Exploring a Graph Regression Problem in River Networks*

Benjamin Fankhauser[1][0000−0002−7982−2669], Vidushi Bigler[2][0000−0001−6043−8264], and Kaspar Riesen[1][0000−0002−9145−3157]

[1] Institute of Computer Science, University of Bern, Bern, Switzerland
{benjamin.fankhauser,kaspar.riesen}@unibe.ch
[2] Institute for Optimisation and Data Analysis, Bern University of Applied Sciences, Biel, Switzerland
vidushi.bigler@bfh.ch

Abstract. In order to monitor climate change in Switzerland, the Federal Office for the Environment has been measuring river water temperature at 81 water stations for over 40 years. Based on these measurements and the underlying river network, we create a novel graph regression problem with a total of 3,480 graphs. The task is to predict the water temperature of output nodes by using a subset of input nodes. Depending on the subset, this creates challenging information bottlenecks and long-range dependencies. In a first contribution, we set RMSE baselines for four standard Graph Neural Networks (GNNs). Namely, we employ and compare GCN, GIN, GAT, and GraphSAGE architectures with up to 7 layers. In a second contribution, we analyze how noise propagates through such networks. In our evaluation, we observe that the GNN models degenerate more severely than alternative architectures. This empirical result shows that the message passing framework can harm models in presence of noise, a rarely mentioned limitation we would like to address in future research.

Keywords: River water temperature · Graph regression · Graph Neural Network · Noise propagation

1 Introduction

River water temperature is an interesting and challenging environmental variable [1, 2]. On the way towards the sea, river water temperatures are affected in several ways. First, there is solar radiation, which heats up the river bed and particles in the water. The smaller the river is (which is usually the case with sidearms of main rivers), the greater the effect of solar radiation becomes. However, these sidearms add up to total catchment areas which in turn influence the main river substantially. Further meteorological influences are, for instance, atmospheric temperature exchange on the surface, rain water, or snow and glacier

* Supported by Swiss National Science Foundation (SNSF) Grant Nr. PT00P2_206252. Data are kindly provided by the Federal Office for the Environment.

melt. The latter mainly affects the Alpine catchment area. A second group of influences are of anthropogenic source. For example, the climate regime shift in the late 1980s lead to a sudden increase in water temperature [3, 4]. Further, rainfall typically transfers a certain amount of the heat of a city into the nearest water body. Other anthropogenic influences are nuclear and hydrological power plants, as well as any construction projects close to, or inside of, rivers.

Switzerland has a rich water body and a broad mixture of the above mentioned influencing factors. In the Alpine regions, for instance, there is large influence of snow and glacier melt, as well as some of the largest hydrological power plants of Europe. Downstream, the landscape is hilly and the central plateau is home to large and slow flowing flatland rivers. On this central plateau we also have larger cities that potentially influence the water temperatures. Further downstream, the rivers merge before they flow into neighboring countries. The four major rivers of Switzerland are called Rhine, Rhone, Ticino and Inn.

The water temperature of rivers in Switzerland has been monitored by the *Federal Office for the Environment* (FOEN) for over 40 years. The FOEN measures the water temperatures at so called water stations in 10 minute intervals and report them as daily averages. The temperature sensors at each water station are regularly controlled and calibrated. In order to improve the monitoring on recent climate change, the FOEN doubled the number of water measurement stations from approximately 40 to 81 between the years 2000 and 2010.

In literature various methods have been proposed to model river water temperature. Air2Stream [5], for instance, is a physically inspired statistical model which uses the air temperature and river discharge to model river water temperature. Since modeling river water temperature is a time series problem, various LSTM [6] models haven been proposed for this scenario. Usually, these LSTMs model the relationship between meteorological variables (usually the air temperature) and river water temperature [7, 8]. With advances in deep learning, more sophisticated architectures have been proposed [9–11], as well as the use of neighboring water stations [12], or spatio-temporal neural networks combining LSTMs with Graph Neural Networks (GNNs) [13].

In general, we observe that the models for water temperature modeling are getting more and more complex in terms of input data and architecture. Yet, we also see that these models only marginally improve the current state of the art. While an improved predictive performance is generally beneficial, it is at least questionable whether the high complexity of the models is worth the little progress. At the same time, we have the impression that GNNs seem to underperform in this specific scenario and that there may still be potential for more substantial improvements.

