
A Submodular Framework for Graph Comparison

Pinar Yanardag

Department of Computer Science
Purdue University
West Lafayette, IN, 47906, USA
ypinar@purdue.edu

S.V.N. Vishwanathan

Department of Computer Science
University of California
Santa Cruz, CA, 95064, USA
vishy@ucsc.edu

Abstract

In this paper, we present a submodular framework for graph comparison. Our framework is inspired by latest state-of-the-art submodular document summarization models and extends the graphlet kernel to encourage *diversity* while avoiding redundancy. Our experiments on several benchmark datasets show that our framework outperforms the graphlet kernel in terms of classification accuracy by using 50% less samples.

1 Introduction

In many real-world domains such as bioinformatics, chemoinformatics [16] and social networks [18], we are often interested in *comparing graphs*. For instance, one might aim to classify chemical compounds by predicting whether a compound is active in an anti-cancer screen or not [17] (see Figure 1). A popular approach to solve this problem is to represent each chemical compound as a graph, and to use a graph kernel to compute a kernel matrix \mathcal{K} where each entry \mathcal{K}_{ij} represents how similar graph i to graph j . The computed kernel matrix is then simply plugged into a kernelized learning algorithm such as Support Vector Machines (SVM) [13] in order to classify graphs.

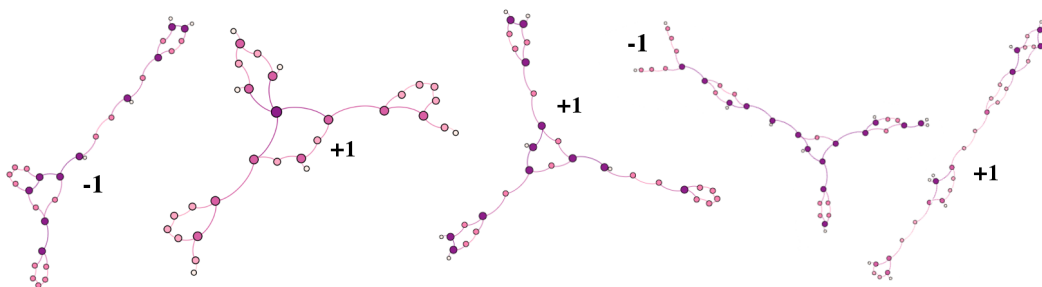


Figure 1: Sample graphs from NCI1 dataset [17] where each graph represents a chemical compound and labeled as +1 (active in an anti-cancer screen) or -1 (non-active in an anti-cancer screen). Graphs are created using Gephi [1].

A commonly used paradigm for representing graphs is to use a vector that contains normalized frequencies of occurrence of certain motifs or sub-graphs. The graphlet kernel of Shervashidze et al. [14] uses induced sub-graphs of k nodes (christened as *graphlets* by Przulj [11]) as motifs in the vector representation, and computes the kernel via a dot product between these vectors. However, existing sampling methods for graphlets suffer from a few drawbacks. Next, we first introduce a brief background on the graphlet kernel, and then motivate our solution to propose a novel strategy to address these drawbacks.

2 Background and Motivation

A *graph* is a pair $G = (V, E)$ where $V = \{v_1, v_2, \dots, v_{|V|}\}$ is an ordered set of *vertices* or *nodes* and $E \subseteq V \times V$ is a set of *edges*. Given $G = (V, E)$ and $H = (V_H, E_H)$, H is a *sub-graph* of G iff there is an injective mapping $\alpha : V_H \rightarrow V$ such that $(v, w) \in E_H$ iff $(\alpha(v), \alpha(w)) \in E$. Two graphs $G = (V, E)$ and $G' = (V', E')$ are *isomorphic* if there exists a bijective mapping $g : V \rightarrow V'$ such that $(v_i, v_j) \in E$ iff $(g(v_i), g(v_j)) \in E'$. *Graphlets* are small, non-isomorphic sub-graphs of a large network, introduced by Przulj [11] to design a new measure of local structural similarity between biological networks. Graphlets up to size five are shown in Figure 2.

