference toward hubs is assessed by the Spearman correlation between attention incoming strength (summing the column of the Attention graph, after filtering the top attended nodes) and betweenness centrality of the network. Negative correlations would suggest that the model consider more important peripheral nodes. Alignment between attention and input features can be checked by correlating incoming strength and the selected attributes, which can build testable hypotheses about the factors driving attention in different parts of the city.
During training, the Attention Graph proved useful as a diagnostic. A few simple

indicators immediately computable from the graph like the typical hop distance of links or the concentration of top-1 targets on a single node, provided information regarding the global attention having a single attractor: many nodes "listen" to the same node, so global information is funneled through a single point, and model does not search for global relations.
Several sanity checks were performed to validate the construction, and examples are available in the (Section 4.3):
All matrices are checked to be row-stochastic. Also when selecting the top-k projections, they get re-normalized to preserve this property.
GAT attention graphs remain sparse and constrained to the base adjacency, whereas Performer attention graphs are dense, as one would have expected. For GAT, the support of A_{agg} lies within L-hop reachability on the base graph, as it should be since each GAT layer only models one hop from the node.

Overall, the (hybrid) Attention Graph offers a concise and quantitative picture of how the models channel information. It is easy to map back onto the city, and it supports both interpretive statements (like where and how far information travels) and practical interventions (what to adjust to improve the model).

3.4.2 Residual stream visualization

While attention is central in the models proposed in this thesis, focusing only on attention misses out important parts of the computation, like nonlinearities from activation functions, normalizations and the contribution of the MLP blocks.
The Attention Graph summarizes *where* the model focuses across the graph, the purpose of the residual stream analysis is to complement it by showing *how* the internal representation changes as it goes through the network.

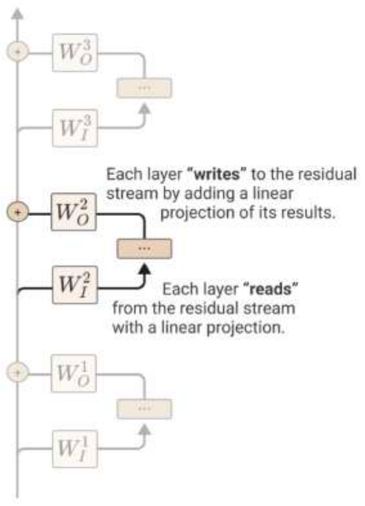

W_O^3
W_I^3
Each layer **"writes"** to the residual stream by adding a linear projection of its results.
W_O^2
W_I^2
Each layer **"reads"** from the residual stream with a linear projection.
W_O^1
W_I^1

Figure 12: residual stream representation from [4, Figure 2]

The tool was completely developed during this thesis work. But, finds its inspiration in the transformer circuit work from Anthropic [4], the main concept used is the one of the residual stream viewed as a linear communication bus.

The authors affirm that all the layers in transformer act according to the following scheme (see Figure 12):

1. Apply a learnable linear transformation in order to "read" the information from the residual stream
2. Apply a nonlinear transformation like attention or the activation function for MLP blocks
3. Apply another linear transformation to the result and then sum back to the residual stream to "write" its contribution

In general this study led the way to various new ways of interpreting transformer, in particular this thesis takes inspiration from works that treat the residual stream as a dynamical system and individual tokens (nodes here) as points in evolution in a higher dimensional space.
The proposed tool in this thesis is to rebuild the whole "story" of the input through the layers until the final representation, and to inspect it through dimensionality reduction.

Concretely, let h^0 denote the embedded input (and positional encoding) through the procedure described in Section 3.3 and h^l the hidden state after layer l. For each layer the residual update is decomposed into specific layer increments:

$$\Delta h^l_{\text{local}}\ \Delta h^l_{\text{global}}\ \Delta h^l_{\text{MLP}} \tag{21}$$

corresponding respectively to the GAT layer, the Performer layer, and the feed-forward block (all taken before the residual addition).
Because all models share the same input embedding and a constant hidden width H, all the update signals are directly comparable across layers, branches, and architectures, enabling a unified, coherent analysis.

Once obtained the whole update "story", numerous analysis ways are possible, the first one that was performed in this thesis work is how the representation of a single "query" node moves through the residual stream relative to two prototype groups defined by the target variable.
For a given node q, the method compares its trajectory to two reference sets:

- the top $k\%$ of nodes by the true target y
- the bottom k% of nodes by the true target y

At each update step the distance (norm) between the query node and the reference sets centroids is recorded.
Two regimes are computed:

- dynamic distance: After every update (Performer/GAT/MLP), the query and all nodes in the two groups are updated using the recorded residual increments, and the high/low centroids are recomputed. The dynamic curves measure co-movement. They show whether the query is moving together with a reference set

- static distance: The final high/low centroids are first computed. The query is then "replayed" through the same sequence of updates, but distances are taken to the fixed final centroids. The static curves measure convergence of the query toward the high/low reference sets, independently of the sets drift.

Since the norm is directly computed from the hidden embedding, the plotted trajectory reflects the exact increments the model applies during inference, no dimensionality reduction happens here.
This way if the distance to the high group goes down while the distance to the low group grows, the model, when processing through the layers, is moving the query toward the manifold associated with high targets. Another piece of information that can be extracted from the inflection points is which component (global, local, or MLP) is responsible for the movement at each stage.
This analysis gives an idea of the residual stream dynamics: it shows when along the network depth the model begins to separate the query from the opposite prototype, and which component drives that separation.
Clear, monotone convergence of the query toward the high prototype, especially when attributable to a specific type of layer, supports the claim that the branch is functionally responsible for that prediction. Flat curves with later separation indicate that later layers carry the discriminative purpose, also it is possible to see when there are cases where attention only analysis would be incomplete.

The second analysis visualizes how node representations move through the network by projecting their full residual trajectories onto a low-dimensional subspace. Given a set of query nodes randomly extracted $Q \subset \{1, ..., N\}$, the procedure reconstructs, for each $q \in Q$, the sequence:

$$h_q^0, h_q^1, ..., h_q^T \tag{22}$$

Where T is total updates of the residual stream.
Successive states are obtained by adding the recorded residual updates in the given model order.
In parallel, two prototype centroids are computed for the top and bottom reference set like in the previous analysis. To obtain a readable visualization, a PCA model is fitted on the union of all trajectory points from the selected queries together with the reference centroid.
The trajectories are then shown as arrowed lines in the first two (or three) principal components, with the two centroids drawn as reference points.
A trajectory that progresses steadily toward the top centroid indicates that the model moves that node into the space associated with high targets values.
It is possible to see whether the model creates a clear separation space between high and low target, or if there are intermediate "directions".

Once again different types of layers contributes can be highlighted: for example long straight segments reflect a dominant component.

Finally, an examination the geometry of the final representation h^T by projecting all nodes embedding with PCA. The scatter in the first two components, colored by the target, provides a fast shot of how the model arranges the city at the end of the computation. A clear color gradient along one principal direction indicates that the target is well aligned with a dominant axis of variation in h^T. To make this assessment quantitative, the scores on PC1 (and PC2) have been correlated with the target and, when relevant, the PC1 direction with the readout vector w of the output head have been compared.
Agreement between these measures, high correlation of PC scores with the target and small angle between PC1 and w, supports the interpretation that the model has learned a largely linear separation in the final space.
This final-geometry view complements the trajectory analyses: while trajectories reveal when and through which components separation emerges, the PCA of h^T shows how that separation is ultimately organized in the representation used for prediction.

In conclusion, this framework is internal and mechanistic: it relies only on quantities produced during a forward pass (residual updates, and hidden states), rather than on any kind of proxy explanations.
In this sense it complements the Attention Graph: the latter explains which relations are focused on among nodes, while the residual stream analysis is able to show when and where those relations actually modify the representation and, ultimately, the prediction.

3.5 Implementation details

All models were trained using the Adam optimizer with an initial learning rate of 5×10^{-3} for the edge-level task and 1×10^{-3} for the node-level task.
A weight decay of 5×10^{-4} was applied to regularize the parameters.
All models used two layers, GAT model used an hidden dimension of 64, while the others of 32, this was done in order to keep the parameters number comparable ().
All models had four attention heads and dropout rates at 0.5. Random walk positional encodings of dimension 30 were added to the node features.
Models were trained for up to 1000 epochs with early stopping triggered by the validation loss.
All experiments were run on a Mac Book Air, with 8GB of RAM and chip M3.

Model	**N. of Parameters**
Hybrid: Layer-wise fusion	21507
Hybrid: Late fusion	21981
GAT model	17409
Performer model	17629
MLP	19085

Table 1: Number of trainable parameters per model (maximum input set)

Chapter 4
Results

This chapter presents the empirical results in two stages. First, the edge-level prediction task was used as a warm-up to the modeling framework: it served primarily to validate the data pipeline and compare architectures (GAT, Performer, and their Hybrid).
Accordingly, Section 4.1 emphasizes predictive performance and city generalization (results inside the domain, but on unseen test nodes and completely held out cities). The goal here is to have a first understanding regarding how well local, global and combined attentional models transfer across different urban street networks.
The second task, introduced in Section 4.2, delivers a deeper analysis, comparisons and also results regarding the interpretability components.
In summary, first task results focuses on the proposed models performance and generalization, second task involves also comparison with baseline models, and the interpretability framework.

4.1 Edge-level traffic-speed prediction

The evaluation of the edge-level models is done on five Italian cities used for training/ validation (Piacenza, Cremona, Pesaro, Trento, Rimini) and assess generalization on two held-out cities (Verona , Ferrara) as described in Section 3.2.1. Targets are the synthetic edge speeds derived from limit speeds modulated by betweenness centrality, traffic lights, and city center buffer (Equation 13).
The optimization is done on mean squared error (MSE) on edge splits (70/15/15). The reported test performance and training details follow what has been described in Section 3.
The models compared here are the one presented in Section 3: GAT model, Performer model, and the two hybrids variants (late fusion and layer-wise).

4.1.1 Quantitative results

The metric used for this task evaluation are the one presented in Section 3.3.5.
Cities are denominated according to the respective province code:

Code	**City**
PC	Piacenza
CR	Cremona
PU	Pesaro
TN	Trento
RN	Rimini
VR	Verona
FE	Ferrara

Table 2: Code and respective city

GAT model results:

City	MSE	RMSE	MAE
PC	48.19	6.94	4.51
CR	49.98	7.07	4.68
PU	56.26	7.50	4.97
TN	55.63	7.46	5.35
RN	42.52	6.52	4.31

Mean	50.52	7.10	4.76
Std dev	5.67	0.40	0.41

Performer model results:

City	MSE	RMSE	MAE
PC	49.64	7.05	4.48
CR	48.71	6.98	4.73
PU	56.99	7.55	5.08
TN	53.70	7.33	5.31
RN	45.01	6.71	4.48

Mean	50.81	7.12	4.82
Std dev	4.64	0.32	0.37

Hybrid model (Late fusion) results:

City	MSE	RMSE	MAE
PC	51.94	7.21	4.36
CR	53.13	7.29	4.87
PU	63.13	7.95	5.53
TN	57.41	7.58	5.41
RN	48.72	6.98	4.87

Mean	54.87	7.40	5.01
Std dev	5.57	0.37	0.47

Hybrid model (Layer-wise fusion) results:

City	MSE	RMSE	MAE
PC	46.37	6.81	4.27
CR	45.64	6.76	4.44
PU	58.13	7.62	5.27
TN	54.76	7.40	5.27
RN	44.98	6.71	4.45

Mean	49.98	7.06	4.74
Std dev	6.04	0.42	0.49

Table 3: Edge-level task results on test sets of cities used for training (transductive learning.)

GAT model results:

City	MSE	RMSE	MAE
VR	66.43	8.15	6.38
FE	81.23	9.01	6.16
Mean	73.83	8.58	6.27
Std dev	7.40	0.43	0.11

Performer model results:

City	MSE	RMSE	MAE
VR	73.10	8.55	6.55
FE	86.05	9.28	6.46
Mean	79.57	8.91	6.51
Std dev	6.47	0.36	0.04

Hybrid model (Late fusion) results:

City	MSE	RMSE	MAE
VR	66.41	8.15	6.15
FE	82.94	9.11	6.34
Mean	74.67	8.63	6.24
Std dev	8.27	0.48	0.09

Hybrid model (Layer-wise fusion) results:

City	MSE	RMSE	MAE
VR	61.88	7.87	6.07
FE	85.54	9.25	6.35
Mean	73.71	8.56	6.21
Std dev	11.83	0.69	0.14

Table 4: Edge-level task results on generalization on held out cities (inductive learning).

Overall, Table 3 summarizes transductive performance for this specific task, on the five training cities. The layer-wise hybrid attains the best average errors (lowest MSE/RMSE/MAE), followed closely by GAT. Performer is competitive but slightly worse on average, while late fusion lags behind.
Trento is an interesting example and is mixed: Performer reduces MSE relative to GAT, whereas the layer-wise hybrid obtains the lowest MAE. Showing that pure global is probably catching some outlier values, thus reducing the quadratic error.
Taken together, these results indicate that for this proxy speed target, largely driven by local features, local attention already provides a strong baseline, and while crossing local with global updates can yield consistent, incremental gains, it may become more advantageous on tasks with stronger global dependencies.

Table 4 shows results across the two held out cities, the GAT model achieves strong performances, while Performer yields the weakest averages, indicating that pure global attention is not sufficient for generalizing this proxy speed target under city shift, which is a reasonable results since no long range rules are incorporated in the creation of the target. GAT and Layer-wise hybrid are essentially tied with the first performing slightly better on absolute errors indicating that the interleaved local and global updates reduce a few large errors (helping MSE) at the cost of slightly larger typical errors (hurting MAE). Instead, GAT keeps typical errors smaller but incurs occasional larger residuals that inflate MSE. This will be seen in the next section on the qualitative analysis.

