Wasserstein Weisfeiler-Lehman Graph Kernels

Matteo Togninalli[*,†], Elisabetta Ghisu[*,†], Felipe-Llinares Lopez[*], Bastian Rieck[*], Karsten Borgwardt[*]
[*]D-BSSE, Machine Learning and Computational Biology Lab, ETH Zurich, Switzerland; [†]Equal Contributions

Introduction

Graph kernels have been very successful in dealing with the complexity of graphs, achieving good predictive performances.

Limitations of existing mehods
(1) Aability to capture complex nonlinear structural characteristics of the graph (Figure 1)
(2) Generalisation to graphs with high-dimensional continuous node attributes.

Our solution
We propose a method that combines the vectorial graph representations with ideas from optimal transport theory [7].

While classical WL-based kernels compress information at the graph level, with simple average or sum, we measure e Wasserstein distance between the distribution of node embeddings.

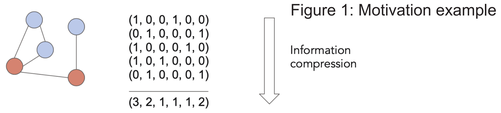


Figure 1: Motivation example

Background

Graph Kernels
Kernels are a class of similarity functions that present attractive properties and can be used for classification and regression in kernelisable machine learning algorithms, such as SVM [5].
Kernels on graphs are generally defined using the R-Convolution framework [3]. The main idea is to decompose the graph G = (V, E) into substructures and define a kernel value k(G, G′) as a combination of their similarities. Most popular existsing approaches rely on subtree-based propagation schemes (WL) [6] or extraction of paths and walk, before performing an aggregation step to obtain the final graphs similarity measure.

Wasserstein Distance
The Wasserstein distance between probability distributions is:

Definition 1 *The L^p-Wasserstein distance for $p \in [1, \infty)$ is defined as*

$$W_p(\sigma, \mu) := \left(\inf_{\gamma \in \Gamma(\sigma,\mu)} \int_{M \times M} d(x,y)^p \, d\gamma(x,y) \right)^{\frac{1}{p}}, \tag{1}$$

where $\Gamma(\sigma, \mu)$ is the set of all transportation plans $\gamma \in \Gamma(\sigma, \mu)$ over $M \times M$ with marginals σ and μ on the first and second factors respectively.

and can be interpreted as the most “inexpensive” way to transport all the probability mass from the distribution σ so as to match the distribution μ .

Method Overview

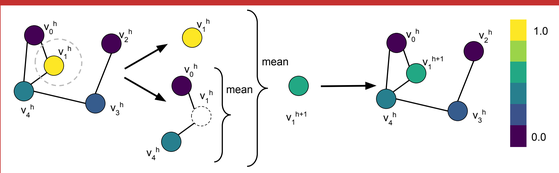


Figure 2: WL continuous embedding scheme

Wasserstein distance on graphs

The main steps of our method can be summarized as follows:

(1) Transform each graph into a set of node embeddings
(2) Measure the Wasserstein distance between each pair of graphs
(3) Compute a similarity matrix to be used in the learning algorithm

First we need to define a scheme to extract node emeddings from a graph, then we define a Wasserstein based distance on graphs to evaluate their simialrity.

Definition 2 (Graph Embedding Scheme) *Given a graph $G = (V, E)$, a graph embedding scheme $f\colon G \to \mathbb{R}^{|V| \times m}$, $f(G) = X_G$ is a function that outputs a fixed-size vectorial representation for each node in the graph. For each $v_i \in V$, the i-th row of X_G is called the node embedding of v_i.*

Definition 3 (Graph Wasserstein Distance) *Given two graphs $G = (V, E)$ and $G' = (V', E')$, a graph embedding scheme $f\colon G \to \mathbb{R}^{|V| \times m}$ and a ground distance $d\colon \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$, we define the Graph Wasserstein Distance (GWD) as*

$$D_W^f(G, G') := W_1(f(G), f(G')). \tag{2}$$

The WL based graph embedding scheme
We propose a WL-based [6] graph embedding scheme that generates node embeddings from the node labels or attributes of the graphs, via a recursive procedure to obtain the features (Figure 2).

Definition 4 (WL Features) *Let $G = (V, E)$ and let H be the number of WL iterations. Then, for every $h \in \{0, \dots, H\}$, we define the WL features as*

$$X_G^h = [x^h(v_1), \dots, x^h(v_{n_G})], \tag{3}$$

where $x^h(\cdot) = \ell^h(\cdot)$ for categorically labelled graphs and $x^h(\cdot) = a^h(\cdot)$ for continuously attributed graphs. We refer to $X_G^h \in \mathbb{R}^{n_G \times m}$ as the node features *of graph G at iteration h. Then, the node embeddings of graph G at iteration H are defined as:*

$$\begin{aligned} f^H &\colon G \to \mathbb{R}^{n_G \times (m(H+1))} \\ &\quad G \mapsto \text{concatenate}(X_G^0, \dots, X_G^H). \end{aligned} \tag{4}$$

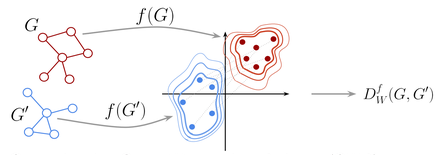


Figure 3: Generating the Graph Wasserstein Distance (GWD)

In the case of continuous attributes a(v), for each node v in G, we use a mean based recursive shceme to update the node embedding, as shown in Figure 2:

$$a^{h+1}(v) = \frac{1}{2} \left(a^h(v) + \frac{1}{\deg(v)} \sum_{u \in \mathcal{N}(v)} w\left((v,u)\right) \cdot a^h(u) \right). \tag{5}$$

Once the node embeddings are generated by the graph embedding scheme, we evaluate the pairwise Wasserstein distance between graphs (Definition 3), by computing the ground distances between each pair of node (Definition 1). This can be either the Hamming or Euclidean,in the case of categorical or continuous labels, respectively.