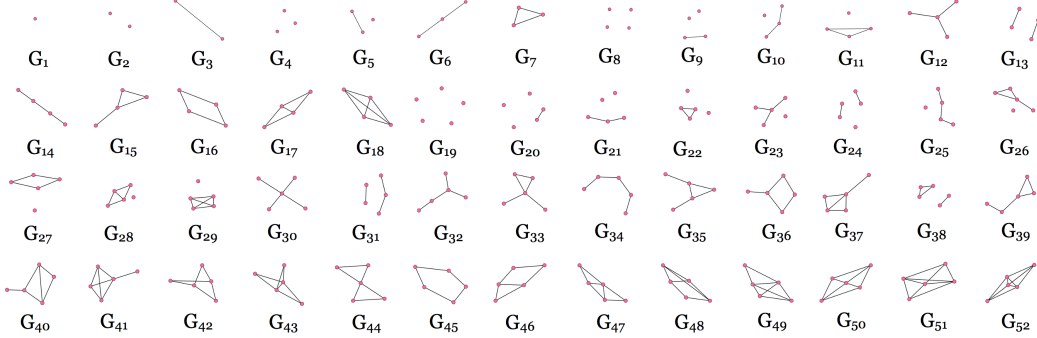


Figure 2: Graphlets of size $k \leq 5$. Plots are generated with networkX library [6].

2.1 Graphlet Sampling

Let $\mathbf{G}_k = \{G_1, G_2, \dots, G_{n_k}\}$ be the set of size- k graphlets where n_k denotes the number of unique graphlets of size k . Given a graph \mathcal{G} , we define $f_{\mathcal{G}}$ as a normalized vector of length n_k whose i -th component corresponds to the frequency of occurrence of G_i in \mathcal{G} :

$$f_{\mathcal{G}} = \left(\frac{c_1}{\sum_j c_j}, \dots, \frac{c_{n_k}}{\sum_j c_j} \right)^T. \quad (1)$$

where c_i denotes number of times G_i occurs as a sub-graph of G . Given two graphs \mathcal{G} and \mathcal{G}' , the graphlet kernel [14] k_g is defined as:

$$k_g(\mathcal{G}, \mathcal{G}') := f_{\mathcal{G}}^T f_{\mathcal{G}'}, \quad (2)$$

which is simply the dot product between the normalized graphlet-frequency vectors.

2.2 Graphlet Sampling

Given two graphs \mathcal{G} and \mathcal{G}' , determining whether \mathcal{G} contains a subgraph isomorphic to \mathcal{G}' is NP-complete¹. However, the subgraph isomorphism can be tested in polynomial time $O(n_{\mathcal{G}'}! \cdot n_{\mathcal{G}'}^2 \cdot \binom{n_{\mathcal{G}}}{n_{\mathcal{G}'}})$ if graph \mathcal{G} on nodes $n_{\mathcal{G}}$ is input and graph \mathcal{G}' on $n_{\mathcal{G}'}$ is fixed [12]. This requires an exhaustive search by iterating over all subsets of $n_{\mathcal{G}'}$ nodes of \mathcal{G} and prohibitively expensive even for moderate sized graphs. Thus, [14] proposed a sampling scheme that randomly searches graphlets in a given graph. However, this random sampling scheme does not take several important observations into account as follows.

- **Redundancy:** The graphlet frequency distribution exhibits a power-law behavior, that is, the frequency of certain graphlets grows as a power of others. This becomes a significant problem when the frequency of informative graphlets are overwhelmed by graphlets that do not carry any discriminating power across graphs. Figure 3 illustrates such an example where graphlet G_{20} occurs significantly higher than more informative graphlets such as G_{32} and G_{34} . An ideal framework should take this observation into account and avoid selecting redundant graphlets.

¹It can be shown that this problem includes Hamilton path/cycle and maximum clique as special cases [5].



Figure 3: A sample graph from MUTAG [3] dataset that illustrates the decomposed graphlets where size of each graphlet is positively correlated with its frequency in the graph.

- **Diversity:** It can be observed that graphlets of a given size k are related to each other [18]. For instance, graphlets G_{36} , G_{37} and G_{40} only differ by one node and one edge. Therefore, a sampling scheme that does not take the inherent similarity between graphlets does not respect the diversity among the samples. An ideal framework should encourage diverse graphlets when performing sampling.

Our Solution: Our algorithm takes advantage of the inherent similarity between graphlets of size $\leq k$ and $k + 1$. The key observation of our proposal is that, one can use this relationship to represent a graphlet of size $k + 1$ as a *probability distribution* over size $\leq k$ graphlets. Let G_i represent a graphlet of size $k + 1$, G_j represent a graphlet of size k and n_{ij} denote the number of times G_j occurs as a sub-graph of G_i . Computation of n_{ij} is done by deleting a node of G_i and counting how many times graphlet G_j is produced as a result [19]. Repeating the same process for all graphlets of size $1 \leq l \leq k$ and normalizing the frequencies, we obtain a distribution for graphlet G_i by means of smaller-sized graphlets (see Figure 4 for an example decomposition).

