GraphHSCN: Heterogenized Spectral Cluster Network for Long Range Graph Data

Sirui Tao
University of California San Diego
La Jolla, CA, USA
s1tao@ucsd.edu

Alison Camille Dunning
University of California San Diego
La Jolla, CA, USA
adunning@ucsd.edu

Zhishang Luo
University of California San Diego
La Jolla, CA, USA
zluo@ucsd.edu

ABSTRACT

Graph Neural Networks (GNNs) have gained tremendous popularity for their potential to effectively learn from graph-structured data, commonly encountered in real-world applications. However, most of these models, based on the message-passing paradigm (interaction within a neighborhood of a few nodes), can only handle local interactions within a graph. When we enforce the models to use information from far away nodes, we will encounter two major issues — oversmoothing [7] & oversquashing [12]. Architectures such as the transformer and diffusion models are introduced to solve this. Yet, these models are not tested on large graph datasets containing graphs with large diameters. Although transformers are powerful, they require significant computational resources for both training and inference, thereby limiting their scalability, particularly for graphs with long-term dependencies. Hence, this paper proposes GraphHSCN—a Heterogenized Spectral Cluster Network, a message-passing-based approach specifically designed for capturing long-range interaction information (when prediction depends on representations of distant nodes interacting with each other)[2].

INTRODUCTION

General discussion of graph data

All varieties of real-world data can be represented as graphs, and the complicated interactions between the node instances in the graph data are usually very valuable information to learn. The recent developments of Graph Neural Networks showed significant success in learning graph data. However, sometimes in the graph data, the interactions between two node instances might be too complicated to learn. One situation is when the two nodes are very far from each other but interacting. Our project tries to help the GNNs learn this long-range interaction.

Common GNN architectures

Three models are first learned during this quarter. These models are Graph Convolutional Network [4], Graph Attention Network [13], and Graph Isomorphism Network [15]. The idea of GCNs is to learn a function of features on a graph with a feature matrix as X and a representation of the graph structure in matrix form, which typically is adjacency matrix A. Combine these together, and we will have node-level output as a matrix. We can also, from there, have graph-level output through global pooling. In this case, each layer in the GCN is a function that propagates information forward with a weight matrix. Compared to GCN, Graph Attention Network [4], or GAT, used the attention mechanism. One key difference is that, by using the attention mechanism, the more important node during the aggregation will have a higher weight. And instead of multiple channels used in GCN, GAT used multi-head attention. As shown in the graph below: [13] Graph Isomorphism Network [15], or GIN, used the idea of isomorphism to build a GNN that generalizes the Weisfeiler-Lehman graph isomorphism test. In the paper, the authors introduced a theory of "deep multisets." They showed that with universal multiset functions, aggregation schemes over nodes and the multiset of their neighbors could be built. And this function can be learned through MLPs. Another type of GNN depends on the idea of transformers. Introduced by the paper "Rethinking Graph Transformers with Spectral Attention" [5], Spectral Attention Network or SAN used the Transformers and its self-attention mechanism to try to encode the graph structure. SAN attempts to solve the issue of message-passing GNNs that can't learn the long-range interactions in graphs.

Major Challenges

As mentioned in the abstract, two major challenges exist to prevent models under the "message passing" paradigm from performing well on long-range graph datasets: oversmoothing [7] & oversquashing [12].

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

Oversmoothing

In the context of long-range interaction graph datasets, oversmoothing in Graph Neural Networks (GNNs) refers to a situation where the model loses information about the original graph structure and flattens the node embeddings to be indistinguishable from each other because the information got diluted when transmitting across a long chain of nodes. This can result in decreased accuracy and poor performance on downstream tasks.

In other words, GNNs operate by aggregating and updating the representations of neighboring nodes iteratively. However, if the number of iterations is too high or the aggregation operation is too strong, the embeddings of all nodes become similar, making it difficult for the model to distinguish between them.

Oversquashing

The phenomena of "oversquashing" occurs in Message Passing Neural Networks (MPNNs) when the learned task requires long-range dependencies and the structure of the graph results in exponentially many long-range neighboring nodes. In such situations, messages coming from non-adjacent nodes need to be propagated and compressed into fixed-size vectors, causing a phenomenon of information oversquashing. The structural characteristics of the graph responsible for oversquashing are referred to as "bottlenecks".

