AFFINITY-BASED HOMOPHILY: CAN WE MEASURE HOMOPHILY OF A GRAPH WITHOUT USING NODE LABELS?

Indranil Ojha, Kushal Bose & Swagatam Das
Electronics & Communication Sciences Unit
Indian Statistical Institute
203, B.T. Road, Kolkata - 700108
oindranil@yahoo.co.in, kushalbose92@gmail.com, swagatamdas19@yahoo.co.in

ABSTRACT

The homophily (heterophily) ratio in a graph represents the proportion of edges connecting nodes with similar (dissimilar) class labels. Existing methods for estimating the homophily ratio typically rely on knowing the class labels of each node in the graph. While several algorithms address both homophilic and heterophilic graphs, they necessitate prior knowledge of the homophily ratio to choose the appropriate one. To address this limitation, we propose a novel metric for measuring homophily ratio without needing information about node labels. In our approach, we define learnable affinity vectors for each node, characterizing the expected feature relationships with its neighbors. Our method, Affinity-based Homophily, derives the homophily ratio using these affinity vectors, eliminating the need for prior node label information. We conducted experiments on various benchmark homophilic and heterophilic graphs, demonstrating the commendable performance of our homophily measure.

1 INTRODUCTION AND RELATED WORKS

Graph Neural Networks (GNNs) gain popularity due to its immense success on the graph-structured data. The existing GNN models like GCN (Kipf & Welling (2016), GraphSage Hamilton et al. (2017), SGC Wu et al. (2019), GCNII Chen et al. (2020)) etc. rely on message passing mechanism which mixes the representation of a node with the features of its neighbours. The models assume the homophily nature of the graph where connected node are having identical class labels. The performances of those models decline when they are applied on the heterophilic graphs where connected may have different class labels. Various methods were developed for tackling both homophilic and heterophilic graphs like (Pan & Kang (2023), Bo et al. (2021), Cavallo et al. (2023), Chen et al. (2023)). Homophily ratio may play the decisive role for choosing most well-suited algorithm for the input graph. But in real-world semi-supervised settings, we only have class labels of a few nodes, making it impossible to estimate the homophily ratio. The fact motivates to design a novel approach **A**ffinity-based **H**omophily or **AH** which estimates homophily ratio without having prior information regarding the node labels.

The features of the connected node pairs may not sufficiently represent the actual relationship between them. Therefore, we define a set of learnable vectors attributed to represent the true characteristics of the corresponding neighbors. These vectors are termed as `affinity vectors`. The affinity vectors are sourced from the same domain of the raw input features. In addition, affinity vector should represent the characteristics of the neighbors and input features denotes the properties of the node itself. A more detailed discussion on affinity vectors is available in Appendix A. The affinity vectors are employed to compute self-affinity values of individual nodes i.e. cosine similarity between affinity vector and feature vector. The final homophily ratio is estimated as the combination of self-affinity values of the individual nodes.

Contribution We design a novel and efficient technique to measure homophily ratio by learning affinity vectors. Our method does not require any knowledge regarding the class labels of the nodes.

2 PROPOSED METHOD

Consider an attributed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges, with $|\mathcal{V}| = n$. Let $A \in \mathbb{R}^{n \times n}$ be the adjacency matrix and $X \in \mathbb{R}^{n \times m}$ is the feature matrix containing m-dimensional feature vectors for each node.

Let us assume $Z \in \mathbb{R}^{n \times m}$ as the affinity matrix containing affinity vectors for each node. We define affinity score between i and j is $x_i z_j^\top$ which is the cosine similarity between feature vector of ith node and affinity vector of jth node. The closed form is represented as $XZ^\top \in \mathbb{R}^{n \times n}$. We reformulate our affinity matrix as $\mathcal{F} = (XZ^T + 1)/2$ where similarity scores are scaled to $[0, 1]$ range. The diagonal entries of affinity matrix are the self-affinity scores of the nodes, and they indicate the tendency of nodes to connect to nodes of similar features, i.e. the homophily of the dataset. Note that the diagonals of affinity matrix should not take part in the optimization process, otherwise the self-affinity values will tend to zero. The affinity matrix should capture the structure of the graph topology in a softer way in order to learn an effective representations. Therefore, the affinity vectors are learned by minimizing the following function.

$$\mathcal{L} = 1 - \text{soft F-score}((\sigma(\mathcal{F}) \odot (\mathbb{1} - \mathrm{I}), A), \tag{1}$$

where $\sigma(.)$ denotes the Sigmoid activation function, I is the identity matrix, $\mathbb{1}$ is matrix of all ones. The homophily ratio can be estimated by averaging the self-affinity scores i.e. diagonal elements of the optimized affinity matrix. The direct averaging of self affinity scores may not be a good idea because nodes may higher average degree i.e. tendency to connect with other nodes is higher even for heterophilic graphs. Finally, the Affinity-based Homophily (AH) is estimated as following.

$$\text{AH} = \frac{\log(1 + \mu_d/\mu_{nd})}{\log(1 + (n^2 - n)/2e)}, \tag{2}$$

where μ_d and μ_{nd} are the means of diagonal and non-diagonal entries of affinity matrix respectively, and $e = |\mathcal{E}|$ is the number of edges. We explain the details of the derivations in the Appendix B and Appendix C respectively.

