Graph Random Neural Features for Distance-Preserving Graph Representations

Daniele Zambon [1] **Cesare Alippi** [1 2] **Lorenzo Livi** [3 4]

Abstract

We present Graph Random Neural Features (GRNF), a novel embedding method from graph-structured data to real vectors based on a family of graph neural networks. The embedding naturally deals with graph isomorphism and preserves the metric structure of the graph domain, in probability. In addition to being an explicit embedding method, it also allows us to efficiently and effectively approximate graph metric distances (as well as complete kernel functions); a criterion to select the embedding dimension trading off the approximation accuracy with the computational cost is also provided. GRNF can be used within traditional processing methods or as a training-free input layer of a graph neural network. The theoretical guarantees that accompany GRNF ensure that the considered graph distance is metric, hence allowing to distinguish any pair of non-isomorphic graphs.

1. Introduction

Inference on graph-structured data is one of the hottest topics in machine learning, thanks to successes achieved in several scientific fields, like neurosciences, chemistry, computational biology and social sciences (Elton et al., 2019; Battaglia et al., 2018; Li et al., 2017). One of the major research challenges there consists of building a practical solution able to process graphs, yet managing the graph isomorphism problem. A way to address this latter problem passes through metric distances and complete kernels; however, it has been shown to be at least as hard as deciding whether two graphs are isomorphic or not (Gärtner et al., 2003).

[1]Universitá della Svizzera italiana, Lugano, Switzerland [2]Politecnico di Milano, Milano, Italy [3]University of Manitoba, Winnipeg, Canada [4]University of Exeter, Exeter, United Kingdom. Correspondence to: Daniele Zambon <daniele.zambon@usi.ch>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

When data come as real vectors, the seminal paper by Rahimi & Recht (2008a) provides an efficient method to approximate radial basis kernels, exposing a parameter —the embedding dimension— trading off computational complexity with approximation accuracy. The Random Kitchen Sinks (Rahimi & Recht, 2009) technique builds on (Rahimi & Recht, 2008a) by adding a linear trainable layer and achieves, via convex optimization, an optimal estimation error on regression tasks (Rahimi & Recht, 2008b); see also (Principe & Chen, 2015; Rudi & Rosasco, 2017) for discussions. More recently, significant efforts aim at moving the research to the graph domain (Oneto et al., 2017), with most contributions focusing on graph neural network properties, especially on their ability to discriminate between non-isomorphic graphs (Chen et al., 2019; Maron et al., 2019a; Xu et al., 2019). Recent research also provided neural architectures granting the universal approximation property for functions on the graph domain (Maron et al., 2019c; Keriven & Peyré, 2019); the property holds asymptotically with the number of neurons.

Here, we propose Graph Random Neural Features (GRNF), a training-free embedding method that provides a map $\mathbf{z} : \mathcal{G} \to \mathbb{R}^M$ from attributed graphs to numeric vectors and manages the graph isomorphism problem. We prove that GRNF is able to discriminate between any pair of non-isomorphic graphs in probability and approximately preserves the metric structure of the graph domain in the embedding space. Notably, GRNF can also be employed as the first layer of a graph neural network architecture. The main idea is to construct the map $\mathbf{z}$ from a family $\mathcal{F} = \{\psi(\,\cdot\,;\mathbf{w}) \mid \mathbf{w} \in \mathcal{W}\}$ of *graph neural feature* maps $\psi(\,\cdot\,;\mathbf{w}) : \mathcal{G} \to \mathbb{R}$, which are node-permutation-invariant graph neural networks with a single hidden-layer and a scalar output (Maron et al., 2019b), separating[1] graphs in $\mathcal{G}$. Parameter vector $\mathbf{w} \in \mathcal{W}$ is randomly sampled from a suitable distribution P and encodes the parameters associated with each neuron; as such, no training is requested. Figure 1 provides a visual description.

The major –novel– theoretical contributions provided here

[1]Family $\mathcal{F}$ is said to separate graphs of $\mathcal{G}$ when for any pair of distinct (non-isomorphic) graphs $g_1, g_2 \in \mathcal{G}$, there exists a parameter vector $\mathbf{w} \in \mathcal{W}$ so that $\psi(g_1;\mathbf{w}) \neq \psi(g_2;\mathbf{w})$.

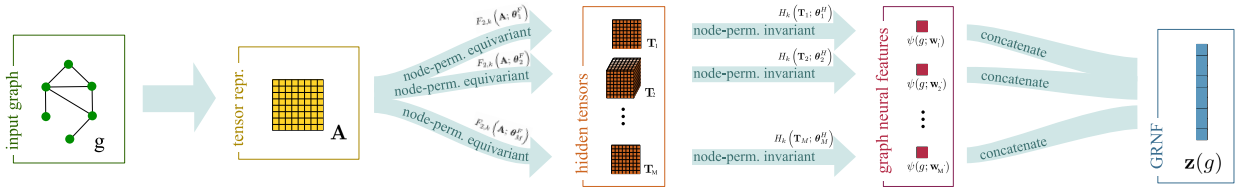

Figure 1. Scheme of a GRNF map $\mathbf{z} : \mathcal{G} \to \mathbb{R}^M$. An n-node graph $g \in \mathcal{G}$ is firstly represented as a weighted adjacency matrix $\mathbf{A} \in \mathcal{T}^2$ (Section 2). A collection of M graph neural features $\{\psi(g; \mathbf{w}_m)\}_{m=1}^M$ is then computed, weighted and, finally, concatenated in vector $\mathbf{z}(g)$. As described in Section 3, each graph neural feature map $\psi(\,\cdot\,; \mathbf{w}_m)$ is the composition of a node-permutation equivariant function, that maps matrix $\mathbf{A}$ to a tensor $\mathbf{T}_m \in \mathcal{T}^k$ of (potentially different) order k, and a node-permutation invariant one, that maps $\mathbf{T}_m$ to a scalar value $\psi(g; \mathbf{w}_m) \in \mathbb{R}$.

are associated with Theorems 1 and 2. Briefly,

- Theorem 1 shows that, given a suitable distribution P over parameter set $\mathcal{W}$, distance

$$d_P(g_1, g_2) = \left(\mathbb{E}_{\mathbf{w} \sim P} \left[\left(\psi(g_1; \mathbf{w}) - \psi(g_2; \mathbf{w}) \right)^2 \right] \right)^{\frac{1}{2}}, \tag{1}$$

  is metric. This implies that $d_P(g_1, g_2) > 0$ if and only if the two graphs are non-isomorphic.

- Theorem 2 proves that the squared distance $|\mathbf{z}(g_1) - \mathbf{z}(g_2)|_2^2$ between graph representations converges, by increasing the embedding dimension M, to $d_P(g_1, g_2)^2$ as $O(M^{-\frac{1}{2}})$, and constructively suggests an embedding dimension $M_{\epsilon,\delta}$ that guarantees the discrepancy between the two to be less than an arbitrary value ε with probability at least $1 - \delta$. This guides the designer in selecting the embedding dimension M.