The process is highlighted in Figure 3.

From Wasserstein DIstance to Kernels

Definition 5 (Wasserstein Weisfeiler-Lehman) *Given a set of graphs $\mathcal{G} = \{G_1, \dots, G_N\}$ and the GWD defined for each pair of graph on their WL embeddings, we define the Wasserstein Weisfeiler-Lehman (WWL) kernel as:*

$$K_{\text{WWL}} = e^{-\lambda D_W^{f_{WL}}}. \tag{6}$$

This is a Laplacian kernel, which offers favourable conditions for positive definiteness in case of non-Euclidean distances [2]. The Wasserstein distance in its general form is not isometric to an L2-norm, therefore it is not necessarily possible toderive a PSD kernel from the Wasserstein. Nevertheless, we can show that, in the setting of categorical node labels, the obtained kernel is PSD.

Theorem 1 *The categorical WWL kernel is positive definite for all $\lambda > 0$.*

By contrast, for the continuous case, establishing the definiteness of the obtained kernel remains an open problem. Therefore, to ensure the theoretical and practical correctness of our results, we employ recently developed methods for learning with indefinite kernels which utilise tools from theory of Krein space [4].

Results and Discussion

Table 1: Classification accuracies on graphs with categorical node labels. Comparison of Weisfeiler--Lehman kernel (WL), optimal assignment kernel (WL-OA), and our method (WWL).

Method	ENZYMES	PROTEINS	IMDB-B	BZR	COX2	BZR-MD	COX2-MD
VH-C	47.15±0.79	60.79±0.12	71.64±0.49	74.82±2.13	48.51±0.63	66.58±0.97	64.89±1.06
RBF-WL	68.43±1.47	75.43±0.28	72.06±0.34	80.96±1.67	75.45±1.53	**69.13±1.27**	71.83±1.61
HGK-WL	63.04±0.65	75.93±0.17	73.12±0.40	78.59±0.63	**78.13±0.45**	**68.94±0.65**	74.61±1.74
HGK-SP	66.36±0.37	75.78±0.17	73.06±0.27	76.42±0.72	72.57±1.18	66.17±1.05	68.52±1.00
GH	65.65±0.80	74.78±0.29	72.35±0.55	76.49±0.99	76.41±1.39	**69.14±2.08**	66.20±1.05
WWL	**73.25±0.87***	**77.91±0.80***	**74.37±0.83***	**84.42±2.03***	**78.29±0.47**	**69.76±0.94**	**76.33±1.02**

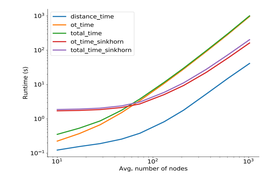
We compare WWL with state-of-the-art graph kernel and relevant baselines, all trained on the same splits. As a classifier we use an SVM (or a KSVM [4] for WWL) and then report the average accuracy of a 10-fold cross-validation, repeting each step 10 times.

Results
Tables 1 and 2 show that WWL outperforms the competitors in most of the datasets. Evaluating the runtime of our approach we observe that it can benefit from fast approximations [1]

Method	Rank
WWL	1
HGK-WL	2.86
RBF-WL	3.29
HGK-SP	4.14
VH-C	5.86

Table 2: Ranking of methods

Figure 2: Runtime



Conclusions
Our experiments show that WWL graph kernels constitute the new state of the art for graph classification in the scenario of continuous node attributes. We see a great potential in multiple lines of research for future work, including: runtime improvements; theoretical bound on the positive definiteness of continuos WWL; extensions to graph neural networks.

References

[1] J. Altschuler, J. Weed, and P. Rigollet. Near-linear time approximation algorithms for optimal transport via sinkhorn iteration. NeurIPS, 2017.
[2] A. Feragen, F.L., S.H. Geodesic exponential kernels: When curvature and linearity conflict IEEE Conference on Computer Vision and Pattern Recognition, 2015
[3] D. Haussler. Convolution kernels on discrete structures. Technical report, University of California, 1999.
[4] G. Loosli, S. Canu, C. S. Ong. Learning SVM in Krein spaces. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2015.
[5] B. Schölkopf and A. J. Smola. Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT press, 2002
[6] N. Shervashidze, P. Schweitzer, E.J. v.Leeuwen, K. Mehlhorn, and K.M. Borgwardt. Weisfeiler-Lehman graph kernels. JMLR, 2011
[7] C. Villani. Optimal transport: old and new. Number 338 in Grundlehren der mathematischen Wissenschaften. Springer, 2008