ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-7605-5/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICASSP39728.2021.9414377

LEARNING THE RELEVANT SUBSTRUCTURES FOR TASKS ON GRAPH DATA

Lei Chen, Zhengdao Chen, Joan Bruna

New York University

ABSTRACT

Focusing on graph-structured prediction tasks, we demonstrate the ability of neural networks to provide both strong predictive performance and easy interpretability, two properties often at odds in modern deep architectures. We formulate the latter by the ability to extract the relevant substructures for a given task, inspired by biology and chemistry applications. To do so, we utilize the Local Relational Pooling (LRP) model, which is recently introduced with motivations from substructure counting. In this work, we demonstrate that LRP models can be used on challenging graph classification tasks to provide both state-of-the-art performance *and* interpretability, through the detection of the relevant substructures used by the network to make its decisions. Besides their broad applications (biology, chemistry, fraud detection, etc.), these models also raise new theoretical questions related to compressed sensing and to computational thresholds on random graphs.

Index Terms— Graph, pooling, substructure

1. INTRODUCTION

In data analysis, strong predictive performance is important, as is the ability to interpret the predictions made by the algorithm. In modern deep learning architectures, these two aspects are often in tension. In typical DL application domains – such as computer vision or natural language processing – the quest for stronger performances has often overpowered the need to provide interpretable decisions, despite considerable research to provide meaningful interpretation of pre-trained large neural networks.

Graph-structured prediction tasks offer a new opportunity to re-evaluate this tradeoff. Graph Neural Networks [1] (GNN) are a class of neural architectures that encode the symmetries of the graph-structured inputs by design, such as permutation invariance. A powerful paradigm to provide both predictive power and interpretability is based on sparsity: a high-dimensional observation is expressed as a sparse combination of features out of a (possibly very large) dictionary of elements. In the context of graph-structured tasks, a natural candidate for providing such dictionaries are *graph substructures*, defined as attributed subgraphs contained in a given input graph.

While GNNs based on the Weisfeler-Lehman (WL) framework are amongst the most popular nowadays (such as Message-Passing Neural Networks [2], or Graph Attention Network [3]), they are provably unable to detect or count attributed graph substructures of more than 2 nodes [4], and are therefore unable to be interpreted in terms of relevant substructures. An alternative GNN framework based on Local Relational Pooling [5, 4] uses instead generic non-linear representations of local egonets, that are symmetrised to ensure the resulting model to be permutation-invariant. While WL-based networks cannot count substructures, LRP models are by design able to identify substructures contained in the local egonets, leading to improved empirical performances [4].

In this work, we demonstrate another benefit of LRPs by introducing an algorithm to extract task-specific relevant substructures, similar in spirit to Class Activation Mapping [6]. This algorithm is tested empirically on a challenging molecular prediction task, and connections to random graph theory and compressed sensing are discussed informally.

2. LOCAL RELATIONAL POOLING (LRP)

We consider an *attributed* graph $G = (V, E, x, e) \in \mathcal{G}$, where typically $V = [n] := \{1, \dots, n\}$ is the vertex set, $E \subset V \times V$ is the edge set, $x_i \in \mathcal{X}$ represents the node feature (or attribute) of node i, and $e_{i,j} \in \mathcal{E}$ represents the edge feature of edge (i, j) if $(i, j) \in E$. We define the node degree vector $D \in \mathbb{N}^n$ with D_i being the degree of node i. For convenience of tensor formulation, we assume that $\mathcal{X}$ and $\mathcal{E}$ are subsets of $\mathbb{R}^d$. For simplicity, we only consider undirected graphs without self loops. Note that an *unattributed* graph can be viewed as an attributed graph with identical node and edge features.