To counteract this trend, we propose in the present paper a novel graph regression dataset that is focused on the relationship of river water temperature at different water stations. In this dataset, we get rid of all temporal and meteorological variables in the proposed problem. To achieve this, we define a graph for different points in time in the time series, with the corresponding water temperatures labeled on each node, and thus obtain thousands of node-labeled graphs.

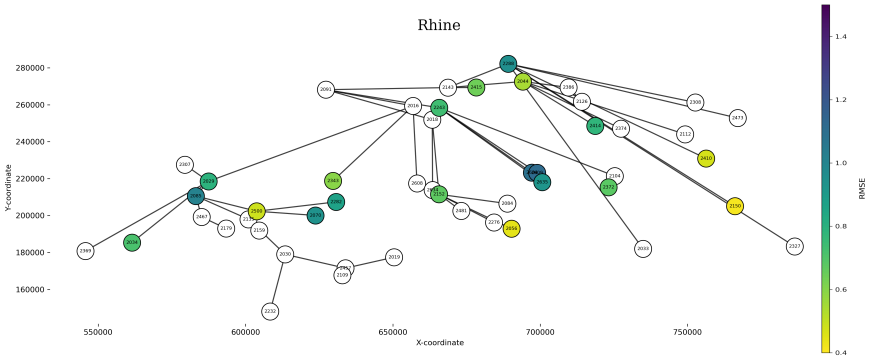

Fig. 1: Water temperature stations of the river Rhine represented as a graph structure. Non-colored nodes are input nodes, while the output nodes are colored. The darker the node is, the more difficult it is to predict.

The problem is then formulated in such a way that missing node labels are to be derived on the basis of existing node labels. Solutions to the here stated problem will be of great importance both for applications in modeling the river water temperature as well as for graph-based pattern recognition in general.

The remainder of this paper is organized as follows. In Section 2, we introduce the novel graph regression problem and the corresponding dataset. In Section 3, we briefly review standard architectures for GNNs, which are in turn assessed in Section 4. Also in Section 4, we provide an ablation study where we thoroughly analyze the behavior of different models under noise, i.e. in situations when a node does not behave as it did during training. Finally, we conclude our work and propose future work in Section 5.

2 Graph Regression on the Swiss River Network

As proposed in [12], the flow of water in a river forms a natural graph structure that describes the connections between different water stations located among the river by means of nodes and edges. Each node represents a water station and edges between two nodes mean that the corresponding stations are connected by a river. In this work, a graph $G = (V, E, X)$ represents the state of a complete water body on a single day. Each node $v \in V$ in the graph is assigned a feature vector $X(v) = (x, y, Y, D, wt) \in \mathbb{R}^5$, which includes the x- and y-coordinates of the water station in LV95[3] format, the year of measurement (Y), the day of the year (D) ranging from 1 to 365[4], and the water temperature wt. In Fig. 1, for instance, we illustrate the graph structure for the river Rhine.

For the river Rhine, the daily measurements from 1980 to 2021 result in 14,975 graphs, each containing the same 49 nodes and the same edge structure.

[3] LV95 is the Swiss coordinate system where 1 unit corresponds to 1 meter.
[4] Respectively, 366 on leap years.

To ensure completeness, all graphs with at least one node where the water temperature is missing are excluded. This yields in a final dataset of 3,480 graphs where each node $v \in V$ has a valid label $X(v)$.

The set of nodes V for each graph G is divided into two disjoint subsets, viz. input nodes $V^{(i)}$ and output nodes $V^{(o)}$, where $V^{(i)} \cap V^{(o)} = \emptyset$. The goal of the proposed regression task is to predict the water temperature of the output nodes using the measured water temperatures at the input nodes operating on the complete graph $G = (V, E, X)$. Formally, the task is to model a function $f : \mathbb{G} \to \hat{\mathbb{G}}$ such that for each output node $v^{(o)} \in V^{(o)}$, the predicted water temperature $\hat{wt}$ matches the true water temperature wt as accurate as possible.

The difficulty of the proposed task crucially depends on the choice of the input-output split. For this study, we define a fixed split with 40% of the nodes belonging to $V^{(o)}$. Moreover, we assure that the number of the edges of the shortest path from any output node to its nearest input node is maximum 3. This setup introduces specific challenges, such as long-range dependencies and potential information bottlenecks. Fig. 1 illustrates this fixed split into input and output nodes and highlights the different degrees of difficulties in prediction with different colors.

