Link Prediction for Flow-Driven Spatial Networks

Bastian Wittmann
University of Zurich
bastian.wittmann@uzh.ch

Johannes C. Paetzold
Technical University of Munich
johannes.paetzold@tum.de

Chinmay Prabhakar
University of Zurich
chinmay.prabhakar@uzh.ch

Daniel Rueckert
Technical University of Munich
daniel.rueckert@tum.de

Bjoern Menze
University of Zurich
bjoern.menze@uzh.ch

Abstract

Link prediction algorithms aim to infer the existence of connections (or links) between nodes in network-structured data and are typically applied to refine the connectivity among nodes. In this work, we focus on link prediction for flow-driven spatial networks, which are embedded in a Euclidean space and relate to physical exchange and transportation processes (e.g., blood flow in vessels or traffic flow in road networks). To this end, we propose the Graph Attentive Vectors (GAV) link prediction framework. GAV models simplified dynamics of physical flow in spatial networks via an attentive, neighborhood-aware message-passing paradigm, updating vector embeddings in a constrained manner. We evaluate GAV on eight flow-driven spatial networks given by whole-brain vessel graphs and road networks. GAV demonstrates superior performances across all datasets and metrics and outperformed the state-of-the-art on the ogbl-vessel benchmark at the time of submission by 12% (98.38 vs. 87.98 AUC). All code is publicly available on GitHub.[1]

1. Introduction and Motivation

Networks (or graphs) can serve as efficient representations of real-world, ultra-complex systems and can be further classified into different categories. A prominent category is represented by undirected networks embedded in a Euclidean space constrained by geometry, called spatial networks [5]. In this work, we are focusing on spatial networks, where a form of physical exchange or *flow* can be used to describe characteristic functional properties of the underlying physical system. Examples include road networks, water bodies, and global exchange networks, but they can also be found in biology (*e.g.*, vascular system, lymphatic system, and connectome). We will refer to such networks as *flow-driven spatial networks* (see Fig. 1).

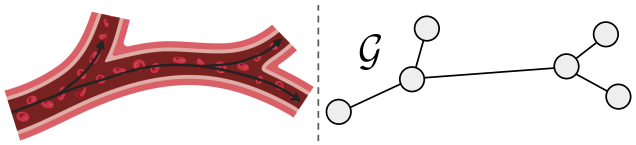

Figure 1. Flow-driven spatial network $\mathcal{G}$ (right), representing vasculature (left). $\mathcal{G}$'s nodes are embedded in a Euclidean space and represent spatial positions specified by x-, y-, and z-coordinates.

Predominantly, network representations of physical systems originate from imaging methodologies, such as nanometer-scale microscopy in biology or regional to continental scale satellite remote sensing for road networks. The generation of compact network representations from these images is a multi-stage and imperfect process (segmentation, skeletonization, and subsequent graph pruning), which introduces noise and artifacts. However, for many relevant downstream tasks operating on flow-driven spatial networks, such as blood flow modeling [34] - a crucial component of investigating neurovascular brain disorders - or traffic forecasting [20], a flawless network representation is of utmost importance. In this context, the task of link prediction presents itself as a meaningful measure to identify absent connections and reduce local noise arising from the erroneous graph generation process [17, 32]. Link prediction algorithms tailored to flow-driven spatial networks, however, remain heavily under-explored.

Therefore, we bring a simplistic yet general definition of the principle of physical flow, characterized by a direction and magnitude, to link prediction in graph representation learning. Our *hypothesis* is that for flow-driven spatial networks, link prediction algorithms should heavily benefit from considering known functional properties, such as the aforementioned *physical flow*, which are defined by the structural properties of the network (*e.g.*, bifurcation angles [35]). To this end, we propose the *Graph Attentive*

[1] https://github.com/bwittmann/GAV

Vectors (GAV) link prediction framework. GAV operates on *vector embeddings* representative of the network's structural properties and updates them in a constrained manner, imitating simplified dynamics of physical flow in spatial networks (*e.g.*, blood flow in the vascular system or traffic flow in road networks). We summarize our core contribution as follows:

1. We propose an attentive, neighborhood-aware message passing layer, called GAV layer, which updates vector embeddings, mimicking the (change in) direction and magnitude of physical flow in spatial networks.
2. We introduce a readout module that aggregates vector embeddings in a physically plausible way and thus facilitates the interpretability of results.
3. We prove the above-mentioned hypothesis by demonstrating state-of-the-art performance across all metrics in extensive experiments on eight flow-driven spatial networks, including the Open Graph Benchmark's [17] ogbl-vessel benchmark[2] (98.38 vs. 87.98 AUC).

2. Related Works

This section discusses previous work on link prediction algorithms and message-passing layers. Particular emphasis is placed on methods featured in our experiments.

2.1. Link Prediction

Link prediction algorithms are applied in various fields, such as social network analysis [10, 26, 30], bioinformatics [21, 25, 36], recommender systems [2, 18, 37, 53], and supply chain improvement [28]. Broadly speaking, different link prediction algorithms try to estimate link existence between two nodes either via heuristic or learned methods.

Heuristic algorithms employ predefined heuristics to encode the similarity between nodes. Some prominent candidates are represented by common neighbors, resource allocation [54], preferential attachment [4], Adamic-Adar [1], Jaccard [19], Katz [22], and average commute time [13]. However, all heuristic link prediction algorithms suffer from the same underlying issue. They exploit predefined, simple heuristics, which can not be modified to account for different network types. *E.g.*, common neighbors has been developed for social networks and hence yields underwhelming results when applied to molecular graphs.

On the other hand, learned algorithms do not rely on predesigned heuristics but rather learn a more complex, data-driven heuristic utilizing neural networks. Thus, learned algorithms can easily adapt to different network types while typically outperforming their heuristic counterparts. SEAL [49, 51] represents a prominent, learned link prediction framework, defining link prediction as a subgraph-level classification task by training a binary GNN-based classifier to map from subgraph patterns to link existence. To this end, SEAL first extracts a local subgraph around the link of interest, which is subsequently forwarded to DGCNN [50] for classification. Moreover, SEAL's authors introduce an additional node labeling technique, known as labeling trick [51], to enhance the expressiveness of node features obtained from GNNs. SIEG [14] builds upon SEAL and introduces, inspired by Graphormer [48], a pairwise structural attention module between two nodes of interest to capture local structural information more effectively. This results in state-of-the-art performances and allows SIEG to simultaneously overcome Graphormer's issue of exploding computational complexity when applied to ultra-large graphs. SUREL+ [46] introduces the use of node sets to represent subgraphs. To complement the loss of structural information, SUREL+ provides set samplers, structure encoders, and set neural encoders. SUREL+ outperforms the baseline SEAL on various link prediction benchmarks. It should be mentioned that Cai *et al*. [7] investigate the use of line graphs for link prediction. Please note that none of the above-mentioned methods are tailored to flow-driven spatial networks.