When transferring to unseen cities, local attention GAT, generalizes most reliably for this proxy speed target, while global attention and hybrids add value in being more able to catch outliers, but fail on to reach the same levels of GAT on typical nodes. Table 4 shows results across the two held-out cities. The layer-wise hybrid attains the best averages but is almost equal to the GAT. The late-fusion hybrid, which had the worst results in transductive learning, is intermediate on MSE and even slightly better than GAT on MAE, while Performer is the weakest on all three metrics. The layer-wise hybrid stronger averages come with higher between-city variance, indicating sensitivity to topology shift. GAT is more stable.
These patterns are consistent with what said for the proxy target. Pure global attention is not sufficient on its own, while interleaving local and global updates can reduce large errors in some cities but may over-adjust in others. This trade-off, smaller typical errors for GAT versus better outlier control for the hybrid, will emerge clearly in the qualitative analysis.

4.1.2 Qualitative results

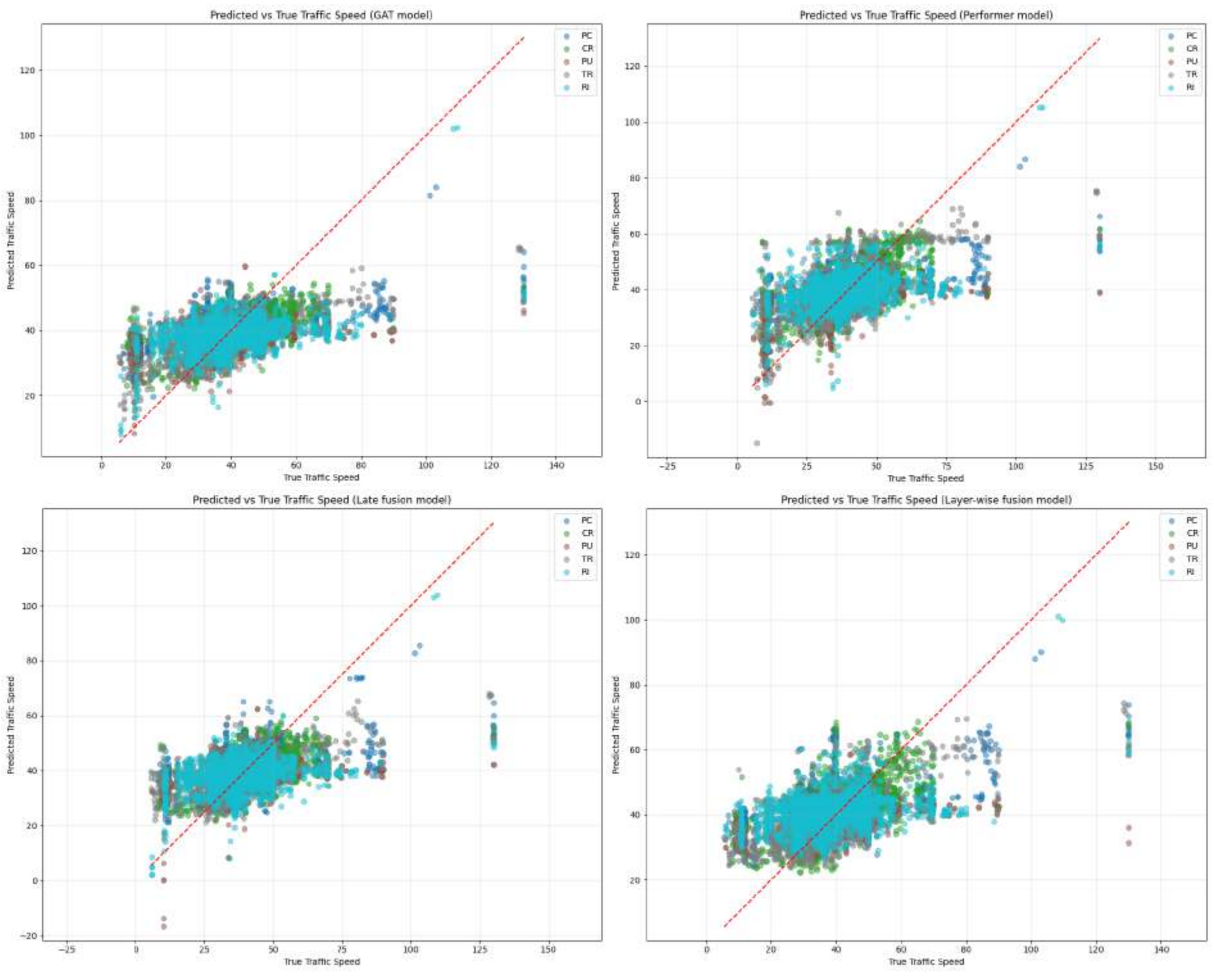


Figure 13: scattering of predicted vs true target

The scatter plots of predicted vs. true edge speeds (Figure 13) reveal consistent patterns across cities and models.

The vast majority of predictions concentrate in a narrow band ($\approx$ 30– 60 km/h) with a visible saturation around $\sim$ 60 km/h. This is evident in the systematic underestimation of high speed edges ($\approx$ 80– 130 km/h).

Error grows with the true speed, while the low-speed regime is in comparison more tight.

The city color overlay shows error which are not completely different across cities, suggesting that the learned bias is largely model/feature-driven rather than city specific, consistent with what seen in the quantitative generalization results (Table 4).

GAT model has a decent alignment to the identity line, typical predictions are calibrated.

Layer-wise hybrid shows slightly broader dynamic range than the other models.

This aligns with the quantitative trade-off with this model showing the lowest errors.

Late fusion hybrid and Performer model include some unrealistic negative predictions, suggesting that the late head is less calibrated.

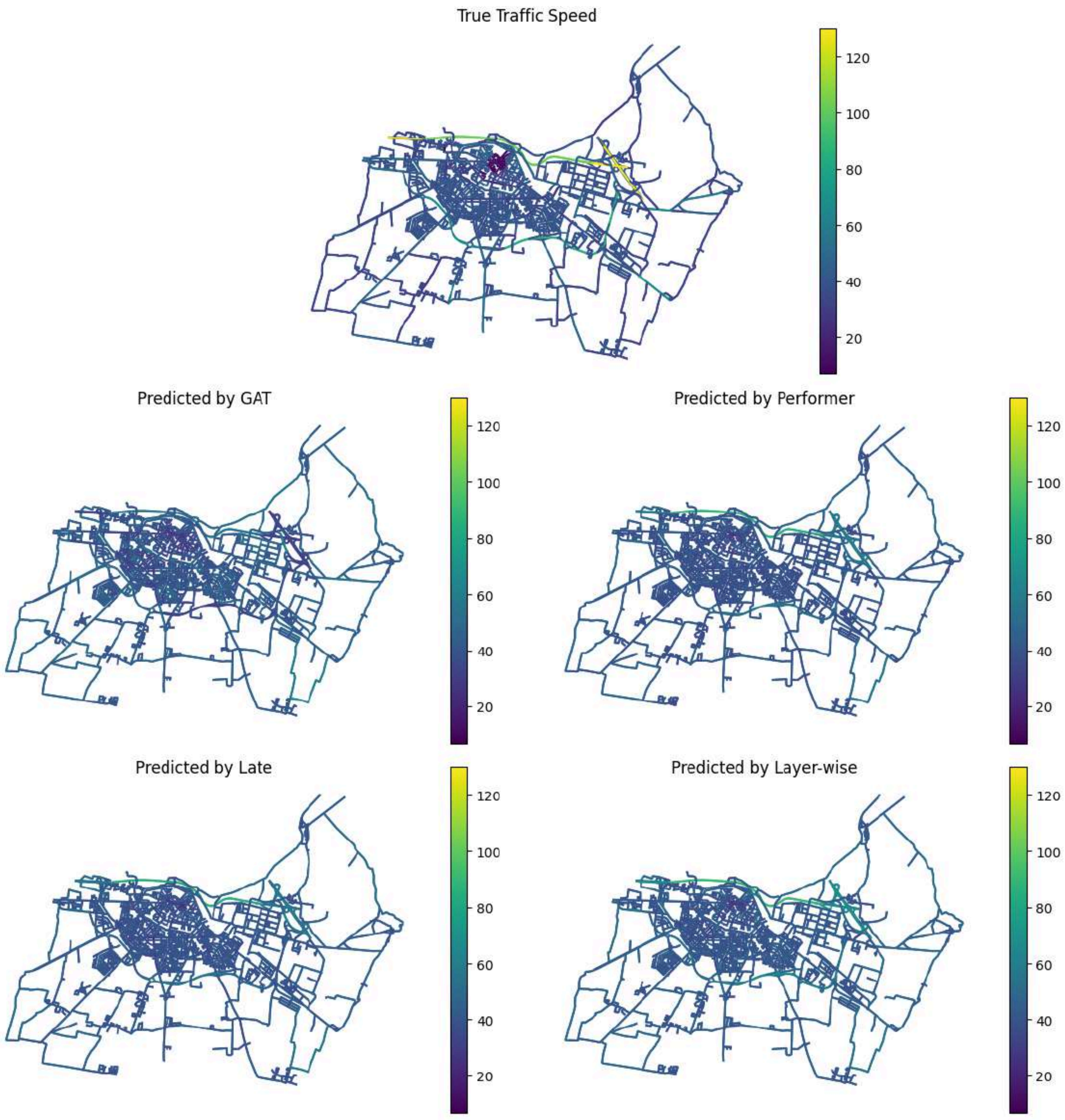


Figure 14: Predictions on city of Piacenza of all the proposed models

In Figure 14, all models track the main urban regime reasonably well. Late fusion hybrid model has the overall better results, qualitatively it looks like the combination of local and global contribution in this case helped distinguish the "city center" with additional speed deficit.

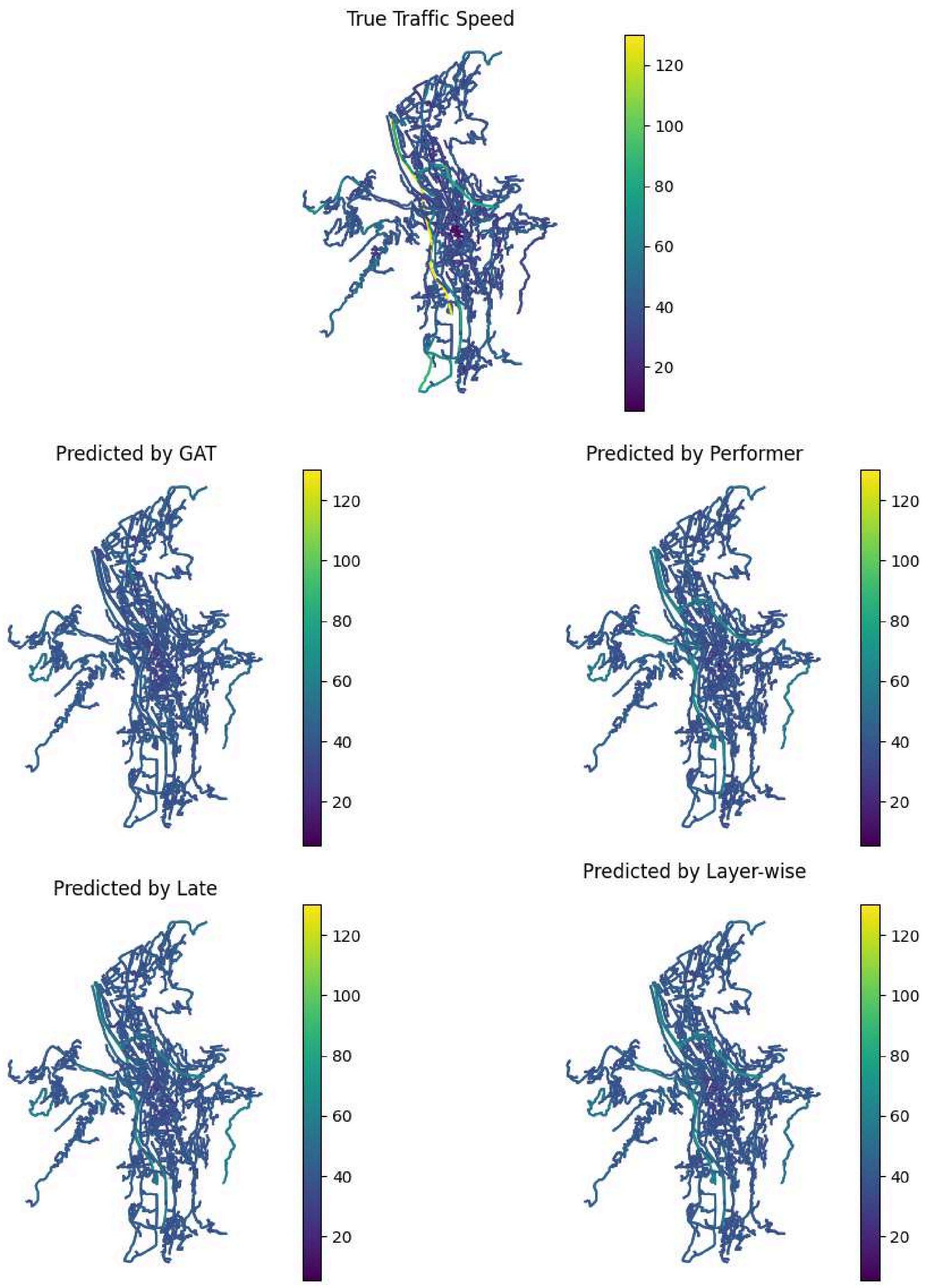


Figure 15: Predictions on city of Trento of all the proposed models

Performer model and hybrids model slightly better the mid to high speeds than GAT here, consistent with the good MSE in this structurally different city. Clearly the long motorway crossing the city is better captured by the global attentional models.

The similar plots across models and the compression toward the dominant speed band and underestimation of the very specific high-speed edges is consistent with:
- label imbalance, since there are few motorway segments,
- MSE used as loss function in training, which favors the most dense region of the target distribution.

After all, the qualitative evidence make a stronger case for the already stated intuitions: GAT already has the most well-calibrated typical predictions, but, layer-wise fusion offers selective improvements on outliers. Late fusion and Performer remain more compressed toward the modal target but still show to be able to, sometimes, model outliers behaviors.