3 Related Work

The problem formulated in Section 2 corresponds to a transductive learning problem on the node labels of a graph. Due to the irregular structure of the underlying graph, message passing GNNs might be a suitable choice for setting first benchmark results [14]. Hence, we present in this paper the results of four standard GNNs briefly reviewed in the next subsections.

All GNNs used in the present paper use specific versions of the generalized message passing framework defined by

$$\boldsymbol{x}_i^{(m)} = \text{update}(\boldsymbol{x}_i^{(m-1)}, \text{aggregate}_{v_j \in \mathcal{N}(v_i)}(g(\boldsymbol{x}_j^{(m-1)}, \boldsymbol{x}_i^{(m-1)}))). \tag{1}$$

For each node $v_i \in V$ the state is initialized with the original node features $\boldsymbol{x}_i^{(0)} = X(v_i)$. A new state $\boldsymbol{x}_i^{(m)}$ is computed by the update function depending on the previous state $\boldsymbol{x}_i^{(m-1)}$ and an aggregation of the states of the neighbors $v_j \in \mathcal{N}(v_i)$. The aggregation function itself is usually implemented as mean or sum of the transformation function g.

3.1 Graph Convolution Neural Network (GCN)

The GCN [15] uses the following implementation of the message passing algorithm (with an edge weight of 1).

$$\boldsymbol{x}_i^{(m)} = \boldsymbol{W}^T \sum_{v_j \in \mathcal{N}(v_i) \cup \{v_i\}} \frac{1}{\sqrt{\hat{d}_j \hat{d}_i}} \boldsymbol{x}_j^{(m-1)},$$

where $\hat{d}_i = 1 + deg(v_i)$ is used for the degree-normalization and $\boldsymbol{W}$ is a matrix with learnable weights.

3.2 Graph Isomorphism Neural Network (GIN)

The GIN [16] improves the expressiveness of the GCN by approximating an injective multiset aggregation function using Multilayer Perceptrons (MLPs). In its reduced form it uses the following implementation of the message passing algorithm.

$$\boldsymbol{x}_i^{(m)} = MLP\left((1+\epsilon)\boldsymbol{x}_i^{(m-1)} + \sum_{v_j \in \mathcal{N}(v_i)} \boldsymbol{x}_j^{(m-1)}\right),$$

where $\epsilon \in \mathbb{R}$ is a free parameter.

3.3 Graph Attention Networks (GAT)

The GAT [17] extends the message passing framework by incorporating attention mechanisms to compute dynamic weights for neighbors during the aggregation step as follows.

$$\boldsymbol{x}_i^{(m)} = \sigma\left(\sum_{v_j \in \mathcal{N}(v_i) \cup \{v_i\}} \alpha_{ij} \boldsymbol{W} \boldsymbol{x}_j^{(m-1)}\right),$$

where $\alpha_{ij} = softmax(e_{ij})$, and $e_{ij} = \text{LeakyReLU}\left(\boldsymbol{a}^T \left[\boldsymbol{W}\boldsymbol{x}_i^{(m-1)} \parallel \boldsymbol{W}\boldsymbol{x}_j^{(m-1)}\right]\right)$.

Here, $\boldsymbol{W}$ is a shared learnable weight matrix applied to transform node features, $\boldsymbol{a}$ is the attention vector, $\parallel$ denotes concatenation, and α_{ij} is the normalized attention coefficient that determines the importance of node v_j to v_i.

Further, GAT introduces multi-head attention, where K independent attention mechanisms are applied, and their outputs are concatenated as follows.

$$\boldsymbol{x}_i^{(m)} = \Big\|_{k=1}^{K} \sigma\left(\sum_{v_j \in \mathcal{N}(v_i) \cup \{v_i\}} \alpha_{ij}^{(k)} \boldsymbol{W}^{(k)} \boldsymbol{x}_j^{(m-1)}\right)$$

3.4 Graph Sample and Aggregate (GraphSAGE)

GraphSAGE [18] introduces an inductive approach to learning node embeddings by sampling and aggregating features from a fixed number of neighbors rather than using the entire neighborhood. Formally,

$$\boldsymbol{x}_i^{(m)} = \sigma\left(\boldsymbol{W} \cdot \text{aggregate}_{v_j \in \tilde{\mathcal{N}}(v_i)}\left(\boldsymbol{x}_j^{(m-1)}\right)\right) ,$$

Here, the key distinction lies in the neighborhood set $\tilde{\mathcal{N}}(v_i)$, which is fixed in size and uniformly drawn from the neighbors $\mathcal{N}(v_i)$. For the aggregation function three strategies are proposed, viz. Mean Aggregation, Pooling Aggregation, and LSTM Aggregation (in our evaluation we use Mean Aggregation).