2.2. Message-Passing Layers

GNNs utilize the concept of message-passing to encode semantically rich features within network-structured data. Over time, multiple variations of message-passing layers have been proposed [11, 12, 16, 43]. For instance, GCN's message-passing layer [24] weighs each incoming message with a fixed coefficient, the node degree, before aggregation. In contrast, GAT's message-passing layer [6] learns aggregation weights dynamically based on attention scores. GraphSAGE's message-passing layer [15] does not directly aggregate central node features with incoming messages. Instead, it distinguishes these two kinds of features and learns two different transformations, one on the central node and another on incoming messages. EdgeConv [41] aggregates the feature difference between the central node and its neighbors combined with the central node's features. Thus, EdgeConv draws parallels to aggregating spatial vectors if the nodes embed spatial positions. However, our proposed GAV layer differs significantly from EdgeConv, as we explicitly constrain the update of vector embeddings to imitate the simplified dynamics of physical flow in flow-driven spatial networks. Importantly, only a few works tried to adapt the message-passing paradigm to spatial networks [9, 52].

3. The Graph Attentive Vectors Framework

Our proposed Graph Attentive Vectors (GAV) link prediction framework is depicted in Fig. 2. GAV represents a simple yet effective end-to-end trainable framework tailored to the task of link prediction for flow-driven spatial net-

[2] https://ogb.stanford.edu/docs/linkprop

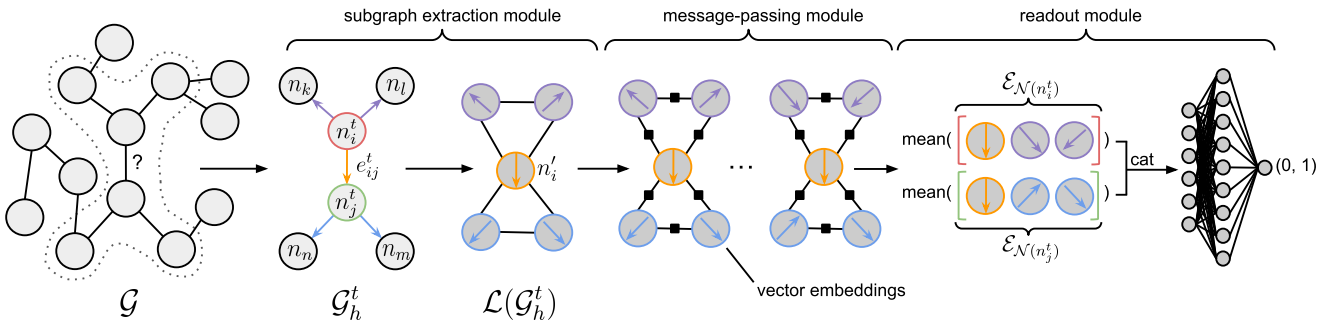

Figure 2. Overview of the GAV link prediction framework. GAV is divided into three modules, namely the subgraph extraction module, the message-passing module, and the readout module. First, an h-hop enclosing subgraph $\mathcal{G}_h^t$ is extracted around the target nodes $\{n_i^t, n_j^t\}$ (red and green) affiliated to the target link e_{ij}^t (orange) and subsequently transformed into a line graph representation $\mathcal{L}(\mathcal{G}_h^t)$ to construct vector embeddings representative of the network's structural properties; second, we perform iterative message-passing among vector embeddings in $\mathcal{L}(\mathcal{G}_h^t)$ via k GAV layers, modeling simplified physical flow in spatial networks; and, third, a final subgraph-level readout module aggregates refined vector embeddings and predicts the probability of link existence with regard to the target link e_{ij}^t. To provide a concise visualization, h was set to 1. We would like to draw the reader's attention to color coding.

works. It predicts the probability of link (or edge) existence between two nodes in a graph $\mathcal{G}$ based on a binary classifier $\mathcal{F}$, composed of a novel message-passing and readout module operating on vector embeddings (see Sections 3.2 and 3.4). To this end, $\mathcal{F}$ should be able to differentiate between positive (real) and negative (sampled) links by assigning high probabilities of existence to plausible and low probabilities of existence to implausible links. Following most competitive approaches [14, 49, 51], we treat link prediction as a subgraph classification task. To determine the probability of existence of an individual target link between two target nodes, we, therefore, first extract an enclosing subgraph describing the target link's local neighborhood in a subgraph extraction module (see Section 3.1). Subsequently, the subgraph is transformed into a line graph representation to construct vector embeddings and forwarded to $\mathcal{F}$, resulting in an iterative link prediction scheme predicting the existence of target links one at a time. In the following, we elaborate extensively on GAV's individual components (see Fig. 2). For ease of reference, we provide a notation lookup table in the supplementary (see Supp. A).

3.1. Subgraph Extraction Module

The undirected input graph $\mathcal{G} := (\mathcal{V}, \mathcal{E})$ is defined by a set of nodes $\mathcal{V}$ and a set of corresponding edges $\mathcal{E}$. While a node $n_i \in \mathcal{V}$ contains a specific spatial position given by coordinates ($n_i \in \mathbb{R}^{d_{\text{spatial}}}$), an edge $e_{ij} \in \mathcal{E}$ indicates a connection between nodes n_i and n_j.

Similar to most competitive approaches [14, 49, 51], we extract as a first step an h-hop enclosing subgraph $\mathcal{G}_h^t$ around the nodes $\{n_i^t, n_j^t\}$ affiliated to the target link e_{ij}^t from the original graph representation $\mathcal{G}$ (please note that we refer to the target in our notations as t). This results in an expressive and efficient representation of the target link's local neighborhood, including relevant structural patterns necessary to determine link existence. Further, the subgraph extraction significantly reduces space complexity, which is crucial for link prediction in ultra-large graphs (see Table 1).