These phenomenons are extensively observed empirically, but the theoretical understanding of them is rather limited.

Relevant works for our architecture

In order to deal with the above-mentioned problem, we studied existing literature in this domain. Below, we will quickly go over their main ideas.

Positional Encoding

To begin with, SignNet [6] is designed to be invariant to sign flips of eigenvectors, which is an important property for spectral graph representation learning because eigenvectors can have both positive and negative values.

SignNet is built using a feedforward neural network with a series of linear and non-linear transformations. The input to the network is the Laplacian eigenvectors of a graph, and the output is a vector representing a graph embedding. The network is trained to minimize a loss function that measures the similarity between the predicted embeddings and the true embeddings of the graph.

SignNet is shown to be universal under certain conditions, meaning that it can approximate any continuous function of eigenvectors with the desired invariances. The authors also demonstrate that SignNet outperforms existing spectral graph representation methods on several benchmark datasets.

Global shortcuts

Then, the paper "FoSR" [3] proposes a computationally efficient algorithm for graph neural networks (GNNs) that prevents two common problems in GNNs: oversquashing and oversmoothing.

As mentioned in the "Major Challenges" sections, oversquashing is a problem that arises when GNNs pass messages along the edges of the graph, leading to inefficient information propagation for certain graph topologies. This problem is linked to the curvature and spectral gap of the graph. Oversmoothing, on the other hand, occurs when adding edges to the message-passing graph can lead to increasingly similar node representations.

The proposed algorithm uses spectral expansion to systematically add edges to the graph and prevent oversquashing, while a relational architecture is used to preserve the original graph structure and prevent oversmoothing. The authors demonstrate experimentally that their algorithm outperforms existing graph rewiring methods in several graph classification tasks.

In summary, "FoSR" proposes an algorithm for GNNs that addresses two common problems in GNNs - oversquashing and oversmoothing - by using spectral expansion and a relational architecture.

Spectral clustering

In order to solve the issue of transformers being too expansive to compute, especially for larger graphs, we explored the method from "Spectral Clustering with Graph Neural Networks for Graph Pooling" [1] to decrease the workload for the attention-based procedures.

This paper proposes a new graph clustering method that can be used to implement pooling operations in Graph Neural Networks (GNNs). Though Spectral clustering (SC) is a popular technique to find strongly connected communities on a graph, but it can be expensive to compute the eigendecomposition of the Laplacian, and clustering results are graph-specific, meaning that pooling methods based on SC must perform a new optimization for each new sample.

To address these limitations, the paper proposes a continuous relaxation of the normalized minCUT problem and trains a GNN to compute cluster assignments that minimize this objective. The proposed method is differentiable, does not require computing the spectral decomposition, and learns a clustering function that can be quickly evaluated on out-of-sample graphs.

The authors use this clustering method to design a graph pooling operator that overcomes some important limitations of state-of-the-art graph pooling techniques and achieves the best performance in several supervised and unsupervised tasks.

In summary, the paper proposes a new differentiable graph clustering method that can be used in GNNs to implement pooling operations without requiring the expensive computation of spectral decomposition. The proposed pooling operator achieves state-of-the-art performance in several tasks.

Heterogeneous graph

Later, we found the Heterogeneous graph architecture [11] to help our model learn from both the original graph and the virtual nodes and edged created from our clustering results.

The Heterogeneous graph paper describes an approach for analyzing and mining knowledge from interconnected, multi-typed data, including relational databases, that form complex and semi-structured information networks.

The authors argue that most network science researchers focus on homogeneous networks without distinguishing different types of objects and links, which is not suitable for complex, multi-typed data in the real world.

The paper proposes a structural analysis approach for mining useful knowledge from such networks by leveraging the rich semantic meaning of structural types of objects and links in the networks. The authors summarize a set of methodologies that can effectively and efficiently mine knowledge from such information networks and point out some promising research directions.

In summary, the paper presents a framework for analyzing and mining knowledge from complex, heterogeneous information networks, which are common in the real world but often neglected in network science research. The proposed approach leverages the rich semantic meaning of structural types of objects and links in the networks and provides a set of methodologies for effectively and efficiently mining useful knowledge from such networks.

