FAST GRAPH KERNEL WITH OPTICAL RANDOM FEATURES

Hashem Ghanem *Nicolas Keriven* *Nicolas Tremblay*

CNRS, GIPSA-lab, FR-38402 Saint Martin d'Heres Cedex, France

ABSTRACT

The graphlet kernel is a classical method in graph classification. It however suffers from a high computation cost due to the isomorphism test it includes. As a generic proxy, and in general at the cost of losing some information, this test can be efficiently replaced by a user-defined mapping that computes various graph characteristics. In this paper, we propose to leverage *kernel random features* within the graphlet framework, and establish a theoretical link with a mean kernel metric. If this method can still be prohibitively costly for usual random features, we then incorporate *optical* random features that can be computed in *constant time*. Experiments show that the resulting algorithm is orders of magnitude faster that the graphlet kernel for the same, or better, accuracy.

Index Terms— Optical random features, Graph kernels

1. INTRODUCTION

In mathematics and data science, graphs are used to model a set of objects (the *nodes*) and their interactions (the *edges*). Given a set of pre-labeled graphs ($\mathcal{X} = \{\mathcal{G}_1, \dots, \mathcal{G}_n\}, \mathcal{Y} = \{y_1, \dots, y_n\}$), where each graph $\mathcal{G}_i$ belongs to the class with label y_i, graph classification consists in designing an algorithm that outputs the class label of a new graph. For instance, proteins can be modeled as graphs: amino acids are nodes and the chemical links between them are edges. They can be classified to enzymes and non-enzymes [1]. In social networks analysis, post threads can be modeled with graphs whose nodes are users and edges are replies to others' comment [2]. One task is then to discriminate between discussion-based and question/answer-based threads [3]. In addition to the graph structure, nodes and edges may have extra features. While it has been shown that node features are important to obtain high classification performance [4], here we focus on the case where one has only access to the graph structure.

Structure-based graph classification has been tackled with many algorithms. Frequent subgraphs based algorithms [5] analyze the graph dataset $\mathcal{X}$ to catch the frequent and discriminative subgraphs and use them as features. Kernel-based algorithms [6] can be used by defining similarity functions (kernels) between graphs. An early and popular example is the *graphlet kernel*, which computes frequencies of subgraphs. It is however known to be quite costly to compute [7], in particular due to the presence of graph isomorphism tests. While possible in particular cases [7], accelerating the graphlet kernel for arbitrary datasets remains open. Finally, graph neural networks (GNNs) [8, 9] have recently become very popular in graph machine learning. They are however known to exhibit limited performance when node features are unavailable [10].

In kernel methods, random features are an efficient approximation method [11, 12]. Recently, it has been shown [13] that *optical computing* can be leveraged to compute such random features in *constant time* in *any dimension* – within the limitations of the current hardware, here referred to as Optical Processing Units (OPUs). The main goal of this paper is to provide a proof-of-concept answer to the following question: can OPU computations be used to reduce the computational complexity of a combinatorial problem like the graphlet kernel? Drawing on a connection with mean kernels and Maximum Mean Discrepancy (MMD) [14], we show, empirically and theoretically, that a fast and efficient graph classifier can indeed be obtained with OPU computations.

2. BACKGROUND

First, we present the concepts necessary to define the graphlet kernel. We represent a graph of size v by the adjacency matrix $\mathbf{A} \in \{0,1\}^{v \times v}$, such that $a_{i,j} = 1$ if there is an edge between nodes $\{i, j\}$ and 0 otherwise. Two graphs are said to be isomorphic ($\mathcal{G} \cong \mathcal{G}'$) if we can permute the nodes' labels of one such that their adjacency matrices are equal [15].

2.1. Isomorphic graphlets

In this paper, we will, depending on the context, manipulate two different notions of k-graphlets (that is, small graphs of size k): with or without discriminating isomorphic graphlets. We denote by $\bar{\mathfrak{H}} = \{\bar{\mathcal{H}}_1, ..., \bar{\mathcal{H}}_{\bar{N}_k}\}$ with $\bar{N}_k = 2^{\frac{k(k-1)}{2}}$ the set of all size-k graphs, where isomorphic graphs are counted multiple times, and $\mathfrak{H} = \{\mathcal{H}_1, ..., \mathcal{H}_{N_k}\} \subset \bar{\mathfrak{H}}$ the set of all non-isomorphic graphs of size k. Its size N_k has a (quite verbose) closed-from expression [16], but is still exponential in k. In the course of this paper, we shall manipulate mappings $\varphi(\mathcal{H})$ and probability distributions (histograms) over graphlets. By default the underlying space will be $\bar{\mathfrak{H}}$, however when the mapping φ is *permutation-invariant*, then the