An alternate way to interpret this probability distribution is as follows: each graphlet G_i of size $k + 1$ *covers* graphlets of size $\leq k$ with a certain amount. Thus, whenever we sample a graphlet G_i , we also *cover* some portion of other graphlets. For instance, whenever we pick graphlet G_{41} , we cover the following graphlets with associated probabilities: $G_{41} = \{G_2 : 0.21, G_3 : 0.49, G_5 : 0.07, G_6 : 0.07, G_7 : 0.09, G_{11} : 0.01, G_{15} : 0.03, G_{18} : 0.01\}$. Therefore, one can view this as a *maximum coverage* problem where our main objective is to select m subsets from a collection $S = \{S_1, S_2, \dots, S_n\}$ of n subsets such that the union of the selected sets $\bigcup_{i=1}^m S_i$ has the maximum coverage. In other words, we would like to maximize the following objective function:

$$A^* = \operatorname{argmax}_{A \subseteq S: |A| \leq m} f(A)$$

This problem is well-known to be NP-hard [4]. However, it can be formulated in terms of *submodular* functions where a greedy algorithm is guaranteed to approximate the optimum solution within a factor of $(1 - 1/e) \approx 0.63$ [10]. Submodular functions exhibits a natural *diminishing returns* property which is suitable for our task, that is, whenever we *cover* a graphlet, we want its gain to *diminish* over time. Next, we introduce some background on submodularity.

3 Methodology

In this section, we first introduce a brief background on submodularity, and then we propose our framework.

3.1 Submodularity

Submodularity is a discrete optimization method that shares similar characteristics with concavity, while resembling convexity. Submodularity appears in a wide range of application areas including social networks, viral marketing [8] and document summarization [9]. Submodular functions exhibit













	 G ₂	 G ₃	 G ₅	 G ₆	 G ₇	 G ₁₅	 G ₁₀	 G ₁₁	 G ₁₄	 G ₁₈
 G ₃₅	0.35	0.35	0.09	0.09	0.02	0.02	0.02	0.0	0.01	0.0
 G ₄₁	0.21	0.49	0.07	0.07	0.09	0.03	0.0	0.01	0.0	0.01

Figure 4: An example for graphlets G_{35} and G_{41} that shows the decomposed graphlets of size $\leq k$, associated with normalized frequencies. As can be seen from the table, both graphlets cover graphlets $G_2, G_3, G_5, G_6, G_7, G_{15}$ with different weights.

Algorithm 1 Greedy submodular function maximization with budget constraint

Require: V, k

Ensure: Selected set of neighborhoods

- 1: Initialize $S \leftarrow \emptyset$
 - 2: **while** $|S| \leq k$ **do**
 - 3: $v \leftarrow \operatorname{argmax}_{z \in V \setminus S} (f(S \cup \{z\}) - f(S))$
 - 4: $S \leftarrow S \cup \{v\}$
 - 5: **end while**
 - 6: **return** S
-

a natural diminishing returns property, that is, given two sets S and T , where $S \subseteq T \subseteq V \setminus v$, the incremental *value* of an item v decreases as the context in which v is considered grows from S to T .

More formally, submodularity is a property of set functions, *i.e.*, the class of functions $f : 2^V \rightarrow \mathbb{R}$ that maps subsets $S \subseteq V$ to a value $f(S)$ where V is a finite ground set. The function f maps any given subset to a real number. The function f is called normalized if $f(\emptyset) = 0$, and it is monotone if $f(S) \leq f(T)$, whenever $S \subseteq T$. The function f is called submodular if the following equation holds for any $S, T \subseteq V$:

$$f(S \cup T) + f(S \cap T) \leq f(S) + f(T) \quad (3)$$

Another equivalent definition of the submodularity property is given as follows:

$$f(S \cup \{v\}) - f(S) \leq f(R \cup \{v\}) - f(R) \quad (4)$$

where f is submodular if $R \subseteq S \subseteq V$ and $v \in V \setminus S$. This form of submodularity directly satisfies the property of diminishing returns; the *value* of v never increases when the context gets larger.