Attention

After the extraction of cluster-level features, we also want to explore if we can use the attention mechanism to find additional connections between clusters. Here, for larger clusters, we used sparse attention [17] to make the computation more efficient. However, some graph datasets have rather small clusters after the second procedure so we also explored whether global attention mechanisms such as set2set [14] could further improve the performance. We will give more details about what sparse attention and set2set are in the following two paragraphs.

Here, sparse attention refers to Sparse Graph Attention Networks (SGATs)[17], which are a type of Graph Neural Network (GNN) that improve the performance of graph learning tasks by learning to assign sparse attention coefficients over a graph's neighbors. This sparsity is achieved through an L_0 -norm regularization, which allows SGATs to identify noisy or task-irrelevant edges and perform feature aggregation on the most informative neighbors. SGATs have been shown to outperform traditional GNNs, such as Graph Attention Networks (GATs), on both assortative and disassortative graphs, while removing about 50-80% of edges from large assortative graphs without sacrificing classification accuracy. SGATs are the first graph learning algorithm to demonstrate significant redundancies in graphs, and their edge-sparsified graphs can achieve similar or sometimes even higher predictive performance than original graphs.

Set2Set [14] is a neural network architecture that aggregates information from node-level features in a graph to make graph-level predictions. It does this by adding a permutation-invariant operation that can aggregate information from all node features into a fixed-length vector, which is used to make the final prediction. It works by iteratively refining a set-level representation of the input graph, which is initialized with the node-level features.

Following the general flow of GraphGPS (General Powerful Scalable Graph Transformers) [8], we concatenate the output from the sparse attention output on the cluster level with the GCN output feature on the node level. These concatenated results are passed through a Multi-layer Perceptron to create a final prediction.

Dataset

Our benchmark datasets, described below, span both levels of graph learning. We compare our architecture's performance to those of these common message-passing graph neural networks: GCN, GAT, and GIN. Additionally, we compare it with SAN, [5], a transformer architecture, the first of which to consider the full spectrum of eigenvalues for positional encoding.

Node Classification: Resampled Citation Datasets

Our first benchmark considers the transductive semi-supervised node classification task on citation networks. The Cora, CiteSeer, and PubMed networks [16] node features are bag-of-words representations of documents and edges represent citation links. The goal of this task is to assign a class to each document. To build the labeled training dataset, 20 instances of each class are randomly sampled. 1,000 instances are sampled for the test dataset, and the remaining 500 for the validation set. Our benchmarking method repeats training on three different seeds of splitting on the datasets. The citation networks are summarized in Table 1.

With the aim of tailoring these datasets to the task of long-range model benchmarking, we adopt the resampling scheme introduced in the Hierarchical Graph Network paper [9]. This scheme retains the process of selecting 20 examples from each class for training, but rather than doing so uniformly at random, for a drawn node, it "sanitizes" its k-hop neighborhood of labels. Due to the nature of a citation network, these datasets have high homophily, meaning that most of a node's first-order neighbors belong to the same class. Because of this, it is necessary to ensure the model carries a large enough receptive field to reach beyond this neighborhood, so that correct labels of the k-th order neighbors aren't "imprinted" in their representations. In this study, we employ a buffer of $k = 1$ and later hope to evaluate with $k = 2$. These are the same values as in the HGNet paper.

Graph-Level Prediction: Peptides Functional and Structural Dataset

Peptides are short chains of amino acids, which are abundant in nature and serve various important biological functions. Despite being shorter than proteins, peptides have a molecular structure that is significantly larger than that of a small drug-like molecule, since each amino acid contains multiple heavy atoms. The peptides were sourced from the SATPdb database [10], which provides comprehensive information on the sequence, 3D structure, function, and molecular graph of each peptide. The graphs represent one-dimensional chains of amino acids, emphasizing the need for the model to accurately identify the location of each amino acid within the graph.

Peptides functional is for Graph Classification task and Peptides Structural for Graph Regression task. The graphs in both

Table 1. Citation network dataset statistics.

