Empirical comparison of the interpretability of graphlet and symmetry kernels

Yannick Kuhar[1], **Uroš Čibej**[1]

[1]*University of Ljubljana; Faculty of Computer and Information Science*
E-mail: yannick.kuhar@fri.uni-lj.si, uros.cibej@fri.uni-lj.si

Abstract

Graph datasets have become a valuable resource of knowledge. To successfully mine knowledge from them, we need interpretable machine learning methods. One concept that remains underexplored in this context is graph automorphisms or graph symmetries. We want to explore graph symmetries as they offer an interpretable way to mine knowledge from graphs. Our primary goal in this paper is to assess both symmetry-based and traditional graph kernels from the perspective of interpretability. We compare explanations generated by machine learning models trained on various graph kernels across datasets from multiple domains. By analyzing the similarities and differences in these explanations, we aim to pinpoint the scenarios in data domains and model types where symmetry-based kernels offer unique insights. Ultimately, our findings guide further research of more interpretable and effective graph learning methods.

1 Indroduction

Machine learning on complex data structures has become a popular area of research. In many domains, such as chemistry, bioinformatics, social network analysis, and computer vision, data naturally appears as graphs [1]. We need these machine learning methods to be interpretable to better analyze graphs and gain knowledge from graph datasets. Methods that offer interpretable features are graph kernels. They are functions that map graphs to vectors and use them to measure the similarity between graphs. Researchers have proposed various graph kernels [1]. Some utilize subgraph structure as the basis of the mapping, while others use random walks. Our goal for this paper is to explore graph symmetry-based kernels and how interpretable their features are compared to those of their predecessor, the graphlet kernel.

Graph neural networks (GNNs) were proposed to model graph-structured data, but their non-interpretability remains an issue [4]. Given the appropriate model and explanation, much knowledge could be mined from various graph datasets. GNNs may not be the appropriate tool for such a task. On the other hand, graph kernels construct features based on graph properties, which makes them a more appropriate tool for knowledge mining, as the features represent something. In many cases, the cells of a graph's feature vector represent counts of some pattern that occurs in a given graph. Measuring the significance of these patterns could provide greater insight into a domain consisting of graph-structured data.

Graph symmetries are a graph property that has not yet been fully explored in graph kernel interpretability. We will test symmetry-based kernels on various graph datasets and compare their explanations to the graphlet kernel. The explanations will be constructed using Permutation Feature Importance (PFI) [12].

2 Related Work

Graph symmetries have been studied in the field of graph compression. Two algorithms have been developed that exploit them to compress undirected graphs [2]. This research shows that symmetries carry information about the graph's structure, thus making them valuable for graph classification.

Over the years, researchers have proposed many graph kernels based on different graph properties. Random walk kernels were among the first developed [6]. Perhaps the most well-known kernels were the Graphlet kernel [7] that uses the graphlets distribution and the Weisfeiler-Lehman kernel [8] that works on a dataset of node-labeled graphs and uses a label refinement algorithm.

In recent years, graph neural networks [3] have become a popular alternative tool to graph kernels as they achieve strong performances. In this paper, we will not use graph neural networks, as our main objective is to investigate the use of symmetries in the context of graph kernels. This would lay the foundation for further research to investigate the amount of information carried by graph symmetries. To this end, we will use a well-known tool to explain the models.

The aim of interpretability as a field is to understand why certain decisions and predictions were made [11]. These insights can help us better understand the relationship between the features and the targets, which can help guide further development and scientific discovery. SHapley Additive exPlanations (SHAP) [5] has proven a powerful model-agnostic method to provide feature importance. However, it is not scalable to larger datasets due to its high computational complexity. On the other hand, Permutation Feature Importance (PFI) [12] accomplishes a similar task but much faster, this is the main reason we

chose to use this method in our work.

3 Preliminaries

This section introduces the basic concepts of graphs and describes their relevant graph kernels.

3.1 Basic concepts

A graph is defined as $G = (V, E)$, where $V(G)$ is the set of vertices and $E(G)$ is the set of edges. Graph G is undirected if $(v_i), v_j) \in E(G) \Leftrightarrow (v_j, v_i) \in E(G)$. Otherwise, it is a directed graph. An edge (v_i, v_i) is called a *loop*. In general, vertices can be connected to more than one edge. A simple graph has no loops and no multiple edges. In this paper, we will only use simple undirected graphs.

Graph symmetries are a synonym for graph automorphisms. Their standard representation is a permutation of the vertex set or edge set. Let us look at vertex symmetries as an example. We can write them as permutations of the vertex set $V(G)$, i.e., a bijective function $\pi : V(G) \to V(G)$ that preserves connectivity ($(u, v) \in E(G) \Leftrightarrow (\pi(u), \pi(v)) \in E(G)$). All such permutations form a permutation group $Aut(G)$ [2], an example is shown in Figure 1.

