Graph Filtration Kernels

Till Schulz[1], **Pascal Welke**[1], **Stefan Wrobel**[1,2,3]
[1] Universität Bonn, Germany
[2] Fraunhofer IAIS, Sankt Augustin, Germany
[3] Fraunhofer Center for Machine Learning, Sankt Augustin, Germany
{schulzth, welke, wrobel}@cs.uni-bonn.de

Abstract

The majority of popular graph kernels is based on the concept of Haussler's $\mathcal{R}$-convolution kernel and defines graph similarities in terms of mutual substructures. In this work, we enrich these similarity measures by considering graph filtrations: Using meaningful orders on the set of edges, which allow to construct a sequence of nested graphs, we can consider a graph at *multiple granularities*. A key concept of our approach is to track graph features over the course of such graph resolutions. Rather than to simply compare frequencies of features in graphs, this allows for their comparison in terms of *when* and for *how long* they exist in the sequences. In this work, we propose a family of graph kernels that incorporate these existence intervals of features. While our approach can be applied to arbitrary graph features, we particularly highlight Weisfeiler-Lehman vertex labels, leading to efficient kernels. We show that using Weisfeiler-Lehman labels over certain filtrations strictly increases the expressive power over the ordinary Weisfeiler-Lehman procedure in terms of deciding graph isomorphism. In fact, this result directly yields more powerful graph kernels based on such features and has implications to graph neural networks due to their close relationship to the Weisfeiler-Lehman method. We empirically validate the expressive power of our graph kernels and show significant improvements over state-of-the-art graph kernels in terms of predictive performance on various real-world benchmark datasets.

Introduction

Graph-structured data is prevalent in countless domains such as social networks, biological interaction graphs and molecules. A central task on this kind of data is the classification of graphs. Perhaps the most established machine learning methods for graph classification are based on graph kernels which, even in the advent of neural network approaches, remain highly relevant due to their remarkable predictive performance (Kriege, Johansson, and Morris 2020). The majority of graph kernels are instances of Haussler's $\mathcal{R}$-convolution kernel (Haussler 1999) which define graph similarity in terms of pairwise similarities between the graphs' substructures. Some well-known representatives are the Weisfeiler-Lehman graph kernels (Shervashidze et al. 2011), the shortest-path kernel (Borgwardt and Kriegel 2005) and the cyclic pattern kernel (Horváth, Gärtner, and Wrobel 2004). Generally, such kernels revert to simply comparing counts of mutual substructures. We refer to these kernels as *histogram kernels*. While they prove to be successful in many classification tasks, their notion of comparing features is often too rigid.

Motivated by this limitation, we introduce a family of graph kernels which regard a graph at multiple levels of resolution. This is realized using the concept of graph filtrations, which define sequences of nested subgraphs that differ only in the sets of edges. Such a sequence can be viewed as incremental refinements that construct a graph by gradually adding sets of edges. An example depicting this concept is found in Fig. 1. Clearly, with changing sets of edges, the graph features change as well. That is, features occurring at some point in the sequence may disappear at a later moment. In this work, we track such feature existence intervals which allows for a comparison not only in terms of feature frequency but also by *when* and for *how long* they exist. This comparison of feature occurrence distributions is realized using the Wasserstein distance which, using recent results, allows for proper kernel functions on this kind of information. A benefit of this approach is the kernels' ability to handle continuous edge attributes.

Our main contributions are summarized as follows:

1. We introduce a general graph kernel framework which defines similarity by comparing graph feature occurrence distributions over sequences of graph resolutions and show that such kernels generalize histogram kernels.
2. We particularly consider the well-known Weisfeiler-Lehman subtree features and show that there exist filtration kernels using such features which *strictly increase the expressive power* over the ordinary Weisfeiler-Lehman subtree kernel.
3. We empirically validate our theoretical findings on the expressive power of our kernels and furthermore provide experiments on real-world benchmark datasets which show a favorable performance of our approach compared to state-of-the-art graph kernels.