It has been shown that submodular function minimization can be solved in polynomial time [7], while submodular function maximization is an NP-complete optimization problem and intractable. However, it has been shown by [10] that the maximization of a monotone submodular function under a cardinality constraint can be solved near-optimally using a greedy algorithm. In submodular function maximization, we are interested in solving the following optimization problem:

$$A^* = \operatorname{argmax}_{A \subseteq V: |A| \leq k} f(A)$$

subject to a cardinality constraint k . If a function f is submodular, takes only non-negative values, and is monotone, then even though the maximization is still NP complete, we can use a greedy algorithm (see Algorithm 1) to approximate the optimum solution within a factor of $(1 - 1/e) \approx 0.63$ [10].

3.2 Framework

We formulate our task as a submodular optimization problem and adapt our framework from document summarization task of [9]. In order to apply our framework, we changed the random sampling approach slightly as follows: whenever we sample a graphlet, we also sample its immediate neighbors. Thus, our sampling procedure is still random, but we also have a chance to capture the graphlets within similar neighborhoods. Let S represent a set of neighborhoods of size k graphlets, $P_{<k}$ represent the set of unique graphlet types of size $< k$, p represent an arbitrary graphlet in $P_{<k}$, i represent an arbitrary neighborhood and j represent a graphlet in that neighborhood. We then define our submodular objective function as follows:

$$F(S) = \sum_{p \in P_{<k}} \left(\sum_{i \in S} \sum_{j \in i} r_{pj} \right)^\alpha \quad (5)$$

where r_{pj} is the weight of graphlet p in j and α is curvature parameter that determines the rate that reward diminishes over time. Interpretation of this framework is as follows: we measure the diversity of the selected set S in terms of graphlets of size $< k$. For each graphlet p in the set of unique graphlets of size $< k$, we quantify the amount that is already covered by the selected neighborhoods in S . Thus, for each selected neighborhood i in the set S , we sum how much we already covered the $< k$ -sized graphlet p .

3.3 Proof

In order to show that $F(S)$ is submodular, we will use the *composition property* of submodular functions with concave functions:

Theorem 1 Given functions $F : 2^V \rightarrow R$ and $f : R \rightarrow R$, the composition $F' = f \circ F : 2^V \rightarrow R$ is non-decreasing submodular if f is non-decreasing concave and F is non-decreasing submodular.

Proof Equation 5 is submodular. $(x)^\alpha$ is a non-decreasing concave function. Inside of $(x)^\alpha$, we have a modular function with non-negative weights, thus monotone. Applying $(x)^\alpha$ to such a monotone submodular function yields a submodular function, and summing submodular functions retains submodularity property. Therefore, $F(S)$ is submodular. ■

4 Experiments

In this section, we first introduce the datasets used for our experiments. Then, we discuss our experimental setup and introduce our results.

4.1 Datasets

We applied our framework to benchmark graph kernel datasets, namely, MUTAG, PTC and ENZYMES and NCI1. MUTAG is a dataset of 188 mutagenic aromatic and heteroaromatic nitro compounds [3] with 7 discrete labels. PTC [15] is a dataset of 344 chemical compounds that reports the carcinogenicity for male and female rats and it has 19 discrete labels. ENZYMES is a balanced dataset of 600 protein tertiary structures obtained from [2] and has 3 discrete labels. NCI109 [17] dataset has 4127 nodes, made publicly available² by the National Cancer Institute (NCI), is a balanced data sets of chemical compounds screened for ability to suppress or inhibit the growth of a panel of human tumor cell lines, having 38 discrete labels. See Table 1 for detailed statistics of the datasets.

²<http://pubchem.ncbi.nlm.nih.gov>

Table 1: Properties of datasets used in graph kernel experiments.

Dataset	# of graphs	# of classes	# of nodes (avg)	# of labels
MUTAG	188	2 (125 vs 63)	17.9	7
PTC	344	2 (152 vs 192)	25.5	19
ENZYMES	600	6 (100 each)	32.6	3
NCI109	4127	2 (2079 vs 2048)	29.6	38

5 Experimental Setup

We compare our framework against graphlet kernel. Both kernels are coded in Python and normalized to have a unit length in the feature space. Moreover, we use 10-fold cross validation with a binary C -Support Vector Machine (SVM) where the C value for each fold is independently tuned using training data from that fold. In order to exclude random effects of the fold assignments, this experiment is repeated 10 times and average prediction accuracy of 10 experiments with their standard deviations are reported.

In order to achieve a fair comparison, we first sampled 100 neighborhoods of graphlets from a given dataset. Then, we feed exactly the same set of graphlets into our framework and submodularly selected 50 of them. Thus, our framework uses 50% less information than the graphlet kernel in the following experiments.