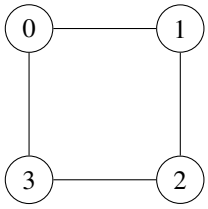


Figure 1: A simple undirected graph G with 4 nodes and 4 edges. $Aut(G)$ = {(1,3), (0,2), (0,1)(2,3), (0,2)(1,3), (0,3)(1,2), (0,1,2,3), (0,3,2,1)}

Each vertex-symmetry π induces an edge-symmetry $\overline{\pi}(u, v) = (\pi(u), \pi(v))$. Graph symmetries have been successfully used in graph compression [2], but in this study, we will explore how interpretable they are as machine learning features.

3.2 Graph kernels

Kriege et al. [1] define graph kernels as follows: Let $\mathcal{G}$ be a non-empty, finite set of graphs, and let $k : \mathcal{G} \times \mathcal{G} \to \mathbb{R}$ be a function. The function k is called a kernel on $\mathcal{G}$ if there exists a Hilbert space $\mathcal{H}_k$ and a feature map $\phi : \mathcal{G} \to \mathcal{H}_k$ such that, for any graphs $G, G' \in \mathcal{G}$, the kernel satisfies:

$$k(G, G') = \langle \phi(G), \phi(G') \rangle$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product in $\mathcal{H}_k$. Additionally, the kernel must be symmetric (i.e., $k(G, G') = k(G', G)$) and positive semi-definite [1].

3.2.1 Graphlet kernel

The graphlet kernel (GK) compares graphs based on the distribution of small subgraphs, known as graphlets. In this evaluation, we restrict the graphlets to those having 3, 4, or 5 vertices [7].

The feature map $\phi : \mathcal{G} \to \mathbb{R}^N$ is defined such that the i-th component of the vector $v \in \mathbb{R}^N$ represents the count of occurrences of the i-th graphlet, denoted as $graphlet(i)$, in the input graph G. The resulting count vector is normalized to account for differences in graph sizes. We use the ORCA algorithm [10] to compute the graphlet distributions efficiently.

3.2.2 Symmetry kernel

To define the symmetry kernel, we first establish the type of symmetries we aim to capture. We consider graph symmetries as permutations of the vertex set. However, we focus on their permutation types instead of counting specific permutations.

A permutation type is the integer partition of the number of elements n, corresponding to the lengths of cycles in the permutation. For a given permutation π, we denote its type as $\tau(\pi)$. For example, if $\pi = (123)(45)(6)$, then $\tau(\pi) = (3, 2, 1)$, with the cycle lengths ordered in non-increasing order.

As an interesting variation, we can explore truncated permutation types by ignoring cycles of length one. This results in a more compact feature space. Using the previous example, the truncated type would be $\tau((123)(45)(6)) = (3, 2)$. We name the vertex symmetry variant the Vertex Symmetry Kernel (VSK) and the variant with ones the Vertex Symmetry Kernel with Ones (VSKO).

3.2.3 Permutation feature importance

The idea behind this method is to measure the importance of a feature by breaking its relationship to the target [12]. The method is straightforward. First, it trains a model on the input data and measures its error. Then, we select the feature whose importance we want to measure, and the method randomly shuffles its values. A new model is trained on the manipulated dataset. A feature is important if the new model's error increases.

4 Results and Interpretations

This section describes the datasets used in our experiments and presents the results in terms of model quality and the interpretations of the models.

4.1 Experimental Setting

This paper aims to show the difference between graphlets and symmetries. Therefore, we trained different machine learning models on the graphlet and symmetry counts and explained the importance of their features using PFI. Then we compared the similarities and differences between the resulting explanations.

The first pipeline we designed ourselves. We use SVM, Random Forest, and Adaboost machine learning algorithms. First, we split each dataset according to the standard 70:30 split. Then, we fine-tune each model's parameters using grid search cross-validation. We fine-tune the regularization parameter, the internal kernel type, and the kernel coefficient for SVM. For Random Forest, we fine-tune the number of decision trees and the function that

measures the split quality. For Adaboost, we fine-tuned the number of estimators and the learning rate.

Our experiments used well-known graph kernel benchmarking datasets: AIDS, Mutagenicity, NCI1, NCI109, PROTEINS, BZR, DHFR, MUTAG, PTC-FM, PTC-FR, PTC-MM, OHSU, REDDIT-BINARY, IMDB-BINARY, and github-stargazers [13].