Background

Graph kernels. *Kernels* are functions of the form $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ which define similarity measures on non-empty

Copyright © 2022, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

sets $\mathcal{X}$. More precisely, k is a kernel if there exists a mapping $\varphi : \mathcal{X} \to \mathcal{H}_k$ with $k(x, y) = \langle \varphi(x), \varphi(y) \rangle$ where $\langle \cdot, \cdot \rangle$ is the inner product in the associated Hilbert space $\mathcal{H}_k$. With Haussler's work on *convolution kernels* over discrete structures (Haussler 1999), kernel methods became widely applicable on graphs. The concept of $\mathcal{R}$-convolution kernels provides a general framework which can be used to construct graph kernels by defining graph similarity in terms of their aggregated substructure similarities. More formally, let there be a function decomposing the graph G into the set of its substructures $\mathcal{X}_G$. Then, for graphs G, G', the (simplified) $\mathcal{R}$-convolution kernel is defined by

$$k(G, G') = \sum_{(x,y) \in \mathcal{X}_G \times \mathcal{X}_{G'}} \kappa(x, y) \tag{1}$$

where $\kappa(x, y)$ is a kernel on the substructures. To guarantee convergence, we assume $\mathcal{X}_G$ to be finite for each G. As this decomposition into graph substructures can be perceived as a feature extraction process, we will commonly refer to such substructures as *features*. In a majority of graph kernels the function κ is simply the Dirac delta which amounts to 1 if the features x and y are equivalent and 0 otherwise.[1] Thus, $k(G, G')$ essentially measures graph similarity by counting pairs of equivalent features. Some well-known examples are the Weisfeiler-Lehman subtree kernel (Shervashidze et al. 2011) and cyclic pattern kernel (Horváth, Gärtner, and Wrobel 2004). Such kernels can alternatively be described by the inner product of explicit feature vectors

$$\varphi(G) = [c(f_1(G)), c(f_2(G)), \ldots] \tag{2}$$

where $c(f_i(G))$ indicates the count of feature $f_i \in \mathcal{F}$ in graph G over some fixed feature domain $\mathcal{F}$. We refer to this kind of kernels as *histogram* kernels. Note that histogram kernels can equivalently be expressed as a sum of feature frequency products, i.e.,

$$K_H^{\mathcal{F}}(G, G) = \sum_{f \in \mathcal{F}} c(f(G))\, c(f(G')) \ . \tag{3}$$

Wasserstein distance. The Wasserstein distance is a distance function between probability distributions based on the concept of optimal mass transportation. In the following, we specifically consider the 1-Wasserstein distance for discrete distributions, i.e., histograms, and refer to it as the Wasserstein distance. For more general definitions see e.g. Peyré and Cuturi (2019). Intuitively, the Wasserstein distance can be viewed as the minimum cost necessary to transform one pile of earth into another. It is, therefore, also known as the *earth movers* distance or *optimal transportation* distance. More formally, given two vectors $x \in \mathbb{R}^n$ and $x' \in \mathbb{R}^{n'}$ with $\|x\|_1 = \|x'\|_1$ and a cost matrix $C_d \in \mathbb{R}^{n \times n'}$ containing pairwise distances between entries of x and x', the *Wasserstein distance* is defined by

$$\mathcal{W}_d(x, x') = \min_{T \in \mathcal{T}(x, x')} \langle T, C_d \rangle \tag{4}$$

with $\mathcal{T}(x, x') \subseteq \mathbb{R}^{n \times n'}$ and $T\mathbf{1}_{n'} = x$, $\mathbf{1}_n^\top T = x'$ for all $T \in \mathcal{T}(x, x')$, where $\langle \cdot, \cdot \rangle$ is the Frobenius inner product. The function d defining the costs in C_d between entries of x and x' is called *ground distance*. If the ground distance is a metric, then the Wasserstein distance is a metric (Peyré and Cuturi 2019, Sec. 2.4).

[1] We note that kernels generally allow an individual weighting of features. However for reasons of simplicity, we omit this aspect in the further discussions.

From Filtrations to Distances

We now outline the concept of tracking features over sequences of different graph resolutions and define distance measures between graphs using this kind of information.

Feature Persistence

The general idea of *feature persistence* in graphs is derived from persistent homology, which refers to a method in computational topology that aims at measuring topological features at various levels of resolution (Edelsbrunner and Harer 2010). Persistent homology is applied in a wide range of topological data analysis tasks and has recently become a popular tool for analyzing topological properties in graphs (Aktas, Akbas, and Fatmaoui 2019). In this work, we adopt some of its basic concepts and specifically fit them to describe the idea of feature persistence.

