GLAudio Listens to the Sound of the Graph

Aurelio Sulser [*1] **Johann Wenckstern** [*2] **Clara Kuempel** [*1]

Abstract

We propose GLAudio: Graph Learning on Audio representation of the node features and the connectivity structure. This novel architecture propagates the node features through the graph network according to the discrete wave equation and then employs a sequence learning architecture to learn the target node function from the audio wave signal. This leads to a new paradigm of learning on graph-structured data, in which information propagation and information processing are separated into two distinct steps. We theoretically characterize the expressivity of our model, introducing the notion of the receptive field of a vertex, and investigate our model's susceptibility to over-smoothing and over-squashing both theoretically as well as experimentally on various graph datasets.

1. Introduction

With the advent of the popular architectures GCN (Kipf & Welling, 2017), GAT (Velickovic et al., 2017), and GIN (Xu et al., 2018), Graph Neural Networks (GNNs) have emerged as a powerful tool for learning on relational data. Despite the theoretical importance of depth for the expressivity of neural networks, most GNNs encountered in applications are relatively shallow. This is related to two fundamental issues impairing the expressivity of deep GNNs: over-squashing (Alon & Yahav, 2020), (Topping et al., 2021) and over-smoothing (Li et al., 2018), (Oono & Suzuki, 2019). A comprehensive theoretical framework to assess the representational capabilities of GNNs was introduced in (Xu et al., 2018). It established that GNNs can possess a representational power at most equivalent to the Weisfeiler-Leman graph isomorphism test when applied to featureless graphs. This revelation prompts an intriguing inquiry: what types of functions are learnable by GNNs when node features are included? A detailed characterization of these functions for most GNNs remains elusive, instead, the two primary limitations to their expressive power have been extensively explored: over-smoothing and over-squashing. The concept of over-squashing was first introduced by (Alon & Yahav, 2020). They observed that for a prediction task involving long-range interactions between nodes separated by a distance k, a GNN requires an equivalent number of k layers to capture these interactions effectively. As the k-th neighborhood of a vertex expands exponentially with k, the exploding volume of data must be compressed into a fixed-size vector at each vertex v, leading to over-squashing. In (Giovanni et al., 2023), it was explored how over-squashing reduces the class of functions learnable by GNNs. In (Di Giovanni et al., 2023), it was noted that standard GNNs are guided by the heat equation, a smoothing process. For such heat-equation guided GNNs, over-smoothing and over-squashing are intrinsically related to the spectral gap of the graph Laplacian, resulting in an inevitable trade-off between these two issues (Giraldo et al., 2023). In recent years, there has been a significant effort to introduce new architectures addressing these performance impairments, moving beyond heat-equation-guided GNNs. A promising approach are continuous-time GNNs that are often physically inspired (Chamberlain et al., 2021), (Bodnar et al., 2022). One such architecture is GraphCON (Rusch et al., 2022) that models the node features as controlled and damped oscillators, coupled via the adjacency structure of the underlying graph. Inspired by the great success of the Transformer Model in natural language processing, a completely new approach called Graph Transformers was proposed (Kreuzer et al., 2021), (Dwivedi & Bresson, 2020), (Rampášek et al., 2022). Unlike GNNs, graph transformers do not propagate the node features over the connectivity structure of the graph instead the node features are augmented by a laplacian encoding of the node's position in the graph. In that sense, the encoding of the connectivity structure is separated from the learning task, mitigating the phenomena over-smoothing and over-squashing.

Contribution. Continuing this idea of separating the encoding of the connectivity structure from the learning task, we propose GLAudio. GLAudio propagates the node features

[*]Equal contribution [1]Department of Computer Science, ETH Zürich, Switzerland [2]Department of Mathematics, ETH Zürich, Switzerland. Correspondence to: Aurelio Sulser <asulser@ethz.ch>.

Proceedings of the Geometry-grounded Representation Learning and Generative Modeling at 41st International Conference on Machine Learning, Vienna, Austria. PMLR Vol Number, 2024.

through the graph network according to the discrete wave equation (I) and then uses sequence learning architectures like LSTM or the Transformer model to learn the node function $\mathbf{Y}_v : \mathbb{R}^{|V(G)|\times d_0} \to \mathbb{R}^{d_1}$ at each vertex $v \in V(G)$ from the wave signal received at v. In that sense, GLAudio separates the feature propagation and signal processing into two distinct steps. Unlike the heat equation, the wave equation preserves the Dirichlet energy. Thus, the node features can be propagated over long distances without smoothing out. Moreover, the compression of information takes place during the encoding of node features and graph structure into a wave signal. Consequently, with increasing resolution the signal received at a vertex in GLAudio should be significantly less compressed compared to the fixed-size vector in standard GNNs. These two facts suggest that over-smoothing and over-squashing are mitigated. We are able to characterize the function class learnable by GLAudio and address these two phenomena.

Motivation for Using GLAudio Over Traditional GNNs. GLAudio offers notable advantages over traditional GNNs, particularly in mitigating over-smoothing and over-squashing, handling long-range dependencies, and performing well on heterophilic graphs. Traditional GNNs often suffer from feature indistinguishability and loss of information over multiple layers. In contrast, GLAudio uses the discrete wave equation to preserve the Dirichlet energy, maintaining feature distinctiveness over long-range propagation. This makes GLAudio more suitable for large-scale graphs with complex connectivity, such as biological and social networks, heterogeneous structures, and applications requiring detailed feature propagation, such as molecular graphs. We support these capabilities with theoretical analysis and empirical evidence and justify the additional computational overhead in scenarios where traditional GNNs fall short.