Since GAV operates on *vector embeddings*, we subsequently transform $\mathcal{G}_h^t$ into a line graph representation $\mathcal{L}(\mathcal{G}_h^t) := (\mathcal{V}', \mathcal{E}')$. In $\mathcal{L}(\mathcal{G}_h^t)$, each node $n_i' \in \mathcal{V}'$ represents an edge $e_{ij} \in \mathcal{E}$ via an individual vector embedding, while an edge $e_{ij}' \in \mathcal{E}'$ indicates adjacency between two edges in $\mathcal{G}_h^t$ iff they are incident. To create vector embeddings, we encode edges as vectors between the involved nodes (*e.g.*, $n_i' = n_j - n_i$). Therefore, $\mathcal{L}(\mathcal{G}_h^t)$'s node embeddings are defined as vectors (see Fig. 2) of unique length and direction representative of edges in $\mathcal{G}_h^t$ (see Supp. C for implementation details). The line graph $\mathcal{L}(\mathcal{G}_h^t)$ formed around the target link e_{ij}^t is finally forwarded to the message-passing module.

3.2. Message-Passing Module and GAV Layer

To adjust link prediction to flow-driven spatial networks, we propose a novel message-passing layer, termed GAV layer. We perform k iterations of message-passing among the nodes of the line graph $\mathcal{L}(\mathcal{G}_h^t)$, obtained from our subgraph extraction module, in our message-passing module. The GAV layer's message-passing relies on a straightforward intuition inspired by principles of physical flow. To be precise, we treat nodes in $\mathcal{L}(\mathcal{G}_h^t)$ as vector embeddings and update them in a constrained manner, imitating simplified dynamics of physical flow in spatial networks. The detailed structure of a single GAV layer is visualized in Fig. 3.

In order to update an individual vector embedded in a node $n_i' \in \mathbb{R}^{d_{\text{spatial}}}$, we first project it together with a matrix $N_i \in \mathbb{R}^{|\mathcal{N}(n_i') \cup n_i'| \times d_{\text{spatial}}}$, consisting of directly neighboring nodes and the node itself, into a higher dimensional space d_{message}; the projection is through a learnable func-

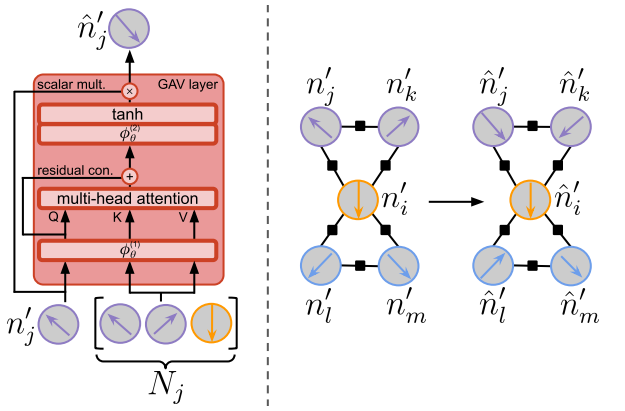

Figure 3. Visualization of a single GAV layer updating the vector embedding of node n'_j. We forward the vector embedding of node n'_j together with $N_j \in \mathbb{R}^{|\mathcal{N}(n'_j) \cup n'_j| \times d_{\text{spatial}}}$, which represents the set of n'_j and its neighbors n'_k and n'_i, to the GAV layer. In this specific example, the vector embedding has been flipped (see $\hat{n}'_j$), which can be interpreted as a change in direction of physical flow.

tion $\phi_\theta^{(1)} : \mathbb{R}^{d_{\text{spatial}}} \to \mathbb{R}^{d_{\text{message}}}$. Subsequently, $\phi_\theta^{(1)}(n'_i)$ and $\phi_\theta^{(1)}(N_i)$ are forwarded to a multi-head attention operation, where $\phi_\theta^{(1)}(n'_i)$ represents a single query, while $\phi_\theta^{(1)}(N_i)$ represents the key and value sequence.

$$\tilde{n}'_i = \text{MultiHeadAttn}(\phi_\theta^{(1)}(n'_i),\ \phi_\theta^{(1)}(N_i),\ \phi_\theta^{(1)}(N_i)) \quad (1)$$

This results in an intermediate node representation $\tilde{n}'_i \in \mathbb{R}^{d_{\text{message}}}$, which incorporates not only information of the node itself but also its local structural neighborhood via the concept of attention. In the next step, we apply a residual connection for increased gradient flow and forward the result to the learnable function $\phi_\theta^{(2)} : \mathbb{R}^{d_{\text{message}}} \to \mathbb{R}$, followed by a tanh non-linearity.

$$s_i = \tanh(\phi_\theta^{(2)}(\tilde{n}'_i + \phi_\theta^{(1)}(n'_i))) \quad (2)$$

Finally, the scalar value $s_i \in (-1, 1)$ is utilized to update the original node representation n'_i via scalar multiplication.

$$\hat{n}'_i = s_i \cdot n'_i \quad (3)$$

Hence, after one layer of message-passing, the updated, refined node representation is given by $\hat{n}'_i \in \mathbb{R}^{d_{\text{spatial}}}$. Importantly, $\hat{n}'_i$ preserves relevant structural properties of the original node representation n'_i. This is because scalar multiplication with $s_i \in (-1, 1)$ restricts the modification of vector embeddings given by nodes in $\mathcal{L}(\mathcal{G}_h^t)$. In essence, our message-passing paradigm can be geometrically interpreted as a scaling combined with a potential flipping operation of vectors. These constraints imposed by our message-passing align with our principal idea of modeling simplified dynamics of physical flow in spatial networks via potential, constrained changes in the direction and magnitude of vector embeddings, preserving the network's structural properties.

3.3. Labeling Trick

Following common practices [7, 14, 51], we apply a labeling trick to enable the message-passing module to distinguish between the relevance of different vector embeddings and, hence, learn an expressive structural representation of the target link's local neighborhood. The labeling trick ensures that vector embeddings created from the target link e_{ij}^t and edges connected to the target nodes $\{n_i^t, n_j^t\}$ are identifiable by a distinct label. Labels generated by our interpretation of the labeling trick are shown in Fig. 4.

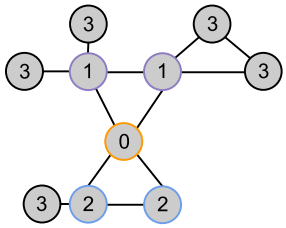

Figure 4. Labels generated by the labeling trick for an exemplary line graph (h set to two). Our interpretation of the labeling trick assigns the label 0 to the vector embedding representing the target link (orange), the labels 1 and 2 to vector embeddings representing edges connected to the target nodes n_i^t and n_j^t (purple and blue), and the label 3 to remaining vector embeddings.

The additional labels generated by the labeling trick are concatenated to $\mathcal{L}(\mathcal{G}_h^t)$'s vector embeddings in the message-passing module (before message-passing).