underlying space can be thought of as $\mathfrak{H}$. Also note that, assuming each isomorphic copies has equal probability, a probability distribution over $\bar{\mathfrak{H}}$ can be *folded* into one over $\mathfrak{H}$, and both distributions *contain the same amount of information*.

2.2. The graphlet kernel

The traditional graphlet kernel is defined by computing histograms of subgraphs over non-isomorphic graphlets $\mathfrak{H}$. We define the matching function $\varphi_k^{match}(\mathcal{F}) = \left[1_{(\mathcal{F} \cong \mathcal{H}_i)}\right]_{i=1}^{N_k} \in \{0,1\}^{N_k}$, where $\mathcal{F}$ is a graph of size k. In words, $\varphi_k^{match}(\mathcal{F})$ is a one-hot vector of dimension N_k identifying $\mathcal{F}$ up to isomorphism. Note that the cost of evaluating φ_k^{match} once is $O\left(N_k C_k^{\cong}\right)$, where $C_k^{\cong}$ is the cost of the isomorphism test between two graphs of size k, for which no polynomial algorithm is known [17]. Given a graph $\mathcal{G}$ of size v, let $\mathfrak{F}_{\mathcal{G}} = \{\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_{\binom{v}{k}}\}$ be the collection of subgraphs induced by all size-k subsets of nodes. The following representation vector is called the k-spectrum of $\mathcal{G}$:

$$\mathbf{f}_{\mathcal{G}} = \binom{v}{k}^{-1} \sum_{\mathcal{F} \in \mathfrak{F}_{\mathcal{G}}} \varphi_k^{match}(\mathcal{F}) \in \mathbb{R}^{N_k} \tag{1}$$

For two graphs $\mathcal{G}, \mathcal{G}'$, the graphlet kernel [7] is then defined as $\mathbf{f}_{\mathcal{G}}^T \mathbf{f}_{\mathcal{G}'}$. For a graph of size v, the computation cost of $\mathbf{f}_{\mathcal{G}}$ is $C_{gk} = \mathcal{O}\left(\binom{v}{k} N_k C_k^{\cong}\right)$. This cost is usually prohibitively expensive, since each three terms are exponential in k.

Subgraph sampling is generally used as a first step to accelerate (and sometimes modify) the graphlet kernel [7]. Given a graph $\mathcal{G}$, we denote by $S_k(\mathcal{G})$ a sampling process that yields a random subgraph of $\mathcal{G}$, seen as a probability distribution over $\bar{\mathfrak{H}}$. Then, sampling s subgraphs $\hat{\mathfrak{F}}_{\mathcal{G}} = \{F_1, ..., F_s\}$ $i.i.d.$ from $S_k(\mathcal{G})$, we define the estimator:

$$\hat{\mathbf{f}}_{\mathcal{G},S_k} = s^{-1} \sum_{F \in \hat{\mathfrak{F}}_{\mathcal{G}}} \varphi_k^{match}(F). \tag{2}$$

and its expectation $\mathbf{f}_{\mathcal{G},S} = \mathbb{E}_{F \sim S_k(\mathcal{G})}\ \varphi_k^{match}(F)$, which is nothing more than the *folding* of the distribution $S_k(\mathcal{G})$ over non-isomorphic graphlets $\mathfrak{H}$. For any sampler, we refer to these expectations as graphlet kernels. In fact, in all generality, any choice of sampling procedure S_k yields a different definition of graphlet kernel. For instance, if one considers uniform sampling (S^{unif}: independently samples k nodes of $\mathcal{G}$ without replacement), then one obtains the original graphlet kernel of Eq. (1): $\mathbf{f}_{\mathcal{G},S^{\text{unif}}} = \mathbf{f}_{\mathcal{G}}$. Other choices of sampling procedures are possible [18]. In this paper, we will also use the random walk (RW) sampler, which, unlike uniform sampling, tends to sample connected subgraphs, which may be more informative about the graph structure.