2. GLAudio

2.1. Wave Signal Encodes Features and Graph Structure

The discrete wave equation on a graph G with Laplacian matrix $\mathbf{L}$ and initial resting configuration $\mathbf{x} \in \mathbb{R}^{|V(G)|\times d_0}$ reads

$$\begin{cases} \ddot{\mathbf{X}}(t) = -\mathbf{L}\cdot\mathbf{X}(t) \\ \mathbf{X}(0) = \mathbf{x} \quad \dot{\mathbf{X}}(0) = 0. \end{cases} \tag{I}$$

It is well known that the unique solution is given by the continuous-time signal $\mathbf{X}(t) = \cos\big(\mathbf{L}^{1/2}\cdot t\big)\cdot\mathbf{x}$. The following theorem proves that this signal encodes much of the information about the features and the graph structure.

Theorem 2.1. *Given two graphs G, H on the same vertex set with initial features $\mathbf{x}_G, \mathbf{x}_H$, then we have for the two corresponding wave signals $\mathbf{X}_G(t), \mathbf{X}_H(t), \forall t > 0$*

$$\mathbf{X}_G|_{[0,t]} = \mathbf{X}_H|_{[0,t]} \iff \forall n \in \mathbb{N}_0 : \mathbf{L}_H^n \cdot \mathbf{x}_H = \mathbf{L}_G^n \cdot \mathbf{x}_G$$

Motivated by this fact, we propose to use the signal $\mathbf{X}(t)$ as an encoding of the graph's features and structure.

2.2. Model Architecture

Encoder. To implement the encoder, we solve the differential equation (I) in discrete time using an implicit-explicit ODE scheme (Norsett & Wanner, 1987). Let N denote the number of discrete time steps and T the stopping time. Let $h = \frac{T}{N}$ be the step size. We denote the approximation of $\mathbf{X}(ih)$ by $\mathbf{X}^i$ for $i = 0, ..., N$. The encoder architecture then reads

$$\begin{cases} \mathbf{X}^{i+1} = \mathbf{X}^i + h\mathbf{V}^{i+1} \\ \mathbf{V}^{i+1} = \mathbf{V}^i - h\mathbf{L}\cdot\mathbf{X}^i \\ \mathbf{X}^0 = \mathbf{x} \quad \mathbf{V}^0 = 0. \end{cases} \tag{II}$$

where $\mathbf{V}^i$ is an auxiliary "velocity" variable. We denote the step function provided by the numerical scheme by $\hat{\mathbf{X}}(\mathbf{x}, t) = \mathbf{X}^i \cdot 1_{[0,h]}(t) + \sum_{i=2}^{N} \mathbf{X}^i \cdot 1_{((i-1)h,ih]}(t)$. It is well known that as N increases $\hat{\mathbf{X}}(\mathbf{x}, t)$ converges in the function space $L^\infty([0, T]; \mathbb{R}^{|V(G)|\times d_0})$ to the true solution $\mathbf{X}(\mathbf{x}, t)$ uniformly over all possible initial features of some fixed compact set $C \subseteq \mathbb{R}^{|V(G)|\times d_0}$.

Decoder. For the decoder model, we can use any sequence learning architecture. We have tested RNN decoders, in particular, LSTM (Gers et al., 2000) and CoRNN (Rusch & Mishra, 2020). But it might also be interesting to investigate how the Transformer model (Vaswani et al., 2017) or State Space Models (Smith et al., 2022), (Gu & Dao, 2023) perform as a decoder. For sequence learning architectures, universal approximation theorems have been established (Schäfer & Zimmermann, 2006), (Lanthaler et al., 2023), (Yun et al., 2019), (Wang & Xue, 2024). In the following, we will make use of the universality of RNNs (Schäfer & Zimmermann, 2006) to characterize the expressivity of GLAudio. A simple RNN on an input sequence $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^{d_0}$ is given by for all $1 \le i \le n$

$$\begin{aligned} \mathbf{s}_i &= \sigma(\mathbf{W}\cdot\mathbf{s}_{i-1} + \mathbf{U}\cdot x_i) \\ \mathbf{y}_i &= \mathbf{V}\cdot\mathbf{s}_i, \end{aligned} \tag{III}$$

where $\mathbf{W} \in \mathbb{R}^{s\times s}, \mathbf{V} \in \mathbb{R}^{s\times d_1}, \mathbf{U} \in \mathbb{R}^{d_0\times s}$ and σ is a non-linearity, $\mathbf{s}_i$ are the hidden states and $\mathbf{y}_i$ are the outputs.

Theorem 2.2 ((Schäfer & Zimmermann, 2006))**.** *For any dynamical system $\boldsymbol{S}_{i+1} = g(\boldsymbol{S}_i, \boldsymbol{X}_{i+1}), \boldsymbol{Y}_{i+1} = h(\boldsymbol{S}_{i+1})$, where $g : \mathbb{R}^s \times \mathbb{R}^{d_0} \to \mathbb{R}^s$ measurable and $g : \mathbb{R}^s \to \mathbb{R}^{d_1}$ continuous, there exists an arbitrary good approximation of $\boldsymbol{Y}_n$ by some $\boldsymbol{y}_n$ of the form (III) uniformly for any input sequence $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$ of some fixed compact set $C \subseteq \mathbb{R}^{n\times d_0}$.*

Table 1. Node classification test accuracy obtained on network graph datasets. Means are obtained from 10 random initializations over fixed publicly available train/val/test split. Baselines for GCN, GAT and GraphCON are taken from (Rusch et al., 2022).