3.4. Readout Module

Our proposed readout module consists of a custom node aggregation operation followed by a learnable function $\phi_\theta^{(3)} : \mathbb{R}^{2 \cdot d_{\text{spatial}}} \to \mathbb{R}$ and processes refined vector embeddings obtained from the message-passing module. While the node aggregation operation aims to distill pertinent information from the refined graph representation in a fashion invariant to node ordering and quantity, $\phi_\theta^{(3)}$ predicts the probability of link existence with regard to the target link.

Equation 4 summarizes the readout module's functionality, where $\mathcal{E}_{\mathcal{N}(n_i^t)}$ defines the set of refined vector embeddings originally created from edges adjacent to n_i^t (see Fig. 2), $\mathcal{E}_{\mathcal{N}(n_j^t)}$ the set of refined vector embeddings originally created from edges adjacent to n_j^t (see Fig. 2), $\|$ the concatenation operation, and $\hat{y}_{ij}^t$ the predicted probability of target link existence.

$$\hat{y}_{ij}^t = \phi_\theta^{(3)}(\text{mean}(\mathcal{E}_{\mathcal{N}(n_i^t)}) \,\|\, \text{mean}(\mathcal{E}_{\mathcal{N}(n_j^t)})) \quad (4)$$

Thus, our node aggregation operation constructs mean vector embeddings (see $\text{mean}(\mathcal{E}_{\mathcal{N}(n_i^t)})$ and $\text{mean}(\mathcal{E}_{\mathcal{N}(n_j^t)})$) representative of our simplified definition of flow in the two target nodes and intends to exploit their relationship to predict whether target nodes should be connected or not.

Table 1. Properties of the raw datasets. Each dataset consists of exactly one ultra-large graph.

Dataset Name	# Nodes	# Edges	Node Degree	Node Features	d_{spatial}	Edge Features	Description
ogbl-vessel [17]	3,538,495	5,345,897	3.02	x-, y-, z-coordinates	3	–	BALB/c mouse strain[1]
c57-tc-vessel [32]	3,820,133	5,614,677	2.94	x-, y-, z-coordinates	3	–	C57BL/6 mouse strain[1]
cd1-tc-vessel [32]	3,645,963	5,791,309	3.18	x-, y-, z-coordinates	3	–	CD-1 mouse strain[1]
c57-cc-vessel [39]	6,650,580	9,054,100	2.72	x-, y-, z-coordinates	3	–	C57BL/6 mouse strain[2]
belgium-road [3]	1,441,295	1,549,970	2.15	x-, y-coordinates	2	–	Belgium
italy-road [3]	6,686,493	7,013,978	2.10	x-, y-coordinates	2	–	Italy
netherlands-road [3]	2,216,688	2,441,238	2.20	x-, y-coordinates	2	–	Netherlands
luxembourg-road [3]	114,599	119,666	2.09	x-, y-coordinates	2	–	Luxembourg

[1] tissue clearing (tc) and light-sheet microscopy imaging [2] corrosion casting (cc) and SRμCT imaging

3.5. Loss Function

Since our approach represents a binary classifier $\mathcal{F}$ determining the probability of link existence with regard to the target link, we optimize a binary cross-entropy loss function during training. Here, $y_{ij}^t \in \{0,1\}$ indicates whether the target links are negative (sampled) or positive (real).

$$\mathcal{L}_{\text{BCE}} = \frac{-1}{|\mathcal{E}|} \sum_{ij \in \mathcal{E}} y_{ij}^t \cdot \log(\hat{y}_{ij}^t) + (1 - y_{ij}^t) \cdot \log(1 - \hat{y}_{ij}^t) \quad (5)$$

We would like to highlight that our approach is trained in an entirely end-to-end manner, solely based on the information of link existence. Therefore, intermediate representations, such as the refined vector embeddings, are determined completely data-driven.

4. Experiments and Results

In this section, we demonstrate the performance of our proposed GAV framework on the ogbl-vessel benchmark [17] and on additional datasets sourced from publicly available flow-driven spatial networks. We first elaborate on baseline algorithms and the experimental setup, followed by a detailed description of datasets utilized in our experiments. Finally, we report quantitative results, investigate our design choices by conducting extensive ablation studies, and discuss GAV's interpretability in detail. Additional experiments on non-flow-based benchmarks and roto-translational invariance can be found in the supplementary.

4.1. Baselines and Experimental Setup

To evaluate GAV properly, we not only compare GAV to algorithms submitted to the ogbl-vessel benchmark but also propose a novel *secondary baseline* (SEAL+EdgeConv) combining the SEAL framework [49, 51] with the EdgeConv message-passing layer [41], following recent trends in graph-based object detection from point clouds [8, 40, 45, 47]. Extensive experiments with different baseline algorithms revealed that this provides us with an improved, highly competitive secondary baseline for link prediction in spatial networks. We, therefore, compare GAV on the additionally sourced datasets to SEAL's original version (with tuned hyperparameters), which has shown to deliver results on par with or superior to the state-of-the-art on multiple link prediction benchmarks, and to our introduced secondary baseline SEAL+EdgeConv. Details on the configuration of our secondary baseline can be found in the supplementary (see Supp. H).

GAV was trained using the Adam optimizer [23] with a learning rate of 0.001 and a batch size of 32 on a single Tesla V100 GPU (32 GB) until convergence. An ablation study on the number of hops h in the subgraph extraction module and the number of message-passing iterations k indicates that setting both to one is sufficient (see Table 5). In the GAV layer, the number of heads of the multi-head attention operation is set to 4, while $\phi_\theta^{(1)}$ represents a single linear layer with an output dimension of $d_{\text{message}} = 32$, and $\phi_\theta^{(2)}$ is given by a two-layer MLP with a hidden dimension of 64. The GAV layer makes use of leaky ReLU non-linearities [29] to increase gradient flow and simplify weight initialization. The readout module's learnable function $\phi_\theta^{(3)}$ is represented by a two-layer MLP with a hidden dimension of 128. All hyperparameters were tuned on the validation set of the ogbl-vessel benchmark.

4.2. Datasets

We experiment with multiple 2D and 3D flow-driven spatial networks to demonstrate the generalizability of our approach (see Table 1). In total, we conduct experiments on eight networks, including the ogbl-vessel benchmark and seven additionally sourced datasets, representing whole-brain vessel graphs of different mouse strains and road networks of various European countries. In this context, link prediction can be interpreted as predicting the probability of the existence of blood vessels and road segments. Whole-brain vessel graphs and road networks are graphically visualized in the supplementary (see Supp. E).