Dataset	# Nodes	# Classes	# Edges
Citeseer	3,312	6	4,732
Cora	2,708	7	5,429
PubMed	19,717	3	44,338

datasets are correspond to peptides chain structures and are derived in such way that the graphs have large diameters relative to their sizes. Both datasets have 15535 graphs. [2]

We chose these two datasets in order to evaluate the GNNs' abilities to learn long-range interactions for Graph-level tasks.

METHOD

In the architecture section, we detailed each component within our model and in the dataset & benchmark section, we listed the datasets we used and the SOTA models we compared our model against.

Architecture

SignNet

In the architecture we determined to use SignNet to compute the positional encoding for each graph. We pre-computed the top 50 eigenvalues and corresponding eigenvectors of the laplacian matrix L of each graph. Then we used the computed top 50 eigenvectors (Nx50) as the input for the SignNet. The SignNet will be a function $f\colon \mathbb{R}^{n\times k} \to \mathbb{R}^{n\times d}$ where d is arbitrary dimentions.

Spectral clustering

We trained a GNN model using pytorch-geometric and used the loss functions as the sum of minCUT Loss and orthogonal Loss to compute the cluster assignment for each graphs. We trained for each graph for about 50 iterations with a early stop threshold of loss decrease as 0.001. With the trained model we get the cluster assignments which we used in the next step to create virtual nodes and construct the heterogenous graph dataset.

Heterogeneous graph

From the original graphs after we trained and computed the clusters we will add one new virtual node for each of the cluster. For K clusters, where K is a arbitrary number, we will add K new virtual nodes. For each virtual node, we will add M new edges between all the nodes in one cluster and the new virtual node, where M is the number of the nodes in a cluster. After virtual nodes connect the local nodes, we will fully connect all the virtual nodes. To address the different properties of the edges from local nodes to local nodes, the edges from local nodes to virtual nodes and the edges from virtual nodes to virtual nodes, we applied the idea of Heterogenous Graph Neural Network by using different message passing convolution layers for different types of edges. [11]

Finally, here is the flow of our architecture diagram [Figure 1].

RESULTS

Because of the limited time and computational resources, we could not run the experiment for all the datasets or even one