Datasets	GCN	GRAPHCON-GCN	GAT	GRAPHCON-GAT	**GLAUDIO-CORNN**
CORA	0.815	0.819	0.818	0.832	0.795
CITESEER	0.719	0.729	0.714	0.732	0.686
PUBMED	0.778	0.788	0.787	0.795	0.781
TEXAS	0.551	0.854	0.522	0.822	0.802
WISCONSIN	0.518	0.878	0.494	0.857	0.831
CORNELL	0.605	0.843	0.619	0.832	0.764

For a detailed list of all hyper-parameters and further configuration options, we refer to Appendix C.

2.3. Expressivity of GLAudio

In this section, we provide a thorough characterization of the expressive power of GLAudio. This analysis enables us to articulate how GLAudio potentially alleviates the issues of over-squashing and over-smoothing. To ease the notation, we restrict the discussion to the case $d_0 = 1$. The presented results easily generalize to the case of arbitrary d_0. Let $\{\phi_i\}_i$ be an eigenbasis of $\mathbf{L}$. We define the receptive field $\mathcal{R}_v$ of a vertex v to be the set $\{\phi_i \mid \forall i \in [n] : \langle \mathbf{e}_v, \phi_i \rangle \neq 0\}$. We call a function $f : \mathbb{R}^n \to \mathbb{R}^m$ *supported on a linear subspace* $V \subseteq \mathbb{R}^n$ if for any $\mathbf{u} \perp V$ we have that $\forall \mathbf{x} \in \mathbb{R}^n : f(\mathbf{x} + \mathbf{u}) = f(\mathbf{x})$.

Theorem 2.3. *Provided all eigenvalues of* $\mathbf{L}$ *are unique, if* $\mathbf{Y}_v$ *is supported on* $\mathcal{R}_v$*, and* $\forall \epsilon > 0, \forall C \subset \mathbb{R}^n$ *compact there exists an approximation* $\mathbf{y}_N$ *according to* (III) *on the input sequence* $\boldsymbol{X}^1, \ldots, \boldsymbol{X}^n$ *such that for all initial configurations* $\mathbf{x} \in C$, $\left\| \mathbf{Y}_v(\mathbf{x}) - \mathbf{y}_N(\boldsymbol{X}^1, \ldots, \boldsymbol{X}^N) \right\|_2 \leq \epsilon$.

Over-smoothing There exist a number of measures in the literature on over-smoothing in deep GNNs, e.g. measures based on the Dirichlet energy (Cai & Wang, 2020), (Zhao & Akoglu, 2019) or on the mean-average distance (MAD) (Chen et al., 2020), (Zhou et al., 2020). The survey paper (Rusch et al., 2023) gives a unified, rigorous, and tractable definition.

Definition 2.4. (Rusch et al., 2023) Given a sequence of GNNs on a graph G, where the $(N+1)$-th GNN differs from the N-th GNN by exactly one additional layer. We say that the sequence over-smooths w.r.t. some node similarity μ if $\mu(\mathbf{y}^N) < c_1 e^{-c_2 N}$, where $\mathbf{y}^N$ is the output of the N-th GNN and $c_1, c_2 > 0$.

Note that in our model, as we increase N for fixed stopping time T, the input signal $\dot{\mathbf{X}}(\mathbf{x}, t)$ of the RNN converges towards the true signal $\mathbf{X}(\mathbf{x}, t)$ and thus the output $\mathbf{y}_N$ can only become more accurate. This implies that provided there are two vertices u, v such that $\mathbf{Y}_u(T) \neq \mathbf{Y}_v(T)$ and using Theorem 2.3, $\mu(\mathbf{y}_N)$ cannot converge to 0.

Over-squashing To formalize the concept of over-squashing, (Topping et al., 2021) described it in terms of the impact of a distant node's feature $\mathbf{x}_u$ on the prediction $\mathbf{y}_v$ at node v. They noted an exponential decay in $\left| \frac{\partial \mathbf{y}_v}{\partial \mathbf{x}_u} \right|$ with increasing distance between nodes u and v in standard GNNs. Utilizing Theorem 2.3, we find that the output $\mathbf{y}_v$ reacts similar to $\mathbf{Y}_v$ to changes along $\mathcal{R}_v$ while being insensitive to perturbations perpendicular to $\mathcal{R}_v$. This nuanced understanding enables a more targeted approach to addressing the limitations of GLAudio in handling long-range interactions.

3. Experiments

For details on training methods and hyper-parameters, we refer to Appendix C. The code for all experiments is available on https://github.com/AurelioSulser/GLAudio.

3.1. Node Classification on Network Graphs

For a comparison with other model architectures, we evaluated the performance of GLAudio on six widely popular semi-supervised node classification single-graph datasets: Cora, CiteSeer, PubMed, Texas, Wisconsin and Cornell. Derived from citation networks, the first three are homophilic, i.e., adjacent vertices tend to have the same class label. The latter three are significantly more heterophilic, i.e. adjacent vertices tend to have different labels. Due to their smoothing bias, this property poses a significant challenge for traditional MPNNs like GCN. Average test accuracies for all six datasets are reported in Table 1.