The computation cost per graph of the approximate graphlet kernel of Eq. (2) is $C_{gk+gs} = \mathcal{O}\left(sC_S N_k C_k^{\cong}\right)$, where C_S is the cost of sampling one subgraph. For a fixed error in estimating $\mathbf{f}_{\mathcal{G},S}$, the required number of samples s generally needs to be proportional to N_k [7], which unfortunately still yields an unaffordable algorithm.

Algorithm 1: GSA-φ generic algorithm

Input: labeled graph dataset $\mathcal{X} = (\mathcal{G}_i, y_i)_{i=1,\ldots,n}$
Tools *Graphlet sampler S_k, a function φ, a linear classifier (ex. SVM)*
Hyperparameters *k: graphlet size, s: number of graphlet samples*, m: *number of random features*
Output: Trained model to classify graphs
Algorithm
Random initialization of the SVM weights
for $\mathcal{G}_i$ *in* $\mathcal{X}$ **do**
  $\mathbf{z}_i = \mathbf{0}$ (null vector of size m)
  for $j = 1 : s$ **do**
    $F_{i,j} \leftarrow S_k(\mathcal{G}_i)$
    $\mathbf{z}_i \leftarrow \mathbf{z}_i + \frac{1}{s}\varphi(F_{i,j})$
$\mathcal{D}_\varphi \leftarrow (\mathbf{z}_i, y_i)_{i=1,\ldots,n}$
Train the classifier on this vector-valued dataset $\mathcal{D}_\varphi$

3. GRAPHLET KERNEL WITH OPTICAL MAPS

3.1. Proposed method

In this paper, we focus on the main remaining bottleneck of the graphlet kernel, that is, the function φ_k^{match}. We define a framework where it is replaced with a user-defined map $\varphi : \bar{\mathfrak{H}} \to \mathbb{R}^m$, which leads to the final representation:

$$\hat{\mathbf{f}}_{\mathcal{G},S_k,\varphi} = s^{-1} \sum_{F \in \hat{\mathfrak{F}}_{\mathcal{G}}} \varphi(F). \tag{3}$$

and similarly its expectation $\mathbf{f}_{\mathcal{G},S_k,\varphi}$. The resulting methodology is referred to as *Graphlet Sampling and Averaging* (GSA-φ), and summarized in Alg. **??**. The cost of computing (3) is $C_{GSA-\varphi} = \mathcal{O}\left(sC_S C_\varphi\right)$, where C_φ is the cost of applying φ. Similar methods have been studied with φ as simple graphlets statistics [19], which unavoidably incurs information loss. We see next that choosing φ as *kernel random maps* preserves information for a sufficient number of random features. Some of these maps will *not* be permutation-invariant at the graphlet level, however, in the infinite sample limit, it is easy to see that the representation vector $\mathbf{f}_{\mathcal{G},S_k,\varphi}$ is indeed permutation-invariant at the graph level.

3.2. Kernel random features with $GSA - \varphi$

In the graphlet kernel, the underlying metric used to compare graphs is the Euclidean distance between graphlet histograms. When φ_k^{match} is replaced by another φ, one compares certain *embeddings* of distributions, which is reminiscent of kernel mean embeddings [14]. We show below that this corresponds to choosing φ as kernel random features.

For two objects $\mathbf{x}, \mathbf{x}'$, a kernel κ associated to a random features (RF) decomposition is a positive definite function that can be decomposed as follows [11]:

$$\kappa(\mathbf{x}, \mathbf{x}') = \mathbb{E}_{\mathbf{w} \sim p}[\xi_{\mathbf{w}}(\mathbf{x})^* \xi_{\mathbf{w}}(\mathbf{x}')] \tag{4}$$

where ξ is a real (or complex) function parameterized by $\mathbf{w} \in \Omega$, and p a probability distribution on Ω. A classical example is the Fourier decomposition of translation-invariant kernels [11]. The RF methodology then defines maps:

$$\varphi(\mathbf{x}) = m^{-1/2}(\xi_{\mathbf{w}_j}(\mathbf{x}))_{j=1}^{m} \in \mathbb{C}^m \tag{5}$$

where m is the number of features and the parameters $\mathbf{w}_j$ are drawn iid from p. Then, $\kappa(\mathbf{x}, \mathbf{x}') \approx \varphi(\mathbf{x})^H \varphi(\mathbf{x}') = m^{-1} \sum_j \xi_{\mathbf{w}_j}(\mathbf{x})^* \xi_{\mathbf{w}_j}(\mathbf{x}')$.