4 Experimental Evaluation

In this study we assess the performance of the previously presented standard GNNs on the proposed graph regression problem (stated in Section 2). The goal of our evaluation is twofold. First, we want to use these experiments to disseminate initial benchmark results. Second, we present a specific noise study to show that standard GNNs might have undesirable effects in certain situations when applied to the newly formulated problem.

4.1 Experimental Setup

For each of the basic architectures described in Section 3.1 to 3.4, we create models as follows. The first layer does message passing and maps the input features into the hidden space. Each following layer operates in the hidden space. After the last layer, a linear transformation maps the hidden space onto the water temperature prediction $\hat{wt}$. For the GAT model, which uses multi attention heads, the respective hidden size is adjusted accordingly. This generic construction has two hyperparameters, viz. the amount of layers and the hidden size.

All of the node features are min-max normalized. Further, the water temperature on output nodes – which is predicted by the models – is replaced with -1. The dataset is divided into training, validation, and test sets with relative split sizes of 64%, 16%, and 20%, respectively. For hyperparameter tuning we use an extensive random search based on ASHA scheduling [19]. During training, we minimize the *Mean Square Error* (MSE) on the training set, perform model selection based on the MSE of the validation set and report the *Root Mean Square Error* (RMSE) and the *Nash-Sutcliffe model Efficiency Coefficient* (NSE), both defined in Eq. (2), on the test set.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{wt}_i - wt_i)^2}, \qquad \text{NSE} = 1 - \frac{\sum_{t=1}^{T}(wt^{(t)} - \hat{wt}^{(t)})^2}{\sum_{t=1}^{T}(wt^{(t)} - \bar{wt})^2} \tag{2}$$

4.2 Baseline Models

To compare the performance of the GNN architectures, we employ three baseline models.

Table 1: RMSE performance on the output nodes on the hold out test set. The configuration specifies the architecture of the best performing model on the validation set. The last column is the amount of learnable parameters in the respective model.

Method	**RMSE**	**NSE**	**Configuration**			**Parameters**
			Layers	Hidden Size	Heads	
Average Neighbor	1.36	.887	-	-	-	-
No Edge GNN	1.45	.895	9	220	-	390,501
End-to-end MLP	0.55	.983	2	196	-	36,496
GCN	0.70	.974	7	162	-	159,571
GIN	0.76	.969	7	45	-	27,226
GAT	0.62	.978	5	48	5	235,057
GraphSAGE	0.61	.978	7	110	-	147,181

- Average Neighbor: This model computes the average water temperature of the nearest input nodes $v^{(i)} \in V^{(i)}$ for each output node $v^{(o)} \in V^{(o)}$. The distance used is the shortest path between two nodes.
- No Edge GNN: This MLP model $f : \mathbb{R}^4 \to \mathbb{R}$ predicts the water temperature $\hat{wt}$ based on the node features of a single output node $X(v^{(o)}) = (x, y, Y, D)$. This is an average prediction based on the location and date of the measurement and basically corresponds to a GNN without message passing.
- End-to-end MLP: This model predicts the output nodes by means of a function of all input nodes: $f : \mathbb{R}^{29\times5} \to \mathbb{R}^{20\times1}$. This model is only applicable on this specific split of input and output nodes and represents the very specific situation, where it is possible to not use the graph structure at all.

4.3 Test Results

After model selection on the validation set, we evaluate the best-performing model on the hold out test set (see Table 1). In our experiment the best performing GNN is GraphSAGE closely followed by GAT. In order to deal with the long-range dependencies, all GNN models perform the best with 5 to 7 layers.

The GNN architectures clearly outperform the first two reference systems w.r.t. both RMSE and NSE. However, the end-to-end MLP performs better in terms of RMSE, indicating that the river structure poses a challenging problem for standard GNN Architectures.