3 EXPERIMENTS & RESULTS

We applied AH measure on three homophilic and six heterophilic datasets including two large graphs. Refer figure 1 for detailed performance of our homophily measure. The results show that AH assigns higher values for homophilic datasets and lower values for heterophilic datasets, establishing the effectiveness of our proposed approach. Refer Appendix C for the visualizations of the affinity matrices. Our source code is available at https://github.com/kushalbose92/affinity-based-homophily.

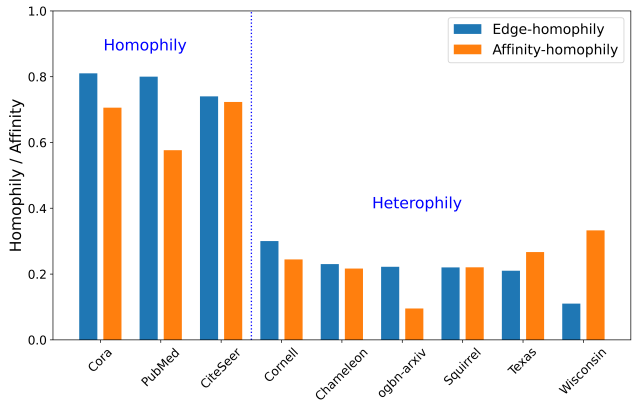


Figure 1: Comparison of homophily ratios assigned by edge-homophily method and our approach AH. The ratios derived by AH follows the identical trend with the ratios of the edge homophily measure.

4 CONCLUSION & FUTURE WORKS

In this work, we designed a novel technique to measure homophily of a graph dataset without using any class labels of the nodes. We applied our measure on several datasets and obtained results which significantly established the efficacy of our measure. As a future work, we should investigate how the proposed homophily measure can help in developing new GNN models for tackling both homophilic and heterophilic graphs.

5 OTHER SECTIONS

URM STATEMENT

The authors acknowledge that at least one key author of this work meets the URM criteria of ICLR 2024 Tiny Papers Track.

REFERENCES

Deyu Bo, Xiao Wang, Chuan Shi, and Huawei Shen. Beyond low-frequency information in graph convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 3950–3957, 2021.

Andrea Cavallo, Claas Grohnfeldt, Michele Russo, Giulio Lovisotto, and Luca Vassio. Gcnh: A simple method for representation learning on heterophilous graphs. *arXiv preprint arXiv:2304.10896*, 2023.

Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *International conference on machine learning*, pp. 1725–1735. PMLR, 2020.

Yuhan Chen, Yihong Luo, Jing Tang, Liang Yang, Siya Qiu, Chuan Wang, and Xiaochun Cao. Lsgnn: Towards general graph neural network in node classification by local similarity. *arXiv preprint arXiv:2305.04225*, 2023.

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

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Erlin Pan and Zhao Kang. Beyond homophily: Reconstructing structure for graph-agnostic clustering. *arXiv preprint arXiv:2305.02931*, 2023.

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

APPENDIX A - MORE ON *Affinity* VECTORS

Homophily ratio plays a pivotal role to understand the characteristics of the underlying graph data. Standard homophily measures are dependent on the similarity between the input node features. In contrast, we define the concept of an *affinity* vector, associated with each node, which is semantically the same as the feature vector, but with different purposes. Please refer to 2 for the detailed illustration.

Every node is equipped with input feature vectors and affinity vectors. Affinity vectors bridge the gap between homophilic and heterophilic datasets. For homophilic datasets, the affinity vector of a node is similar, or close to its feature vector in the feature domain. For heterophily, they will be dissimilar. Once we can learn these affinity vectors, we can use the average similarity of these two vectors for each node (self-affinity) to derive the homophily level of the dataset.

Instead of defining the characteristics of the node itself (what the feature vector does), it actually defines what the node expects in another node in order to establish an edge-connection with it.

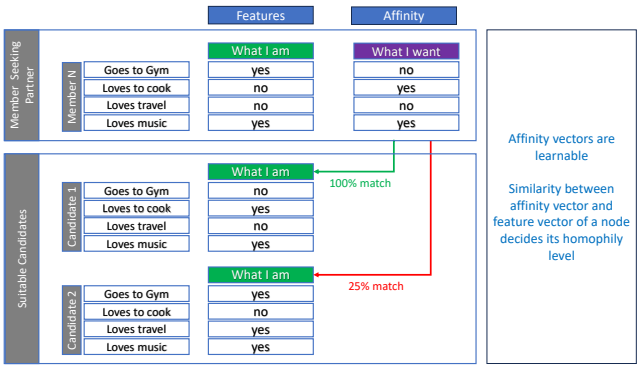


Figure 2: Schematic diagram explaining feature vector and affinity vector.