To compare graphlets and symmetries more effectively, we used the same graphlet indexing as Hočevar et al. [10]. Figure 4 shows some examples of graphlets that frequently occurred in our experiments and the symmetries they translate into.

4.2 Results and Discussion

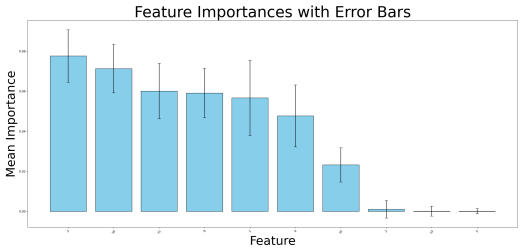
We measured the differences between the explanations yielded by PFI. Two types of differences occurred in our experiments. First, the importance of the symmetry-based features is permuted, such as in the example shown in Figure 2. In this example, the ranking of importance of the detected patterns is swapped. Graphlet 3 is ranked as the most important feature in this experiment. However, the symmetry type that is directly translated into (2)(2) is ranked as the third most important. On the other hand, the most important symmetry type that was detected was (2)(2)(1), which translates to a graphlet with index 9, which is only the fourth most important graphlet. We name this difference type 1.

The second difference we observed was that sometimes symmetry types appear as important features without having graphlets that translate into them. One such example is shown in Figure 3. In this example, graphlets 9 and 14 translate into symmetry (2)(2)(1), which is not present in the list of important symmetries. Also absent are the symmetries (2)(2), (2)(2)(1), and (5) into which graphlets indexed 3, 14, and 15 translate into. The graphlet that appears is graphlet 2, which translates into symmetry (3). Symmetry (2)(1) had the highest PFI, but no graphlet that translates into it. We name this difference type 2.

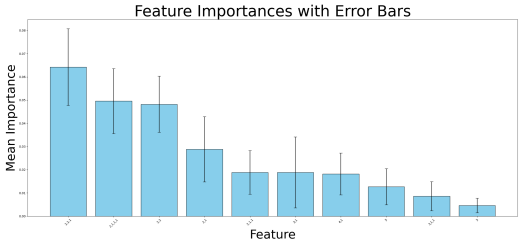
This shows us that not only can symmetry kernels detect different patterns that the graphlet kernel, but they can also rank features differently. Table 1 summarizes the presence or absence of differences between explanations based on graphlet and symmetry counts for each model trained on each dataset. Each cell indicates whether the explanations diverged (1,2, or B) or remained consistent (N) for that specific model–dataset combination. This analysis highlights that explanations vary across models and datasets, with some combinations (e.g., SVM on NCI1, PROTEINS, or BZR) consistently producing distinct results. In contrast, others (e.g., all models on github-stargazers) yielded no differences.

5 Conclusion

In this paper, we investigated the role of graph symmetries in interpretable machine learning. We trained and explained several symmetry-based kernels as well as the graphlet kernel. We then constructed a comparison of

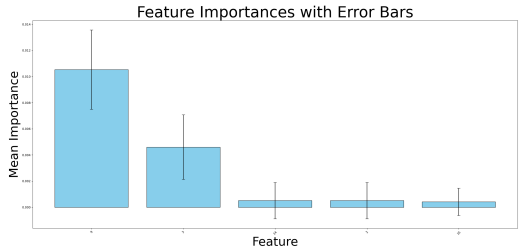


(a) PFI values for the graphlet kernel.

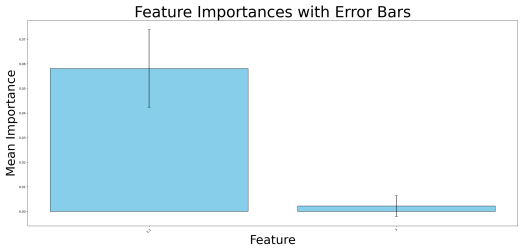


(b) PFI values for the Vertex symmetry kernel with ones.

Figure 2: The PFI histograms on the RandomForest model trained on the PTC-FM dataset.



(a) PFI values for the graphlet kernel.



(b) PFI values for the Vertex symmetry kernel with ones.

Figure 3: The PFI histograms on the Adaboost model trained on the COX2 dataset.

Table 1: The differences between explanations of each model trained on each dataset. If a cell is marked by **N**, differences between the explanations of symmetries and graphlets did not occur. **1** signifies that the difference of type **1** occurred, and **2** means that type 2 occurred. Cells marked with **B** show that both differences occurred simultaneously.