The paper is organized as follows. Section 3 introduces the family $\mathcal{F}$ of graph neural features $\psi(\,\cdot\,; \mathbf{w})$. Section 4 defines distance d_P and proves Theorem 1. The GRNF map $\mathbf{z}$ is presented in Section 6, where Theorem 2 is also proven. Section 7 relates our work with existing literature. Finally, experimental results in Section 8 empirically validate our theoretical developments.

2. Notation

For $k \in \mathbb{N}$, denote the space of order-k tensors with size n on every mode as

$$\mathcal{T}^k = \mathbb{R}^{\overbrace{n \times \ldots n \times n}^{k \text{ times}}};$$

and $\mathbb{R}$ with $\mathcal{T}^0$. Let us denote with $\pi \star \mathbf{T}$ the operation of applying permutation π, with π in the symmetric group S_n, to each mode of tensor $\mathbf{T} \in \mathcal{T}^k$, namely $(\pi \star \mathbf{T})_{i_1,\ldots,i_k} = \mathbf{T}_{\pi(i_1),\ldots,\pi(i_k)}$. In this paper we use the convention to represent with $\mathbf{A}$ tensors of order 2, i.e., matrices, and with $\mathbf{T}$ tensors of generic order $k \in \mathbb{N}_0$.

Let us define a graph g with at most n nodes as a triple $(V_g, E_g, \mathrm{at}_g)$, with $V_g \subseteq \{1, 2, \ldots, n\}$ a set of nodes, $E_g \subset V_g \times V_g$ a set of edges, and attribute map $\mathrm{at}_g : V_g \cup E_g \to \mathbb{R}$ associating nodes and edges with scalar attributes in a bounded subset of $\mathbb{R}$. We denote the set of such graphs with $\mathcal{G}(n)$, and with $\mathcal{G}$ the space $\bigcup_{n \in \mathbb{N}} \mathcal{G}(n)$ of graphs with arbitrarily large, yet finite order. When no self-loops are present, we can represent each graph $g \in \mathcal{G}(n)$ with a tensor $\mathbf{A}_g \in \mathcal{T}^2$, where $(\mathbf{A}_g)_{i,i} = \mathrm{at}_g(i)$, for $i \in V_g$, $(\mathbf{A}_g)_{i,j} = \mathrm{at}_g((i,j))$, for $(i,j) \in E_g$ and $(\mathbf{A}_g)_{i,j} = 0$, otherwise. Different ordering choice of the nodes in V_g results in different representations $\mathbf{A}_g$. In our case, in which nodes are non-identified, this results in a bijection between $\mathcal{G}(n)$ and the quotient space $\mathcal{T}^2{}_{/S_n} = \{[\mathbf{A}]_{/S_n} \mid \mathbf{A} \in \mathcal{T}^k\}$ of equivalence classes $[\mathbf{A}]_{/S_n} = \{\pi \star \mathbf{A} \mid \pi \in S_n\}$.

Please note that the above assumptions are made to simplify the maths, however, extensions to allow for self-loops and attributes of any dimension is straightforward and detailed in Appendix A.

3. Graph Neural Features

Given an integer $n \in \mathbb{N}$, we call *graph feature map* any function $f : \mathcal{T}^2 \to \mathbb{R}$ to the real set which is *invariant* under permutation of the nodes, namely, $f(\mathbf{A}_g) = f(\pi \star \mathbf{A}_g)$, for every $\pi \in S_n$; indeed, having g multiple representations $\mathbf{A}_g \in [\mathbf{A}_g]_{/S_n}$, this property is necessary to make $f(\mathbf{A}_g)$ a proper function of the graph g itself, hence resulting with the same output regardless of the specific representation $\mathbf{A}_g$. For this reason, in the following, we use the notation $f(g)$ and $f(\mathbf{A}_g)$ interchangeably. Conversely, a function $f : \mathcal{T}^l \to \mathcal{T}^k$ is said *equivariant* to node permutation if $f(\pi \star \mathbf{T}) = \pi \star f(\mathbf{T}), \forall \pi \in S_n$.

Recent findings (Maron et al., 2019b) have shown that the set of all linear permutation-invariant functions $\mathcal{T}^k \to \mathbb{R}$ is a vector space of dimension $\mathrm{Bell}(k)$, i.e., the number of partitions of set $\{1, 2, \ldots, k\}$; similarly, the space of linear permutation-equivariant functions is proven to be a $\mathrm{Bell}(k + l)$-dimensional vector space. Denoting with $\{\mathbf{I}_\gamma\}_{\gamma=1}^{\mathrm{Bell}(k)}$ and $\{\mathbf{E}_\gamma\}_{\gamma=1}^{\mathrm{Bell}(k+l)}$ the bases of invariant and

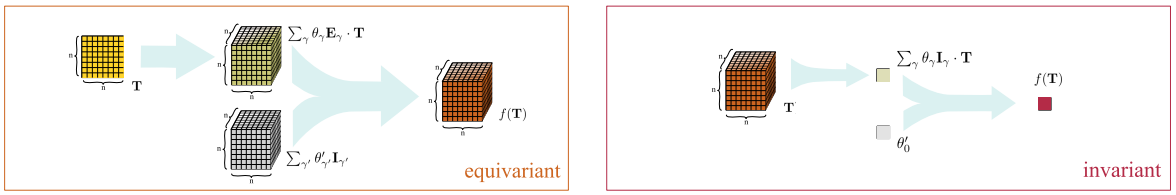

Figure 2. Structure of an affine node-permutation equivariant function $F_{2,K}(\mathbf{T}, \boldsymbol{\theta})$ (left) and a node-permutation invariant one $H_k(\mathbf{T}, \boldsymbol{\theta})$ (right) as linear combinations of the bases $\{\mathbf{I}_\gamma\}$ and $\{\mathbf{E}_\gamma\}$ defined in Equation 2. In this example, $k = 3$.

equivariant linear spaces, respectively, we obtain that every affine invariant and equivariant function can be written in terms of tensor products and sums of the form

$$f(\mathbf{T}) = \begin{cases} H_k(\mathbf{T}; \boldsymbol{\theta}) = \sum_\gamma \theta_\gamma \mathbf{I}_\gamma \mathbf{T} + \theta'_0 \\ F_{l,k}(\mathbf{T}; \boldsymbol{\theta}) = \sum_\gamma \theta_\gamma \mathbf{E}_\gamma \mathbf{T} + \sum_{\gamma'} \theta'_{\gamma'} \mathbf{I}_{\gamma'} \end{cases} \tag{2}$$

where $\{\theta_\gamma\}$ and $\{\theta'_{\gamma'}\}$ are coefficients that relate to the linear and bias parts, respectively, and $\boldsymbol{\theta} = \{\theta_\gamma\} \cup \{\theta'_{\gamma'}\}$. We refer the reader to Appendix A for a proper definition of $\{\mathbf{I}_\gamma\}$ and $\{\mathbf{E}_\gamma\}$; Figure 2 provides a visual representation of these affine functions.

We are ready to define the set of graph neural features as composition of equivariant and invariant affine maps and some (nonlinear) activation functions.