Discussion and Results. We observe that on all six datasets GLAudio is able to successfully learn a classification strategy. On the heterophilic graph datasets, GLAudio outperforms the GCN and GAT architecture, whereas on homophilic graphs of Cora and CiteSeer GCN and GAT achieve higher accuracies. These results match our theoretic understanding: Both GCN and GAT can be seen as time discretizations of the discrete first-order heat equation equipped with learnable parameters causing them to naturally smooth the nodes' features. This bias is useful on homophilic datasets, yet disadvantageous on heterophilic graphs. Derived from the second-order wave equation, GLAudio does not exhibit this flaw, achieving consistently

high accuracies on all datasets.

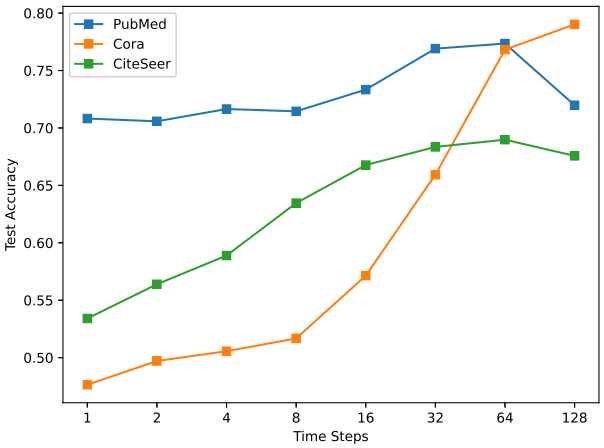

Figure 1. Test accuracy for varying numbers of time steps N. Accuracies are averaged over 10 random initializations.

To verify that GLAudio is not prone to over-smoothing, we measured its accuracy on homophilic datasets by varying the time steps N, keeping other hyper-parameters constant, notably the stopping time T. Contrary to the effect of over-smoothing, GLAudio's performance improved with more time steps and reached optimal levels within the range of 50 to 200, as shown in Figure 1. This contrasts with the findings in (Chamberlain et al., 2021) where GCNs' accuracies diminish significantly below 50% with more than 16 layers.

3.2. Graph Regression on ZINC Dataset

In (Rusch et al., 2022), it was found that the Mean Absolute Error (MAE) of standard GCNs on the ZINC dataset increased with the model's depth, while for GraphCon-GCN, the MAE showed a decreasing trend. This indicates the significance of long-range dependencies within the ZINC dataset. Given the capability of GLAudio to support exceptionally deep models, as highlighted in Figure 1, its potential is explored on ZINC.

The ZINC dataset comprises approximately 12,000 molecular graphs, curated for the regression of a molecular property known as constrained solubility. Each graph represents a molecule, where the node features correspond to the types of heavy atoms present, and the edge features denote the types of bonds connecting these atoms.

Discussion and Results. GLAudio's initial performance on the ZINC dataset appears modest, performing on par with GIN, and GatedGCN, see Table 2. The reason could be that GLAudio might disproportionately emphasize long-range over short-range interactions as models successful on ZINC typically emphasize the latter (Dwivedi et al., 2022). To test this, we combined a 4-layer GCN ($\sim$ 10,000 parameters) and a GLAudio-CoRNN ($\sim$ 90,000 parameters), allowing GCN to focus on short-range and GLAudio on long-range dependencies. The result achieved a test MAE of 0.3157, markedly surpassing standard GNN models and nearing state-of-the-art performance, suggesting GLAudio's proficiency in capturing dependencies missed by shallow GCN.

Table 2. Test MAE on ZINC (without edge features, small 12**k** version) restricted to small network sizes of $\sim 100k$ parameters. Baseline results are taken from (Beaini et al., 2021) and (Rusch et al., 2022).

MODEL	TEST MAE
GIN (XU ET AL., 2018)	0.41
GATEDGCN (BRESSON & LAURENT, 2017)	0.42
DGN (BEAINI ET AL., 2021)	0.22
GCN (KIPF & WELLING, 2017)	0.47
GRAPHCON-GCN (RUSCH ET AL., 2022)	0.22
GAT (VELICKOVIC ET AL., 2017)	0.46
GRAPHCON-GAT (RUSCH ET AL., 2022)	0.23
GLAUDIO-LSTM	0.4342

3.3. Long Range Graph Benchmark: Peptides-struct

Long Range Graph Benchmark is a set of 5 datasets to measure the ability of GNNs and Graph Transformers to solve problems depending on long-range interactions (Dwivedi et al., 2023). We evaluated GLAudio on one of these tasks, namely Peptides-struct, a multi-dimensional graph regression task on molecular graphs.

Table 3. Test mean absolute error (MAE) on Peptides-struct. Baseline results are taken from (Dwivedi et al., 2023).

MODEL	TEST MAE
GCN (KIPF & WELLING, 2017)	0.3496
GIN (XU ET AL., 2018)	0.3547
GATEDGCN (BRESSON & LAURENT, 2017)	0.3420
TRANSFORMER+LAPPE (HAMILTON ET AL., 2017)	0.2529
GLAUDIO-LSTM	0.3569

Discussion and Results. The results in Table 3 show GLAudio achieving on par MAEs with traditional MPNNs but underperforming graph transformers. This suggests that GLAudio might have difficulties capturing the long-range dependencies of Peptides-struct. On the contrary, we observed significant performance enhancements with an increasing number of time steps. Moreover, recent findings from (Tönshoff et al., 2023) reveal that a simple GCN with extensive hyper-parameter tuning can achieve state-of-the-art results (0.2460 MAE) on Peptides-struct. It casts doubt on the suitability of Peptides-struct as a benchmark for long-

range graph interactions.