Relational Pooling [5] generates a permutation-invariant model by symmetrizing a model that is not necessarily permutation-invariant. Let $\mathbf{B} \in \mathbb{R}^{n \times n \times (d+1)}$ be the node-ordering-dependent representation of the graph G defined as

$$\mathbf{B}_{:,:,1} = A, \tag{1}$$

$$\mathbf{B}_{i,i,2:} = x_i, \quad \forall i \in [n], \tag{2}$$

$$\mathbf{B}_{i,j,2:} = e_{i,j}, \quad \forall (i, j) \in E, \tag{3}$$

where $A \in \{0,1\}^{n\times n}$ is the unweighted adjacency matrix of G. Then the Relational Pooling model is defined as

$$f_{\text{RP}}(G) = \frac{1}{|S_n|}\sum_{\pi\in S_n} \vec{f}(\pi \circ \mathbf{B}), \tag{4}$$

where $\vec{f}$ is some function that can be permutation-sensitive and S_n is the permutation group of n elements, and π acts on $\mathbf{B}$ by permuting the first two dimensions of the $\mathbf{B}$. Assuming $\mathcal{X}$ and $\mathcal{E}$ are finite sets, the set of all such f_{RP} is proven to be a universal approximator of permutation-invariant functions [5].

Local Relational Pooling [4] simplifies the permutation on the entire graph in (4) to permutations on egonets [7]. Since egonets are rooted graphs, they further reduce the symmetrization over all permutations in S_n to within a subset $S_n^{\text{BFS}} \subseteq S_n$ or $S_n^{\text{DFS}} \subseteq S_n$, which consist of the permutations compatible with breath-first-search (BFS) or depth-first-search (DFS), respectively, to further reduce the complexity. Below we focus on the latter version, LRP-DFS, which is more aligned with our target.

Similar to (4), we formulate a **one-layer** LRP-DFS-k as

$$\hat{f}_{\text{LRP}}(G,i) = \frac{1}{|S_{i,k}^{\text{DFS}}|}\sum_{p\in S_i^{\text{DFS}}} \vec{f}(p), \tag{5}$$

$$f_{\text{LRP}}(G) = \frac{1}{|V|}\sum_{i\in V} \hat{f}_{\text{LRP}}(G,i), \tag{6}$$

where $\vec{f}$ is some permutation-sensitive function and $S_{i,k}^{\text{DFS}}$ is a *bag of sequences* that is generated by *k-Truncated DFS* from node i in G, to be defined below.

We define k-Truncated DFS from node i as a procedure that returns all depth-k *acyclic* paths starting from node i in the graph G. Thus, all paths that it returns have length at most $(k+1)$. Note that a path p in $S_{i,k}^{\text{DFS}}$ has length $l < k+1$ if and only if all neighbors of the last node of p already appear in p before the last node. Formally, this can be written as

$$\mathcal{N}(p[l]) \setminus p[1,\dots,l-1] = \emptyset, \tag{7}$$

where $\mathcal{N}(\cdot)$ denotes the neighboring set. For example, in a graph G with 4 nodes and $E = \{(1,2),(1,3),(2,3),(3,4)\}$, the bags of sequences from node 1 are

$$\begin{aligned} S_{1,1}^{\text{DFS}} &= \{[1,2],[1,3]\},\\ S_{1,2}^{\text{DFS}} &= \{[1,2,3],[1,3,2],[1,3,4]\},\\ S_{1,k\geq 3}^{\text{DFS}} &= \{[1,2,3,4],[1,3,2],[1,3,4]\}. \end{aligned}$$

In our implementation, we chose the the permutation-sensitive function $\vec{f}$ to be

$$\vec{f}(p) = \vec{f}_1(\mathbf{B}_p) \odot \vec{f}_2(D_p), \tag{8}$$

$$\mathbf{B}_p = \mathbf{B}_{p\otimes p,:} \in \mathbb{R}^{(k+1)^2\times d}, \tag{9}$$

$$[\mathbf{D}_p]_i = D_{p[i]}, \tag{10}$$

where $p\otimes p$ is a Cartesian product of p and itself that returns a 2-dimensional list, and $\vec{f}_1$, $\vec{f}_2$ are two learnable MLPs. We use zero padding when $|p| < k+1$ in (9). [4] designs a multi-layer LRP with initialization $\mathbf{B}^{(0)} = \mathbf{B}$ and iteration $\hat{f}_{\text{LRP}}(G,\cdot) : \mathbf{B}^{(t)} \mapsto \mathbf{B}^{(t+1)}$.