Whole-Brain Vessel Graphs Blood vessels represent fascinating structures forming complex networks that transport oxygen and nutrients throughout the human body. The vascular system is, therefore, intuitively represented as a flow-driven spatial network, where branching points of ves-

Table 2. Quantitative results. GAV outperforms the previous state-of-the-art across all metrics and datasets. We report mean and standard deviation values on the ogbl-vessel benchmark based on ten different random seeds (0 - 9). To compare GAV to existing baseline algorithms, we report quantitative results based on the area under the receiver operating characteristic curve (AUC). We additionally utilize the evaluation metric Hits@k as an additional, stricter performance measure. More details on evaluation metrics can be found in the supplementary (see Supp. G). Please note that the ogbl-vessel benchmark's official evaluation metric is AUC. Therefore, Hits@k values of participating algorithms are not available. We highlight the respective three best quantitative values in shades of teal.

Dataset	Model	# Params ↓	AUC ↑ (%)	Hits@100 ↑ (%)	Hits@50 ↑ (%)	Hits@20 ↑ (%)
ogbl-vessel	GCN [24]	396,289	43.53 ± 9.61	-	-	-
	MLP	1,037,577	47.94 ± 1.33	-	-	-
	Adamic-Adar [1]	0	48.49 ± 0.00	-	-	-
	GraphSAGE [15]	396,289	49.89 ± 6.78	-	-	-
	SAGE+JKNet [44]	273	50.01 ± 0.07	-	-	-
	SGC [42]	897	50.09 ± 0.11	-	-	-
	LRGA [33]	26,577	54.15 ± 4.37	-	-	-
	SEAL [51]	172,610	80.50 ± 0.21	-	-	-
	S3GRL (PoS$^+$) [27]	2,382,849	80.56 ± 0.06	-	-	-
	SUREL+ [46]	56,353	84.96 ± 0.68	-	-	-
	SIEG [14]	752,716	87.98 ± 1.00	-	-	-
	SEAL+EdgeConv (ours)	49,346	97.53 ± 0.32	16.09 ± 10.48	9.37 ± 6.18	4.99 ± 4.24
	GAV (ours)	8,194	98.38 ± 0.02	34.77 ± 0.94	28.02 ± 1.58	19.71 ± 2.31
c57-tc-vessel	SEAL [51]	43,010	78.21	0.12	0.06	0.01
	SEAL+EdgeConv (ours)	49,346	97.23	16.71	10.39	5.01
	GAV (ours)	8,194	98.24	33.26	26.89	21.32
cd1-tc-vessel	SEAL [51]	43,010	83.60	0.27	0.16	0.06
	SEAL+EdgeConv (ours)	49,346	97.91	17.05	11.57	2.98
	GAV (ours)	8,194	98.72	35.82	27.25	17.23
c57-cc-vessel	SEAL [51]	43,010	83.75	0.65	0.44	0.24
	SEAL+EdgeConv (ours)	49,346	97.49	7.21	3.35	1.06
	GAV (ours)	8,194	97.99	18.90	14.58	9.04
belgium-road	SEAL [51]	43,010	86.73	1.25	0.68	0.30
	SEAL+EdgeConv (ours)	49,346	96.98	0.55	0.55	0.51
	GAV (ours)	8,194	99.29	47.44	38.60	22.11
italy-road	SEAL [51]	43,010	90.07	0.32	0.16	0.08
	SEAL+EdgeConv (ours)	49,346	90.24	0.26	0.17	0.07
	GAV (ours)	8,194	99.41	28.49	20.08	11.99
netherlands-road	SEAL [51]	43,010	84.19	0.00	0.00	0.00
	SEAL+EdgeConv (ours)	49,346	96.06	3.91	2.20	1.01
	GAV (ours)	8,194	99.44	37.55	26.97	10.77
luxembourg-road	SEAL [51]	43,010	89.79	11.39	6.15	3.12
	SEAL+EdgeConv (ours)	49,346	97.53	59.79	39.15	19.42
	GAV (ours)	8,194	99.31	85.88	76.84	61.95

sels typically represent nodes embedding x-, y-, and z-coordinates, while edges are defined as blood vessels running between branching points [32]. We report results on the Open Graph Benchmark's ogbl-vessel benchmark [17], which measures the performance of different link prediction algorithms with regard to whole-brain vessel graphs aiming to remove artifacts introduced by the multi-stage graph generation process. The ogbl-vessel benchmark consists of millions of nodes and edges (see Table 1) and describes the murine brain vasculature all the way down to the microcapillary level. However, we not only experiment with the ogbl-vessel benchmark but also source three additional whole-brain vessel graphs of different mouse strains acquired via different imaging methodologies [38, 39] (see Table 1, footnote).

Road Networks Further, we report results on diverse road networks representative of four European countries for a thorough evaluation of our proposed GAV framework's generalizability. To this end, we adopt publicly available road networks introduced in the DIMACS graph partitioning and clustering challenge [3]. These road networks correspond to the largest connected components of OpenStreetMap's [31] road networks and are vastly different in size (*e.g.*, luxembourg-road constitutes roughly 100,000 edges, whereas italy-road has more than 7,000,000). In road networks, intersections and locations with stronger curvature represent nodes in the form of x- and y-coordinates, while connecting roads represent edges.

Preprocessing Link prediction datasets require positive (label 1) and negative links (label 0). Positive links correspond to existent edges in our datasets, whereas negative links represent artificially created, non-existent edges. As link prediction algorithms are commonly employed to improve the graph representation through the identification of

absent connections and the reduction of local noise arising from graph generation, negative links should appear as authentic as possible. In light of the absence of negative links in our sourced datasets, we prepare our sourced datasets in a manner that aligns with the ogbl-vessel benchmark. Therefore, we sample negative links using a spatial sampling strategy. To be precise, we randomly connect nodes in close proximity, taking a maximum distance threshold of $\delta = \overline{e_{ij}} + 2\sigma$ into account. Here, $\overline{e_{ij}}$ denotes the average edge length estimated over the entire graph $\mathcal{G}$ and σ the standard deviation. The number of negative, sampled links corresponds to the number of positive, real links across all datasets. We finally split positive and negative links into training, validation, and test sets (split 80%/10%/10%).