4. Conclusion

We proposed a novel graph learning architecture based on wave propagation. A central distinction from other models lies in the separation of information propagation and information processing into two different steps. This separation allows for deep feature propagation without over-smoothing. Moreover, our theoretical study of expressivity provides a new approach to understanding over-squashing as a miss-alignment between graph structure and task. In our empirical studies, GLAudio was benchmarked against state-of-the-art models using network datasets and molecular graphs. On heterophilic datasets, we empirically validate that GLAudio alleviates over-smoothing. Additionally, we investigated the phenomenon of over-squashing on more complex datasets, namely ZINC and Peptides-struct. Based on our results, it remains inconclusive as to whether GLAudio significantly mitigates over-squashing. Despite not yet achieving state-of-the-art scores on more complex datasets, we believe that our approach of separating information processing from feature propagation represents a promising concept that leaves space for further investigation.

References

Alon, U. and Yahav, E. On the bottleneck of graph neural networks and its practical implications. *arXiv preprint arXiv:2006.05205*, 2020.

Beaini, D., Passaro, S., Létourneau, V., Hamilton, W., Corso, G., and Liò, P. Directional graph networks. In *International Conference on Machine Learning*, pp. 748–758. PMLR, 2021.

Bodnar, C., Di Giovanni, F., Chamberlain, B., Liò, P., and Bronstein, M. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gnns. *Advances in Neural Information Processing Systems*, 35: 18527–18541, 2022.

Bresson, X. and Laurent, T. Residual gated graph convnets. *arXiv preprint arXiv:1711.07553*, 2017.

Cai, C. and Wang, Y. A note on over-smoothing for graph neural networks. *arXiv preprint arXiv:2006.13318*, 2020.

Chamberlain, B., Rowbottom, J., Gorinova, M. I., Bronstein, M., Webb, S., and Rossi, E. Grand: Graph neural diffusion. In *International Conference on Machine Learning*, pp. 1407–1418. PMLR, 2021.

Chen, D., Lin, Y., Li, W., Li, P., Zhou, J., and Sun, X. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 3438–3445, 2020.

Di Giovanni, F., Rowbottom, J., Chamberlain, B. P., Markovich, T., and Bronstein, M. M. Understanding convolution on graphs via energies. *Transactions on Machine Learning Research*, 2023.

Dwivedi, V. P. and Bresson, X. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*, 2020.

Dwivedi, V. P., Joshi, C. K., Luu, A. T., Laurent, T., Bengio, Y., and Bresson, X. Benchmarking graph neural networks, 2022.

Dwivedi, V. P., Rampášek, L., Galkin, M., Parviz, A., Wolf, G., Luu, A. T., and Beaini, D. Long range graph benchmark, 2023.

Gers, F. A., Schmidhuber, J., and Cummins, F. Learning to forget: Continual prediction with lstm. *Neural computation*, 12(10):2451–2471, 2000.

Giovanni, F. D., Rusch, T. K., Bronstein, M. M., Deac, A., Lackenby, M., Mishra, S., and Veličković, P. How does over-squashing affect the power of gnns?, 2023.

Giraldo, J. H., Skianis, K., Bouwmans, T., and Malliaros, F. D. On the trade-off between over-smoothing and over-squashing in deep graph neural networks. CIKM '23, pp. 566–576. Association for Computing Machinery, 2023. doi: 10.1145/3583780.3614997. URL https://doi.org/10.1145/3583780.3614997.

Gu, A. and Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Hamilton, W., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks, 2017.

Kreuzer, D., Beaini, D., Hamilton, W., Létourneau, V., and Tossou, P. Rethinking graph transformers with spectral attention. *Advances in Neural Information Processing Systems*, 34:21618–21629, 2021.

Lanthaler, S., Rusch, T. K., and Mishra, S. Neural oscillators are universal. *arXiv preprint arXiv:2305.08753*, 2023.

Li, Q., Han, Z., and Wu, X.-M. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Norsett, E. H. S. and Wanner, G. Solving ordinary differential equations i: Nonsti problems, 1987.

Oono, K. and Suzuki, T. Graph neural networks exponentially lose expressive power for node classification. *arXiv preprint arXiv:1905.10947*, 2019.

Rampášek, L., Galkin, M., Dwivedi, V. P., Luu, A. T., Wolf, G., and Beaini, D. Recipe for a general, powerful, scalable graph transformer. *Advances in Neural Information Processing Systems*, 35:14501–14515, 2022.

Rusch, T. K. and Mishra, S. Coupled oscillatory recurrent neural network (cornn): An accurate and (gradient) stable architecture for learning long time dependencies. *arXiv preprint arXiv:2010.00951*, 2020.

Rusch, T. K., Chamberlain, B. P., Rowbottom, J., Mishra, S., and Bronstein, M. M. Graph-coupled oscillator networks, 2022.

Rusch, T. K., Bronstein, M. M., and Mishra, S. A survey on oversmoothing in graph neural networks. *arXiv preprint arXiv:2303.10993*, 2023.