4.4 Noise Propagation Analysis

Recently, it has been shown that graph structures are not advantageous for all pattern recognition tasks [20]. Moreover, in [13] it is empirically verified that graph based models underperform on water temperature tasks. In the following

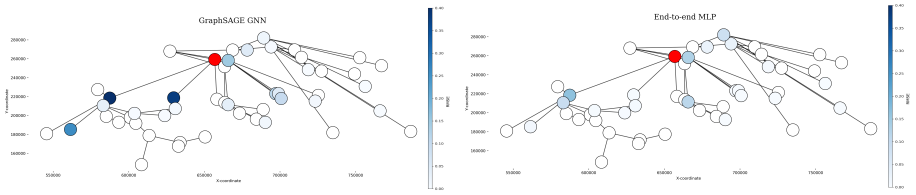

Fig. 2: On the left we show the best performing GraphSAGE model and on the right we show the end-to-end MLP. The water temperature of the input node 2016 (in red) is affected by Gaussian noise with a standard deviation of $\sigma = 2°C$. Visualized is the difference of the RMSE compared to the models without noise. A darker blue corresponds to a larger degeneration.

ablation study we want to analyze the special case of noisy message passing in the test set. While the message passing architecture is expected to improve the predictions of the output nodes, the question remains, how much damage can be caused by a single faulty input node?

In this ablation study we randomly select one specific input node (water station 2016) and introduce various amounts of Gaussian noise on the water temperature of this input node. We then deploy the trained GNNs on this noisy test set and analyze the reported RMSE degeneration over all output nodes.

Fig. 2 shows how a single noisy input node affects several output nodes (we apply a Gaussian noise with a standard deviation $\sigma = 2°C$). On the left, we see the effect of the noise on the neighboring nodes in the GraphSage model. The darker a node is colored, the greater its deviation from the RMSE compared to the model without the faulty node. The propagation of the error is clearly visible. If we compare the result with the end-to-end MLP (shown on the right of Fig. 2), we observe that the nearest nodes are also affected the most, but the end-to-end MLP is more robust under noise. That is, less output nodes are affected and in general less severe than with the GNN.

The quantitative analysis shown in Fig. 3 shows the RMSE (y-axis) as a function of the Gaussian noise (x-axis) for all GNNs and the end-to-end MLP. We increase the level of Gaussian noise by setting the standard deviation to 1, 2, and $4°C$. It is clearly visible that the end-to-end MLP is more robust to noise, while the best-performing GNNs quickly degrade.

5 Discussion and Conclusion

In the present paper, we introduce a novel graph regression task which is not based on temporal or meteorological dependencies (in contrast to common water temperature prediction datasets). The dataset consists of 3,480 graphs with the same set of nodes and edges. Moreover, in a first benchmark analysis, we evaluate the performance of GNN based architectures on the novel task.

The proposed graph regression problem is associated with information bottle necks and long-range dependencies. In our evaluation, we observe the GAT and

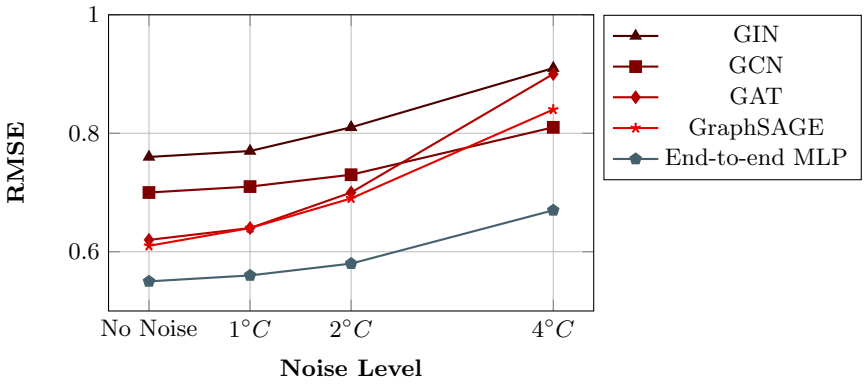

Fig. 3: The RMSE (y-axis) as a function of the Gaussian noise (x-axis) for all GNNs and the end-to-end MLP. We increase the standard deviation of a Gaussian noise from 1, to 2, and finally $4°C$. It can be clearly seen that the end-to-end MLP is more robust to noise, while the best-performing GNNs quickly degenerate.

GraphSAGE models outperform the GCN and GIN variants. Further, we compare the result with an end-to-end MLP, which is not restricted by the graph structure. We note that this simple – but inflexible – model performs slightly better compared to the GNN models.