Assume that we have a base kernel $\kappa(\mathcal{F}, \mathcal{F}')$ between *graphlets*, with a RF decomposition $(\xi_{\mathbf{w}}, p)$, and define φ as in (5). Then, one can show [20, 21] that the Euclidean distance between the embeddings (3) approximates the following *Maximum Mean Discrepancy* (MMD) [14, 22] between distributions on $\bar{\mathfrak{H}}$:

$$\begin{aligned}&\mathrm{MMD}^2(S_k(\mathcal{G}), S_k(\mathcal{G}')) \\ &\qquad = \mathbb{E}_{\mathbf{w}}\Big(\left|\mathbb{E}_{S_k(\mathcal{G})}\xi_{\mathbf{w}}(F) - \mathbb{E}_{S_k(\mathcal{G}')}\xi_{\mathbf{w}}(F')\right|^2 \Big)\end{aligned} \tag{6}$$

The main property of the MMD is that, for so-called *characteristic kernels*, it is a true metric on distributions, *i.e.* $\mathrm{MMD}(S_k(\mathcal{G}), S_k(\mathcal{G}')) = 0 \Leftrightarrow S_k(\mathcal{G}) = S_k(\mathcal{G}')$. Most usual kernels, like the Gaussian kernel, are characteristic [14, 22].

Theorem 1. *Let $\mathcal{G}$ and $\mathcal{G}'$ be two graphs. Assume a random feature map as in* (5)*. Assume that $|\xi_{\mathbf{w}}(F)| \leq 1$ for any $\mathbf{w}, F$. We have for all $\delta > 0$ and with probability at least $1 - \delta$:*

$$\begin{aligned}&\left| \|\hat{\mathbf{f}}_{\mathcal{G},S_k,\varphi} - \hat{\mathbf{f}}_{\mathcal{G}',S_k,\varphi}\|_2^2 - \mathrm{MMD}^2(S_k(\mathcal{G}), S_k(\mathcal{G}')) \right| \\ &\qquad \leq 4m^{-\frac{1}{2}}\sqrt{\log(6/\delta)} + 8s^{-\frac{1}{2}}\left(1 + \sqrt{2\log(3/\delta)}\right)\end{aligned} \tag{7}$$

Proof. See the extended version of this article [**?**]. □

Hence, if two classes of graphs are well-separated in terms of the MMD (6), then, for sufficiently large m, s, GSA-φ has the same classification power. However, according to (7), m should be of the order of s, and we have seen that the latter generally needs to be quite large: most usual random feature scheme, typically in $C_\varphi = O(k^2 m)$, still have a high computation cost. We discuss next the use of *optical hardware*.

3.3. Considered choices of φ_{RF}

Gaussian maps: the RF map of the Gaussian kernel [11].

$$\varphi_{Gs}(\mathcal{F}) = m^{-1/2}\left(\sqrt{2}\cos(\mathbf{w}_j^T \mathbf{a}_{\mathcal{F}} + b_j)\right)_{j=1}^{m} \in \mathbb{R}^m \tag{8}$$

where $\mathbf{a}_{\mathcal{F}} = flatten(\mathbf{A}_{\mathcal{F}})$ is the vectorized adjacency matrix of the graphlet $\mathcal{F}$, the $\mathbf{w}_j \in \mathbb{R}^{k^2}$ are drawn from a Gaussian distribution and $b_j \sim \mathcal{U}([0, 2\pi])$. While using a Gaussian kernel on $\mathbf{a}_{\mathcal{F}}$ is not very intuitive, this will serve as a baseline for other methods. Note that φ_{Gs} is not permutation-invariant.

Graphlet kernel		$O(\binom{v}{k} N_k C_k^{\cong})$
GSA-φ with:	φ_k^{match}	$O(C_S s N_k C_k^{\cong})$
	φ_{Gs}	$O(C_S s m k^2)$
	φ_{Gs+eig}	$O(C_S s(mk + k^3))$
	φ_{OPU}	$O(C_S s)$

Table 1. Per-graph complexities of GSA-φ.