4.2 Node level greenness prediction

This section evaluates the proposed attention architectures on a node-level regression task: predicting the Green View index for nodes, which are H3 cells, using the data sources and preprocessing described in Section 3.
The results report two things:
- comparison between local, global, and hybrid attentional models against non graph baselines
- study how performance scales when progressively enriching the input features.

The results here provide the quantitative basis for the interpretability analysis that follow in (Section 4.3)

Experiments use cities and preprocess data according to Section 3.2.2, with transductive evaluation on the within city test split and inductive evaluation on an held out city, following the problem definition, training protocol and implementation described in Section 3.
The comparison includes the GAT model (local attention), the Performer model (global attention), and the two hybrids (late fusion and layer-wise fusion), alongside simple baselines defined in Section 3.

To better understand the effect of the information given in input, models are trained across an increasing set of features:
- population intensity (`PopSum`)
- points-of-interest (`POI`)
- Number of buildings in a given node (`Building Count`)
- sky-view information (`Sky View Mean`)

And all intermediate combinations, creating an incremental training grid used to compare learning curves across architectures and baselines.

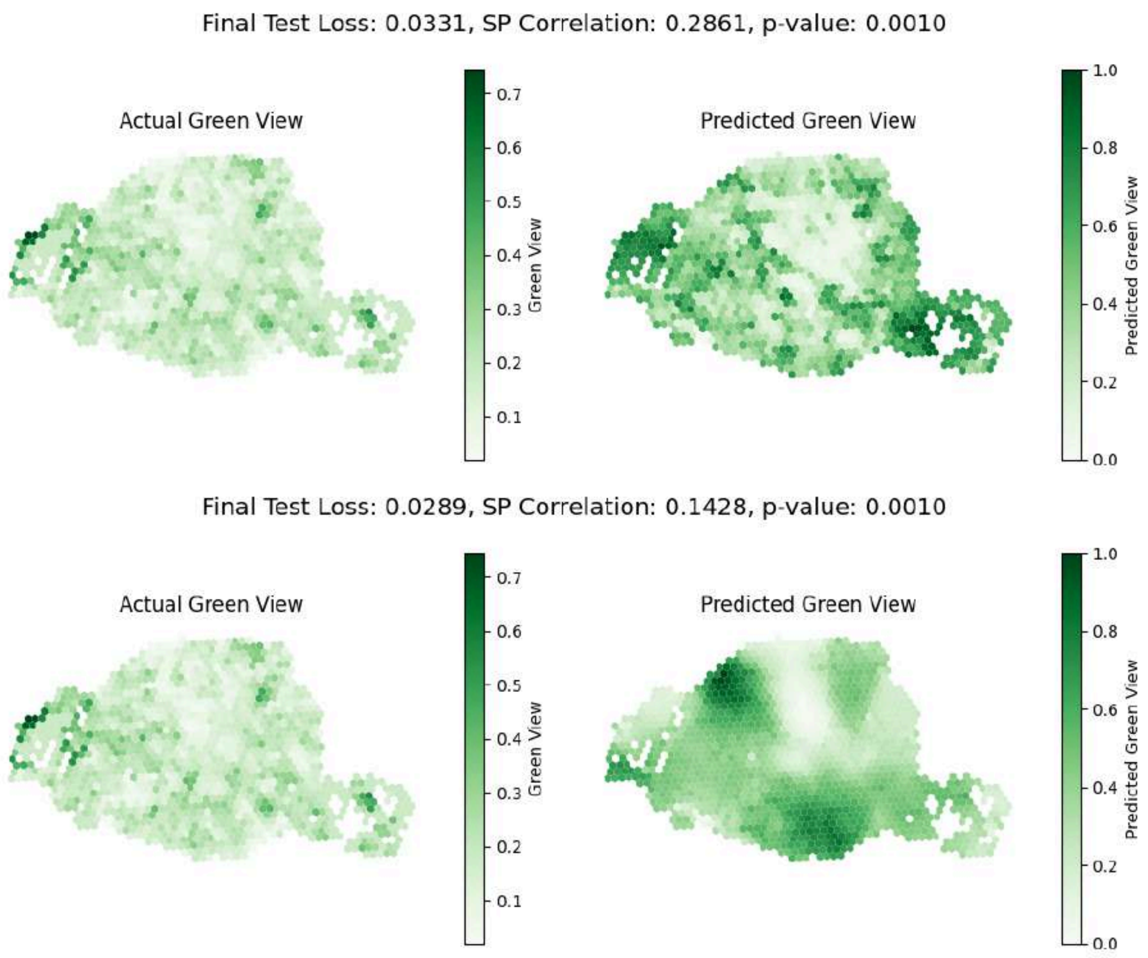


Figure 16: **Top**: prediction with higher MSE but stronger spatial agreement with the ground truth. **Bottom**: prediction with lower MSE yet weaker spatial alignment

The primary error metrics is MSE. Spatial Pearson correlation between real GVI and predicted one, which quantifies how similarly two spatial variables change over space, is introduced in order to know which model respects the spatial structure of the graph most closely.
This metric was introduced because it often happened that, a qualitatively completely different from the original, predicted graph, had a lower loss, especially on the hold out city (Figure 16) Before the results in Figure 17 are reported all the correlations between the four input features and the target variable in the city of Paris, which is the one used as holdout city, used for generalization.

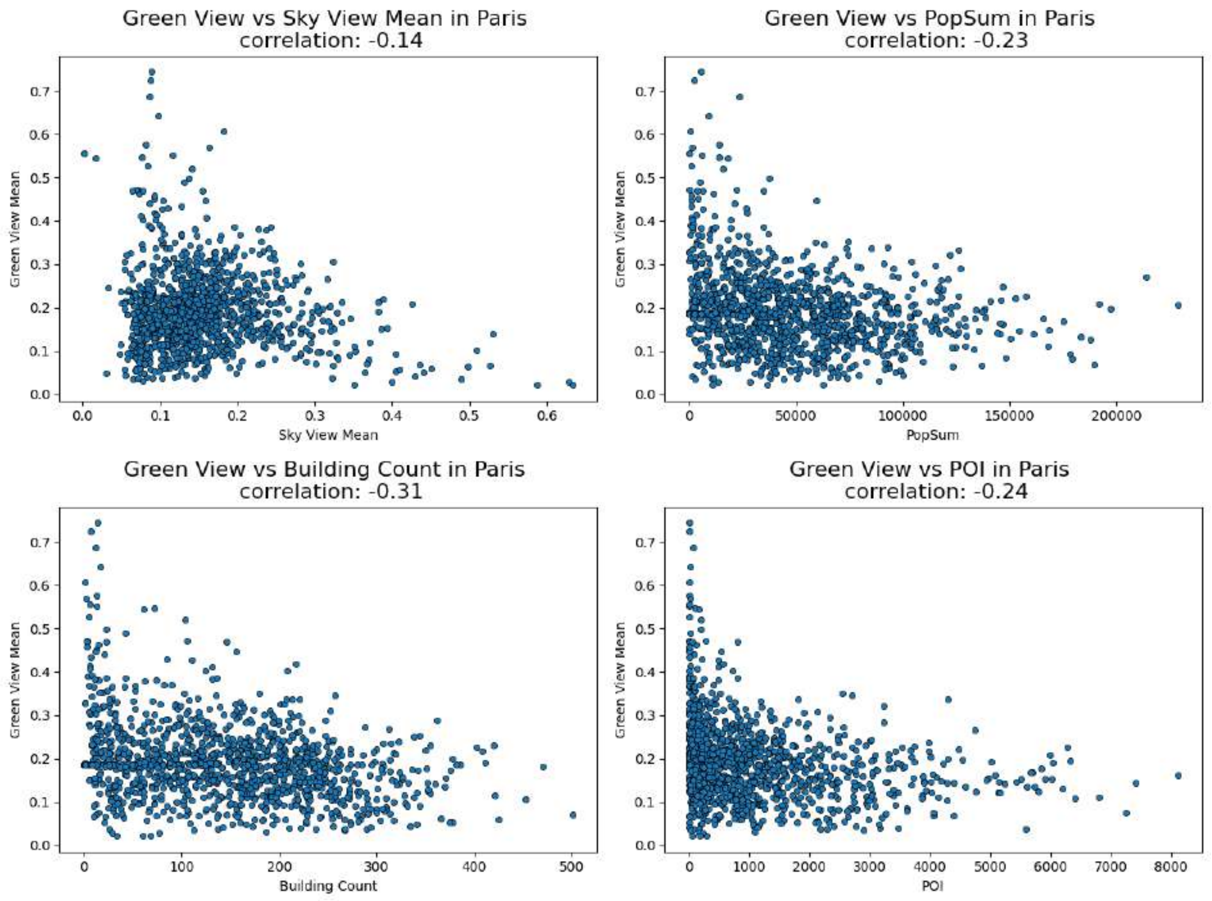


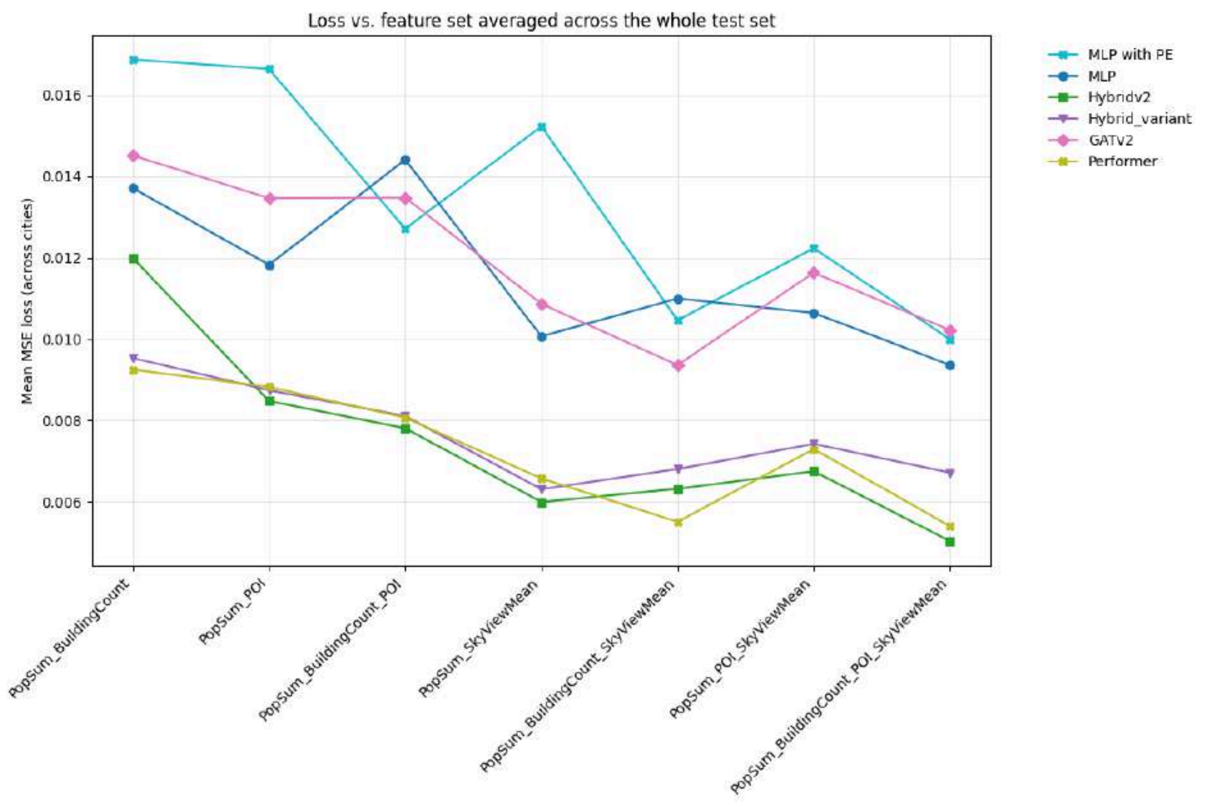
Figure 17: Scattering of input features vs true target

All features show weak to moderate negative correlations, with Building Count being the strongest. However, these low absolute values confirm that none of the features alone explains the target well. In the quantitative results in Figure 18, `Sky View Mean` emerged as one of the most if not the most influential feature despite its low linear correlation, indicating the presence of non linear effects and interactions that simple scatter plots cannot reveal.

Quantitative results of the trained models are summarized by the overview figures:

1. in domain averages of test loss and spatial correlation across feature sets for transductive learning
2. plots of the results for the held out city of every model to compare how each model generalizes.