Definition 1 (Graph neural feature)**.** *We define a* graph neural feature *to be a parametric map* $\psi(\,\cdot\,; \mathbf{w}) : \mathcal{T}^2 \to \mathbb{R}$*, with parameters* $\mathbf{w} = (k, \boldsymbol{\theta}_F, \boldsymbol{\theta}_H)$*, and resulting from the composition*

$$\mathcal{T}^2 \xrightarrow{F_{2,k}(\,\cdot\,;\boldsymbol{\theta}_F)} \mathcal{T}^k \xrightarrow{\rho_e} \mathcal{T}^k \xrightarrow{H_k(\,\cdot\,;\boldsymbol{\theta}_H)} \mathbb{R} \xrightarrow{\rho_i} \mathbb{R},$$

where ρ_i*,* ρ_e *are activation functions applied component wise and* $F_{2,k}(\cdot; \boldsymbol{\theta}_F)$*,* $H_k(\cdot; \boldsymbol{\theta}_H)$ *are affine equivariant and invariant ones, respectively, in the form of Equation 2. The parameter space is*

$$\mathcal{W} = \{\mathbf{w} = (k, \boldsymbol{\theta}_F, \boldsymbol{\theta}_H)\}, \tag{3}$$

for $k \in \mathbb{N}$*,* $\boldsymbol{\theta}_F \in \mathbb{R}^{\mathrm{Bell}(k+2)+\mathrm{Bell}(k)}$*,* $\boldsymbol{\theta}_H \in \mathbb{R}^{\mathrm{Bell}(k)+1}$*.*

Note that by composing component-wise any activation function ρ_i, ρ_e –e.g., the sigmoid– to an affine invariant (equivariant) function gives an invariant (equivariant) function. Therefore, in the end $\psi(\,\cdot\,; \mathbf{w})$ is an invariant function. Without any ambiguity, we can write $\psi(g; \mathbf{w})$ as function of g. We comment that, despite ψ has been introduced for graphs in $\mathcal{G}(n)$ with at most n nodes, it is readily extended to operate on the entire space $\mathcal{G}$ of graphs with arbitrarily large order, in fact, parameter space $\mathcal{W}$ does not change with respect to n (Maron et al., 2019b); this is crucial to compare graphs of different orders. We denote with

$$\mathcal{F}(\rho_i, \rho_e, \mathcal{W}), \tag{4}$$

or simply $\mathcal{F}$, the set of all graph neural feature maps introduced above and defined over the entire $\mathcal{G}$. A generalization to graph with vector attributes associated to nodes and edges is possible, as well (please, see to Appendix A).

As shown in the following Lemma 1, family $\mathcal{F}$ is rich enough to separate graphs of $\mathcal{G}$; in other terms, its expressive power permits to distinguish any pair of non-isomorphic graphs.

Lemma 1. *Let* ρ_e *be a squashing function, and* ρ_i *a non-constant one. Then set* $\mathcal{F}(\rho_i, \rho_e, \mathcal{W})$ *is rich enough to separate graphs of* $\mathcal{G}$*, namely, for any pair of distinct (non-isomorphic) graphs* $g_1, g_2 \in \mathcal{G}$*, there exists a parameter configuration* $\mathbf{w}^* \in \mathcal{W}$ *so that* $\psi(g_1; \mathbf{w}^*) \neq \psi(g_2; \mathbf{w}^*)$*.*

Note that the sigmoid function $\sigma(x) = (1 + e^{-x})^{-1}$ is both squashing and non-constant, therefore family $\mathcal{F}(\sigma, \sigma, \mathcal{W})$ of graph neural features

$$\psi(\cdot, \mathbf{w}) = \sigma \circ H_k(\cdot, \boldsymbol{\theta}_H) \circ \sigma \circ F_{2,k}(\cdot, \boldsymbol{\theta}_F)$$

fulfills the hypothesis of Lemma 1.

The proof proceeds with the same strategy adopted by Hornik et al. (1989) to exhibit approximation capabilities of multi-layer perceptrons and, recently, by Keriven & Peyré (2019) to graph neural networks; see Appendix B for details.

4. A Metric Distance for Graphs

The family $\mathcal{F}$ of graph neural features (4) allows to define the distance $d_P(g_1, g_2)$ of Equation 1. The distance assesses the expected discrepancy between graph neural features associated with two graphs $g_1, g_2 \in \mathcal{G}$; it is only requested that $\mathbb{E}_{\mathbf{w}}[\psi(g; \mathbf{w})^2] < \infty$ for every $g \in \mathcal{G}$. Notice that the distance depends on the distribution P. As such, a change in P results in a different distance. In a supervised setting, this is a "parameter" that can be tuned (see subsequent Section 6). One of the simplest examples of distance d_P originates from considering a uniform distribution over a finite set $\{\tilde{\mathbf{w}}_r\}_{r=1}^R$ of $R \in \mathbb{N}$ parameters. This specific choice gives

$$d_P(g_1, g_2) = \left(\frac{1}{R} \sum_r \left(\psi(g_1; \tilde{\mathbf{w}}_r) - \psi(g_2; \tilde{\mathbf{w}}_r) \right)^2 \right)^{\frac{1}{2}},$$

and results in a pseudo-metric distance, as it is positive and symmetric, satisfies the triangular inequality. However, the *identifiability* property

$$g_1 = g_2 \iff d_P(g_1, g_2) = 0, \qquad \forall\, g_1, g_2 \in \mathcal{G} \tag{5}$$

does not hold. It can be proved[2] that the resulting distance d_P is always at least a pseudo-metric, regardless of the choice of P.

Remarkably, a principled choice of P, such that its support $\mathrm{supp}(P)$ is the entire set $\mathcal{W}$, ensures that d_P is metric, as Theorem 1 guarantees.

Theorem 1. *Consider set $\mathcal{F}(\rho_i, \rho_e, \mathcal{W})$ of graph neural features on the graph set $\mathcal{G}$. Define a probability distribution P on $\mathcal{W}$, and the corresponding graph distance d_P according to (1). Under the following assumptions, space $(\mathcal{G}, d_P)$ is metric.*

- ***(A1)*** *Functions $\rho_i, \rho_e : \mathbb{R} \to \mathbb{R}$ are continuous, with ρ_e being a squashing function and ρ_i a non-constant one;*
- ***(A2)*** *Support $\mathrm{supp}(P)$ of P covers $\mathcal{W}$;*
- ***(A3)*** *There exists a positive constant $C_{\mathcal{G}}$ such that the fourth momentum $\mathbb{E}_{\mathbf{w}}[\psi(g;\mathbf{w})^4] < C_{\mathcal{G}}$, $\forall\, g \in \mathcal{G}$.*

Notice that all these assumptions are "only" sufficient conditions, and are rather mild. In fact, one practical choice to satisfy them all at once is to build distribution P over a Poisson distribution to select the tensor order k and a corresponding multivariate Gaussian distribution for parameter vectors $\boldsymbol{\theta}_H, \boldsymbol{\theta}_F$, and consider the sigmoid function for both ρ_i and ρ_e. Moreover, without loss on generality, we can assume $C_{\mathcal{G}} = 1$.