Gaussian maps applied on the sorted eigenvalues: We consider a permutation-invariant alternative to the first case. For a graphlet $\mathcal{F}$ we denote the vector of its *sorted* eigenvalues by $\boldsymbol{\lambda}(\mathcal{F}) \in \mathbb{R}^k$ and $\varphi_{Gs+eig}(\mathcal{F}) = \varphi_{Gs}(\boldsymbol{\lambda}(\mathcal{F}))$ (with $\mathbf{w}_j$ of dimension k). Note that the existence of co-spectral graphs, that is, non-isomorphic graphs with the same set of eigenvalues, implies a loss of information when computing $\boldsymbol{\lambda}(\mathcal{F})$.

Optical random feature maps: Due to high-dimensional matrix multiplication, Gaussian RFs cost $\mathcal{O}(mk^2)$ and are notably expensive to compute in high-dimension (in this case, large m). To solve this, OPUs (Optical Processing Units) were recently developed to compute a specific random features mapping in *constant time* $\mathcal{O}(1)$ using light scattering [13] – within the physical limits of the OPU, currently of the order of a few millions for both input and output dimensions. Here we again consider the flattened adjacency matrix for simplicity. The OPU computes an operation of the type:

$$\varphi_{OPU}(\mathcal{F}) = m^{-1/2}\left(|\mathbf{w}_j^T \mathbf{a}_{\mathcal{F}} + \mathbf{b}_j|^2\right)_{j=1}^{m}$$

with $\mathbf{b}_j$ a random bias and $\mathbf{w}_j$ a complex vector with Gaussian real and imaginary parts. Both $\mathbf{w}_j, \mathbf{b}_j$ are here incurred by the physics and are unknown, however the corresponding kernel $\kappa(\mathcal{F}, \mathcal{F}')$ has a closed-form expression [13]. Table 1 summarizes the complexities of the mappings φ examined.

4. EXPERIMENTS

4.1. Datasets

In order to first compare performances of different methods in a controlled setting, we consider a synthetic dataset generated by a *Stochastic Block Model (SBM)* [23]. We generate 300 graphs, 240 for training and 60 for testing. Each graph has $v = 60$ nodes divided equally in six communities. Moreover, graphs are divided into two classes $\{0, 1\}$. For each class we fix p_{in} (resp. p_{out}) which is the probability of generating an edge between any two nodes in the same (resp. different) community. Besides, to prevent the classes from being easily discriminated by the average degree, the pairs $(p_{in,i}, p_{out,i})_{i=0,1}$ are chosen such that nodes in graphs of both classes have the same expected degree (set to 10). Having one degree of freedom left, we fix $p_{in,1}$ to 0.3, and vary $r = (p_{in,1}/p_{in,0})$ the inter-class similarity parameter: the closer r is to 1, the more similar both classes are, and thus the harder it is to discriminate them.

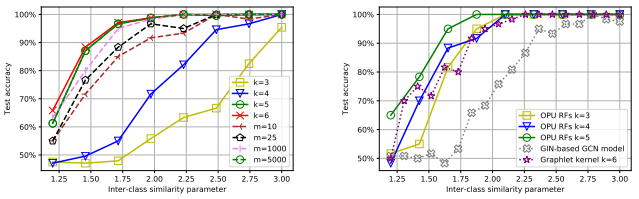

Fig. 1. (left) performance of $GSA-\varphi_{OPU}$ with uniform sampling ($s = 2000$) for different values of k (m fixed at 5000) and m (k fixed to 6). (right) comparison of the performance of $GSA - \varphi_{OPU}$ with RW sampling for different values of k, versus $GSA - \varphi_k^{match}$ with same number of samples and random features $s = 2000$ and $m = 5000$; as well as the GIN-based model consisting of 5 GIN layers followed by 2 fully connected layers, the dimension of hidden layers= 4.*remove the hard-encoded titles in the pdf as well as the (ab). btw, the curve $m = 5000$ is not plotted on the left figure (which makes sense as it should be the same as $k = 6$): remove $m = 5000$ from the legend.*

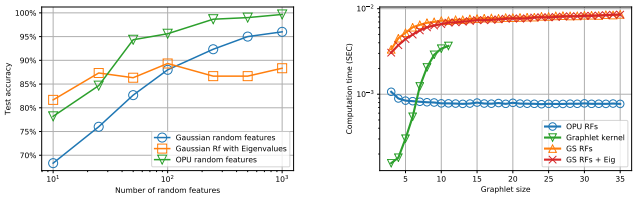

Fig. 2. (left) Test accuracy versus m, for different maps φ in $GSA-\varphi$. (right) Computation time versus k for $GSA-\varphi$ and the graphlet kernel. These figures are for $r = 1.1$, $s = 2000$, $m = 5000$ and a Gaussian map variance $\sigma^2 = 0.01$.