4.2.1 In domain results



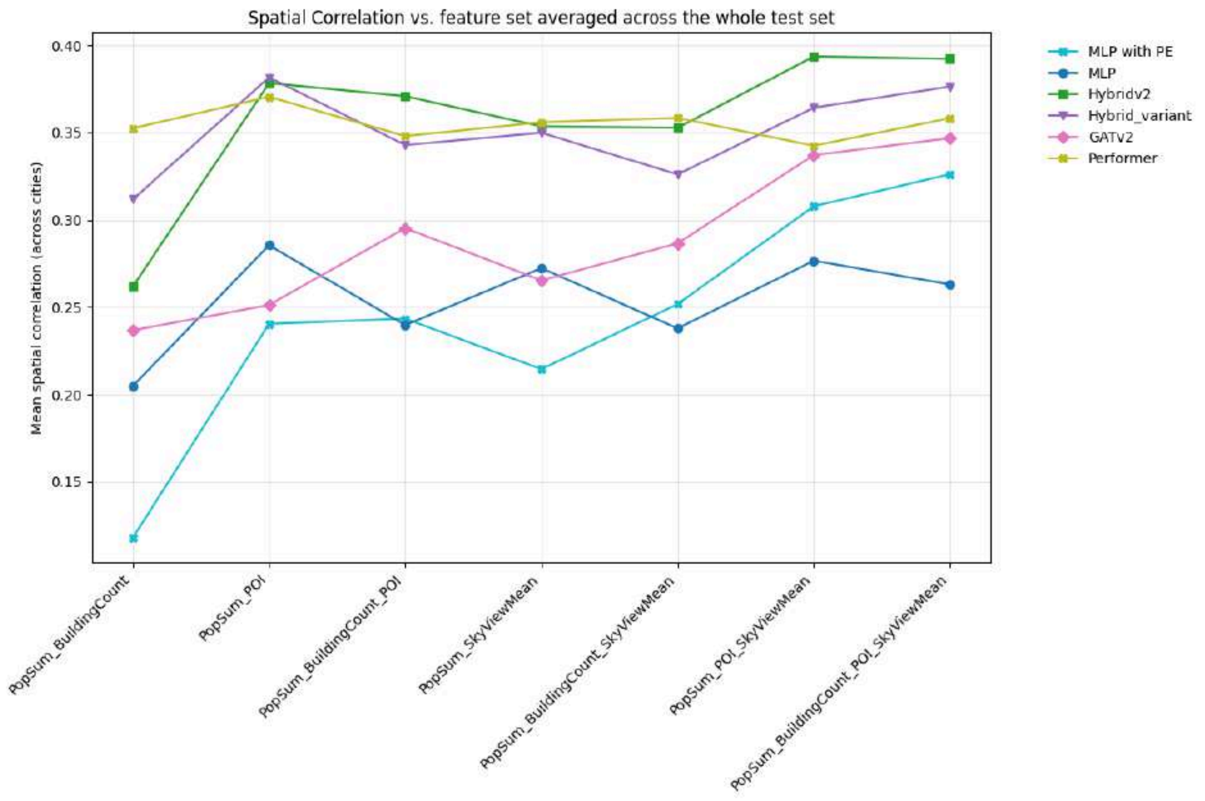


Figure 18: Loss (**Top**) and Spatial correlation (**Bottom**) as feature set is enriched, averaged across all the test nodes of the in domain cities.

In Figure 18, it is possible to see how the error (MSE) decreases as input is enriched, with the largest gains appearing once `Sky View Mean` is introduced.

Unlike in the first task (Section 4.2), the two hybrid models clearly lead in efficiency, followed by Performer, then GAT.
MLP with PE in terms of loss is even worse then its graph “uninformed” counterpart, and both are comparable with GAT but behind the three best graph based models. The curves also show diminishing returns after adding `Sky View Mean`, suggesting that most improvements are given by the combination of `PopSum` + `Sky View Mean` and that `POI/Building Count` mainly provide secondary refinements. There are some non monotonicities (e.g. GAT around `PopSum_BuildingCount_POI`), that can indicate interaction effects between features and inductive bias.
Spatial alignment between predictions and ground truth also increases with feature richness. The hybrid models achieve the highest spatial correlation and improve steadily as inputs are augmented. These results suggest that combining local and global attention best preserves the structure of the original graphs.
Performer maintains moderately high and stable alignment, while GAT model benefits from added features but remains below the hybrids. MLP with PE attains higher spatial correlation than a plain MLP, and like for loss achieves comparable results with pure GAT model, yet both baselines are behind graph models.

On the node-level task, enriching inputs, especially with Sky View Mean, benefits all models. However, hybrid attention consistently delivers the best combination of low loss and high spatial correlation, suggesting a good use of both local structure and long range signals encoded in the features.

4.2.2 Out of domain results

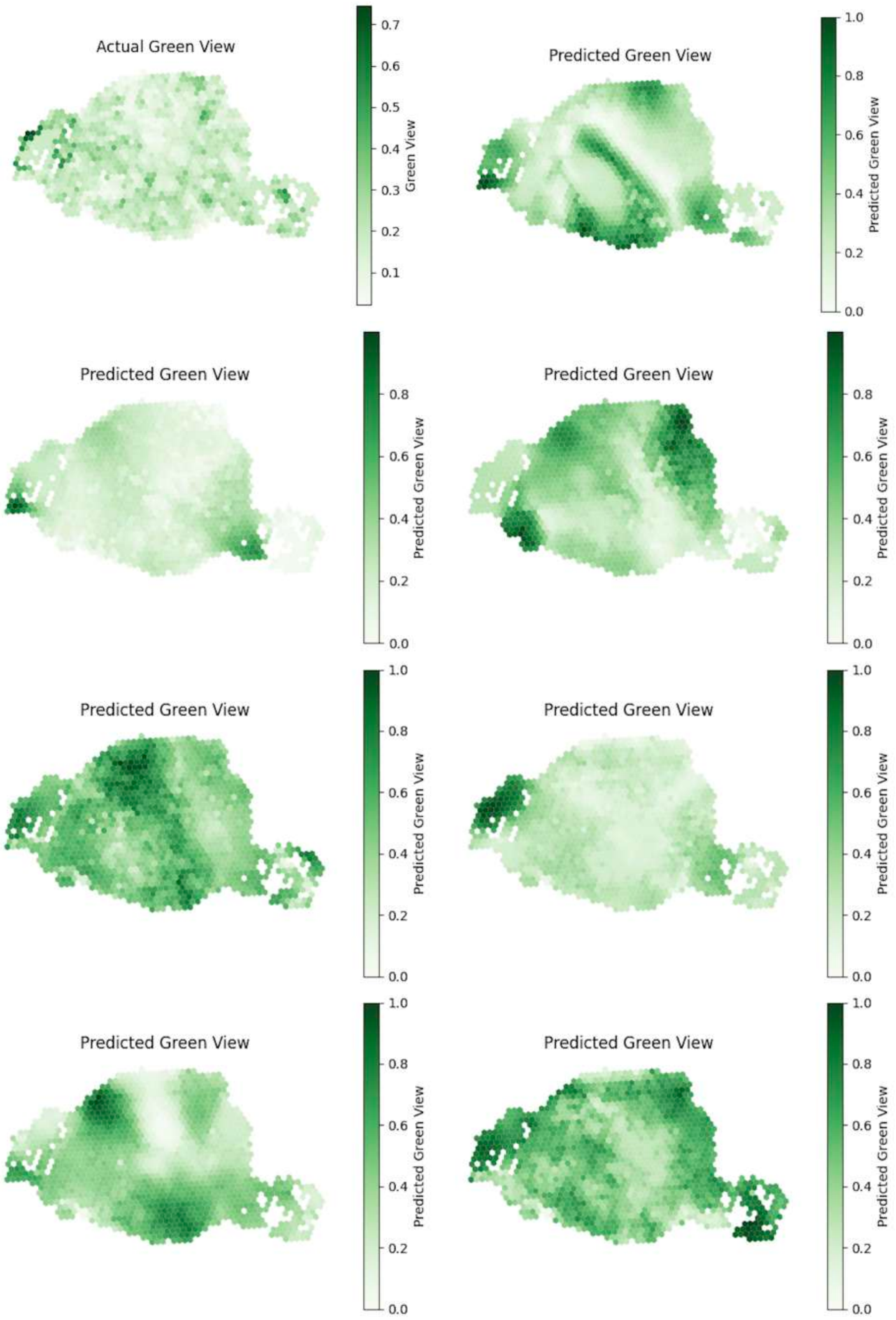


Figure 19: Predictions of MLP model on held out city, from ground truth, to all input features, in increasing order of inputs (from left to right and from up to down)

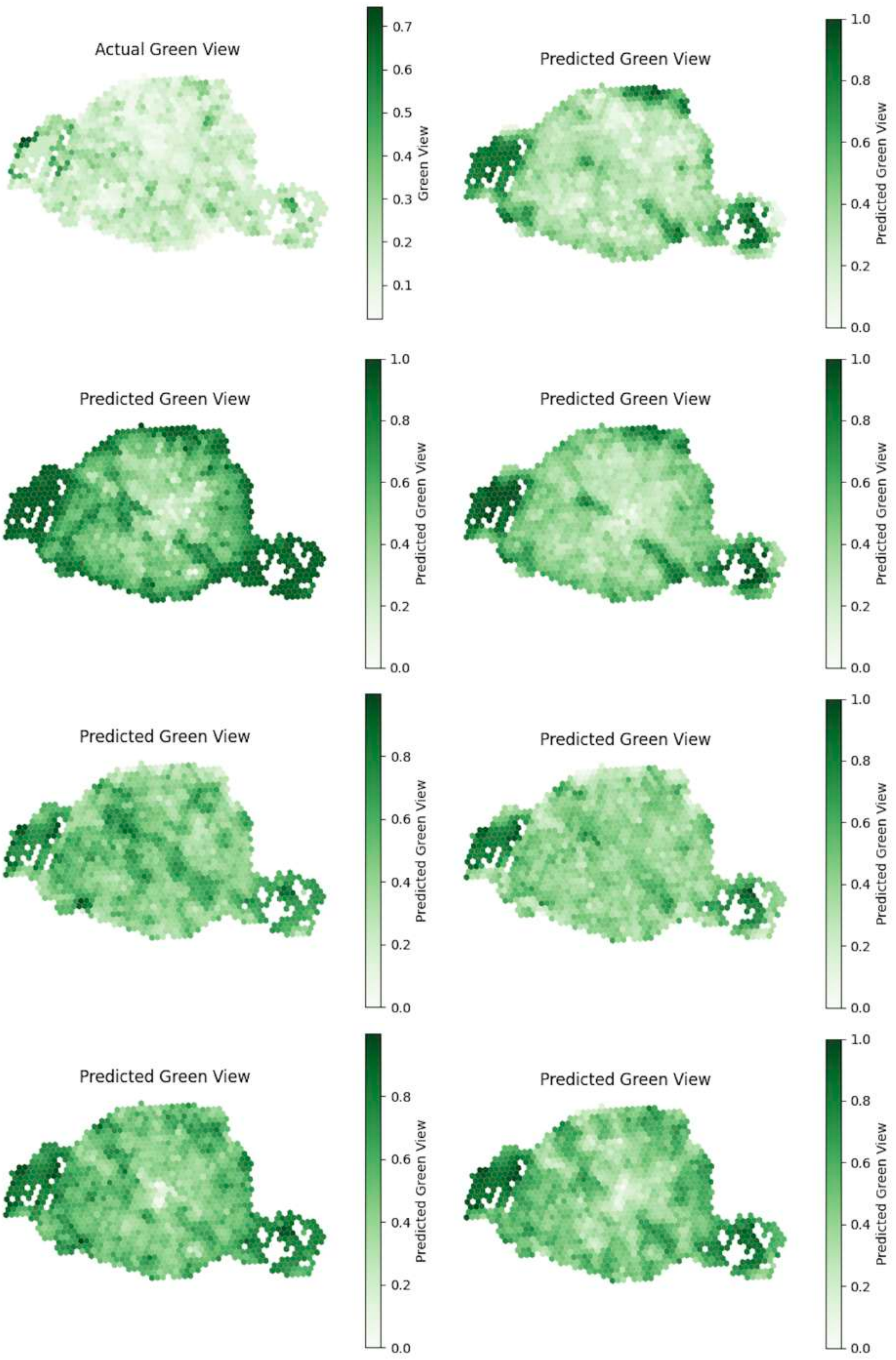


Figure 20: Predictions of GAT model on held out city, from ground truth, to all input features, in increasing order of inputs (from left to right and from up to down)

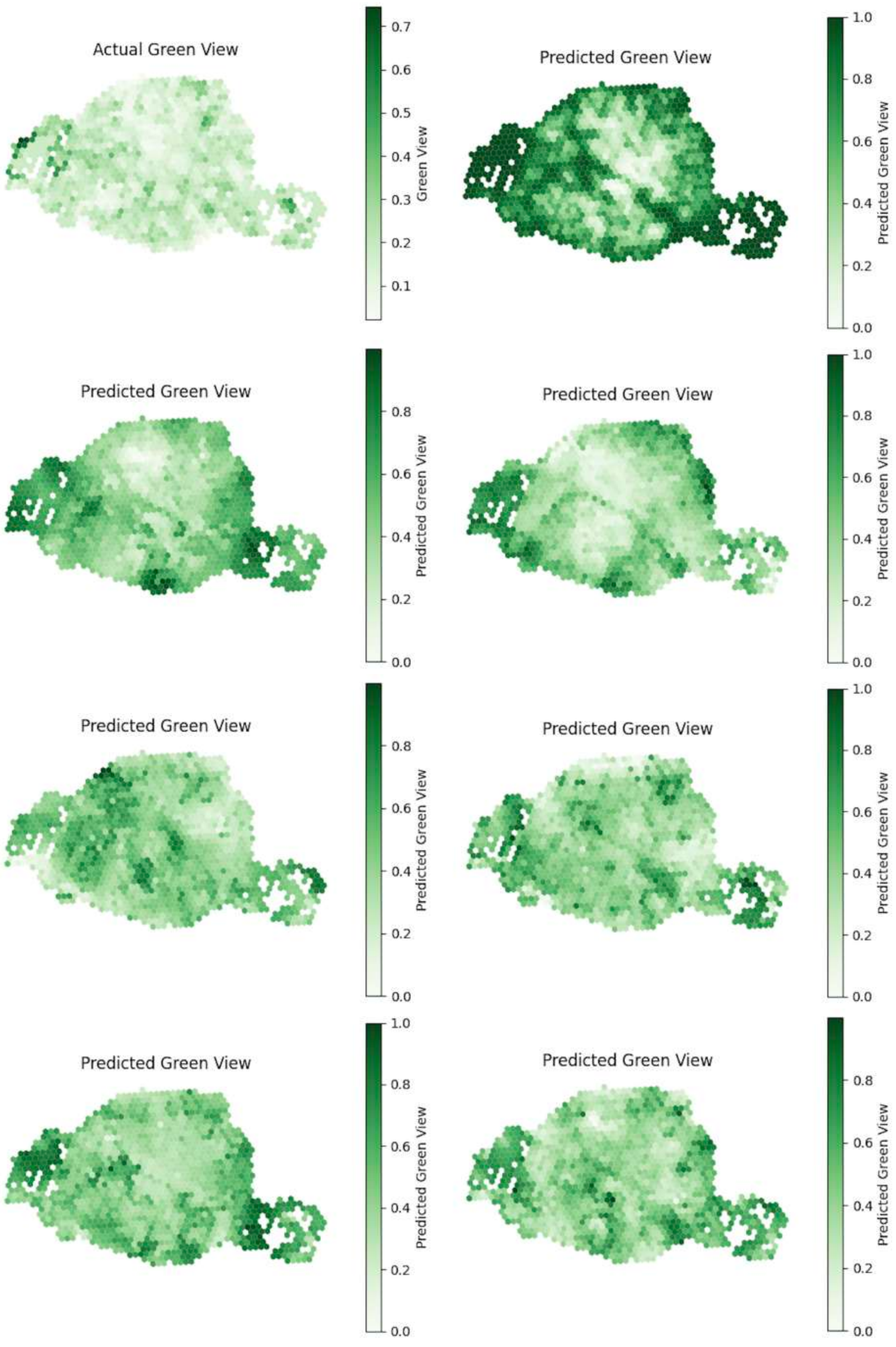


Figure 21: Predictions of Performer model on held out city, from ground truth, to all input features, in increasing order of inputs (from left to right and from up to down)