3. UNDERSTANDING THE RELEVANT SUBSTRUCTURES USING LRP

We consider a graph classification task with training data $\mathcal{D} = \{(G_i, y_i)\}_i$, where $y_i \in [c]$ is a classification label with c as the number of classes. For simplicity, we omit i for most notations in this section. Assuming we have a trained model f_{LRP} with a choice of $\vec{f}$ that is capable of fitting the training data, our next step is to interpret the output of $\vec{f}$ to extract substructures that make large contributions to the prediction. Specifically, we are interested in two tasks in a pipeline: locating and summarizing.

3.1. Locating Relevant Substructures

Given a trained model f_{LRP} and a grpah G, we define the locating task as looking for elements in the power set of V, denoted as $\mathcal{P}(V)$, that make significant contributions to the graph label y. We define a function $H : S\mathcal{P}(V) \to \mathcal{P}(V)$, where $S\mathcal{P}(V)$ is a power set with ordering induced by permutation. $H(\cdot)$ maps an ordered sequence p in (5) to its permutation-invariant node subset. Moreover, we can define equivalence between sequences, induced by $H(\cdot)$, as $p_1 \overset{H}{\sim} p_2 \iff H(p_1) = H(p_2)$.

Given a graph G, a label y and a trained function $\vec{f}$,we define the **L**ocating contribution function C_L as

$$C_L(H(p);(G,y),\vec{f}) = -R\left(y, \sum_{i\in V}\sum_{p'\in S_{i,k}^{\text{DFS}}} \mathbb{1}[p' \overset{H}{\sim} p]\cdot \vec{f}(p')\right), \tag{11}$$

where $R(\cdot,\cdot)$ is cross entropy loss and $\mathbb{1}[\cdot]$ is indicator function. Our next step to sort the support of $C_L(\cdot;(G,y),\vec{f})$ according the function values in an descending order. The leading node subsets are the locations of significant substructures in a certain graph G with label y for the model $\vec{f}$.

3.2. Summarizing Relevant Substructures

We define the summarizing task as looking for small graphs in the space $\mathcal{G}$ that contributes the most in each class. Compared with the locating tasks, this task is in a larger scale than the locating task in two senses: 1) it is across graphs in the same class, and 2) it is across different trained models to return reliable substructures.

Similar to $H(\cdot,\cdot)$ in Section 3.1, we define another function $K : S\mathcal{P}(V) \to \Omega(\mathcal{G})$, where $\Omega(\mathcal{G})$ is orbits in the space

$\mathcal{G}$ with respect to graph isomorphism. We say $p_1 \overset{K}{\sim} p_2 \iff K(p_1) = K(p_2)$, which means subgraphs induced by these two sequences are isomorphic. Therefore, the summarizing task is to find elements in $\Omega(\mathcal{G})$ that contribute the most in each class.

Given a graph G, a label y and a trained function $\vec{f}$, we define the **S**ummarizing contribution function C_S as

$$C_S(K(p);(G,y),\vec{f}) = -R\left(y, \sum_{i\in V}\sum_{p'\in S_{i,k}^{\text{DFS}}} \mathbb{1}[p' \overset{K}{\sim} p]\cdot \vec{f}(p')\right). \tag{12}$$

Then, we sort the support $C_S(\cdot;(G,y),\vec{f})$, get top-m substructures, and build a query function $\mathsf{r}(\cdot;(G,y),\vec{f})$ that returns the substruture's ranking in $C_S(\cdot;(G,y),\vec{f})$. If a graph G_s is not among the top-m, we set $\mathsf{r}(G_s;(G,y),\vec{f}) = m+1$. Finally, we obtain a summary, denoted as $\bar{\mathsf{r}}$, of a substructure G_s in $\mathcal{D}_j$ across models in $\{\vec{f}\}$, as

$$\bar{\mathsf{r}}(G_s;j) = \underset{\substack{(G,y)\in\mathcal{D}_j \\ \vec{f}\in\{\vec{f}\}}}{\text{average}} \mathsf{r}(G_s;(G,y),\vec{f}), \tag{13}$$

where j denotes a class and $\mathcal{D}_j$ is a subset of $\mathcal{D}$ corresponding to class j. Hence, $\bar{\mathsf{r}}(\cdot;j)$ summarizes the top substructures contributing in class j.