6 Results

We compare graphlet kernel with our method (see Table 2). As can be seen from the results, our framework is able to outperform base kernel with statistically significant improvements (shown in bold) while achieving a smaller standard error on most of the datasets.

Table 2: Comparison of classification accuracy for the graphlet kernel with our method where **STD** standard deviation, and **SE** represents standard error.

DATA SET	GRAPHLET KERNEL	SUBMODULAR GRAPHLET KERNEL
MUTAG	77.11 (STD: 1.54, SE: 0.48)	80.22 (STD: 1.08, SE: 0.34)
PTC	55.82 (STD: 1.10, SE: 0.35)	57.14 (STD: 1.35, SE: 0.42)
ENZYMES	23.35 (STD: 1.30, SE: 0.41)	25.10 (STD: 0.92, SE: 0.29)
NCI109	62.15 (STD: 0.28, SE: 0.09)	62.28 (STD: 0.22, SE: 0.07)

7 Conclusion

In this paper, we propose a novel framework to incorporate diversity when performing graph comparison. Even though we restricted ourselves to graph kernel literature in this paper, our framework introduces a new perspective to graph sampling and summarization. For instance, our framework can be easily adoptable to summarize diverse aspects of a given graph for exploration or visualization purposes.

As a future work, we consider modifying our framework to a setting where we dynamically explore the graph in the direction that maximizes our objective function.

8 Acknowledgments

This work is supported by the National Science Foundation under grant No. #1219015.

References

- [1] M. Bastian, S. Heymann, and M. Jacomy. Gephi: an open source software for exploring and manipulating networks. In *ICWSM*, pages 361–362, 2009.
- [2] K. M. Borgwardt, C. S. Ong, S. Schönauer, S. V. N. Vishwanathan, A. J. Smola, and H.-P. Kriegel. Protein function prediction via graph kernels. In *Proceedings of Intelligent Systems in Molecular Biology (ISMB)*, Detroit, USA, 2005. <http://www.stat.purdue.edu/~vishy/papers/BorOngSchVisetal05.pdf>.
- [3] A. K. Debnath, R. L. Lopez de Compadre, G. Debnath, A. J. Shusterman, and C. Hansch. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *J Med Chem*, 34:786–797, 1991.
- [4] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM (JACM)*, 45(4): 634–652, 1998.
- [5] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Series of Books in Mathematical Sciences. W. H. Freeman, 1979.
- [6] A. Hagberg, P. Swart, and D. S. Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Laboratory (LANL), 2008.
- [7] S. Iwata, L. Fleischer, and S. Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *JACM*, 2001.
- [8] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *SIGKDD*, pages 137–146. ACM, 2003.
- [9] H. Lin and J. Bilmes. A class of submodular functions for document summarization. In *HLT*, 2011.
- [10] G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming*, 14:265–294, 1978.
- [11] N. Przulj. Biological network comparison using graphlet degree distribution. In *2006 European Conference on Computational Biology (ECCB)*, September 2006.
- [12] N. Przulj. Biological network comparison using graphlet degree distribution. *Bioinformatics*, 23(2):e177–e183, Jan 2007.
- [13] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.
- [14] N. Shervashidze, S. V. N. Vishwanathan, T. Petri, K. Mehlhorn, and K. Borgwardt. Efficient graphlet kernels for large graph comparison. In M. Welling and D. van Dyk, editors, *Proc. Intl. Conference on Artificial Intelligence and Statistics*. Society for Artificial Intelligence and Statistics, 2009.
- [15] H. Toivonen, A. Srinivasan, R. D. King, S. Kramer, and C. Helma. Statistical evaluation of the predictive toxicology challenge 2000-2001. *Bioinformatics*, 19(10):1183–1193, July 2003.
- [16] S. V. N. Vishwanathan, N. N. Schraudolph, I. R. Kondor, and K. M. Borgwardt. Graph kernels. *Journal of Machine Learning Research*, 2010. URL <http://www.stat.purdue.edu/~vishy/papers/VisSchKonBor10.pdf>. In press.
- [17] N. Wale, I. A. Watson, and G. Karypis. Comparison of descriptor spaces for chemical compound retrieval and classification. *Knowledge and Information Systems*, 14(3):347–375, 2008.
- [18] P. Yanardag and S. Vishwanathan. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1365–1374. ACM, 2015.
- [19] P. Yanardag and S. Vishwanathan. A structural smoothing framework for robust graph comparison. *NIPS*, 2015.