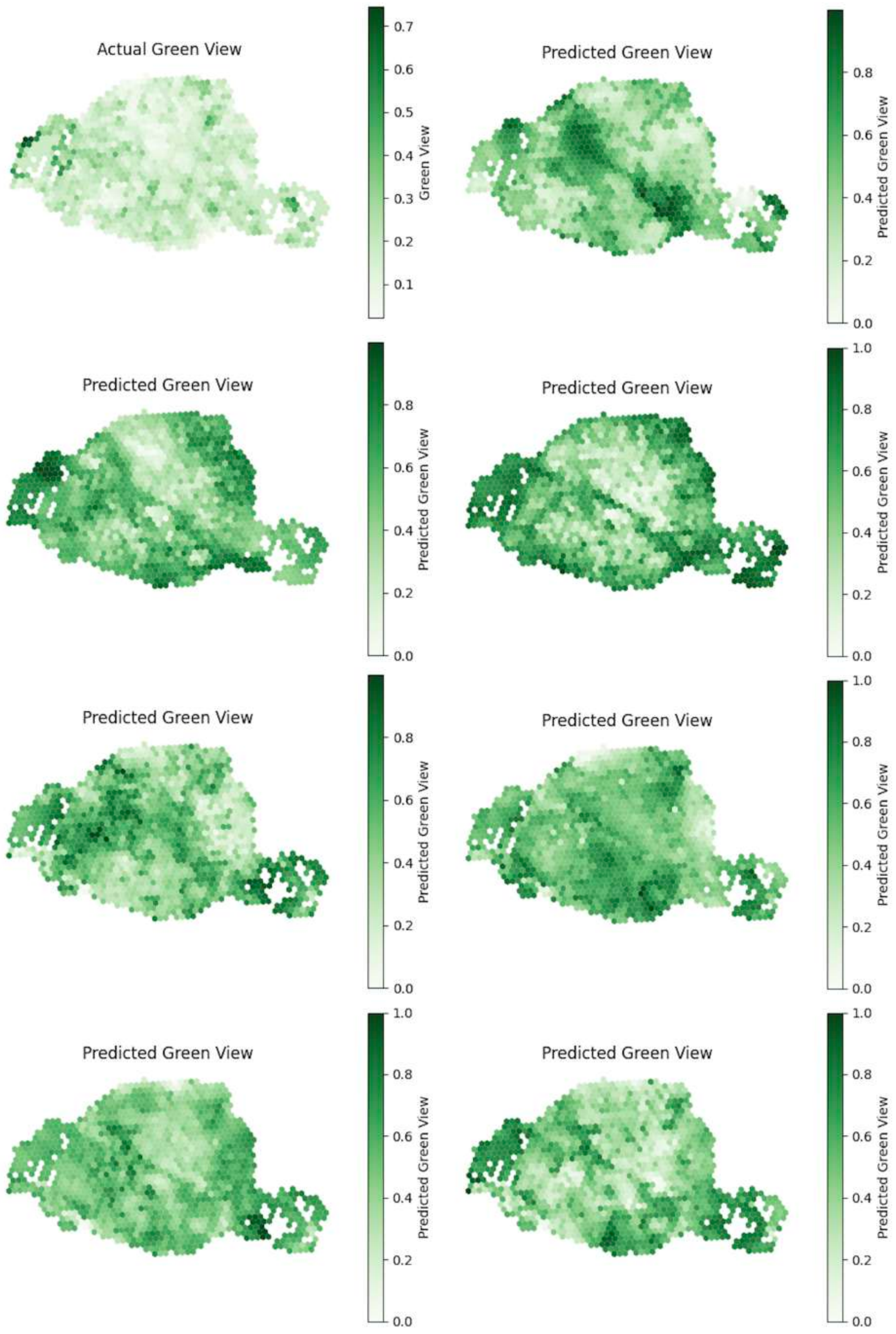


Figure 22: Predictions of late fusion hybrid model on held out city, from ground truth, to all input features, in increasing order of inputs (from left to right and from up to down)

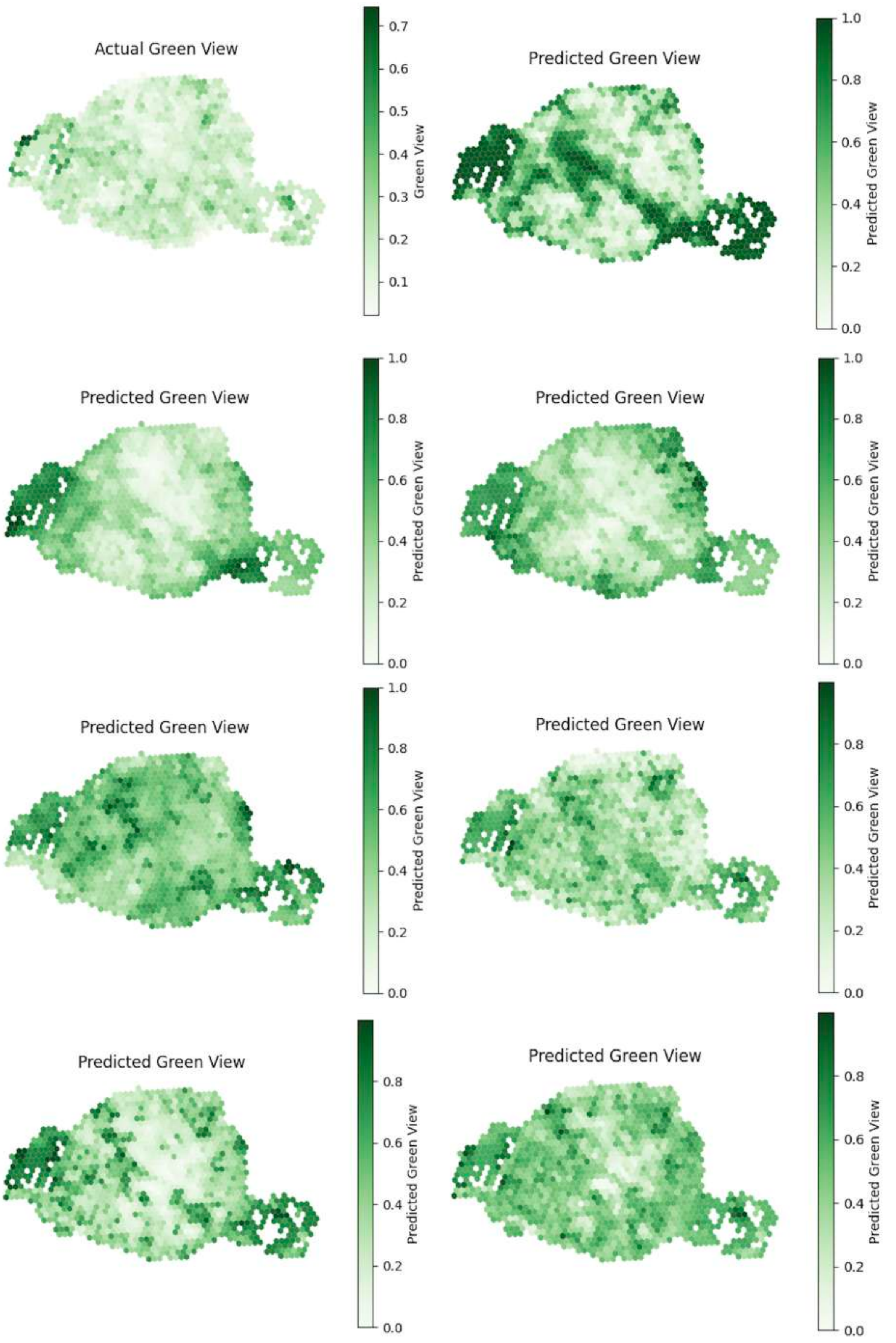


Figure 23: Predictions of layer-wise fusion hybrid model on held out city, from ground truth, to all input features, in increasing order of inputs (from left to right and from up to down)

The preceding panels show the results on inductive generalization, since Paris was unseen at training time.
Across feature levels, every model improves as input is enriched with new features, as for transductive learning, and especially in the first four images in the panel, then the last three are often comparable between the other version of the same model. The layer wise hybrid generalizes best, maps show clear separation between green and non green areas, with strong contrast and limited spurious connections.

The late-fusion hybrid is close but visibly more diffused. Boundaries are smoother and nearby zones are sometimes connected while they are separate in the ground truth. The Performer model benefits from global context, contrast is high and the two main green masses on opposite sides of the city are recovered, but there are frequent false positives in non-green areas and occasional bridging across neighborhoods. GAT model creates lower in contrast and more diffuse maps: large parks are usually detected and many secondary green areas are recovered, but the separation between green and non green areas is less sharp. The MLP baseline (with positional encoding, which was the best one) has the weakest spatial organization, forming green regions that lack correspondence with the reference map. It misses the two main parks in most feature settings, improving only in the full feature panel.

Overall, hybrids, more so the layer wise variant, show the most consistent performance on the unseen city. The results indicate that global context helps creating distinct areas, as seen with Performer model, and that combining global and local attention produces even better results. GAT model identifies the main areas of interest better then Performer but tends to spread predictions across space. MLP baseline is the least aligned with the underlying graph structure, which reflects the absence of graph structure injected in the model.

A final remark is that this node level task, which unlike the edge level one does not come with a synthetic target, already shows benefits from combining local and global attention. At the same time, it does not present clear long range dependencies, since the main signal remains local. Larger gains are therefore expected on problems where information must propagate across many steps, for example predictions shaped by metro lines, rails, or other fast travel modes that connect distant areas of the city.

4.3 Interpretability results

This section reports some interpretability outputs for the node level task.
The goal is to verify that the Attention graph and the residual stream tools behave as expected, and illustrate what each tool reveals.

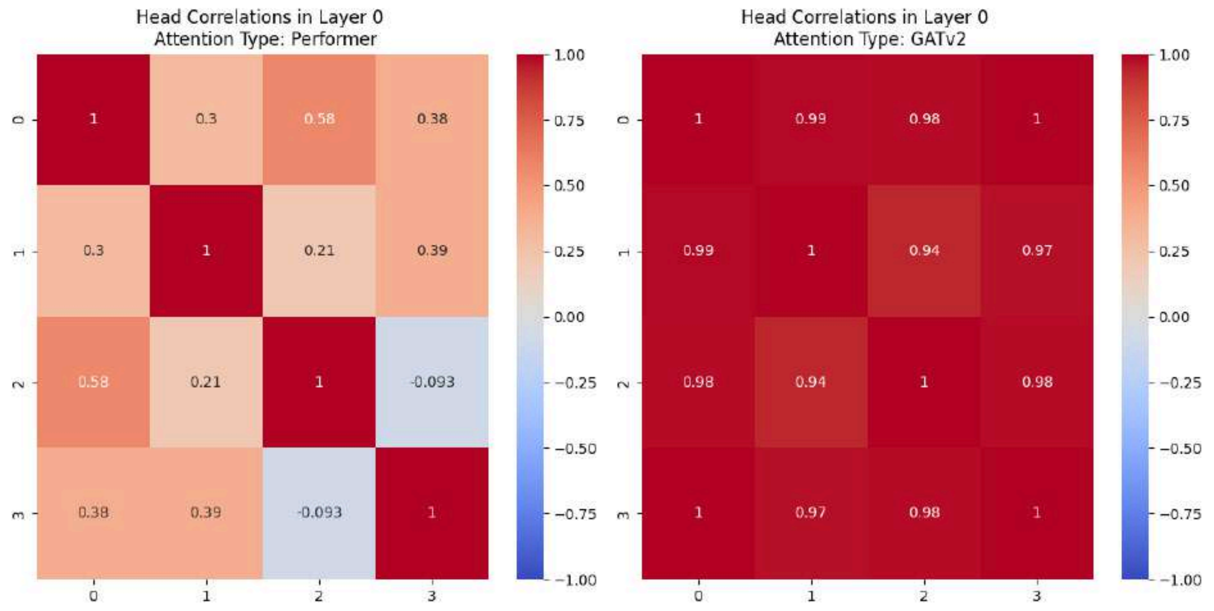


Figure 24: Head head Pearson correlations within a representative layer for GAT and Hybrid (Performer layer of the layer-wise fusion hybrid).

Head wise attention maps (Figure 24) are strongly and consistently positively correlated within GAT layers, indicating complementary learning. In contrast, Performer layers exhibit a mixed correlation structure, including near zero and negative head pairs.

Studying the attention layers inside the layer wise fusion model across all cities, it came out that Performer layers have an average correlation between heads of $\approx 31\%$ in layer 0 which decrease to $\approx 10\%$ in layer 1 this indicates an increasing head specialization, deeper heads capture more diverse attention patterns. Respective GAT layers have $\approx 99\%$ average correlation.

Such diversification supports the proposed correlation aware intra-layer aggregation rule (seen in Equation 19).

Regarding the attention graph, first it was checked that all the produced attention matrices are row-stochastic, checking that each row sums to one.