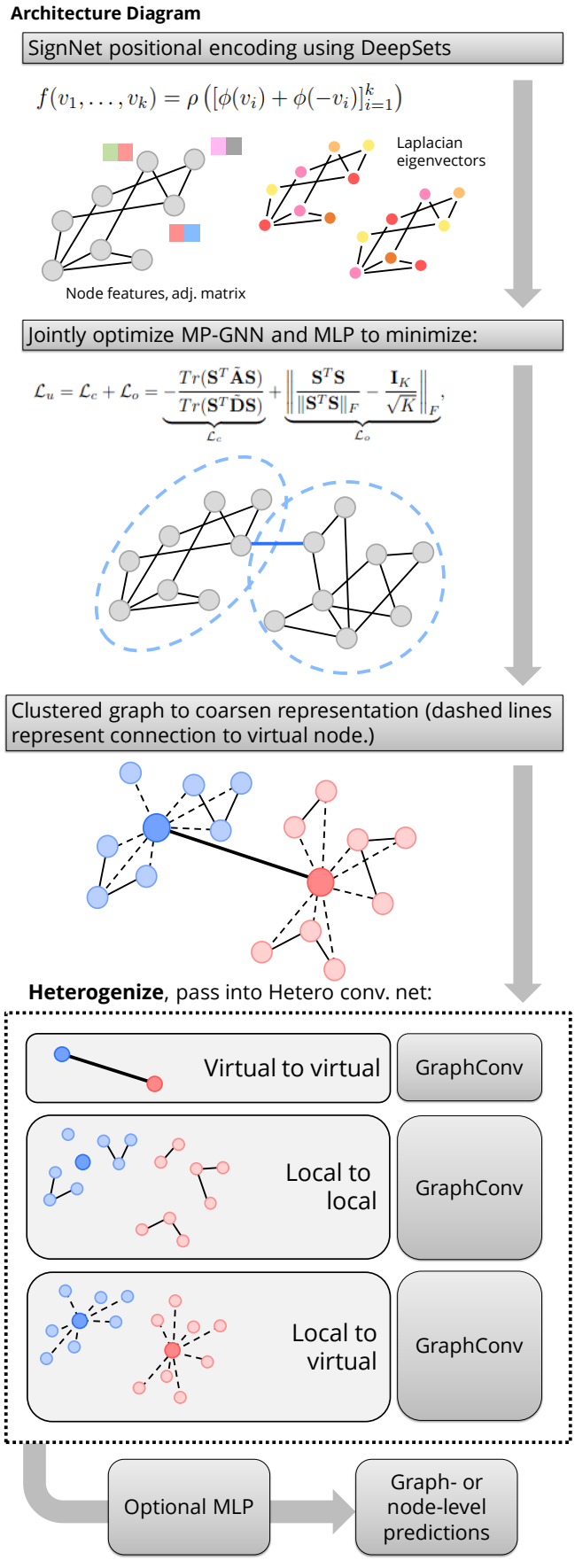

Figure 1. Architecture diagram for GraphHSCN

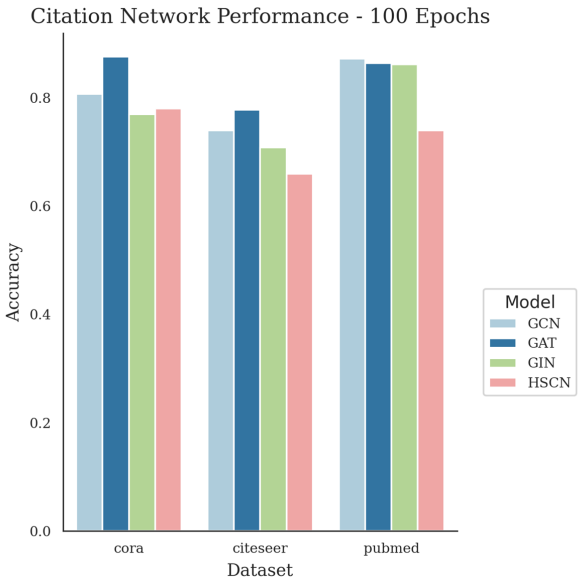

Figure 2. Results on the resampled citation datasets

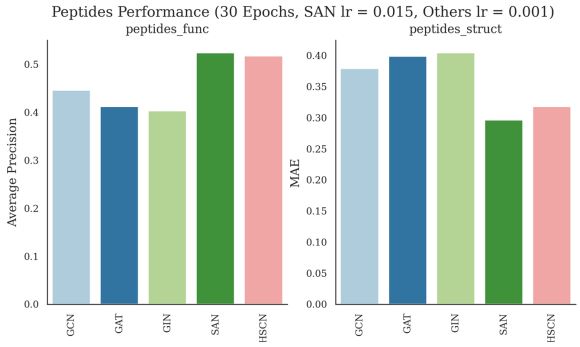

Figure 3. Results on the peptides datasets

whole dataset. Therefore, we chose the dataset Peptides-func with a task of multi-label graph classification and citation dataset. We used Average Precision as suggested by the authors to analyze the results of our experiments. Our baseline models are GCN, GAT, and GIN with the resampled citation dataset graph [Figure 2].

After that, we also measure the models' performance in two of the datasets (Peptides-func & Peptides-struct) from LRGB [2]. Figure 3 shows their performance.

As figures indicate, we achieved a similar performance as SAN for peptides datasets, which is better than other common Message-passing GNNs. Yet, we are way more efficient than SAN.

DISCUSSION

As our model is not performing as well on the resampled citation datasets, we proposes the following potential causes:

- The Cora dataset originally might not contain very important long-range interactions, therefore after re-sample it's still not sure if the long-range interactions now contained in the graphs are strong enough to have great influences on the predictions.
- Cora dataset contains one single graph for node-level tasks. While our models work well for graph-level tasks, the cluster-method with the fully connected virtual nodes might lead the graph to another side of over-smoothing. While we used heterogenous GNN to try to address this possible issue, that might still has an influence on node-level predictions tasks.

There are still some limitations too. First of all, it will probably not scale well when there are many clusters. For a larger number of cluster, we could use sparse attention to better model the interaction between these clusters. Second, while we used several different datasets, the graphs' sizes in peptides datasets are relatively small. Last limitation is that the number of the clusters has a large influence on the final performance of the model and currently we have no better method but cross-validation to chose a best cluter number. In the future we will add the sparse attention mechanism along with more experiments on graphs that are larger. For the cluster numbers K we need a method to analyze the exact affect of the number of the clusters to the performances of the models on different graphs.