3.3. Additional Remarks

The pipeline we proposed for locating and summarizing is analogous to Class Activation Mapping (CAM) [6] that focuses on locating active regions on images, while our methods manage to locate "active" regions (node sequences) on graphs.

In (11) and (12) we set $R(\cdot,\cdot)$ as cross entropy loss for classification tasks. In practice, we would like to divide the model outputs by a constant factor $\gamma > 1$ to observe loss differences more obviously, otherwise top supports might always have zero losses. In our experiments, we set $\gamma = 100$. On the other hand, we would like to point out that $R(\cdot,\cdot)$ is also compatible with regression tasks, with $R(y,O) = \left\| y * \sum_{i\in V}|S_{i,k}^{\text{DFS}}| - O\right\|_2$, where the multiplier $\sum_{i\in V}|S_{i,k}^{\text{DFS}}|$ is due to two average operations in (5) and (6).

We would like to point out some future directions: 1) to get richer bags of sequences, we can involve more general search algorithms, such as search with early stopping, hybrid search combining BFS and DFS; 2) for larger-scale tasks, sampling is a promising way to alleviate the computation cost; 3) it might help the community understand the hierarchy of graph structures through developing a pipeline with a multi-layer LRP model.

4. EXPERIMENTS

Before we conduct tasks of locating and summarizing, we implement a one-layer LRP-DFS-5 on the graph classification task for MUTAG, which is a 2-class dataset of 188 mutagenic aromatic and heteroaromatic nitro compounds with 7 node categories [8]. We follow the setting in [9] that tests the model with 10 folds, which also gives us 10 trained models to summarize substructures. The graph classification results on MUTAG is shown in Table 1. It turns out the 1-layer LRP-DFS-5 outperforms the baselines. Hence, it justifies the assumption that the trained LRP model captures task-relevant information in the dataset to a reasonable extent.

Model	GIN [9]	PPGN [10]	Ring-GNN [11]	LRP-DFS-5†
Test acc.	89.4±5.6	89.44±8.05	86.8±6.4	90.0±6.8

Table 1. Graph Classificaiton Results on the dataset MUTAG. †: LRP-DFS-5 is a 1-layer model run by us.

4.1. Locating Substructures

We randomly pick two graph with different labels in MUTAG and, following the approach described in Section 3.1, locate their top-9 significant substructures given a trained model $\vec{f}$, as shown in Figure 1. It turns out that the leading locations in the graph with label 0 form some 2-paths, 3-paths and 6-paths with subtle differences in losses R, while the leading locations in the graph with label 1 form 6-cycles with larger differences in R. Therefore, we can conclude that among our bags of sequences, 6-cycle makes the most significant contribution to the label 1 for the graph in Figure 1(b).

4.2. Summarizing Substructures

We set the 10 trained models, whose results are reported in Table 1, as our model set $\{\vec{f}\}$. We split $\mathcal{D}$ into $\mathcal{D}_0$ and $\mathcal{D}_1$ according to the labels, as described in Section 3.2, with $|\mathcal{D}_0| = 125$ and $|\mathcal{D}_1| = 63$. We set $m = 9$. Following the approach described in Section 3.2, we summarize important substructures for each class across 10 models, as shown in Table 2. It turns out that, among our bags of sequences, the most significant substructure in graphs of class 0 is a 6-carbon path while that in graphs of class 1 is a 6-carbon cycle, or a benzene ring. Another important pattern in graphs of class 0 is NO_2 group. The observations of the benzene ring and the NO_2 group coincide with results from [12].

5. DISCUSSION

We have introduced a method based on Local Relational Pooling to extract class-specific relevant substructures, while providing state-of-the-art prediction results. In addition to the

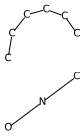
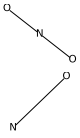
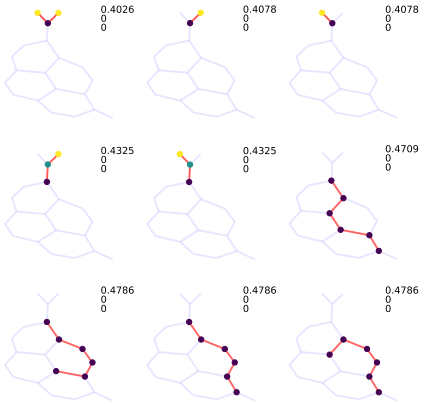
	pattern	$\bar{r}$	pattern	$\bar{r}$
Class 0		2.00		3.94
		4.87		5.36
Class 1		1.91		5.53
		4.82		8.21