The core of the proof aims at showing that d_P possesses the identifiability property (5). To sketch the proof, notice that Assumption **(A1)** enables Lemma 1 and ensures that for any pair of graphs $g_1 \neq g_2$ there exists a particular $\tilde{\mathbf{w}} \in \mathcal{W}$ for which $\psi(g_1;\tilde{\mathbf{w}}) \neq \psi(g_2;\tilde{\mathbf{w}})$. On the other hand, Assumption **(A2)** grants that every feature map in $\mathcal{F}$ is taken into account by (1), and together with **(A1)**, that we can find a neighbourhood $U(\tilde{\mathbf{w}})$ of non-null probability for which

$$\psi(g_1;\mathbf{w}) \neq \psi(g_2;\mathbf{w}), \qquad \forall \mathbf{w} \in U(\tilde{\mathbf{w}}).$$

Property (5) follows from $(\psi(g_1,\cdot) - \psi(g_2,\cdot))^2 \geq 0$.

Finally, showing that d_P is always positive, symmetric, and satisfies the triangular inequality is easier. Observe in fact that for every random variable X_1, X_2, we have $\mathbb{E}[(X_1 + X_2)^2] \leq (\sqrt{\mathbb{E}[X_1^2]} + \sqrt{\mathbb{E}[X_2^2]})^2$, and in particular for $X_1 = \psi(g_1;\mathbf{w}) - \psi(g_3;\mathbf{w})$ and $X_2 = \psi(g_3;\mathbf{w}) - \psi(g_2;\mathbf{w})$. A detailed proof is provided in Appendix B.

[2] Please, see proof of Theorem 1.

5. A Complete Kernel for Graphs

The family $\mathcal{F}$ of graph neural feature maps in (4) allows to define also a kernel function for graphs, hence to process graph in the well-grounded theory of reproducing kernel Hilbert spaces.

Following the same rationale used for distance d_P in Section 4, we define the following positive-definite kernel

$$\kappa_P(g_1, g_2) := \mathbb{E}_{\mathbf{w}\sim P}\left[\xi(g_1;\mathbf{w})\,\xi(g_2;\mathbf{w})\right], \tag{6}$$

where $\xi(g_1, \mathbf{w}) = \psi(g_1;\mathbf{w}) - \psi(\mathbf{0}_g;\mathbf{w})$ and $\mathbf{0}_g$ is a "null" graph, which can be represented by the adjacency $\mathbf{A} = 0 \in \mathbb{R}^{1\times 1}$ of a graph with a single node. Kernel κ_P is intimately related with d_P, in fact:

$$d_P(g_1, g_2)^2 = \kappa_P(g_1, g_1) - 2\kappa_P(g_1, g_2) + \kappa_P(g_2, g_2), \tag{7}$$

or, alternatively, we can express k_P as induced by $(d_P)^2$

$$\kappa_P(g_1, g_2) = \frac{1}{2}\left(d_P(g_1, \mathbf{0})^2 + d_P(g_2, \mathbf{0})^2 - d_P(g_1, g_2)^2\right). \tag{8}$$

Proposition 1 shows that κ_P in (6) is a *complete* kernel, meaning that the canonical embedding $\phi : \mathcal{G} \to \mathcal{H}$ to the reproducing kernel Hilbert space $\mathcal{H}$, and for which we can write $\kappa(g_1, g_2) = \langle \phi(g_1), \phi(g_2) \rangle_{\mathcal{H}}$, is injective (Gärtner et al., 2003), thus it maps distinct graphs to different points in $\mathcal{H}$.

Proposition 1. *Under Assumptions **(A1)–(A3)**, we have:*

- *d_P is of* negative type*, i.e., for every $S \in \mathbb{N}$, any set of graphs $\{g_s\}_{s=1}^S$ and any set of scalars $\{c_s\}_{s=1}^S$, with $\sum_s c_s = 0$, it follows $\sum_i \sum_j c_i\, c_j\; d_P(g_i, g_j) \leq 0$;*
- *κ_P is a* complete *graph kernel.*

The proof follows from the work of Sejdinovic et al. (2013), where if κ_P can be written in the form of (8) and $(d_P)^2$ is a semi-metric of negative type, then (i) d_P is of negative type, as well [Prop. 3], and (ii) κ_P is a complete kernel [Prop. 14].

We now need to show that the semi-metric $(d_P)^2$ is of negative type. Please, notice that kernel κ_P is positive semi-definite, in fact, for any choice of $S, \{c_s\}, \{g_s\}$

$$\sum_{i,j=1}^{S} c_i c_j\, k_P(g_i, g_j) = \mathbb{E}_{\mathbf{w}\sim P}\left[f(\mathbf{w})^2\right] \geq 0. \tag{9}$$

where $f(\mathbf{w}) = \sum_{s=1}^S c_s \xi(g_s;\mathbf{w})$. From relation (7), and when $\sum_s c_s = 0$,

$$\sum_{i,j=1}^{S} c_i c_j d_P(g_i, g_j)^2 = 0 - 2\sum_{i,j=1}^{S} c_i c_j \kappa_P(g_i, g_j) + 0 \leq 0,$$

thanks to (9). This concludes the thesis.

6. Graph Random Neural Features

The definition of graph distance d_P in (1) as an expectation over the feature maps in $\mathcal{F}$ allows us to create a random mapping that approximately preserves the metric structure of graph space $(\mathcal{G}, d_P)$.

Definition 2 (Graph Random Neural Features (GRNF))**.** *Given probability distribution P defined over $\mathcal{W}$ and an embedding dimension $M \in \mathbb{N}$, we define* Graph Random Neural Features map *a function $\mathbf{z} : \mathcal{G} \to \mathbb{R}^M$ that associates to graph $g \in \mathcal{G}$ the vector*

$$\mathbf{z}(g; \mathbf{W}) := \frac{1}{\sqrt{M}} \left[\psi(g; \mathbf{w}_1), \ldots, \psi(g; \mathbf{w}_M)\right]^\top, \quad (10)$$

where $\mathbf{W} = \{\mathbf{w}_m\}_{m=1}^M$ are drawn independently from P, and $\{\psi(\cdot; \mathbf{w}_m)\}_{m=1}^M \subseteq \mathcal{F}(\rho_i, \rho_e, \mathcal{W})$ are graph neural features (Definition 1).

In the following, we may omit the explicit dependence from $\mathbf{W}$ in $\mathbf{z}(g; \mathbf{W})$, when clear from the context, and use the compact notation $\mathbf{z}(g)$. The computational complexity of (10) is studied in Appendix D. Figure 1 provides a visual representation of the map.