To assess the impact of noise in the test set, we run an ablation study where one faulty node – of our choice and control – is placed in the test set. Eventually, we measure the performance degeneration over all nodes. We observe that on our dataset even small amounts of noise degenerates the GNNs performance quickly. Moreover, the end-to-end MLP seems to be more robust under noise compared to the GNN models and the best performing GNNs degenerates the most. The second observation can be interpreted as overfitting on the message passing step.

For future work we would like to investigate on GNN architectures which achieve the same predictive performance and noise robustness as the specific and inflexible end-to-end MLP model.

References

1. Caissie, D.: The thermal regime of rivers: a review. Freshwater biology 51(8), 1389–1406 (2006)
2. Piccolroaz, S., Calamita, E., Majone, B., Gallice, A., Siviglia, A., Toffolon, M.: Prediction of river water temperature: a comparison between a new family of hybrid models and statistical approaches. Hydrological Processes 30(21), 3901–3917 (2016)
3. Reid, P.C., Hari, R.E., Beaugrand, G., Livingstone, D.M., Marty, C., Straile, D., Barichivich, J., Goberville, E., Adrian, R., Aono, Y., et al.: Global impacts of the 1980s regime shift. Global change biology 22(2), 682–703 (2016)

4. Woolway, R.I., Dokulil, M.T., Marszelewski, W., Schmid, M., Bouffard, D., Merchant, C.J.: Warming of central european lakes and their response to the 1980s climate regime shift. Climatic Change 142, 505–520 (2017)
5. Toffolon, M., Piccolroaz, S.: A hybrid model for river water temperature as a function of air temperature and discharge. Environmental Research Letters 10(11), 114011 (2015)
6. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation 9(8), 1735–1780 (1997)
7. Qiu, R., Wang, Y., Rhoads, B., Wang, D., Qiu, W., Tao, Y., Wu, J.: River water temperature forecasting using a deep learning method. Journal of Hydrology 595, 126016 (2021)
8. Piotrowski, A.P., Napiorkowski, M.J., Napiorkowski, J.J., Osuch, M.: Comparing various artificial neural network types for water temperature prediction in rivers. Journal of Hydrology 529, 302–315 (2015)
9. Jia, X., Zwart, J., Sadler, J., Appling, A., Oliver, S., Markstrom, S., Willard, J., Xu, S., Steinbach, M., Read, J., et al.: Physics-guided recurrent graph model for predicting flow and temperature in river networks. In: Proceedings of the 2021 SIAM International Conference on Data Mining (SDM). pp. 612–620. SIAM (2021)
10. Moshe, Z., Metzger, A., Elidan, G., Kratzert, F., Nevo, S., El-Yaniv, R.: Hydronets: Leveraging river structure for hydrologic modeling. arXiv preprint arXiv:2007.00595 (2020)
11. Padrón, R.S., Zappa, M., Bernhard, L., Bogner, K.: Extended range forecasting of stream water temperature with deep learning models. EGUsphere 2024, 1–27 (2024)
12. Fankhauser, B., Bigler, V., Riesen, K.: Graph-based deep learning on the swiss river network. In: International Workshop on Graph-Based Representations in Pattern Recognition. pp. 172–181. Springer (2023)
13. Fankhauser, B., Bigler, V., Riesen, K.: Spatio-temporal graph neural networks for water temperature modeling. In: Structural, Syntactic, and Statistical Pattern Recognition. pp. 31–40. Springer Nature Switzerland, Cham (2025)
14. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: International conference on machine learning. pp. 1263–1272. PMLR (2017)
15. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net (2017), https://openreview.net/forum?id=SJU4ayYgl
16. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net (2019), https://openreview.net/forum?id=ryGs6iA5Km
17. Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y., et al.: Graph attention networks. stat 1050(20), 10–48550 (2017)
18. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. Advances in neural information processing systems 30 (2017)
19. Li, L., Jamieson, K., Rostamizadeh, A., Gonina, E., Ben-Tzur, J., Hardt, M., Recht, B., Talwalkar, A.: A system for massively parallel hyperparameter tuning. Proceedings of Machine Learning and Systems 2, 230–246 (2020)
20. Gillioz, A., Riesen, K.: Graph-based vs. vector-based classification: A fair comparison. In: International Workshop on Graph-Based Representations in Pattern Recognition. pp. 25–34. Springer (2023)