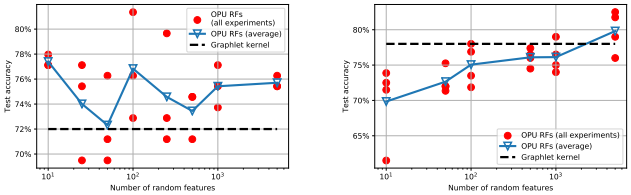

Fig. 3. $GSA - \varphi$ vs the graphlet kernel on real datasets. (left) D&D. (right) Reddit-Binary. With $s = 4000$, and $k = 7$.

In addition, we perform experiments on two real-world datasets: D&D [24] and Reddit-Binary [3] are of size $n = 1178$ and $n = 2000$ respectively. We recall that even though D&D dataset is labeled, the graphs are classified based on their structure only and all other information is discarded. Python codes can be found on `https://github.com/hashemghanem/OPU_Graph_Classifier.git`

4.2. Varying m, k and S_k in $GSA - \varphi_{OPU}$

From Fig. 1 (left), we observe that as k and/or m increase, the performance of $GSA-\varphi_{OPU}$ associated to uniform sampling increases, saturating in this SBM dataset for $m = 5000$ and $k = 6$. From the right figure, and as expected, we note that RW sampling provides better results than the uniform version: the smaller k, the larger the improvement.

4.3. Choice of feature map φ

Comparison of random features. Fig 2 (left) shows that, for sufficiently large m, $GSA-\varphi_{OPU}$ outperforms both $GSA-\varphi_{Gs+Eig}$ and $GSA - \varphi_{Gs}$ (whose variance σ^2 is chosen so as to maximize the validation accuracy).

Comparing $GSA - \varphi_{OPU}$ and $GSA - \varphi_k^{match}$. From Fig 1 (right) we observe that with $s = 2000$ and $m = 5000$, $GSA-\varphi_{OPU}$ with both uniform sampling ($k = 6$) and RW sampling ($k = 5$) clearly outperforms the graphlet kernel with $k = 6$.

Computational time. Fig 2 (right) compares computation time per subgraph, with respect to the subgraph size k. As expected, the execution time is exponential with k for $GSA-\varphi_k^{match}$, roughly polynomial for $GSA - \varphi_{Gs}$ and $GSA - \varphi_{Gs+Eig}$, and constant for $GSA - \varphi_{OPU}$.

To summarize, $GSA - \varphi_{OPU}$ outperforms traditional methods both in accuracy and computation time.

4.4. Comparing $GSA - \varphi_{OPU}$ and GIN-based model

In Fig 1, we see that $GSA - \varphi_{OPU}$ with either RW sampling ($k \geq 4$) or uniform sampling ($k \geq 5$) performs better than the GIN-based graph convolutional model of [10]. We do not report the computational time for GIN, as it is highly dependent on high-speed graphical processing units (GPUs) for training.

4.5. $GSA - \varphi_{OPU}$ on real datasets

Fig 3 shows the test accuracy of $GSA - \varphi_{OPU}$ versus m, for two real datasets. For each value of m we conduct the experiment 3 times on D&D and 4 times on Reddit-Binary dataset and take the average accuracy. For D&D, although results of the 3 experiments get more concentrated as m increases, no clear and steady improvement in the average accuracy is observed. This might be accentuated by the fact that node features are ignored. However, this average is still better than the accuracy obtained by $GSA - \varphi_k^{match}$. For Reddit-Binary, the variance of experiments also decreases slightly with m. More importantly, the average accuracy is monotonically increasing, and outperforms $GSA - \varphi_k^{match}$ for $m \geq 5000$.

5. CONCLUSION

We proposed a generic framework that can deploy OPUs random features in graph classification, since OPUs compute

such features in $\mathcal{O}(1)$ in both input/output dimensions. Then, we showed a concentration of the random embedding around the MMD metric. Our experiments showed that our algorithm is significantly faster than the graphlet kernel with graphlet sampling and performs better while concentrating around the MMD metric. Moreover, it outperformed a state-of-the-art graph convolutional network on graph classification.