Along the same lines of random Fourier features (Rahimi & Recht, 2008a; Li et al., 2019) —but focusing on a distance rather than on a kernel— we have that the squared norm $|\mathbf{z}(g_1) - \mathbf{z}(g_2)|_2^2$, which can be thought as a sample mean of different graph neural features, is an unbiased estimator of the squared distance $d_P(g_1, g_2)^2$. Moreover, thanks to the law of large numbers, we have

$$|\mathbf{z}(g_1) - \mathbf{z}(g_2)|_2^2 \xrightarrow{p} d_P(g_1, g_2)^2, \quad (11)$$

where the convergence is in probability and as $M \to \infty$, i.e., for any $\varepsilon > 0$

$$\lim_{M\to\infty} \mathbb{P}\left(\left||\mathbf{z}(g_1) - \mathbf{z}(g_2)|_2^2 - d_P(g_1, g_2)^2\right| > \varepsilon\right) = 0.$$

By continuity, we also have $|\mathbf{z}(g_1) - \mathbf{z}(g_2)|_2 \xrightarrow{p} d_P(g_1, g_2)$. Since we proved that distance d_P is metric, the above convergence shows the ability of GRNF to distinguish all non-isomorphic graphs. The convergence (11) is of order $\sqrt{M}$, as one can see by

$$\begin{aligned}
&\mathrm{Var}\left[|\mathbf{z}(g_1) - \mathbf{z}(g_2)|_2^2\right] = \\
&\quad = \frac{1}{M^2} \sum_{m=1}^{M} \mathrm{Var}\left[\left(\psi(g_1; \mathbf{w}) - \psi(g_2; \mathbf{w})\right)^2\right] \\
&\leq \frac{8}{M}\left(\mathbb{E}\left[\psi(g_1; \mathbf{w})^4\right] + \mathbb{E}\left[\psi(g_2; \mathbf{w})^4\right]\right) \overset{(A3)}{\leq} \frac{16}{M}, \quad (12)
\end{aligned}$$

thanks[3] to the convexity of $g(x) = x^4$. Finally, by exploiting (12) and Chebyshev's inequality, the following Theorem 2 follows.

Theorem 2. *Under Assumption (A3), for any value $\varepsilon > 0$ and $\delta \in (0, 1)$, inequality*

$$\mathbb{P}\left(\left||\mathbf{z}(g_1) - \mathbf{z}(g_2)|_2^2 - d_P(g_1, g_2)^2\right| \geq \varepsilon\right) \leq \delta$$

holds with embedding dimension $M \geq \frac{16}{\delta\,\varepsilon^2}$.

Theorem 2 allows to identify an embedding dimension M that ensures to fulfill some application requirements expressed in terms of ε and δ. An analogous result:

$$\mathbb{P}\left(|\widetilde{\kappa}_P(g_1, g_2) - \kappa_P(g_1, g_2)| \geq \varepsilon\right) \leq \delta$$

is proven also for the approximated kernel $\widetilde{\kappa}_P(g_1, g_2) = (\mathbf{z}(g_1) - \mathbf{z}(\mathbf{0}_g))^\top(\mathbf{z}(g_2) - \mathbf{z}(\mathbf{0}_g))$, provided that $M \geq \frac{1}{\delta\,\varepsilon^2}$. Again, this is a consequence of convergence

$$\widetilde{\kappa}_(g_1, g_2) \xrightarrow{p} \kappa_P(g_1, g_2), \quad M \to \infty. \quad (13)$$

Remarkably, we can obtain the same convergence results also when sampling the parameters $\mathbf{W} = \{\mathbf{w}_1, \ldots, \mathbf{w}_M\}$ from a distribution $\overline{P}$ different from P. It is only necessary to appropriately weight each component of map $\mathbf{z}$. In compliance with (Li et al., 2019), we call *Weighted GRNF*

$$\overline{\mathbf{z}}(\,\cdot\,; \mathbf{W}) := \left[\ldots, \sqrt{\frac{p(\mathbf{w}_m)}{M\,\overline{p}(\mathbf{w}_k)}}\psi(\,\cdot\,; \mathbf{w}_m), \ldots\right]^\top \quad (14)$$

where p and $\overline{p}$ are the mixed-type probability distributions[4] associated with P and $\overline{P}$, respectively. We refer the reader to Appendix C for further details on this interesting setting.

The specific choice of distribution P induces different distances d_P (and kernel functions κ_P), and a principled choice of P can make the difference. In practice, we can exploit the trick of sampling from a predefined distribution $\overline{P}$, as in (14), and, a posteriori, identify a suitable distribution P that best serves the task to be solved (Sinha & Duchi, 2016). Specifically, once M parameter vectors $\{\mathbf{w}_m\}_{m=1}^M$ are sampled from $\overline{P}$, we have that in (14) the scalars $\{p_m = p(\mathbf{w}_m)\}_{m=1}^M$ become free parameters.

To conclude the section, we stress that GRNF can trade-off metric distances and complete kernels for practical computability, as described by Theorem 2, in agreement with Gärtner et al. (2003).

[3] Being x^p convex, we have $\left(\frac{X}{2} + \frac{Y}{2}\right)^p \leq \frac{X^p}{2} + \frac{Y^p}{2}$, therefore $\frac{1}{2^p}\mathbb{E}\left[(X+Y)^p\right] \leq \frac{1}{2}(\mathbb{E}[X^p] + \mathbb{E}[Y^p])$.

[4] We consider that random variable $\mathbf{w} = (k, \boldsymbol{\theta}_H, \boldsymbol{\theta}_F) \sim P$ is composed a discrete component k with probability mass function $p_k(k)$ and a continuous part $(\boldsymbol{\theta}_H, \boldsymbol{\theta}_F)$ with probability density function $p_\theta(\boldsymbol{\theta}_H, \boldsymbol{\theta}_F|k)$. Finally, we define with $p(\mathbf{w}) = p_k(k)p_\theta(p_\theta(\boldsymbol{\theta}_H, \boldsymbol{\theta}_F|k)$.

7. Related Work

The process of random sampling features takes inspiration from the work on random Fourier features (Rahimi & Recht, 2008a). There, shift-invariant kernels $\kappa(\mathbf{x}_1, \mathbf{x}_2) = \kappa(\mathbf{x}_1 - \mathbf{x}_2)$ are written in the form of an expected value

$$\mathbb{E}_{\omega \sim p}[\zeta(\mathbf{x}_1; \boldsymbol{\omega})\zeta(\mathbf{x}_2; \boldsymbol{\omega})^*] \tag{15}$$

with p being the Fourier transform of $\kappa(\cdot)$ and $\zeta(\mathbf{x}; \boldsymbol{\omega}) = e^{i\boldsymbol{\omega}^\top \mathbf{x}}$ (Bochner's theorem (Rudin, 1991)). By random sampling parameters $\{\boldsymbol{\omega}_m\}_{m=1}^M$ from p and $\{b_m\}_{m=1}^M$ from the uniform distribution $U([0, 2\pi))$, we have that

$$\mathbb{E}\left[\frac{1}{M}\sum_m \cos(\boldsymbol{\omega}_m^\top \mathbf{x}_1 + b_m)\cos(\boldsymbol{\omega}_m^\top \mathbf{x}_2 + b_m)\right] = \kappa(\mathbf{x}_1, \mathbf{x}_2),$$

hence providing a Monte Carlo approximation method for the kernel. Clearly, being able to compute the Fourier transform p is crucial, however, it is not always possible; see, e.g., Vedaldi & Zisserman (2012) for some examples. Extensions of this approach consider dot-product kernels $\kappa(\mathbf{x}_1, \mathbf{x}_2) = \kappa(\mathbf{x}_1^\top \mathbf{x}_2)$ (Kar & Karnick, 2012). The work by Yu et al. (2016) proposed to keep an orthogonal structure in the matrix of sampled weights in order to achieve better approximations with a smaller number of features. Alternatives to random sampling employ Gaussian quadrature to approximates the kernel with higher a convergence rate (Dao et al., 2017). Finally, we mention the works by Wu et al. (2018; 2019) which, relying on a distance measure between data points, can apply the same rationale also to non-numeric data.