4.3. Quantitative Results

GAV demonstrates excellent, superior performances on the task of link prediction across all metrics and datasets, as can be observed in Table 2. We outperform the current state-of-the-art algorithm SIEG on the ogbl-vessel benchmark by **12% (98.38 vs. 87.98 AUC)** while requiring a significantly smaller amount of parameters (8,194 vs. 752,716). However, GAV not only drastically outperforms all algorithms submitted to the ogbl-vessel benchmark but also our introduced strong, secondary baseline, combining the SEAL framework with EdgeConv (SEAL+EdgeConv). The excellent performance and superiority of GAV is even more pronounced when considering the strict evaluation metric of Hits@k. Quantitative results reported in Table 2 additionally indicate the strong performance of our introduced secondary baseline (see Section 4.1).

It is of note that the luxembourg-road dataset's test set contains only 12,000 negative links. We, therefore, compare predictions of its positive links to 12,000 rather than 100,000 negative links (see supplementary, Supp. G). This explains the comparatively strong Hits@k performances on the luxembourg-road dataset.

4.4. Ablation Studies

To further validate GAV, we conduct detailed ablation studies on the validation set of the ogbl-vessel benchmark. Additional ablation studies on the GAV layer and its design choices can be found in the supplementary (Supp. F). Table 3 investigates the importance of the readout module, the message-passing module, and the labeling trick. First,

Table 3. Ablations on main design choices.

Readout Module	Message-Passing	Labeling Trick	AUC ↑	Δ
✓	✓	✓	98.39	–
✗	✓	✓	98.28	-0.11
✓	✗	✓	80.56	-17.83
✓	✓	✗	96.00	-2.39

we exchange our readout module with a SortPooling layer followed by two convolutional layers and an MLP, resembling SEAL's readout operation. We note that our readout module is more applicable to flow-driven spatial networks, as it leads to a modest AUC increase of 0.11. Second, we completely deactivate the message-passing module by forwarding $\mathcal{L}(\mathcal{G}_h^t)$ directly to the readout module. We observe a drastic AUC decrease of 17.83, indicating the importance of modifying the vector embeddings via our proposed GAV layer. Finally, we evaluate the impact of the labeling trick. Excluding the additional labels generated by the labeling trick results in an AUC decrease of 2.39, proving the significance of link identification via additional, distinct labels.

In a second ablation study, we experiment with different message-passing layers (see Supp. L), including EdgeConv, in our message-passing module. We report our findings in Table 4. Our proposed GAV layer outperforms the other

Table 4. Ablations with different message-passing layers.

Message-Passing Layer	AUC ↑	Hits@100 ↑	Hits@50 ↑	Hits@20 ↑
GAV layer (ours)	98.39	34.46	26.30	19.81
EdgeConv [41]	97.43	17.30	5.97	0.78
GAT layer [6]	96.44	4.58	2.55	1.59
SAGE layer [15]	93.53	0.77	0.11	0.03
GCN layer [24]	89.31	0.39	0.22	0.16

message-passing layers across all metrics by a considerable amount, proving our rationale.

Lastly, we vary the number of hops h used to generate $\mathcal{G}_h^t$ and the number of message-passing iterations k (see Table 5). We observe that simultaneously increasing k and h

Table 5. Ablations on k and h.

k & h	AUC ↑	Hits@100 ↑	Hits@50 ↑	Hits@20 ↑
1	98.39	34.46	26.30	19.81
2	98.39	34.00	26.93	21.21
3	98.39	34.25	25.99	17.84

results in no discernible differences in performance. This finding is in line with the γ-decaying theory [49], proving the approximability of high-order heuristics from locally restricted subgraphs.

4.5. Interpretability and Analysis of Results

In Fig. 5, we visualize the behavior of our proposed GAV layer, attempting to facilitate interpretability and provide a clearer insight on how it creates superior representations for link prediction in flow-driven spatial networks. An analysis based on a toy example and additional visualizations similar to Fig. 5 are provided in the supplementary (Supp. D & B).

Scalar multiplication via s_i enables GAV to flip individual vector embeddings (mimicking *change in direction of flow*). We observe that this results in physically implausible, learned sink/source flow between the two target nodes (red and green) of implausible formations (see Fig. 5, second