Schäfer, A. M. and Zimmermann, H. G. Recurrent neural networks are universal approximators. In *Artificial Neural Networks–ICANN 2006: 16th International Conference, Athens, Greece, September 10-14, 2006. Proceedings, Part I 16*, pp. 632–640. Springer, 2006.

Smith, J. T., Warrington, A., and Linderman, S. W. Simplified state space layers for sequence modeling. *arXiv preprint arXiv:2208.04933*, 2022.

Topping, J., Di Giovanni, F., Chamberlain, B. P., Dong, X., and Bronstein, M. M. Understanding over-squashing and bottlenecks on graphs via curvature. *arXiv preprint arXiv:2111.14522*, 2021.

Tönshoff, J., Ritzert, M., Rosenbluth, E., and Grohe, M. Where did the gap go? reassessing the long-range graph benchmark, 2023.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y., et al. Graph attention networks. *stat*, 1050 (20):10–48550, 2017.

Wang, S. and Xue, B. State-space models with layer-wise nonlinearity are universal approximators with exponential decaying memory. *Advances in Neural Information Processing Systems*, 36, 2024.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

Yun, C., Bhojanapalli, S., Rawat, A. S., Reddi, S. J., and Kumar, S. Are transformers universal approximators of sequence-to-sequence functions? *arXiv preprint arXiv:1912.10077*, 2019.

Zhao, L. and Akoglu, L. Pairnorm: Tackling oversmoothing in gnns. *arXiv preprint arXiv:1909.12223*, 2019.

Zhou, K., Huang, X., Li, Y., Zha, D., Chen, R., and Hu, X. Towards deeper graph neural networks with differentiable group normalization. *Advances in neural information processing systems*, 33:4917–4928, 2020.

A. Proof of Theorem 2.1

We remark that the power series representation of the signal $\mathbf{X}_H(t)$ is given by

$$\mathbf{X}_H(t) = \cos\Big(\mathbf{L}_H^{1/2} \cdot t\Big) \cdot \mathbf{x}_H = \sum_{n=0}^{\infty} \frac{(-1)^n \mathbf{L}_H^n t^{2n}}{(2n)!} \cdot \mathbf{x}_H = \sum_{n=0}^{\infty} \frac{(-1)^n \cdot (\mathbf{L}_H^n \cdot \mathbf{x}_H) \cdot t^{2n}}{(2n)!}.$$

It is thus immediate that $\forall n \in \mathbb{N}_0 : \mathbf{L}_H^n \cdot \mathbf{x}_H = \mathbf{L}_G^n \cdot \mathbf{x}_G$ implies that

$$\mathbf{X}_H(t) = \mathbf{X}_G(t).$$

For the other implication, we note that $\mathbf{X}_G|_{[0,t]} = \mathbf{X}_H|_{[0,t]}$ implies that

$$\forall n \in \mathbb{N}_0 : \frac{d^{(2n)}}{dx^{(2n)}} \mathbf{X}_G(0) = \frac{d^{(2n)}}{dx^{(2n)}} \mathbf{X}_H(0).$$

According to the power series representation of $\mathbf{X}_G(t)$ and $\mathbf{X}_H(t)$, this is equivalent to

$$\mathbf{L}_H^n \cdot \mathbf{x}_H = \mathbf{L}_G^n \cdot \mathbf{x}_G.$$

B. Proof of Theorem Theorem 2.3

Before we present the actual proof, we introduce the following definition.

We call an operator $\Phi : (L^\infty([0,1];\mathbb{R}), \|\cdot\|_{L^\infty}) \to (L^\infty([0,1];\mathbb{R}), \|\cdot\|_{L^\infty})$ causal if for any two input signals $u, v \in L^\infty([0,1];\mathbb{R})$

$$u|_{[0,t]} = v|_{[0,t]} \longrightarrow \Phi(u)|_{[0,t]} = \Phi(v)|_{[0,t]}.$$

We begin the proof with the claim that if there exists a continuous (with respect to the L^∞-norm on the input-/output-signals), causal operator Φ such that $\Phi(\mathbf{X}_v)(T) = \mathbf{Y}_v(\mathbf{x})$, then we can conclude.

Proof of the claim. As a preliminary step, we define $K = \{\mathbf{X}_v^i(\mathbf{x}) | \forall 1 \leq i \leq N, \text{ for all initial configuration } \mathbf{x} \in C\}$. It follows from compactness of C that K is compact. In the following, we will give a description of $\Phi(\hat{\mathbf{X}}_v)(T)$ as a dynamical system and apply Theorem 2.2 to this description together with the compact set K.

For an arbitrary input sequence $\mathbf{X}_1, \ldots, \mathbf{X}_N$, set $\mathbf{S}_i = (i, \mathbf{X}_i, \mathbf{V}_i)$, where $\mathbf{V}_i = \mathbf{V}_{i-1} - h \cdot \mathbf{L} \cdot \mathbf{X}_{i-1}$ and $\mathbf{V}_0 = 0$. It is clear that $\mathbf{S}_{i+1}$ is computable from $\mathbf{S}_i, \mathbf{X}_{i+1}$, hence there exists a measurable function g such that $\mathbf{S}_{i+1} = g(\mathbf{S}_i, \mathbf{X}_{i+1})$. We note that for the special input sequence $\mathbf{X}_1 = \mathbf{X}_v^1, \ldots, \mathbf{X}_N = \mathbf{X}_v^N$, we have that $\mathbf{S}_i = (i, \mathbf{X}_v^i, \mathbf{V}_v^i)$.