Our proposal builds on the framework of random features, but it works somehow in the opposite direction. Firstly, it defines a parametric family of features $\mathcal{F} = \{\psi(\,\cdot\,; \mathbf{w}) | \mathbf{w} \in \mathcal{W}\}$ that separates graphs of $\mathcal{G}$ —playing the role of $\{\zeta(\,\cdot\,; \boldsymbol{\omega})\}$ in (15)— and, subsequently, it provides a distance (and kernel) function by selecting a probability distribution P on parameter space $\mathcal{W}$. This choice has two major advantages: (1) it does not require to compute distribution P from a κ, and (2) allows the selection of the most appropriate distribution based on the task at hand. Moreover, the same rationale can be applied to other families of features that separate graphs.

8. Experimental Validation

The experimental campaign is divided into two parts. Section 8.1 gives empirical evidence about the claimed convergence as the embedding dimension M grows. Secondly, Section 8.2 shows that our method can be effectively used as a layer of a neural network and achieves results comparable to the current state of the art on classification tasks.

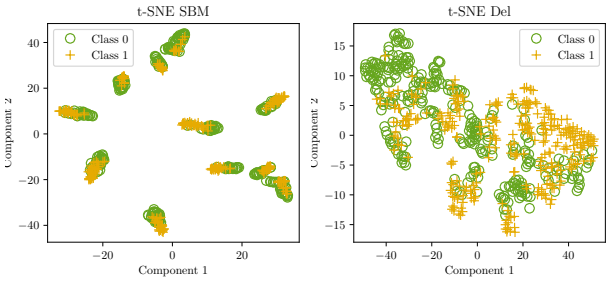

Figure 3. 2-dimensional visualization of the embedding vectors $\mathbf{z}_*(g_i)$ for $i = 1, \ldots, 600$ drawn with t-SNE.

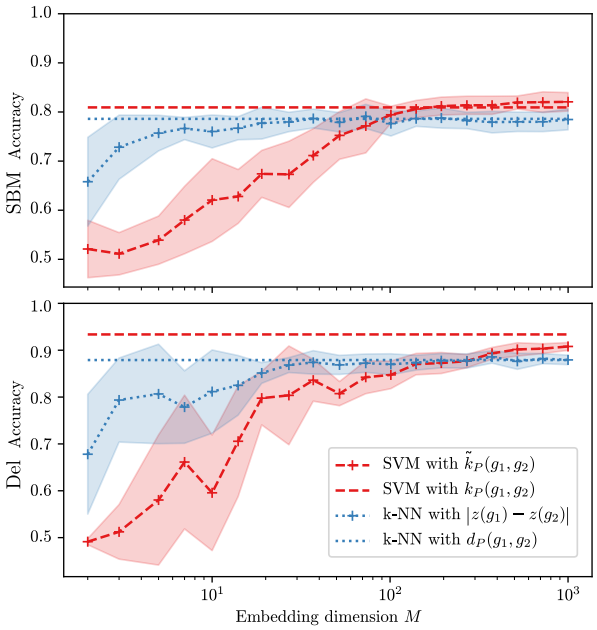

Figure 4. Classification performance in terms of accuracy using different embedding dimensions M. First and second rows correspond to data sets SBM and Del, respectively. Each reported accuracy value is an average across 10 repetitions and the shaded region represents one standard deviation from the average.

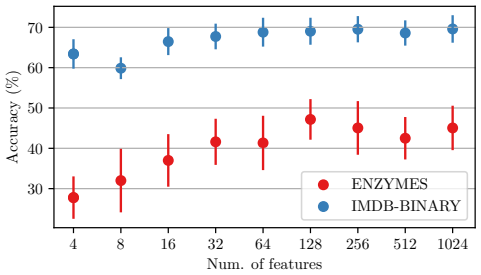

Figure 5. Classification performance on ENZYMES and IMDB-BINARY data sets in terms of accuracy using different embedding dimensions M. Each reported accuracy value is an average across 10 repetitions and the bar indicate one standard deviation from the average.

Table 1. We report accuracy and standard deviation estimated on 10-fold cross-validation, where in each run we consider the optimal hyper-parameter configuration assessed on a validation set. Results from Baseline, DGCNN, DiffPool, ECC, GIN and GraphSAGE are reported from (Errica et al., 2020). (Notice that the authors created two versions of the social data sets, one with no node attributes and the other augmented with node degrees as node feature. Here, we considered the former set up).

	NCI1	PROTEINS	ENZYMES	IMDB-BINARY	IMDB-MULTI	COLLAB
Baseline	69.8±2.2	75.8±3.7	65.2±6.4	50.7±2.4	36.1±3.0	55.0±1.9
DGCNN	76.4±1.7	72.9±3.5	38.9±5.7	53.3±5.0	38.6±2.2	57.4±1.9
DiffPool	76.9±1.9	73.7±3.5	59.5±5.6	68.3±6.1	45.1±3.2	67.7±1.9
ECC	76.2±1.4	72.3±3.4	29.5±8.2	67.8±4.8	44.8±3.1	—
GIN	80.0±1.4	73.3±4.0	59.6±4.5	66.8±3.9	42.2±4.6	75.9±1.9
GraphSAGE	76.0±1.8	73.0±4.5	58.2±6.0	69.9±4.6	47.2±3.6	71.6±1.5
GRNF	66.7±2.4	75.1±2.6	45.9±6.6	69.7±3.8	44.4±3.9	73.2±2.5

8.1. Convergence Increasing the Embedding Dimension

We considered two data sets with two classes: one contains simple graphs with no attributes, and the other one has graphs with bi-dimensional attributes associated to each node. Figure 3 shows a visualization by means of t-SNE (Maaten & Hinton, 2008) to perceive the complexity of the classification problem.

Graphs from the stochastic block model (SBM). We generated two classes of 300, 12-node graphs from the stochastic block model (Holland et al., 1983). Class 0 has a single community with edge probability 0.4, while class 1 has two communities of 6 nodes with 0.8 probability of connecting two nodes of the same community, while the probability of connecting nodes of different communities equals 0.1.

Delaunay's triangulation graphs (Del). We generated two classes of 300, 12-node Delaunay's triangulation graphs (Zambon et al., 2017; Grattarola et al., 2019). The graphs of a single class have been generated starting from a collection of 6 planar points, called seed points (the two classes are determined by different collections of seed points). Seed points are then perturbed with Gaussian noise. Each point corresponds to a node of the graph and its coordinates are considered as node attributes. Finally, the Delaunay's triangulation of the perturbed points gives the topology of the graph.