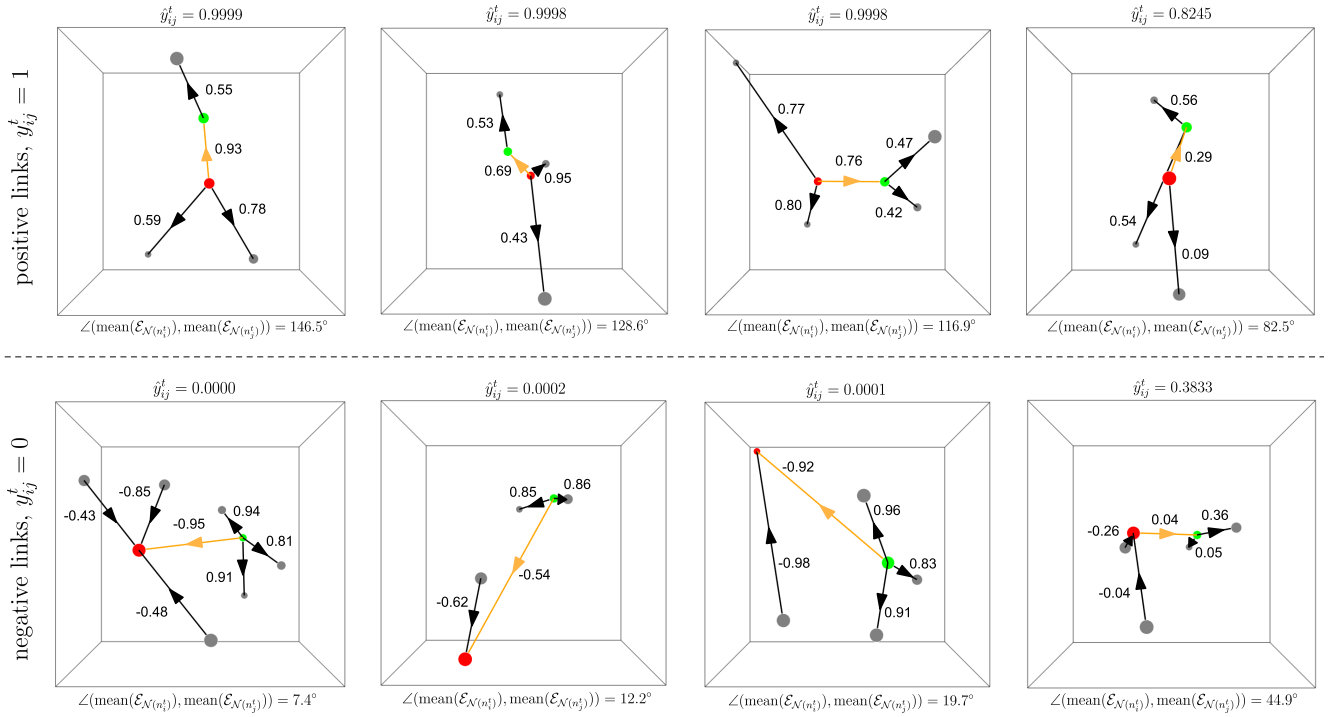

Figure 5. Visualization of the effect of our GAV layer on vector embeddings (ogbl-vessel). We visualize subgraph representations $\mathcal{G}_h^t$ (h set to one) of four positive, plausible (first row) and four negative, implausible target links (second row) together with the GAV layer's predicted scalar values $s_i \in (-1, 1)$. The scalar values s_i used to update vector embeddings in $\mathcal{L}(\mathcal{G}_h^t)$ have been projected to $\mathcal{G}_h^t$ to provide an interpretable visualization. The directionality of edges already incorporates potential shifts in direction enforced by our GAV layer. Please note that following the color coding scheme of Fig. 2, the target link e_{ij}^t is depicted in orange, whereas the two target nodes n_i^t and n_j^t are displayed in red and green. We additionally report the angle $\angle$ between vector embeddings aggregated around the two target nodes (see Section 3.4) and the predicted probability of link existence $\hat{y}_{ij}^t$. The last column contains more challenging cases.

row; Fig. 7, supplementary) with a consistency of 94.26%, which stands in drastic contrast to the behavior of physical flow in spatial networks. The learned sink/source flow between target nodes of negative links, in turn, leads to drastically different representations for implausible and plausible formations (see Fig. 5, first row) that can be effortlessly classified in our readout module. The difference in representation is also reflected in the angle $\angle$ between vector embeddings aggregated around the two target nodes represented by $\text{mean}(\mathcal{E}_{\mathcal{N}(n_i^t)})$ and $\text{mean}(\mathcal{E}_{\mathcal{N}(n_j^t)})$, constructed in our readout module (see Section 3.4). We find that larger angles are more frequently associated with positive and smaller angles with negative predictions (see Fig. 5).

We further attempt to interpret the effect of scaling vector embeddings (mimicking *change in magnitude of flow*). In this context, we hypothesize that small $|s_i|$ may indicate that GAV is uncertain whether to flip vector embeddings, which is a necessity for sink/source flow. Therefore, $|s_i|$ could be interpreted as a measure of certainty (see Fig. 5, last column). This is confirmed by statistical analysis of $|s_i|$, demonstrating that high values ($\mu_{|s_i|} = 0.62$) are assigned to more ($\hat{y}_{ij}^t > 0.9$ or < 0.1) and low values ($\mu_{|s_i|} = 0.13$) to less certain predictions ($0.4 < \hat{y}_{ij}^t < 0.6$).

5. Outlook and Conclusion

In this work, we propose the simple yet effective Graph Attentive Vectors (GAV) link prediction framework. GAV relies on the idea of modeling simplified physical flow by updating vector embeddings in a constrained manner, which intuitively models the underlying physical process and, therefore, presents a strong inductive bias for link prediction in flow-driven spatial networks. This allows GAV to outperform the previous state-of-the-art by an impressive margin on all metrics across multiple whole-brain vessel and road network datasets while requiring a significantly smaller amount of trainable parameters. GAV's imitation of the dynamics of physical flow, however, represents a simplified concept, which is not entirely representative of physical principles from, *e.g.*, fluid dynamics (see Fig. 5). Future work should, therefore, aim to further investigate GAV's parameters and extend its assumptions by incorporating different physical principles, such as conservation of mass and momentum, resulting in vector embeddings highly representative of physical flow in flow-driven spatial networks.

References

[1] Lada A Adamic and Eytan Adar. Friends and neighbors on the web. *Social networks*, 25(3):211–230, 2003. 2, 6

[2] Jun Ai, Yayun Liu, Zhan Su, Hui Zhang, and Fengyu Zhao. Link prediction in recommender systems based on multi-factor network modeling and community detection. *Europhysics Letters*, 126(3):38003, 2019. 2

[3] David A Bader, Henning Meyerhenke, Peter Sanders, and Dorothea Wagner. 10th dimacs implementation challenge-graph partitioning and graph clustering, 2011. 5, 6

[4] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999. 2

[5] Marc Barthélemy. Spatial networks. *Physics reports*, 499(1-3):1–101, 2011. 1

[6] Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? *arXiv preprint arXiv:2105.14491*, 2021. 2, 7

[7] Lei Cai, Jundong Li, Jie Wang, and Shuiwang Ji. Line graph neural networks for link prediction. *IEEE TPAMI*, 44(9):5103–5113, 2021. 2, 4

[8] Yuning Chai, Pei Sun, Jiquan Ngiam, Weiyue Wang, Benjamin Caine, Vijay Vasudevan, Xiao Zhang, and Dragomir Anguelov. To the point: Efficient 3d object detection in the range image with graph convolution kernels. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2021. 5

[9] Tomasz Danel, Przemysław Spurek, Jacek Tabor, Marek Śmieja, Łukasz Struski, Agnieszka Słowik, and Łukasz Maziarka. Spatial graph convolutional networks. In *Neural Information Processing: 27th International Conference, ICONIP 2020, Bangkok, Thailand, November 18–22, 2020, Proceedings, Part V*, pages 668–675. Springer, 2020. 2

[10] Nur Nasuha Daud, Siti Hafizah Ab Hamid, Muntadher Saadoon, Firdaus Sahran, and Nor Badrul Anuar. Applications of link prediction in social networks: A review. *Journal of Network and Computer Applications*, 166:102716, 2020. 2

[11] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016. 2

[12] Matthias Fey, Jan Eric Lenssen, Frank Weichert, and Heinrich Müller. Splinecnn: Fast geometric deep learning with continuous b-spline kernels. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 869–877, 2018. 2

[13] Francois Fouss, Alain Pirotte, Jean-Michel Renders, and Marco Saerens. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Transactions on knowledge and data engineering*, 19(3):355–369, 2007. 2

[14] GitHub. SIEG: Structural Information Enhanced Graph Representation. github.com/anonymous20221001/SIEG_OGB, 2023. 2, 3, 4, 6

[15] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017. 2, 6, 7

[16] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 639–648, 2020. 2

[17] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *arXiv preprint arXiv:2005.00687*, 2020. 1, 2, 5, 6

[18] Zan Huang, Xin Li, and Hsinchun Chen. Link prediction approach to collaborative filtering. In *Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*, pages 141–142, 2005. 2