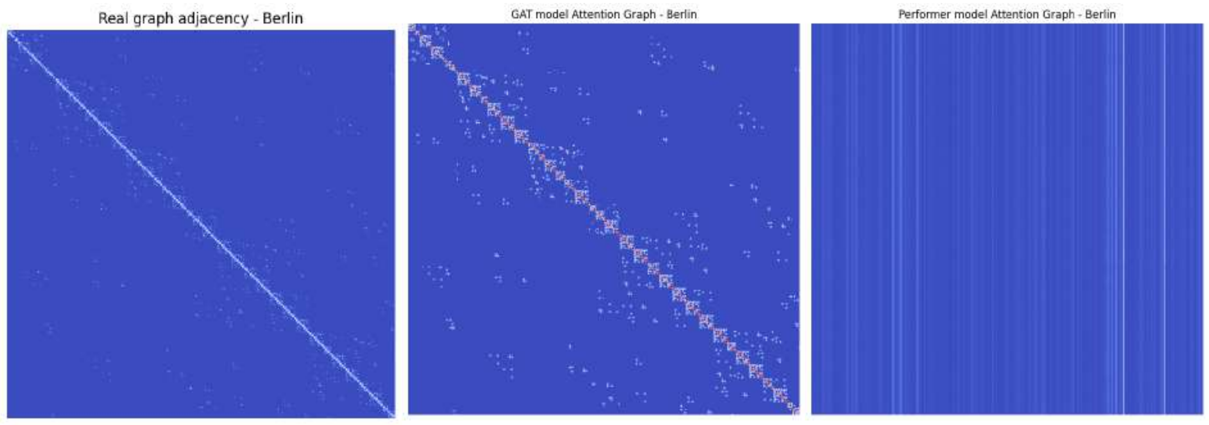


Figure 25:
Left: Graph connectivity, white edges are the one existing in the real graph
Center: GAT model attention graph, white link have non zero attention, red link are the strongest
Right: Performer model attention graph, more intense white means higher attention link

Then it was checked the general structure of the Attention graph: the ones produced by local or global attentional model should look completely different, as shown by Figure 25, GAT model produces a sparse matrix, following the exact structure of the original graph, and increasing the connectivity, by modelling at most two hops (since the model has two layers), while the Performer model has a completely different and dense structure, since it does not consider the original graph connectivity at all and all nodes attend to each other. In the figure it is possible to see how the attention graph assumes the typical structure described by [3], where some nodes are attended by all the other nodes (lighter columns), and take the role of "reference nodes".

Beyond the qualitative checks, the distribution of link lengths in term of hop distance was computed on attention graphs, filtered to retain, for each node, the top 25 outgoing targets. The resulting curves, although city dependent, consistently fall into three distinct regimes (Figure 26):

- GAT model: three clear steps at hop distances $0, 1, 2$, reflecting the depth constraint: with L layers the model can propagate information at most L hops away.
- Performer model: an approximately bell shaped profile centered at a larger, city specific distance, with a long tail of city spanning links.
- Hybrid models: a mixed profile with a dominant peak at short distances and a continuous long distance tail, consistent with the combination of local and global attention.

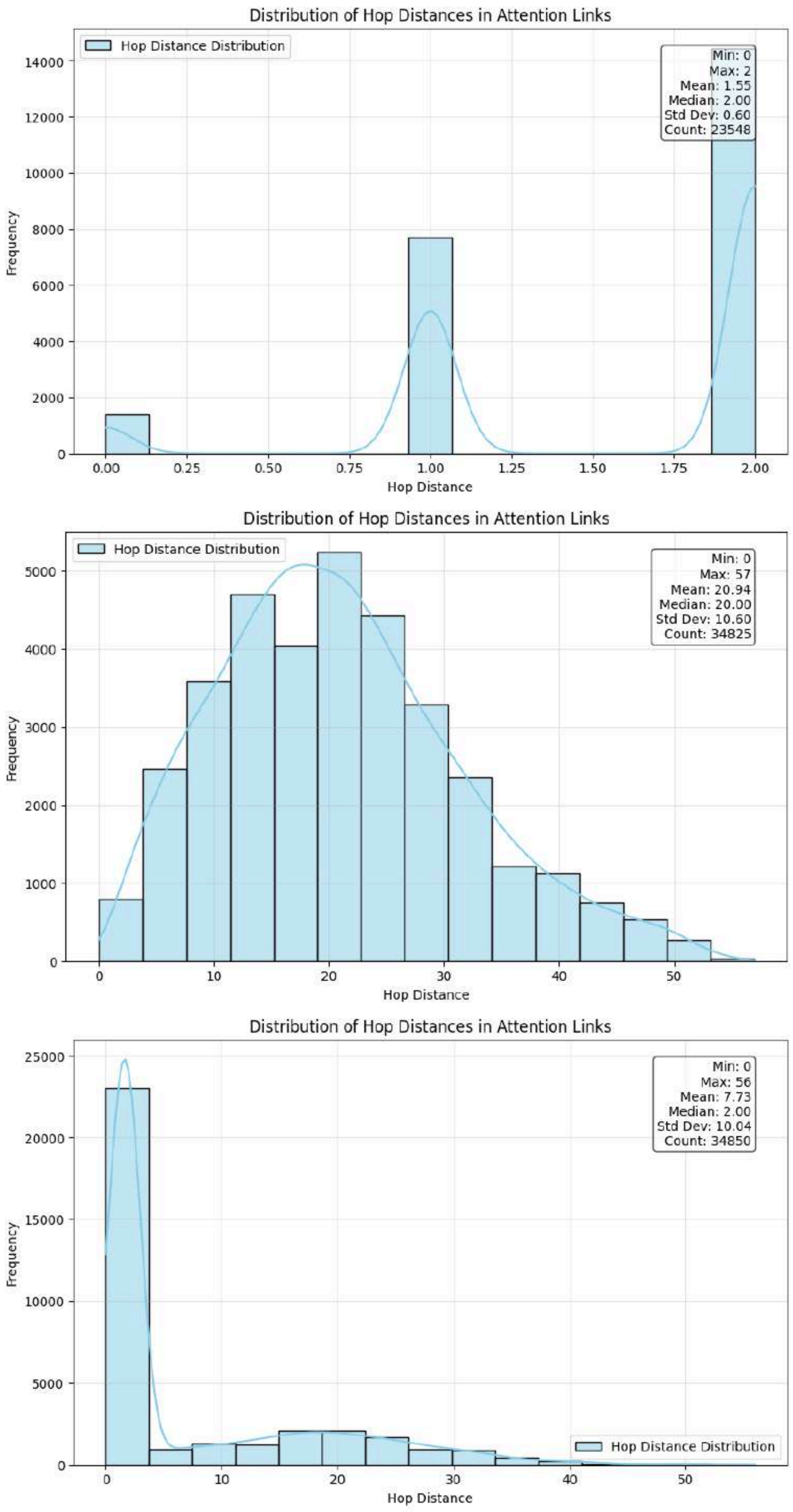


Figure 26: **Top**: GAT Attention graph link length distribution
Center: Performer model Attention graph link length
Bottom: Hybrid model Attention graph link length

Once computed, the Attention graph is a large matrix that is most useful for general diagnostics rather than specific explanations. For example it can reveal whether a global attention model forms meaningful links across the network or concentrates on a small set of reference nodes that act as attractors. In the experiments here (Figure 27), the global attention concentrates on one dominant node, meaning many nodes have their strongest attention to the same target and predictions are effectively compared against that single node. This concentration reduces the diversity of information learned and, for this task, might cause weaker generalization. This phenomenon often happens in unconstrained global attention [3].
In contrast, the same graph but from a local model shows a completely different situation, Attention graph decomposes the city into many small, connected components, indicating distributed propagation with multiple micro attractors.

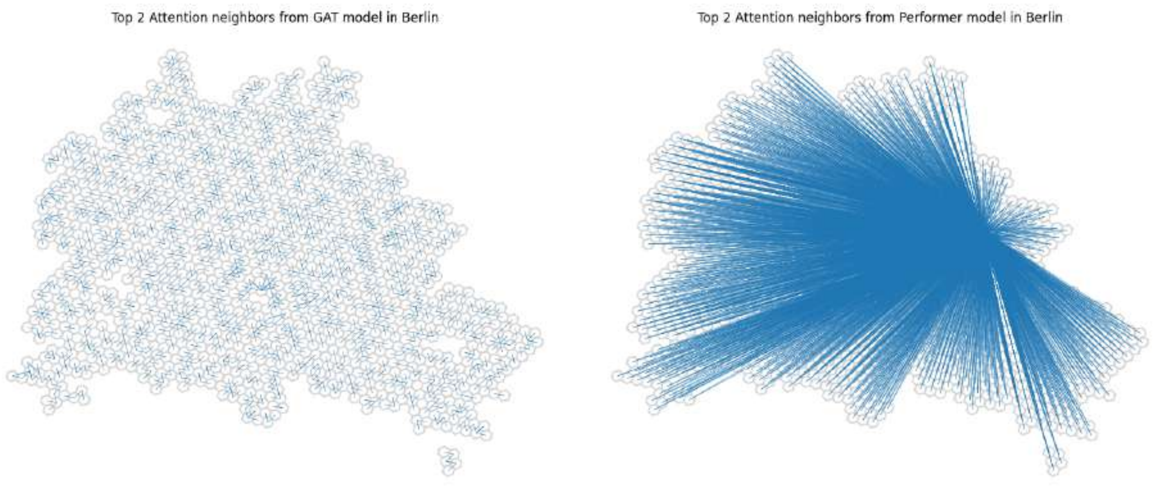

Top 2 Attention neighbors from GAT model in Berlin
Top 2 Attention neighbors from Performer model in Berlin

Figure 27: City map with the top two attention links plotted per each node (Berlin).
Left: GAT model **Right**: Performer model

This different behavior of the two attentions makes it possible to go back at the results from Section 4.2, and try to relate the node task results to what the attention graph is showing:

- GAT propagates information only along the original graph and aggregates from immediate neighbors at each layer. Since GAT updates behave like a graph based averaging filter[6]. This induces a sort of smoothing of node features: values at adjacent nodes become more similar, reducing contrast between adjacent areas, but each node prediction inherits neighborhood context and becomes more spatially coherent. Because the target is largely determined by local cues in this task, and the model sends information among multiple local attractors, this distributed, way of propagation is consistent with the better inductive transfer seen on Paris: local patterns recur across cities and remain valid under shift, while reliance on a small set of global references can misclassify areas that differ completely from the chosen

[6]as seen in Equation 6, from section 2.2.2, GAT updates are weighted neighbor averages

nodes. All of this is an empirical association between stable inductive transfer across cities, at the cost of softer boundaries.

- Performer, by contrast, is unconstrained by the graph and can place most of the attention mass on a few distant reference nodes. When this concentration occurs, like in Figure 27, the model behaves as if it were comparing each node to one or a small set of prototypes. This produces high visual contrast, but also creates spurious assignments depending on how representative the selected references are, which would explain the plentitude of false positives shown by the Performer model in Figure 21, but also why while producing prediction in cities inside the training domain this model worked so good as seen in Figure 18. The reference nodes were directly learned during training.

According to this interpretation hybrid, especially layer wise version, best results could be explained by a combined action of both attention, creating a "best of both worlds" situation where local information are propagated and contrast is still produced thanks to global attention comparing with reference nodes.

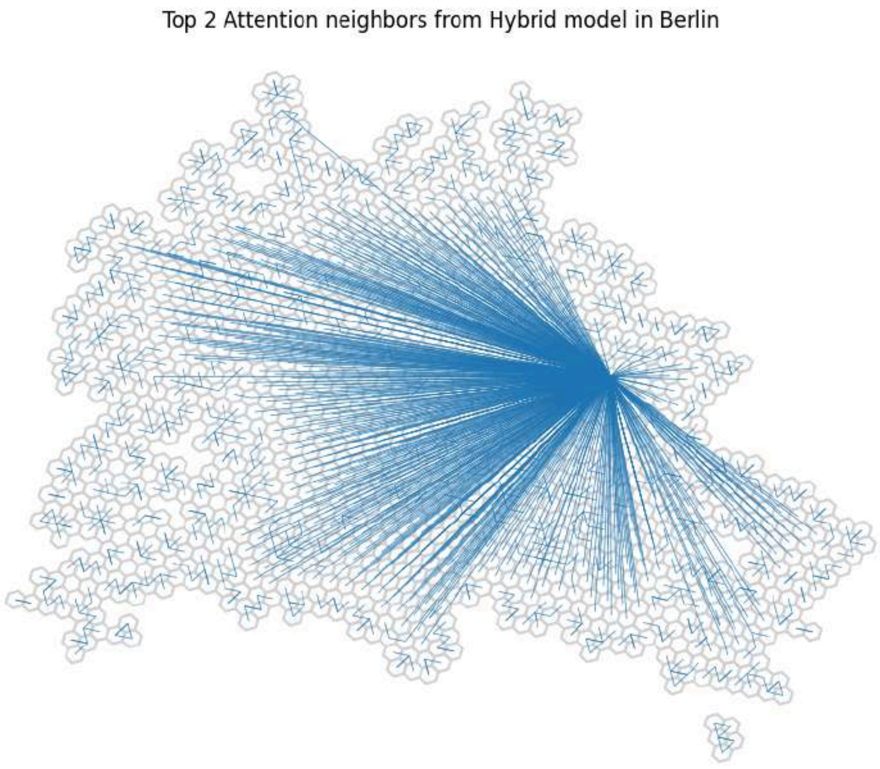
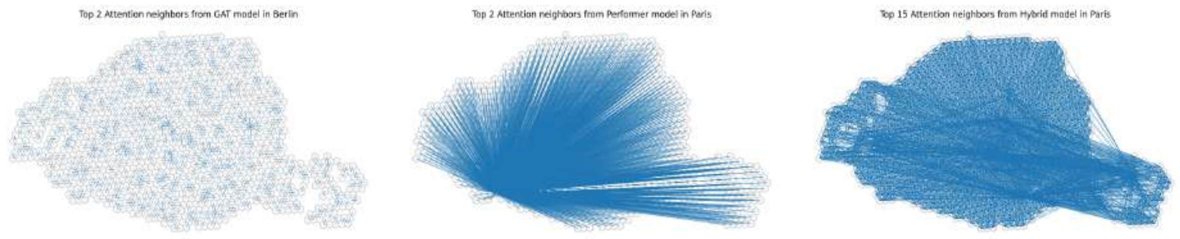


Figure 28: City map with the top two attention links of layer wise hybrid model plotted per each node (Berlin).