The first experiment provides empirical evidence of the validity of the bounds in Theorem 2. Since it is not always possible to compute the true value of d_P, we make use of two GRNF, $\mathbf{z}(\,\cdot\,;\mathbf{W}_1)$ and $\mathbf{z}(\,\cdot\,;\mathbf{W}_2)$, both with embedding dimension M. Let be $\Delta(\mathbf{W}) = |\mathbf{z}(g_1,\mathbf{W}) - \mathbf{z}(g_2,\mathbf{W})|_2^2$, then[5]

$$\mathbb{P}\left(|\Delta(\mathbf{W}_1) - \Delta(\mathbf{W}_2)| \geq \varepsilon\right) \leq \frac{128}{M\,\varepsilon^2} =: \delta_M. \tag{16}$$

We also compared the above M-dimensional approximation Δ with a better estimate $\Delta_* := |\mathbf{z}_*(g_1) - \mathbf{z}_*(g_2)|_2^2$ based on a M_*-dimensional map $\mathbf{z}_*$ with $M_* = 10^6 \gg M$. Assuming that equation $\Delta_* = d_P(g_1,g_2)^2$ holds

$$\mathbb{P}\left(|\Delta - \Delta_*| \geq \varepsilon\right) \leq \frac{16}{M\,\varepsilon^2} =: \delta_*. \tag{17}$$

Finally, we have performed a comparison with the estimate provided by the central limit theorem, i.e., assuming that the left-hand side in (17) is equal to

$$2\Phi\left(-\sqrt{M}\frac{\varepsilon}{\sigma}\right) =: \delta_{clt} \tag{18}$$

where Φ is the cumulative density function of standard Gaussian distribution and $\sigma^2 = \mathrm{Var}[(\psi(g_1;\mathbf{w}) - \psi(g_2;\mathbf{w}))^2]$. Graph g_1, g_2 are randomly selected from the two classes of SBM data set. Results in Figure 6 show that the empirical assessments of the left-hand sides in (16) and (17) are smaller then their respective bounds on the right-hand side of the inequalities, hence confirming the theoretical predictions.

The second experiment is conducted by comparing the performance drop in adopting the approximations (11) and (13) with varying embedding dimension M. The task is a binary classification, and it is performed using support vector machine and k-nearest neighbour classifiers, as standard kernel- and distance-based methods. Figure 4 shows the achieved classification accuracy. We see that the accuracy obtained with GRNF empirical converges to the accuracy obtained with $M_* = 10^4$ features.

[5] For any $\alpha \in (0,1)$, it holds true that $\mathbb{P}(|A-B| \geq \varepsilon) \leq \mathbb{P}(|A-C|+|C-B| \geq \varepsilon) \leq \mathbb{P}(|A-C| \geq \varepsilon\,\alpha) + \mathbb{P}(|C-B| \geq \varepsilon\,(1-\alpha))$. By substituting $\alpha = \frac{1}{2}$, $A = \Delta(\mathbf{W}_1)$, $B = \Delta(\mathbf{W}_2)$ and $C = d_P(g_1,g_2)^2$, we obtain (16).

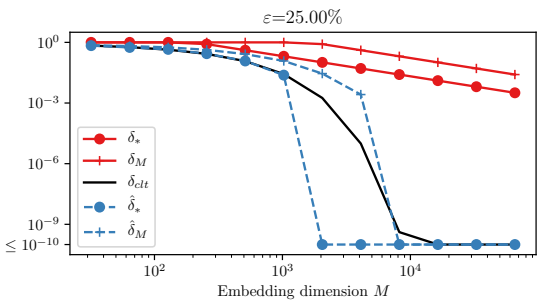

Figure 6. Empirical verification of Theorem 2. δ_M, δ_* and δ_{clt} defined as in Eq. 16, 17 and 18, respectively. $\hat{\delta}_M$ and $\hat{\delta}_*$ denote the empirical computation of the left-hand side in Eq. 16 and 17, respectively. Value of ε is set to the 25% of the squared graph distance computed with M_*.

8.2. GRNF as a Layer of a Neural Net

In this experiment we consider a graph network composed of GRNF as first, untrained layer of a graph neural network and we provide empirical evidence that our proposal is in line with current state-of-the-art performance in graph classification.

To this purpose, we considered the benchmark setup provided by Errica et al. (2020) with data sets from chemical and social domains. Specifically, we considered NCI1, PROTEINS, ENZYMES, IMDB-BINARY, IMDB-MULTI and COLLAB, all available to the public (Kersting et al., 2016) and commonly used for benchmarking. The major differences between the two categories of graph are that graphs of chemical compounds come with node attributes, while the social graphs have generally higher edge density.

Our model combines a GRNF (untrained) layer, with a linear one followed by an output layer to perform classification tasks. All intermediate layers have the rectified linear unit function $x \mapsto \max\{0, x\}$ as activation function. We build features with $k = 1, 2$ tensor orders and embedding dimension $M = 512$. In Table 1 we compare our results with those achieved by some of the most common and varied graph networks: DGCNN (Zhang et al., 2018), DiffPool (Ying et al., 2018), ECC (Simonovsky & Komodakis, 2017), GIN (Xu et al., 2019), GraphSAGE (Hamilton et al., 2017)). Finally, we consider a baseline[6] that has no access to the graph topology (Errica et al., 2020).

Noteworthy, the performance of GRNF is comparable with that of the considered methods and, in most of the classification problems, is substantially better then the baseline. This shows that our proposal is in line with the current state of the art and can exploit the topological information of the graphs.

For completeness, in Figure 5 we report the performance of GRNF on ENZYMES and IMDB-BINARY letting the embedding dimension M vary. Here, we observe that we reach a plateau in the performance with smaller embedding dimension than in the SBM and Del data set, hence fewer features were actually sufficient.

[6]We actually have two baseline models: one for PROTEINS and NCI1 which considers molecular fingerprints, and one for ENZYMES, IMDB-BINARY, IMDB-MULTI and COLLAB employing a more generic layer to aggregate node attributes.

9. Conclusions and Future Work

The present paper proposes a graph embedding method that we called Graph Random Neural Features (GRNF). The method provides a way to generate expressive graph representations that preserve, with arbitrary precision, the metric structure of the original graph domain. Moreover, GRNF does not require a training phase; nonetheless, it is possible to search for a distribution P that best suits the data and task at hand. GRNF, besides providing an explicit embedding method for graphs, can be used as layer of a larger graph neural network. Finally, by approximating graph distances and kernels, GRNF can also be used in conjunction with distance- and similarity-based machine learning methods.

GRNF is based on a family $\mathcal{F} = \{\psi(\cdot\,; \mathbf{w}) : \mathcal{G} \to \mathbb{R}\}$ of graph neural networks, parametrized by vector $\mathbf{w} \in \mathcal{W}$, that separates graphs of $\mathcal{G}$. By defining a probability distribution P over $\mathcal{W}$, we can sample and weight the importance of M graph neural features, obtaining the proposed GRNF map $\mathbf{z} : \mathcal{G} \to \mathbb{R}^M$. Our results show that a distance for graphs can be obtained as the expectation of the squared discrepancy $(\psi(g_1; \mathbf{w}) - \psi(g_2; \mathbf{w}))^2$; similarly, $\mathbb{E}_{\mathbf{w}}[\xi(g_1; \mathbf{w})\, \xi(g_2; \mathbf{w})]$ leads to a positive-definite kernel function for graphs. Theorem 1 states that, when $\mathrm{supp}(P) = \mathcal{W}$ the resulting graph distance is a metric: this implies that, in principle, it is possible to distinguish between any pair of non-isomorphic graphs. Secondly, Theorem 2 proves that the Euclidean distance $|\mathbf{z}(g_1) - \mathbf{z}(g_2)|_2$ between embedding vectors $\mathbf{z}(g_1), \mathbf{z}(g_2)$ converges to the actual graph distance $d_P(g_1, g_2)$, and provides a criterion to select an embedding dimension M ensuring to preserve the metric structure of the original graph domain up to a prescribed error and probability.