To define the function $h(\mathbf{S}_i)$, let us introduce vectors $\tilde{\mathbf{X}}_1, \ldots, \tilde{\mathbf{X}}_i, \tilde{\mathbf{V}}_1, \ldots, \tilde{\mathbf{V}}_i$ (that are in some sense approximations of $\mathbf{X}_1, \ldots, \mathbf{X}_i, \mathbf{V}_1, \ldots, \mathbf{V}_i$) given by

$$\begin{aligned} \tilde{\mathbf{X}}_j &= \tilde{\mathbf{X}}_{j+1} - h\tilde{\mathbf{V}}_{j+1} \\ \tilde{\mathbf{V}}_j &= \tilde{\mathbf{V}}_{j+1} + h \cdot \mathbf{L} \cdot \tilde{\mathbf{X}}_j \\ \tilde{\mathbf{X}}_i &= \mathbf{X}_i \quad \tilde{\mathbf{V}}_i = \mathbf{V}_i. \end{aligned}$$

In the special case where $\mathbf{X}_1 = \mathbf{X}_v^1, \ldots, \mathbf{X}_N = \mathbf{X}_v^N$, we have that $\tilde{\mathbf{X}}_1 = \mathbf{X}_v^1, \ldots, \tilde{\mathbf{X}}_N = \mathbf{X}_v^N$. We define the continuous function $f : \mathbb{R}^{|V(G)|} \times \mathbb{R}^{|V(G)|} \to L^\infty([0,1];\mathbb{R})$ given by $f(\mathbf{X}_i, \mathbf{V}_i)(t) = \tilde{\mathbf{X}}_i \cdot 1_{[0,h]}(t) + \sum_{j=2}^{i} \tilde{\mathbf{X}}_i \cdot 1_{((i-1)h,ih]}(t)$. This allows us to define $h(\mathbf{S}_i) = \Phi(f(\mathbf{X}_i, \mathbf{V}_i))(ih)$. We note that in the special case that $\mathbf{X}_1 = \mathbf{X}_v^1, \ldots, \mathbf{X}_N = \mathbf{X}_v^N$ that $f(\mathbf{X}_i, \mathbf{V}_i)|_{[0,ih]} = \hat{\mathbf{X}}_v|_{[0,ih]}$ and hence by causality of Φ, we have that $h(\mathbf{S}_i) = \Phi(\hat{\mathbf{X}}_v)(ih)$. Applying Theorem 2.2 to this dynamic system with compact set K, we get the output of some RNN $\mathbf{y}_N$ satisfies that $\forall \mathbf{x} \in C : \|\Phi(\hat{\mathbf{X}}_v(\mathbf{x}))(Nh) - \mathbf{y}_N(\mathbf{X}_v^1(\mathbf{x}), \ldots, \mathbf{X}_v^N(\mathbf{x}))\| \leq \epsilon$. Provided we picked N large enough such that $\forall \mathbf{x} \in C : \|\mathbf{Y}_v(\mathbf{x}) - \Phi(\hat{\mathbf{X}}_v(\mathbf{x}))(Nh)\| = \|\Phi(\mathbf{X}_v(\mathbf{x}))(Nh) - \Phi(\hat{\mathbf{X}}_v(\mathbf{x}))(Nh)\| \leq \epsilon$, we can combine the two bounds to obtain the statement of the theorem.

□

All that remains is to argue the existence of such a continuous, causal operator Φ. Let us make the following preliminary remarks. Let $\phi_1, \ldots, \phi_n$ be normalized eigenvectors of $\mathbf{L}$ associated with eigenvalues $\lambda_1, \ldots, \lambda_n$. W.l.o.g. we may assume that $\lambda_1, \ldots, \lambda_n \in \mathbb{Q}$ and let $k \in \mathbb{N}_1$ such that $\forall 0 \leq i \leq n : k \cdot \lambda_i \in \mathbb{N}$. Moreover, we denote the spectral decomposition of $\mathbf{L}$ by $\mathbf{U} \cdot \mathbf{\Lambda} \cdot \mathbf{U}^t$, where $\mathbf{U}$ is orthonormal and $\mathbf{\Lambda}$ is diagonal, and for all $i \geq 1$ we introduce the functions

$$u_i(t) = \frac{\cos(\lambda_i \cdot t)}{\langle \mathbf{e}_v, \phi_i \rangle}.$$

We write $\langle \cdot, \cdot \rangle_t$, where $t > 0$, for the scalar product defined on $C_0([0,1], \mathbb{R})$ by

$$\langle u, v \rangle_t = \frac{1}{\pi k} \int_{[0, 2\pi \cdot k \cdot t]} u\left(\frac{s}{2\pi \cdot k}\right) \cdot v\left(\frac{s}{2\pi \cdot k}\right) ds$$