A major point left open to be analyzed is how to use our algorithm to classify graphs with node features. One promising possibility is to use our algorithm to generate features embeddings on the graph level, and then feed these embeddings with the nodes' features to a deep neural network. On the theoretical side, the properties of the MMD metric could be further analyzed on particular models of graphs to get a concentration with higher certainty.

6. REFERENCES

[1] Giannis Nikolentzos, Polykarpos Meladianos, and Michalis Vazirgiannis, "Matching node embeddings for graph similarity," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[2] 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.

[3] Christopher Morris, Nils M Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann, "Tudataset: A collection of benchmark datasets for learning with graphs," *arXiv preprint arXiv:2007.08663*, 2020.

[4] Chi Thang Duong, Thanh Dat Hoang, Ha The Hien Dang, Quoc Viet Hung Nguyen, and Karl Aberer, "On node features for graph neural networks," *arXiv preprint arXiv:1911.08795*, 2019.

[5] X Yan and J Han, "gspan: Graph-based substructure pattern mining, 2002," *Published by the IEEE Computer Society*, 2003.

[6] Nils M Kriege, Fredrik D Johansson, and Christopher Morris, "A survey on graph kernels," *Applied Network Science*, vol. 5, no. 1, pp. 1–42, 2020.

[7] Nino Shervashidze, SVN Vishwanathan, Tobias Petri, Kurt Mehlhorn, and Karsten Borgwardt, "Efficient graphlet kernels for large graph comparison," in *Artificial Intelligence and Statistics*, 2009, pp. 488–495.

[8] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun, "Spectral Networks and Locally Connected Networks on Graphs," in *ICLR*, 2014, pp. 1–14.

[9] Michael M. Bronstein, Joan Bruna, Yann Lecun, Arthur Szlam, and Pierre Vandergheynst, "Geometric Deep Learning: Going beyond Euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017.

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

[11] Ali Rahimi and Benjamin Recht, "Random features for large-scale kernel machines," in *Advances in neural information processing systems*, 2008, pp. 1177–1184.

[12] Aman Sinha and John C Duchi, "Learning kernels with random features," in *Advances in Neural Information Processing Systems*, 2016, pp. 1298–1306.

[13] Alaa Saade, Francesco Caltagirone, Igor Carron, Laurent Daudet, Angélique Drémeau, Sylvain Gigan, and Florent Krzakala, "Random projections through multiple optical scattering: Approximating kernels at the speed of light," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 6215–6219.

[14] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander J. Smola, "A Kernel Method for the Two-Sample Problem," in *Advances in Neural Information Processing Systems (NIPS)*, 2007, pp. 513–520.

[15] Johannes Kobler, Uwe Schöning, and Jacobo Torán, *The graph isomorphism problem: its structural complexity*, Springer Science & Business Media, 2012.

[16] OEIS Foundation Inc., "The online encyclopedia of integer sequences, https://oeis.org/a000088," 2019.

[17] Anna Lubiw, "Some np-complete problems similar to graph isomorphism," *SIAM Journal on Computing*, vol. 10, no. 1, pp. 11–21, 1981.

[18] Jure Leskovec and Christos Faloutsos, "Sampling from large graphs," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006, pp. 631–636.

[19] Anjan Dutta and Hichem Sahbi, "Stochastic Graphlet Embedding," *IEEE Transactions on Neural Networks and Learning Systems*, 2018.

[20] Nicolas Keriven, Anthony Bourrier, Rémi Gribonval, and Patrick Pérèz, "Sketching for Large-Scale Learning of Mixture Models," *Information and Inference: A Journal of the IMA*, vol. 7, no. 3, pp. 447–508, 2018.

[21] Nicolas Keriven, Damien Garreau, and Iacopo Poli, "NEWMA: a new method for scalable model-free online change-point detection," *arXiv:1805.08061*, pp. 1–22, 2018.

[22] Bharath K. Sriperumbudur, Arthur Gretton, Kenji Fukumizu, Bernhard Schölkopf, and Gert R.G. Lanckriet, "Hilbert space embeddings and metrics on probability measures," *The Journal of Machine Learning Research*, vol. 11, pp. 1517–1561, 2010.