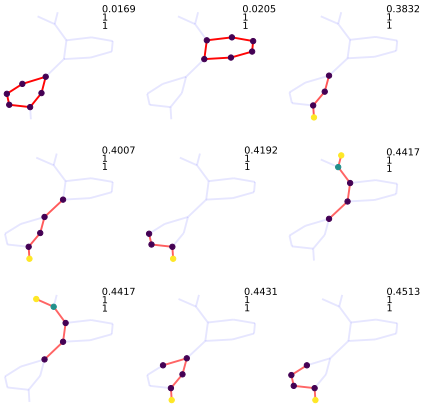
Table 2. Summarizing significant substructures for different classes in MUTAG. Smaller $\bar{r}$ indicates more significance.

broad class of applications where such method could be useful in practice, we raise the following theoretical questions for future work:

- *Compressed sensing of substructures:* In the current version presented, substructures are matched greedily (eq. 11). An interesting alternative is to group candidate substructures together, in order to improve the computational efficiency of the localization algorithm. Individual substructures could afterwards be identified by further relying on incoherence assumptions of the groups, similarly as in compressed censing.
- *Hierarchical substructures and Deep LRP:* In some applications, such as computer graphics or particle physics, graph structures have an important multiscale signature, which limits the efficiency of single-resolution LRP. An important direction of future improvement is to consider Deep LRP models [4] and study whether they can detect multiscale substructures.
- *Random graphs and computational thresholds:* Counting substructures on random graphs, such as cliques, is a canonical problem in theoretical computer science and statistical physics that showcases the so-called statistics-to-computation thresholds. LRP provides a computationally tractable method for a generalised setup, with attributed random graphs and non-linear mappings defining a positive count. An interesting question for future study is thus to first understand relevant statistical limits for such tasks, and whether models like LRP are able to attain those limits.



(a) a graph with label 0



(b) a graph with label 1

Fig. 1. Locating relevant substructures on two graphs belonging to different classes given a trained model. Each upper right corners have three numbers: cross entropy loss R, predicted label, true label. Smaller R indicates more significance. Node colors stand for node categories.

6. REFERENCES

[1] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2008.

[2] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl, "Neural message passing for quantum chemistry," in *International Conference on Machine Learning*, 2017, pp. 1263–1272.

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

[4] Zhengdao Chen, Lei Chen, Soledad Villar, and Joan Bruna, "Can graph neural networks count substructures?," *arXiv preprint arXiv:2002.04025*, 2020.

[5] Ryan Murphy, Balasubramaniam Srinivasan, Vinayak Rao, and Bruno Ribeiro, "Relational pooling for graph representations," in *International Conference on Machine Learning*, 2019, pp. 4663–4673.

[6] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba, "Learning deep features for discriminative localization," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2921–2929.

[7] Victor M Preciado, Moez Draief, and Ali Jadbabaie, "Structural analysis of viral spreading processes in social and communication networks using egonets," *arXiv preprint arXiv:1209.0341*, 2012.

[8] Pinar Yanardag and SVN Vishwanathan, "Deep graph kernels," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, pp. 1365–1374.

[9] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka, "How powerful are graph neural networks?," in *International Conference on Learning Representations*, 2018.

[10] Haggai Maron, Heli Ben-Hamu, Hadar Serviansky, and Yaron Lipman, "Provably powerful graph networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 2156–2167.

[11] Zhengdao Chen, Soledad Villar, Lei Chen, and Joan Bruna, "On the equivalence between graph isomorphism testing and function approximation with gnns," in *Advances in Neural Information Processing Systems*, 2019, pp. 15894–15902.

[12] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec, "Gnnexplainer: Generating explanations for graph neural networks," in *Advances in neural information processing systems*, 2019, pp. 9244–9255.