CONCLUSION

In this research we explored and proposed a new architecture to help Message Passsing Neural Network to learn long range interactions. We utilized the ideas of Spectral Clustering and Heterogenous Graph Neural Network to decrease the diameters of the of the new graph while maintain the original graph's topological structure unchanged. We compared the new results with the baseline models including MLP and other GNNs and confirmed our model's improvements in terms of learning the graphs with long-range interactions. Another contribution for our model is to mitigate the issue of run-time complexity of the models that relied on fully connected graphs' attention mechanism and transformer like SAN. Our model showed comparable performances with SAN but with much smaller run-time complexity.

REFERENCES

[1] Filippo Maria Bianchi, Daniele Grattarola, and Cesare Alippi. 2020. Spectral clustering with graph neural networks for graph pooling. In *International conference on machine learning*. PMLR, 874–883.

[2] Vijay Prakash Dwivedi, Ladislav Rampášek, Mikhail Galkin, Ali Parviz, Guy Wolf, Anh Tuan Luu, and Dominique Beaini. 2022. Long range graph benchmark. *arXiv preprint arXiv:2206.08164* (2022).

[3] Kedar Karhadkar, Pradeep Kr Banerjee, and Guido Montúfar. 2022. FoSR: First-order spectral rewiring for addressing oversquashing in GNNs. *arXiv preprint arXiv:2210.11790* (2022).

[4] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[5] Devin Kreuzer, Dominique Beaini, Will Hamilton, Vincent Létourneau, and Prudencio Tossou. 2021. Rethinking graph transformers with spectral attention. *Advances in Neural Information Processing Systems* 34 (2021), 21618–21629.

[6] Derek Lim, Joshua Robinson, Lingxiao Zhao, Tess Smidt, Suvrit Sra, Haggai Maron, and Stefanie Jegelka. 2022. Sign and basis invariant networks for spectral graph representation learning. *arXiv preprint arXiv:2202.13013* (2022).

[7] Kenta Oono and Taiji Suzuki. 2019. Graph neural networks exponentially lose expressive power for node classification. *arXiv preprint arXiv:1905.10947* (2019).

[8] Ladislav Rampášek, Mikhail Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. 2022. Recipe for a general, powerful, scalable graph transformer. *arXiv preprint arXiv:2205.12454* (2022).

[9] Ladislav Rampášek and Guy Wolf. 2021. Hierarchical graph neural nets can capture long-range interactions. (2021). DOI: http://dx.doi.org/10.48550/ARXIV.2107.07432

[10] Sandeep Singh, Kumardeep Chaudhary, Sandeep Dhanda, Sherry Bhalla, Salman Usmani, Ankur Gautam, Abhishek Tuknait, Piyush Agrawal, Deepika Mathur, and Gajendra Raghava. 2015. SATPdb: a database of structurally annotated therapeutic peptides. *Nucleic Acids Research* 44 (11 2015), gkv1114. DOI: http://dx.doi.org/10.1093/nar/gkv1114

[11] Yizhou Sun and Jiawei Han. 2013. Mining heterogeneous information networks: a structural analysis approach. *Acm Sigkdd Explorations Newsletter* 14, 2 (2013), 20–28.

[12] Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M Bronstein. 2021. Understanding over-squashing and bottlenecks on graphs via curvature. *arXiv preprint arXiv:2111.14522* (2021).

[13] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[14] Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. 2015. Order matters: Sequence to sequence for sets. *arXiv preprint arXiv:1511.06391* (2015).

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

[16] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. 2016. Revisiting Semi-Supervised Learning with Graph Embeddings. *CoRR* abs/1603.08861 (2016). http://arxiv.org/abs/1603.08861

[17] Yang Ye and Shihao Ji. 2021. Sparse graph attention networks. *IEEE Transactions on Knowledge and Data Engineering* 35, 1 (2021), 905–916.

APPENDIX

Github repo: https://github.com/camille-004/Graph-HSCN
Website: https://graphhscn.github.io/