We believe that investigating more sophisticated approximation methods, like Gaussian quadrature, can bring substantial improvement, especially considering the computational overhead that most of the graph processing methods require.

Acknowledgements

This research is funded by the Swiss National Science Foundation project 200021_172671: "ALPSFORT: A Learning graPh-baSed framework FOr cybeR-physical sysTems." The work of L. Livi was supported by the Canada Research Chairs program.

References

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Chen, Z., Villar, S., Chen, L., and Bruna, J. On the equivalence between graph isomorphism testing and function approximation with GNNs. In *Advances in Neural Information Processing Systems*, pp. 15868–15876, 2019.

Dao, T., De Sa, C. M., and Ré, C. Gaussian quadrature for kernel features. In *Advances in Neural Information Processing Systems*, pp. 6107–6117, 2017.

Elton, D. C., Boukouvalas, Z., Fuge, M. D., and Chung, P. W. Deep learning for molecular design–a review of the state of the art. *Molecular Systems Design & Engineering*, 2019. doi: 10.1039/C9ME00039A.

Errica, F., Podda, M., Bacciu, D., and Micheli, A. A fair comparison of graph neural networks for graph classification. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HygDF6NFPB.

Gärtner, T., Flach, P., and Wrobel, S. On graph kernels: Hardness results and efficient alternatives. In *Learning Theory and Kernel Machines*, pp. 129–143. Springer, 2003.

Grattarola, D., Zambon, D., Livi, L., and Alippi, C. Change detection in graph streams by learning graph embeddings on constant-curvature manifolds. *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2019. ISSN 2162-237X. doi: 10.1109/TNNLS.2019.2927301.

Hamilton, W., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. In *Advances in neural information processing systems*, pp. 1024–1034, 2017.

Holland, P. W., Laskey, K. B., and Leinhardt, S. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983. doi: 10.1016/0378-8733(83)90021-7.

Hornik, K., Stinchcombe, M., and White, H. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

Kar, P. and Karnick, H. Random feature maps for dot product kernels. In *Artificial Intelligence and Statistics*, pp. 583–591, 2012.

Keriven, N. and Peyré, G. Universal invariant and equivariant graph neural networks. In *Advances in Neural Information Processing Systems 32*, pp. 7090–7099. Curran Associates, Inc., 2019.

Kersting, K., Kriege, N. M., Morris, C., Mutzel, P., and Neumann, M. Benchmark data sets for graph kernels, 2016. URL http://graphkernels.cs.tu-dortmund.de.

Li, A., Cornelius, S. P., Liu, Y.-Y., Wang, L., and Barabási, A.-L. The fundamental advantages of temporal networks. *Science*, 358(6366):1042–1046, 2017. doi: 10.1126/science.aai7488.

Li, Z., Ton, J.-F., Oglic, D., and Sejdinovic, D. Towards a unified analysis of random Fourier features. In *International Conference on Machine Learning*, pp. 3905–3914, 2019.

Maaten, L. and Hinton, G. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(Nov.):2579–2605, 2008.

Maron, H., Ben-Hamu, H., Serviansky, H., and Lipman, Y. Provably powerful graph networks. In *Advances in Neural Information Processing Systems 32*, pp. 2153–2164. Curran Associates, Inc., 2019a.

Maron, H., Ben-Hamu, H., Shamir, N., and Lipman, Y. Invariant and equivariant graph networks. In *International Conference on Learning Representations*, 2019b. URL https://openreview.net/forum?id=Syx72jC9tm.

Maron, H., Fetaya, E., Segol, N., and Lipman, Y. On the universality of invariant networks. In *International Conference on Machine Learning*, pp. 4363–4371, 2019c.

Oneto, L., Navarin, N., Donini, M., Sperduti, A., Aiolli, F., and Anguita, D. Measuring the expressivity of graph kernels through statistical learning theory. *Neurocomputing*, 268:4–16, 2017.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

Principe, J. C. and Chen, B. Universal approximation with convex optimization: Gimmick or reality? [discussion forum]. *IEEE Computational Intelligence Magazine*, 10 (2):68–77, 2015.

Rahimi, A. and Recht, B. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, pp. 1177–1184, 2008a.

Rahimi, A. and Recht, B. Uniform approximation of functions with random bases. In *46th Annual Allerton Conference on Communication, Control, and Computing*, pp. 555–561. IEEE, 2008b.

Rahimi, A. and Recht, B. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in Neural Information Processing Systems*, pp. 1313–1320, 2009.

Rudi, A. and Rosasco, L. Generalization properties of learning with random features. In *Advances in Neural Information Processing Systems*, pp. 3215–3225, 2017.

Rudin, W. *Functional Analysis*. McGraw-Hill, 1991.

Sejdinovic, D., Sriperumbudur, B., Gretton, A., and Fukumizu, K. Equivalence of distance-based and RKHS-based statistics in hypothesis testing. *The Annals of Statistics*, pp. 2263–2291, 2013.

Simonovsky, M. and Komodakis, N. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3693–3702, 2017.

Sinha, A. and Duchi, J. C. Learning kernels with random features. In *Advances in Neural Information Processing Systems*, pp. 1298–1306, 2016.

Vedaldi, A. and Zisserman, A. Efficient additive kernels via explicit feature maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3):480–492, 2012.

Wu, L., Yen, I. E.-H., Xu, F., Ravikumar, P., and Witbrock, M. D2ke: From distance to kernel and embedding. *arXiv preprint arXiv:1802.04956*, 2018.

Wu, L., Yen, I. E.-H., Zhang, Z., Xu, K., Zhao, L., Peng, X., Xia, Y., and Aggarwal, C. Scalable global alignment graph kernel using random features: From node embedding to graph embedding. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1418–1428. ACM, 2019.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

Ying, Z., You, J., Morris, C., Ren, X., Hamilton, W., and Leskovec, J. Hierarchical graph representation learning with differentiable pooling. In *Advances in neural information processing systems*, pp. 4800–4810, 2018.

Yu, F. X., Suresh, A. T., Choromanski, K., Holtmann-Rice, D., and Kumar, S. Orthogonal random features. In *Advances in Neural Information Processing Systems*, pp. 1975–1983, 2016.

Zambon, D., Livi, L., and Alippi, C. Detecting changes in sequences of attributed graphs. In *IEEE Symposium Series on Computational Intelligence*, pp. 1–7, Nov. 2017. doi: 10.1109/SSCI.2017.8285273.

Zhang, M., Cui, Z., Neumann, M., and Chen, Y. An end-to-end deep learning architecture for graph classification. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.