We remark that given the wave signal $\mathbf{X}(t) = \sum_{j=1}^{n} \phi_j \cos(\lambda_j \cdot t) \langle \phi_j, \mathbf{x} \rangle$ corresponding to the initial configuration $\mathbf{x}$, the scalar product satisfies

$$\langle u_i, \mathbf{X}_v \rangle_1 = \sum_{\phi_j \in \mathcal{R}_v} \langle u_i, \langle \phi_j, \mathbf{e}_v \rangle \cdot \cos(\lambda_j \cdot t) \cdot \langle \phi_j, \mathbf{x} \rangle \rangle_1 \tag{IV}$$

$$= \sum_{\phi_j \in \mathcal{R}_v} \langle \phi_j, \mathbf{x} \rangle \cdot \frac{\langle \phi_j, \mathbf{e}_v \rangle}{\langle \phi_i, \mathbf{e}_v \rangle} \cdot \langle \cos(\lambda_i \cdot t), \cos(\lambda_j \cdot t) \rangle_1 \tag{V}$$

$$= \sum_{\phi_j \in \mathcal{R}_v} \langle \phi_j, \mathbf{x} \rangle \cdot \delta_{ij}. \tag{VI}$$

With these preparations at hand, it is easy to conclude. We note that since $\mathbf{Y}_v$ is supported on $\mathcal{R}_v$, the function $\mathbf{Y}_v$ factorizes into $\mathbf{Y}_v = \tilde{\mathbf{Y}}_v \circ p$, where $p : \mathbb{R}^n \to \mathbb{R}^{|\mathcal{R}_v|}$ is the projection

$$p(\mathbf{x}) = (\langle \mathbf{x}, \phi_i \rangle)_{i \geq 1 : \phi_i \in \mathcal{R}_v}$$

and $\tilde{\mathbf{Y}}_v : \mathbb{R}^{|\mathcal{R}_v|} \to \mathbb{R}$ is continuous.

We define the operator $\Phi : C_0([0,T]; \mathbb{R}) \to C_0([0,T]; \mathbb{R})$ by

$$\Phi(u)(t) = \tilde{\mathbf{Y}}_v((\langle u, u_i \rangle_t)_{i \geq 1 : \phi_i \in \mathcal{R}_v}).$$

We remark that Φ is causal and continuous. Moreover, given the wave signal $\mathbf{X}(t)$ corresponding to the initial configuration $\mathbf{x}$, we have that

$$\Phi(\mathbf{X}_v)(T) = \tilde{\mathbf{Y}}_v((\langle \mathbf{X}_v, u_i \rangle_1)_{i \geq 1 : \phi_i \in \mathcal{R}_v}) = \tilde{\mathbf{Y}}_v((\langle \mathbf{x}, \phi_i \rangle)_{i \geq 1 : \phi_i \in \mathcal{R}_v}) = \tilde{\mathbf{Y}}_v \circ p(\mathbf{x}) = \mathbf{Y}_v(\mathbf{x}).$$

C. Further Experiment and Training Details

Our base architecture admits the following hyper-parameters: Number of layers L, number of time steps N, step size h, hidden dimension size q, stopping time T, and the choice of activation function σ. Note that N, h, and T are coupled by $T = Nh$.

Further, we experimented with initial node-wise linear or non-linear embeddings, different dropout rates, and varying constant or adaptive (ReduceOnPlateau) learning rates. The eigenvalues of the Laplacian $\mathbf{L}$, i.e., the frequencies of our oscillation $\mathbf{X}(t)$ lie in $[0, 2 \max_v \deg(v)]$. Let $\mathbf{D}$ denote the degree matrix. To avoid very high frequencies, difficult to be captured by our numerical approximation of $\mathbf{X}(t)$, we also considered normalizing the Laplacian in (I) by $\mathbf{N} = \mathbf{D}^{-\frac{1}{2}} \mathbf{L} \mathbf{D}^{-\frac{1}{2}}$. The eigenvalues of $\mathbf{N}$ lie in $[0, 2]$. Similarly to avoid very slow frequencies, we implemented the option of adding self-loops, leading to $\mathbf{I} + \mathbf{L}$ or $\mathbf{I} + \mathbf{N}$ instead of $\mathbf{L}$ resp. $\mathbf{N}$ in (I).

We used the Adam optimizer with various learning rates and weight decays. For node classification tasks, we used the multi-class cross-entropy function as a loss function. For regression tasks, we used the $\ell 1$-function as the loss function.

In our training process, each training run of GLAudio for all six network datasets was conducted over a span of 300 epochs. For ZINC the maximum number of epochs was 1000 and for Peptides-struct 300. For ZINC and Peptides-Struct early-stopping at stagnating validation loss was used to avoid overfitting.

Table 4. Overview of best-found hyperparameter configurations for all experiments.

Dataset	L	N	h	Learning Rate	Weight Decay	Activation	Hidden Dim.	Normalized Laplacian	Dropout Rate	Self Loops
Cora	2	200	0.02	0.001	0.005	Leaky ReLU	32	True	0.2	False
Citeseer	1	150	0.01	0.0025	0.005	Leaky ReLU	24	True	0.2	False
Pubmed	1	50	0.06	0.0025	0.005	Leaky ReLU	24	False	0.3	False
Texas	2	10	0.15	0.001	0.01	Leaky ReLU	32	False	0.4	False
Cornell	1	2	0.7	0.0025	0.005	ReLU	48	True	0.4	False
Wisconsin	1	4	0.2	0.01	0.001	Leaky ReLU	32	True	0.3	False
ZINC (CoRNN + GCN)	3	100	0.02	0.005	0.0	Leaky ReLU	150 + 40	True	0.0	True
ZINC (LSTM)	3	20	0.5	0.005	0.0	ReLU	128	True	0.0	True
Peptides-struct (LSTM)	3	20	0.5	0.002	0.0	GeLU	128	True	0.0	False