[19] Paul Jaccard. Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bull Soc Vaudoise Sci Nat*, 37:547–579, 1901. 2

[20] Weiwei Jiang and Jiayun Luo. Graph neural network for traffic forecasting: A survey. *Expert Systems with Applications*, 207:117921, 2022. 1

[21] Chuanze Kang, Han Zhang, Zhuo Liu, Shenwei Huang, and Yanbin Yin. Lr-gnn: a graph neural network based on link representation for predicting molecular associations. *Briefings in Bioinformatics*, 23(1):bbab513, 2022. 2

[22] Leo Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953. 2

[23] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[24] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. 2, 6, 7

[25] Chengwei Lei and Jianhua Ruan. A novel link prediction algorithm for reconstructing protein–protein interaction networks by topological similarity. *Bioinformatics*, 29(3):355–364, 2013. 2

[26] David Liben-Nowell and Jon Kleinberg. The link prediction problem for social networks. In *Proceedings of the twelfth international conference on Information and knowledge management*, pages 556–559, 2003. 2

[27] Paul Louis, Shweta Ann Jacob, and Amirali Salehi-Abari. Simplifying subgraph representation learning for scalable link prediction. *arXiv preprint arXiv:2301.12562*, 2023. 6

[28] Zhi-Gang Lu and Qian Chen. Discovering potential partners via projection-based link prediction in the supply chain network. *International Journal of Computational Intelligence Systems*, 13(1):1253–1264, 2020. 2

[29] Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3. Atlanta, Georgia, USA, 2013. 5

[30] Tsuyoshi Murata and Sakiko Moriyasu. Link prediction of social networks based on weighted proximity measures. In

IEEE/WIC/ACM International Conference on Web Intelligence (WI'07), pages 85–88. IEEE, 2007. 2

[31] OpenStreetMap contributors. Planet dump retrieved from planet.osm.org. openstreetmap.org, 2017. 6

[32] Johannes C Paetzold, Julian McGinnis, Suprosanna Shit, Ivan Ezhov, Paul Büschl, Chinmay Prabhakar, Anjany Sekuboyina, Mihail Todorov, Georgios Kaissis, Ali Ertürk, et al. Whole brain vessel graphs: A dataset and benchmark for graph learning and neuroscience. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. 1, 5, 6

[33] Omri Puny, Heli Ben-Hamu, and Yaron Lipman. Global attention improves graph networks generalization. *arXiv preprint arXiv:2006.07846*, 2020. 6

[34] Franca Schmid, Giulia Conti, Patrick Jenny, and Bruno Weber. The severity of microstrokes depends on local vascular topology and baseline perfusion. *Elife*, 10:e60208, 2021. 1

[35] Matthias Schneider, Johannes Reichold, Bruno Weber, Gábor Székely, and Sven Hirsch. Tissue metabolism driven arterial tree generation. *Medical image analysis*, 16(7):1397–1414, 2012. 1

[36] Zachary Stanfield, Mustafa Coşkun, and Mehmet Koyutürk. Drug response prediction as a link prediction problem. *Scientific reports*, 7(1):40321, 2017. 2

[37] Nitish Talasu, Annapurna Jonnalagadda, S Sai Akshaya Pillai, and Jampani Rahul. A link prediction based approach for recommendation systems. In *2017 international conference on advances in computing, communications and informatics (ICACCI)*, pages 2059–2062. IEEE, 2017. 2

[38] Mihail Ivilinov Todorov, Johannes Christian Paetzold, Oliver Schoppe, Giles Tetteh, Suprosanna Shit, Velizar Efremov, Katalin Todorov-Völgyi, Marco Düring, Martin Dichgans, Marie Piraud, et al. Machine learning analysis of whole mouse brain vasculature. *Nature methods*, 17(4):442–449, 2020. 6

[39] Thomas Wälchli, Jeroen Bisschop, Arttu Miettinen, Alexandra Ulmann-Schuler, Christoph Hintermüller, Eric P Meyer, Thomas Krucker, Regula Wälchli, Philippe P Monnier, Peter Carmeliet, et al. Hierarchical imaging and computational analysis of three-dimensional vascular network architecture in the entire postnatal and adult mouse brain. *Nature Protocols*, 16(10):4564–4610, 2021. 5, 6

[40] Yue Wang and Justin M Solomon. Object dgcnn: 3d object detection using dynamic graphs. *Advances in Neural Information Processing Systems*, 34:20745–20758, 2021. 5

[41] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. 2, 5, 7

[42] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *International conference on machine learning*, pages 6861–6871. PMLR, 2019. 6

[43] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018. 2

[44] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International conference on machine learning*, pages 5453–5462. PMLR, 2018. 6

[45] Honghui Yang, Zili Liu, Xiaopei Wu, Wenxiao Wang, Wei Qian, Xiaofei He, and Deng Cai. Graph r-cnn: Towards accurate 3d object detection with semantic-decorated local graph. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part VIII*, pages 662–679. Springer, 2022. 5

[46] Haoteng Yin, Muhan Zhang, Jianguo Wang, and Pan Li. Surel+: Moving from walks to sets for scalable subgraph-based graph representation learning. *arXiv preprint arXiv:2303.03379*, 2023. 2, 6

[47] Junbo Yin, Jianbing Shen, Chenye Guan, Dingfu Zhou, and Ruigang Yang. Lidar-based online 3d video object detection with graph-based message passing and spatiotemporal transformer attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11495–11504, 2020. 5

[48] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? *Advances in Neural Information Processing Systems*, 34:28877–28888, 2021. 2

[49] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. *Advances in neural information processing systems*, 31, 2018. 2, 3, 5, 7

[50] Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. An end-to-end deep learning architecture for graph classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018. 2

[51] Muhan Zhang, Pan Li, Yinglong Xia, Kai Wang, and Long Jin. Labeling trick: A theory of using graph neural networks for multi-node representation learning. *Advances in Neural Information Processing Systems*, 34:9061–9073, 2021. 2, 3, 4, 5, 6

[52] Zheng Zhang and Liang Zhao. Representation learning on spatial networks. *Advances in Neural Information Processing Systems*, 34:2303–2318, 2021. 2

[53] Zi-Ke Zhang, Chuang Liu, Yi-Cheng Zhang, and Tao Zhou. Solving the cold-start problem in recommender systems with social tags. *Europhysics Letters*, 92(2):28002, 2010. 2

[54] Tao Zhou, Linyuan Lü, and Yi-Cheng Zhang. Predicting missing links via local information. *The European Physical Journal B*, 71:623–630, 2009. 2