APPENDIX B - EXPLANATION OF THE LOSS FUNCTION

As mentioned in the Approach section, there are a few implementation challenges which we addressed in innovative ways. We discuss them below.

1. Comparison of hard adjacency matrix and soft affinity matrix - we addressed this by use of sigmoid function that pushes values above 0.5 towards 1 and values below 0.5 towards zero. So we actually compare A with $sigmoid(k\mathcal{F})$, where k is a hyper-parameter that determines how strongly we want to push the values.

2. The MSE loss between the adjacency and affinity matrices is not an appropriate loss function because of the sparse nature of the two matrices. With MSE loss, the model has a tendency to predict zero for all values so that the error automatically comes close to zero. We applied inverted soft F-score as the loss function to address this.

3. The diagonal elements of the affinity matrix and the adjacency matrix cannot take part in training. Diagonal elements of affinity matrix are self-affinities that are used in AH computation. If they take part in training, they will all be either zeroes (no self loops) or ones (self-loops). To resolve this, we force these values in both matrices to zero just before loss computation. This explains the factor $(\mathbb{1} - \mathrm{I})$ in equation 1. Since both zero values in the two matrices being compared indicates a true negative which does not figure in the F-score formula, non-participation of these values in loss calculation is ensured.

   These explain the loss function $\mathcal{L}$ as defined in equation 1.

APPENDIX C - EXPLANATION OF THE FORMULA FOR AFFINITY BASED HOMOPHILY COMPUTATION

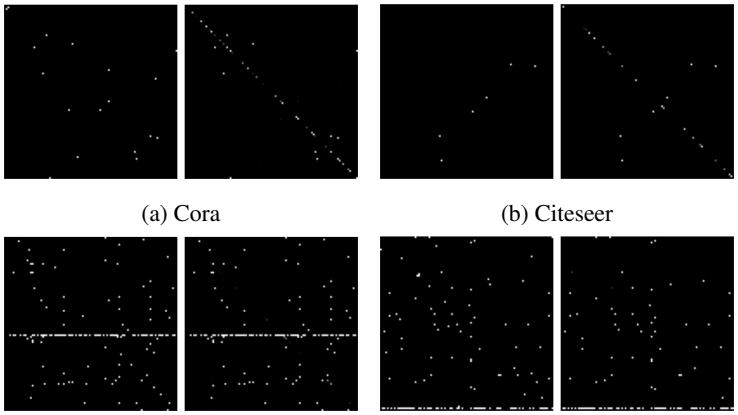
Once the affinity matrix is learnt, an immediate thought is to take the mean of the diagonal as a measure of homophily. But different graphs have different edge-density. Even a heterophilic graph data that has high edge density will have, in general, high node-to-node affinity, and self-affinity will be no exception. So we decided to take a ratio of mean self-affinity vs mean affinity between distinct nodes. But this ratio will not be in the range of [0,1] which gives rise to two issues. Firstly, we cannot compare this new metric with edge-homophily, and also it will not be possible to draw a line between homophilic or heterophilic datasets (we can take 0.5 as that line for edge-homophily). Fortunately, we found that AH is bounded above and we use that bound to force its value to be in the [0,1] range. Finally, we define AH using the formula given in 2. The proof for existence of an upper bound is given below.

Theorem 1 *The ratio of diagonal mean and non-diagonal mean of affinity matrix is upper-bounded by* $\dfrac{n^2-n}{2e}$

Proof: In the ideal case, the affinity matrix of a graph is equal to the adjacency matrix A. For a completely homophilic graph, the self-affinity of each node must be 1. Hence, $\mu_d = 1$. The number of non-diagonal entries in A is $n^2 - n$. Since there are $2e$ entries in non-diagonal part of the adjacency matrix, we have $2e$ entries equal to 1 and rest are zero. So $\mu_{nd} = \dfrac{2e}{n^2-n}$. Hence the maximum value of the ratio is $\dfrac{\mu_d}{\mu_{nd}} = \dfrac{1}{\mu_{nd}} = \dfrac{n^2-n}{2e}$. [Note that $2e$ should be replaced by e for directed graphs.]

APPENDIX D - VISUALIZATION OF AFFINITY MATRIX

One interesting way to visualize the effectiveness of the affinity-based approach is shown in figure 3. Each of the four images (a, b, c and d) contains two plots - the left one is the adjacency matrix and right one is the affinity matrix. After a good training, these two plots should look similar. However, a prominent diagonal appears only for homophilic datasets, indicating high self-affinity of the nodes. We ensured diagonal elements not to take part in training. Even then, the images for Cora and CiteSeer (Figures 3a and 3b) show this phenomenon, while for Texas and Wisconsin (figures 3c and 3d), the diagonal is absent. (For better resolution, only first 100 nodes have been plotted).



(a) Cora (b) Citeseer

(c) Texas (d) Wisconsin

Figure 3: Visualization of affinity matrix - the appearance of diagonal indicates high homophily.