Dataset	SVM	AdaBoost	RandomForest
AIDS	B	B	N
Mutagenicity	2	N	N
NCI1	1	2	1
NCI109	1	B	N
PROTEINS	1	1	1
BZR	2	1	1
COX2	N	2	1
DHFR	2	1	N
MUTAG	1	B	1
PTC-FM	N	2	1
PTC-FR	N	B	1
PTC-MM	1	B	1
OHSU	2	2	N
REDDIT-BINARY	N	N	N
IMDB-BINARY	N	N	1
github-stargazers	N	N	N

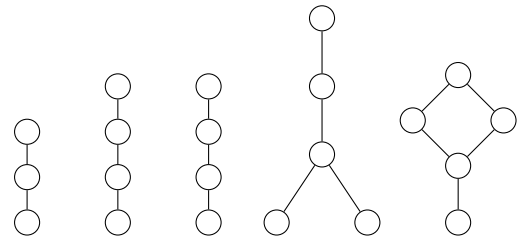


Figure 4: Some of the graphlets with a non-zero PFI value from our experiments. The indices of the graphlets align with those presented in [10]. Graphlet one corresponds to symmetry (2)(1), graphlet 9 to (2)(2)(1), graphlet 3 to (2)(2), graphlets 10 and 16 to (2)(1)(1)(1).

their explanations. Our motivation stemmed from the underexplored potential of graph automorphisms as interpretable features in graph classification tasks. By comparing symmetry-based kernels to the graphlet kernel across multiple datasets and models, we found that the explanations of the resulting models revealed meaningful differences.

Our experiments showed that symmetry-based kernels can rank specific patterns differently from the graphlet kernel. Symmetry-based kernels can also highlight structural patterns not captured by the graphlet kernel. This suggests that symmetries offer complementary information that can enrich our understanding of graph-structured data.

This paper serves as the groundwork for future exploration of symmetries as a tool for studying graphs. In future work, we plan to extend the graphlet kernel using the most significant symmetries: the number of graphlets with more than five vertices skyrockets. A natural extension of the graphlet kernel would be to include larger graphlets, which is impossible due to their large number. However, limiting ourselves to graphlets containing the most important symmetries for a given data domain would save us great work. In essence, we could create a domain-specific graphlet kernel extension. Another interesting research direction would be to incorporate graph symmetries into graph neural networks and how they can be used to manipulate the message passing algorithm. Before we dive deeper into these topics, we want to test these methods on more pattern-rich datasets than the ones we used in this paper.

References

[1] Kriege, Nils M., Fredrik D. Johansson, and Christopher Morris. "A survey on graph kernels." Applied Network Science 5.1 (2020): 1-42.

[2] Čibej, Uroš and Mihelič, Jurij. "Graph automorphisms for compression," Open Computer Science, vol. 11, no. 1, (2021), pp. 51-59. https://doi.org/10.1515/comp-2020-0186

[3] Kipf, Thomas N., and Max Welling. 'Semi-Supervised Classification with Graph Convolutional Networks.' Proceedings of the 5th International Conference on Learning Representations, OpenReview.net, 2017.

[4] Huang, Qiang, et al. "Graphlime: Local interpretable model explanations for graph neural networks." IEEE Transactions on Knowledge and Data Engineering 35.7 (2022): 6968-6972.

[5] Lundberg, Scott M., and Su-In Lee. "A unified approach to interpreting model predictions." Advances in neural information processing systems 30 (2017).

[6] Gärtner, Thomas, Peter Flach, and Stefan Wrobel. "On graph kernels: Hardness results and efficient alternatives." Learning Theory and Kernel Machines: 16th Annual Conference on Learning Theory and 7th Kernel Workshop, COLT/Kernel 2003, Washington, DC, USA, August 24-27, 2003. Proceedings. Springer Berlin Heidelberg, 2003.

[7] Shervashidze, Nino, et al. "Efficient graphlet kernels for large graph comparison." Artificial intelligence and statistics. PMLR, 2009.

[8] Shervashidze, Nino, et al. "Weisfeiler-lehman graph kernels." Journal of Machine Learning Research 12.9 (2011).

[9] Lundberg, Scott M., and Su-In Lee. "A unified approach to interpreting model predictions." Advances in neural information processing systems 30 (2017).

[10] Hočevar, Tomaž, and Janez Demšar. "A combinatorial approach to graphlet counting." Bioinformatics 30.4 (2014): 559-565.

[11] Molnar, Christoph. Interpretable machine learning. Lulu. com, 2020.

[12] Fisher, Aaron, Cynthia Rudin, and Francesca Dominici. "All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously." Journal of Machine Learning Research 20.177 (2019): 1-81.

[13] Morris, Christopher, et al. "Tudataset: A collection of benchmark datasets for learning with graphs." arXiv preprint arXiv:2007.08663 (2020).