In this case the layer-wise hybrid combines dense sets of local links with global links toward a dominant reference as expected.

But if instead of a training city like Berlin, the test city, Paris is used to build the Attention graph, the results show that the situation is not exactly the same as for the previous example.

In Figure 29 it is possible to see that the two pure models behave in the exact same way here, furtherly sustaining the explanation on their different behavior based on this interpretability tool.
The same hybrid model from Figure 28, however, produces on Paris a completely different look, if only two links per nodes were to be showed no long range links would be visible, only after considering the complete local structure, the model uses some specific long links which connect the two green areas both to each other and to central spots in the city.
Seen this, the hypothesis is that while for training cities the model combines both contributions, with a strong emphasis on Performer attention, as seen in Figure 28. In the held-out city, the same model relies primarily on local propagation, basically becoming a GAT model, with only a small number of long-range bridges that enrich the context.


Top 2 Attention neighbors from GAT model in Berlin
Top 2 Attention neighbors from Performer model in Paris
Top 15 Attention neighbors from Hybrid model in Paris
Figure 29: Top two attention links plotted per each node for the first two models, way more links per node are shown in the hybrid model in order to also see global links (Paris).

Finally, the prediction on Paris from the pure GAT model and the layer wise hybrid model are the best on picking the correct green areas, but differ in contrast.
A plausible mechanism is that the presence of few, selective long range links that inject global information into an otherwise local model, sharpen the distinction between green and non green areas.
This mechanism is supported by the Attention graph inspection but further evidence is needed.
To assess the role of non local link, the whole contribution of Performer layers inside the model was disabled by setting to zero all Performer updates to the residual stream. The resulting map (Figure 30) exhibits lower contrast and blurred boundaries compared with the full hybrid, and large green regions are less distinctly separated from their surroundings. This corroborates the hypothesis that this hybrid leverages a small amount of selective long-range signal to sharpen transitions, removing the Performer contribution makes the model revert to a more similar to GAT behavior.

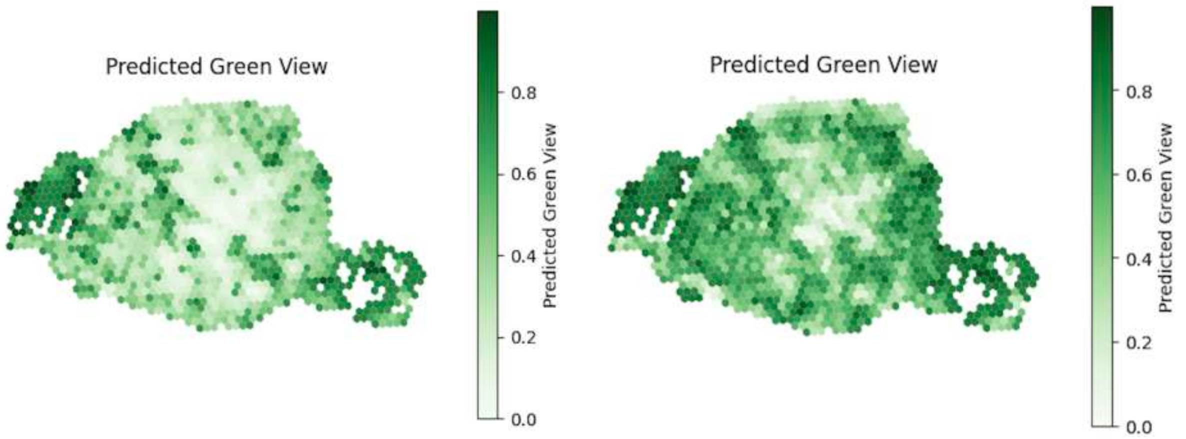


Figure 30: **Left**: Full layer wise hybrid prediction.
Right: Same hybrid prediction with Performer updates turned off in the embedding.
Turning off the global updates reduces map contrast and blurs park boundaries

Building on that hypothesis and complementing the Attention graph analysis, the second interpretability tool, the residual-stream visualization (see Section 3.4.2), is now used to track how node representations move through the network's layers.
Before interpreting trajectories and using it to reason about how the model makes predictions, a brief set of sanity checks ensures the decomposition is reliable:

1. all tensors are captured in evaluation mode so dropout does not confound the measurements.
2. every state can be reconstructed from the extracted updates: it was checked that the embedding of first layer was equal (up to negligible error) to the sum of the extracted initial embedding and the respective extracted updates from the blocks of the first layer (which are model dependant). And the same logic was applied to the second layer, to arrive to the final embedding.
3. the extracted final embedding was fed into the last output layer, verifying the outputs coincide.

With these conditions satisfied, the residual stream is then used to study the previous statement born from the Attention graph analysis.

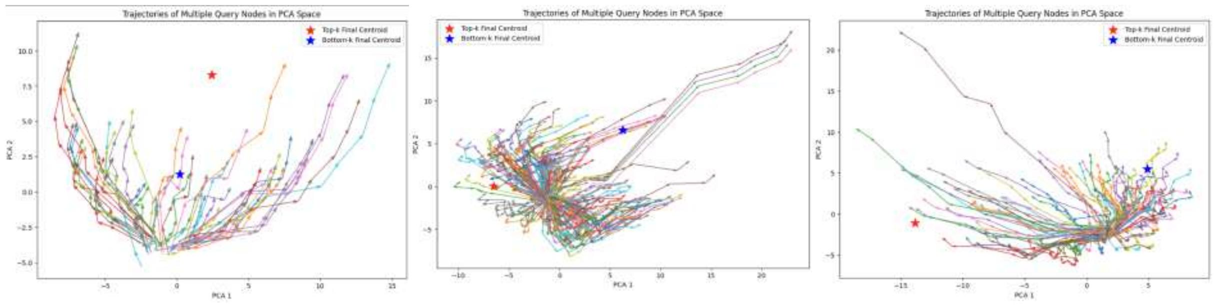


Figure 31: Trajectories of multiple query nodes of Paris:
Left: GAT model, **Center**: Performer model, **Right**: Hybrid model

Figure 31 plots residual-stream trajectories of 500 query nodes randomly selected from a forward pass of the city of Paris inside the models, in the first two principal

components of the latent space, and each arrow is the contribution of a single block of update (local,global, feed forward).
In order to have more visible updates these images are extracted from the same architectures seen until now, but with an increased depth of four layers.
This part is completely qualitative, although the data is exactly coming from the residual stream of the model, the visualization uses a two dimensional PCA projection, which can compress and distort the results, misleading the interpretation.
So, with this in mind, the three pictures show three different behaviors:

- The trajectories generated by GAT model exhibit a contraction toward a narrow, common region followed by a gentle, symmetric spreading, coherent with what has been seen while studying attention, there is not a clear distinction between top and bottom prediction in target, more of a smooth diffusion, going back at the filter view.
- Performer model presents a crowd of nodes that stay near by the top centroid, while some nodes clearly are sent far away. It also could be coherent to what has been seen in the predictions, unlike the local model, this one completely zeros some areas or creates a clear distinction inside the predictions.
- Hybrid model embedding looks like it does not create a clear distinction except for very few nodes, this could be the action of the global information injection, as hypothesized in the previous paragraph.

Another interesting comparison is in the final embedding distributions of the same hydrid model, with and without Performer contribution (Figure 32).
Once again, it looks like this figure is showing the same results from a different perspective. With the full hybrid, high target nodes occupy a more separated regions. When the Performer updates to the residual stream are set to zero, the boundaries between green and non green areas become less distinct. This mirrors the map-level effect observed earlier: removing the global updates creates smoother predictions.

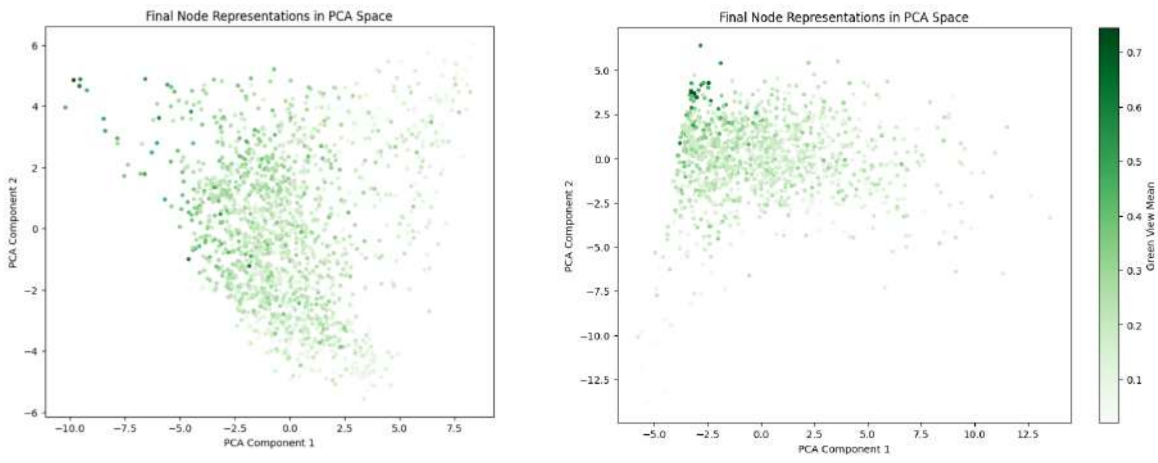


Figure 32: Final node representations of Paris, colored according to the real GVI :
Left: full hybrid model **Right**: hybrid model without Performer contribution

In conclusion this chapter evaluated the proposed architectures on diverse urban spatial graphs and investigated how local and global mechanisms shape representa-

tions. Empirically, the hybrid model delivered the best behavior, balancing between the sharper boundaries promoted by global attention and the smoother, more transferable, results produced by local propagation. Mechanistically, the two tools: attention graph analysis and residual stream visualization, suggested the same picture: GAT neighborhood aggregation supports inductive transfer, while Performer's global comparisons increase contrast.
Layer wise visualizations and the ablation disabling Performer updates validated this: removing the global component reduced separation in the final embedding space.
Beyond model evaluation, there is possibility for an interpretable adjustment of the relative strength of local vs global contribution, trading off contrast against generalization, a choice that may depend on data and the intended use.
The main limitations are the reliance on PCA projections and the fact that the evidences are primarily qualitative. Overall, the results provide both performance evidence and an empirical explanation for why hybrid models might be able to outperform purely local or purely global alternatives on urban networks.

Chapter 5
Conclusions

5.1 Summary of findings

The thesis built and evaluated graph based models for urban street networks on two supervised prediction tasks.
In the first experiment a primal street graphs was constructed for seven Italian cities using features from OpenStreetMap [41] to compute a synthetic proxy for speed. Five cities were used for training and validation while two additional cities were reserved for inductive tests.
The second experiment aggregated street networks into hexagonal H3 cells using the Global Urban Network dataset [48] and derived node features representing population intensity, points of interest, building count and sky view. Green View Index, averaged over the cells, served as the continuous target. Five European big cities were used for training, and one city was left out for unseen city evaluation.
In both tasks nodes were assigned positional encodings based on random walks and training, validation and test splits were created in a way that reduces spatial leakage.

Across tasks the thesis compared a pure Graph Attention Network (local attention), a pure Performer model (global attention) and two hybrid architectures that combine local and global updates in either a late fusion or a layer wise manner.
All models share a common blueprint: an initial embedding combines features with a positional signal. Each attentional block produces an update that is added to a residual stream. A final linear layer maps the accumulated representation to a prediction. To enable analysis of information flow, it has been developed a unified interpretability framework. The framework includes an aggregation and composition procedure to construct Attention graphs that summarize how attention flows across heads and layers, correlation aware rule averages highly correlated heads and preserves divergent heads before composing layers, producing a single flow graph for hybrid models. Also, a residual stream lens that decomposes each layer contribution into local, global and feed forward increments. This allows comparison of local, global and hybrid architectures within a consistent analytic pipeline.

In the edge level task the speed proxy is primarily governed by local features. The layer wise hybrid achieves the lowest average errors on the cities used for training, followed closely by the GAT model. The Performer is competitive but slightly worse

on average, while the late fusion hybrid trails behind. On the two held out cities GAT delivers the strongest averages, Performer is weakest, and the layer wise hybrid ties with GAT but exhibits higher between city variance. These results indicate that local attention provides a strong baseline and that interleaving local and global updates can reduce a few large errors yet may over adjust in some cities. Pure global attention does not suffice for this synthetic target because the target does not incorporate long range effects.
On the node level task, the models predict Green View Index on aggregated H3 cell graphs. When features are progressively enriched the error decreases and spatial correlation between predictions and ground truth increases for all models, with the largest improvement appearing when sky view is introduced. The hybrids obtain the best combination of low loss and high spatial correlation, followed by the Performer and then the GAT. Plain MLP baselines stay behind. In the held out city (Paris in the reported example), the layer wise hybrid produces clear separation between green and non green areas, with strong contrast and limited spurious connections. The late fusion hybrid is similar but presents worse results, boundaries are smoother and nearby zones occasionally merge. The Performer recovers the two main green masses but introduces frequent false positives and bridges across neighborhoods, while GAT identifies consistently the green areas but create smoother, less contrasted maps. MLP with positional encoding shows the weakest spatial organization and often misses important parks until all features are included. These patterns reveal that combining local and global attention leverages both neighborhood coherence and long distance context, whereas pure local models under segment and pure global models over contrast.

The experiments also show that the node level task benefits from hybrid architectures even though the signal remains largely local. It was suggested that larger gains are expected on problems where information must propagate across many steps, such as those influenced by rail or metro networks.