[23] Thorben Funke and Till Becker, "Stochastic block models: A comparison of variants and inference methods," *PloS one*, vol. 14, no. 4, 2019.

[24] Paul D Dobson and Andrew J Doig, "Distinguishing enzyme structures from non-enzymes without alignments," *Journal of molecular biology*, vol. 330, no. 4, pp. 771–783, 2003.

Appendices

A. PROOF OF THEOREM 1

Proof. We decompose the proof in two steps.

Step 1: infinite s, finite m. First we define the random variables $x_j = |\mathbb{E}_{F\sim S_k(\mathcal{G})}\xi_{\mathbf{w}_j}(F) - \mathbb{E}_{F'\sim S_k(\mathcal{G}')}\xi_{\mathbf{w}_j}(F')|^2$, which are: i/independent, ii/have expectation $MMD(\mathcal{G},\mathcal{G}')^2$, /iii are bounded by the interval $[0,4]$ based on our assumption $|\xi_w| \leq 1$. Thus, as a straight result of applying Hoeffding's inequality with easy manipulation: with probability $1-\delta$

$$\left|\frac{1}{m}\sum_{j=1}^{m} x_j - MMD(\mathcal{G},\mathcal{G}')^2\right| \leq \frac{4\sqrt{\log(2/\delta)}}{\sqrt{m}} \tag{9}$$

Step 2: finite s and m. For any *fixed* set of random features $\{w_j\}_{1,\ldots,m}$ and based on our previous assumptions we have: i/ φ_{RF} is in a ball of radius $M = \frac{\sqrt{m}}{\sqrt{m}} = 1$, ii/ $\mathbb{E}_{F\sim S_k(\mathcal{G})}\,\varphi(F) = \mathbb{E}\left(\frac{1}{s}\sum_i \varphi(F_i)\right)$. Therefore, we can directly apply the vector version of Hoeffding's inequality on the vectors $\frac{1}{s}\sum_i \varphi(F_i)$ to get that with probability $1-\delta$:

$$\left\|\mathbb{E}_{F\sim S_k(\mathcal{G})}\varphi(F) - \frac{1}{s}\sum_i \varphi(F_i)\right\| \leq \frac{1+\sqrt{2\log\frac{1}{\delta}}}{\sqrt{s}} \tag{10}$$

Defining $J_{exp}(\mathcal{G},\mathcal{G}') = \|\mathbb{E}_{F\sim S_k(\mathcal{G})}\varphi(F) - \mathbb{E}_{F'\sim S_k(\mathcal{G}')}\varphi(F')\|$ and $J_{avg}(\mathcal{G},\mathcal{G}') = \|\frac{1}{s}\sum_i \varphi(F_i) - \frac{1}{s}\sum_i \varphi(F'_i)\|$, then using triangular inequality followed by a union bound based on (10), we have the following with probability $1-2\delta$,

$$\left|J_{exp}(\mathcal{G},\mathcal{G}') - J_{avg}(\mathcal{G},\mathcal{G}')\right| \leq \frac{2}{\sqrt{s}}\left(1+\sqrt{2\log\frac{1}{\delta}}\right)$$

On the other hand, $J_{exp}(\mathcal{G},\mathcal{G}') + J_{avg}(\mathcal{G},\mathcal{G}') \leq 4$, so with same probability:

$$\left|J_{exp}(\mathcal{G},\mathcal{G}')^2 - J_{avg}(\mathcal{G},\mathcal{G}')^2\right| \leq \frac{8}{\sqrt{s}}\left(1+\sqrt{2\log\frac{1}{\delta}}\right) \tag{11}$$

Since it is valid for any fixed set of random features, it is also valid with *joint* probability on random features and samples, by the law of total probability.

Finally, combining (9), (11) with a union bound and a triangular inequality, we have with probability $1-3\delta$,

$$\left|\|\varphi(\mathfrak{F}_{\mathcal{G}}) - \varphi(\mathfrak{F}_{\mathcal{G}'})\|^2 - MMD(\mathcal{G},\mathcal{G}')^2\right| \leq \frac{4\sqrt{\log(2/\delta)}}{\sqrt{m}} + \frac{8}{\sqrt{s}}\left(1+\sqrt{2\log\frac{1}{\delta}}\right)$$

which concludes the proof by taking δ as $\delta/3$.

□