Head correlation analysis shows a striking difference between local and global attention. Within GAT layers the attention heads are strongly and consistently positively correlated, indicating that heads learn similar patterns. In contrast, Performer layers exhibit mixed correlations with values near zero or negative, and in the layer wise hybrid the average correlation in Performer layers drops from about thirty one percent in the first layer to roughly ten percent in the second layer. These observations support the correlation aware aggregation rule that averages heads when they align and preserves the maximum weight when they diverge.
The Attention graphs derived from the models expose different regimes of information flow. The GAT attention graph is sparse and follows the original connectivity of the street graph, propagating information only within the depth of the network. The Performer attention graph is dense, with nodes attending widely and a bell shaped distribution of hop distances; it tends to concentrate attention on a small set of reference

nodes. Hybrid models display a mixed profile with a dominant peak at short distances and a continuous tail of longer connections. These diagnostics reveal whether a model builds meaningful long distance relationships or collapses onto a few nodes.
In training cities global attention of studied hybrid model focuses on dominant reference nodes, increasing contrast in the prediction, but, reducing diversity and, as it was proposed, weakening generalization. In the held out city, the hybrid relies primarily on local propagation with only a small number of long connections.
The residual stream lens tracks how node representations evolve through the network. When applied to the hybrid model on Paris, disabling the global updates leads to maps with reduced contrast and blurred boundaries, whereas focusing only on global updates creates high contrast but introduces artifacts.
Residual stream trajectories in a deeper model show that GAT embeddings contract toward a common region and then spread gently, Performer embeddings push some nodes far apart creating clear distinctions, and the hybrid embeddings combine these behaviors. These observations suggest that global updates sharpen the separation between high and low green values, while local updates stabilize neighborhoods and prevent spurious predictions.

The results illustrate a trade off between local and global contributions. Local attention provides stability and better generalization across cities but tends to under segment and produce less contrasts. Global attention enhances contrast but risks creating spurious links and single node attractors. Hybrid models balance these effects by combining local smoothing with selective global injections, however, their performance depends on how the contributions are weighted. In the edge level task the synthetic target lacks long distance dependencies and hybrids offer only marginal improvements, whereas in the node level task the hybrids, more so the layer wise version, is clearly superior.

Several limitations must be acknowledged: the speed target in the edge level task is synthetic and constructed from local variables, so conclusions about model performance apply to this specific proxy and may not generalize to real traffic speeds. The node level task already benefits from combining local and global attention, but it does not exhibit strong long range dependencies. Tasks that inherently require information propagation across many steps may produce larger differences. The interpretability tools are descriptive rather than causal. Attention graphs summarize where the model focuses but do not prove that those connections drive predictions. The residual stream analysis relies on a two dimensional principal component projection, which can distort the geometry of the embedding space.
Consequently, mechanistic explanations drawn from these tools should be read as hypotheses for further testing rather than definitive causal proofs.

5.2 Future directions

Looking ahead, the thesis points toward several promising directions.
A first line of research is to move beyond node prediction and use the attention based

machinery to learn network topology itself at a coarser level.
One can treat the Attention Graph as a learned soft connectivity and test whether certain spatial features, such as a high density of points of interest and residential areas, predict the existence of transportation links. Learning complex network topologies in a coarser, more general, hexagonal tilings could lead to better predictions of missing links and to a deeper understanding of how urban form influences connectivity.
A second direction is to train models on multiple cities that include both street networks and metro or light rail layers, with the goal of predicting new lines or variation in existing ones.
Beyond predictive models, the interpretability tools enable controlled ablations: one could zero out the strongest long range links or remove regions with high incoming strength in the Attention Graph and measure the impact on predictions, thereby testing the reliance of the model on specific global paths.
Finally, refining the models themselves—such as experimenting with better suited loss functions, e.g. physics informed, or adding regularization to prevent the formation of single reference nodes, could further improve performances and interpretability.

In conclusion, the thesis presents a coherent framework for modelling and interpreting attention based graph architectures on urban spatial graphs. By building and comparing local, global and hybrid models on two carefully constructed tasks, and by developing tools that show how information flows within these networks, the work shows that local and global mechanisms offer complementary strengths. The balanced perspective emerging from these experiments, together with the proposed future directions, lays the groundwork for new urban analytics that are both predictive and interpretable.

Bibliography

[1] M. M. Bronstein, J. Bruna, T. Cohen, and P. Velickovic, "Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges," *CoRR*, 2021, [Online]. Available: https://arxiv.org/abs/2104.13478

[2] C. K. Joshi, "Transformers are Graph Neural Networks." [Online]. Available: https://arxiv.org/abs/2506.22084

[3] B. El, D. Choudhury, P. Liò, and C. K. Joshi, "Towards Mechanistic Interpretability of Graph Transformers via Attention Graphs." [Online]. Available: https://arxiv.org/abs/2502.12352

[4] N. Elhage *et al.*, "A Mathematical Framework for Transformer Circuits," *Transformer Circuits Thread*, 2021.

[5] M. van Steen, *Graph Theory and Complex Networks: An Introduction.* Maarten van Steen, 2010.

[6] M. Barthélemy, "Spatial networks," *Physics Reports*, vol. 499, no. 1, pp. 1–101, 2011, doi: https://doi.org/10.1016/j.physrep.2010.11.002.

[7] M. Barthélemy, "Betweenness centrality in large complex networks," *European Physical Journal B*, vol. 38, pp. 163–168, 2004, doi: 10.1140/epjb/e2004-00111-4.

[8] M. Batty, *Cities and Complexity.* Cambridge, MA: MIT Press, 2005.

[9] A. Cardillo, S. Scellato, V. Latora, and S. Porta, "Structural properties of planar graphs of urban street patterns," *Physical Review E*, vol. 73, p. 66107, 2006, doi: 10.1103/PhysRevE.73.066107.

[10] S. Lämmer, B. Gehlsen, and D. Helbing, "Scaling laws in the spatial structure of urban road networks," *Physica A: Statistical Mechanics and its Applications*, vol. 363, pp. 89–95, 2006, doi: 10.1016/j.physa.2006.01.051.

[11] H3, "H3 Documentation." Accessed: 2025. [Online]. Available: https://h3geo.org/docs/

[12] G. K. Zipf, "The P1 P2/D Hypothesis: On the Intercity Movement of Persons," *American Sociological Review*, vol. 11, no. 6, pp. 677–686, 1946, Accessed: Aug. 09, 2025. [Online]. Available: http://www.jstor.org/stable/2087063

[13] F. Simini, M. C. González, A. Maritan, and A.-L. Barabási, “A universal model for mobility and migration patterns,” *Nature*, vol. 484, no. 7392, pp. 96–100, 2012, doi: 10.1038/nature10856.

[14] F. Simini, G. Barlacchi, M. Luca, and L. Pappalardo, “A Deep Gravity model for mobility flows generation,” *Nature Communications*, vol. 12, no. 1, p. 6576, 2021, doi: 10.1038/s41467-021-26752-4.

[15] G. Murthy, J. Lu, and L. Rajaram, “Gummada Murthy, Jian Lu, L. Rajaram: Evaluation of Intelligent Transportation System Operations Using Logistic Regression Models. Institute of Transportation Engineers (ITE) Journal, Vol. 83, No. 3 – March 2013.,” *Institute of Transportation Engineers Journal*, vol. 83, p. , 2013.

[16] C.-H. Wu, J.-M. Ho, and D. Lee, “Travel-time prediction with support vector regression,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 5, no. 4, pp. 276–281, 2004, doi: 10.1109/TITS.2004.837813.

[17] Y. Wang, Y. Liu, and X. Yang, “An Empirical Comparison of Urban Road Travel Time Prediction Methods—Deep Learning, Ensemble Strategies and Performance Evaluation,” *Applied Sciences*, vol. 15, no. 14, 2025, doi: 10.3390/app15148075.

[18] M. Luca, G. Barlacchi, B. Lepri, and L. Pappalardo, “Deep Learning for Human Mobility: a Survey on Data and Models,” *CoRR*, 2020, [Online]. Available: https://arxiv.org/abs/2012.02825

[19] R. Liu and S.-Y. Shin, “A Review of Traffic Flow Prediction Methods in Intelligent Transportation System Construction,” *Applied Sciences*, vol. 15, no. 7, 2025, doi: 10.3390/app15073866.

[20] X. Ma, Z. Dai, Z. He, and Y. Wang, “Learning Traffic as Images: A Deep Convolution Neural Network for Large-scale Transportation Network Speed Prediction,” *CoRR*, 2017, [Online]. Available: http://arxiv.org/abs/1701.04245

[21] R. Bellman, *Dynamic Programming*. Dover Publications, 1957.

[22] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, “The Graph Neural Network Model,” *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009, doi: 10.1109/TNN.2008.2005605.

[23] T. N. Kipf and M. Welling, “Semi-Supervised Classification with Graph Convolutional Networks,” *CoRR*, 2016, [Online]. Available: http://arxiv.org/abs/1609.02907

[24] M. Defferrard, X. Bresson, and P. Vandergheynst, “Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering,” *CoRR*, 2016, [Online]. Available: http://arxiv.org/abs/1606.09375

[25] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, “Graph Attention Networks.” [Online]. Available: https://arxiv.org/abs/1710.10903

[26] H. Li *et al.*, “Graph Neural Networks in Intelligent Transportation Systems: Advances, Applications and Trends.” [Online]. Available: https://arxiv.org/abs/2401.00713

[27] N. Kim and Y. Yoon, “Effective Urban Region Representation Learning Using Heterogeneous Urban Graph Attention Network (HUGAT).” [Online]. Available: https://arxiv.org/abs/2202.09021

[28] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. Salakhutdinov, and A. Smola, “Deep Sets.” [Online]. Available: https://arxiv.org/abs/1703.06114

[29] A. Vaswani *et al.*, “Attention is all you need,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, in NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6000–6010.

[30] M. Black, Z. Wan, G. Mishne, A. Nayyeri, and Y. Wang, “Comparing Graph Transformers via Positional Encodings.” [Online]. Available: https://arxiv.org/abs/2402.14202

[31] V. P. Dwivedi, A. T. Luu, T. Laurent, Y. Bengio, and X. Bresson, “Graph Neural Networks with Learnable Structural and Positional Representations.” [Online]. Available: https://arxiv.org/abs/2110.07875

[32] V. P. Dwivedi, C. K. Joshi, A. T. Luu, T. Laurent, Y. Bengio, and X. Bresson, “Benchmarking Graph Neural Networks.” [Online]. Available: https://arxiv.org/abs/2003.00982

[33] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret, “Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention.” [Online]. Available: https://arxiv.org/abs/2006.16236

[34] L. Rampášek, M. Galkin, V. P. Dwivedi, A. T. Luu, G. Wolf, and D. Beaini, “Recipe for a General, Powerful, Scalable Graph Transformer.” [Online]. Available: https://arxiv.org/abs/2205.12454

[35] K. Choromanski *et al.*, “Rethinking Attention with Performers.” [Online]. Available: https://arxiv.org/abs/2009.14794

[36] Z. Shan, F. Yang, X. Shi, and Y. Cui, “Hybrid Learning Model of Global–Local Graph Attention Network and XGBoost for Inferring Origin–Destination Flows,” *ISPRS International Journal of Geo-Information*, vol. 14, no. 5, 2025, doi: 10.3390/ijgi14050182.

[37] P. Zhang *et al.*, “TransGNN: Harnessing the Collaborative Power of Transformers and Graph Neural Networks for Recommender Systems.” [Online]. Available: https://arxiv.org/abs/2308.14355

[38] Y. Ye, Y. Cao, Y. Dong, and H. Yan, “A graph neural network and Transformer-based model for PM2.5 prediction through spatiotemporal correlation,” *Environmental Modelling & Software*, vol. 191, p. 106501, 2025, doi: https://doi.org/10.1016/j.envsoft.2025.106501.

[39] Y. Hou, D. Zhang, M. M. Hasan, and Q. Niu, “PreSTAEFormer: A Hybrid GNN-Transformer Model with Pre-Attention Mechanism for Enhanced Urban Traffic Prediction,” in *2025 6th International Conference on Computing, Networks and Internet of Things (CNIOT)*, 2025, pp. 1–5. doi: 10.1109/CNIOT65435.2025.11070781.

[40] S. Jain and B. C. Wallace, “Attention is not Explanation.” [Online]. Available: https://arxiv.org/abs/1902.10186

[41] OpenStreetMap contributors, “Maps retrieved from OSM .” [Online]. Available: https://www.openstreetmap.org/

[42] S. Ruder, “An overview of gradient descent optimization algorithms.” [Online]. Available: https://arxiv.org/abs/1609.04747

[43] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization.” [Online]. Available: https://arxiv.org/abs/1412.6980

[44] X. Li, C. Zhang, W. Li, R. Ricard, Q. Meng, and W. Zhang, “Assessing street-level urban greenery using Google Street View and a modified green view index,” *Urban Forestry & Urban Greening*, vol. 14, no. 3, pp. 675–685, 2015, doi: https://doi.org/10.1016/j.ufug.2015.06.006.

[45] S. Carotti, “FBK-internship: Keeping track of the work done during the internship.” GitHub, 2025.

[46] G. Boeing, “Modeling and Analyzing Urban Networks and Amenities With OSMnx,” *Geographical Analysis*, no. n/a, p. , doi: https://doi.org/10.1111/gean.70009.

[47] K. Jordahl *et al.*, “geopandas/geopandas: v0.8.1.” [Online]. Available: https://doi.org/10.5281/zenodo.3946761

[48] W. Yap, “The Urbanity Global Network Dataset,” 2023, doi: 10.6084/m9.figshare.22124219.v12.

[49] J. L. Ba, J. R. Kiros, and G. E. Hinton, “Layer Normalization.” [Online]. Available: https://arxiv.org/abs/1607.06450

[50] S. Brody, U. Alon, and E. Yahav, “How Attentive are Graph Attention Networks?.” [Online]. Available: https://arxiv.org/abs/2105.14491