Intuitively speaking, feature persistence tracks the lifespans of graph features in evolving graphs. That is, it records the intervals during which occurrences of a specific feature appear in sequentially constructed graphs. Such a graph sequence is defined by a graph filtration which is essentially an ordered graph refinement that constructs a graph by gradually adding sets of edges. More precisely, given a graph $G = (V, E)$, a *graph filtration* $\mathcal{A}(G)$ is a sequence of graphs

$$G_1 \subseteq G_2 \subseteq \ldots \subseteq G_k = G \ , \tag{5}$$

where $\subseteq$ denotes the subgraph relation and $G_i = (V, E_i \subseteq E)$ is called a *filtration graph* of G. Hence, filtration graphs differ only in the sets of edges and describe a sequence in which the last element is the graph G itself. Without loss of generality, we assume G to be edge-weighted by a function $w : E(G) \to \mathbb{R}_+$ such that the filtration $\mathcal{A}(G)$ is implicated by a sequence of decreasing values

$$\alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_k = 0 \ , \tag{6}$$

where G_i is induced by the set of edges with weights greater or equal α_i, i.e., $G_i = (V, \{e \in E : \ w(e) \geq \alpha_i\})$. Thus, $\mathcal{A}$ is determined by the set of values $\mathcal{A}_\alpha = \{\alpha_1, \ldots, \alpha_k\}$. Furthermore, isomorphic graphs G, G' (considering edge weights) generate equivalent filtrations $\mathcal{A}(G) \equiv \mathcal{A}(G')$.

While traditional persistent homology tracks lifespans of topological features such as connected components and cycles, our notion of feature persistence is concerned with arbitrary graph features. More precisely, given some feature of interest f (e.g., Weisfeiler-Lehman vertex colors, c.f. Fig. 1) and a graph filtration $\mathcal{A}(G)$, feature persistence describes the set of occurrence intervals of f over the sequence $\mathcal{A}(G)$. This concept can very intuitively be depicted using (discrete) persistence barcodes shown in Fig. 1. Each bar in the barcode diagram corresponds to the lifespan of a feature occurrence. Furthermore, considering a graph at different levels of resolutions provides access to features not necessarily present in the graph G itself.

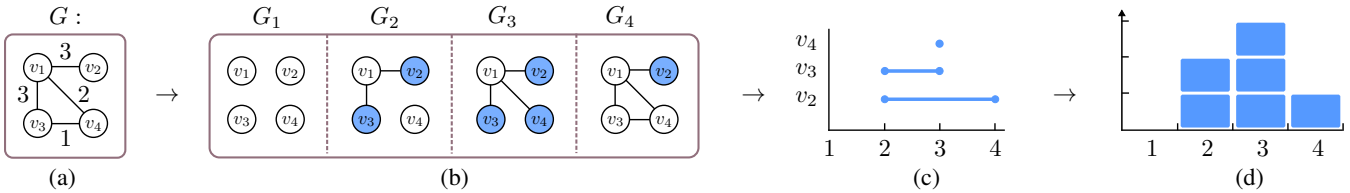

Figure 1: Consider the simple Weisfeiler-Lehman label f corresponding to a vertex having exactly one neighbor. Each occurrence of this feature is individually marked in the filtration graphs $\mathcal{A}(G)$ shown in (b). The barcode in (c) depicts the existence intervals of each such feature occurrence. This information is then aggregated into a filtration histogram $\phi_f^{\mathcal{A}}(G)$ in (d).

Wasserstein Distance on Filtration Histograms

The majority of traditional graph kernels defines the similarity measure in terms of the number of mutual substructures. This comes down to simply comparing frequencies of features. Using the concept of feature persistence, we are able to define much finer similarity measures on graphs. We achieve this by defining a distance function on histograms which aggregate lifespans of feature occurrences. This distance measure compares graphs in terms of *when* and *for how long* a certain feature appears in the filtration. The underlying intuition is that features occurring close to each other in the filtration sequence indicate a higher similarity than those lying farther apart. A natural choice for this distance function is the Wasserstein distance as the aggregated feature occurrence lifespans directly translate into 1-dimensional distributions (see Fig. 1).

To define a distance measure w.r.t. *a single* feature f on graphs G, G', we aggregate the feature persistence information of G and G' into a histogram. This process is visualized in Fig. 1. Such histograms essentially accumulate all feature lifespans of a particular feature and hence reflect the number of feature occurrences in each filtration graph.[2]

Definition 1 (Filtration Histogram)**.** *Given a graph G together with a length-k filtration $\mathcal{A}(G)$ and a feature f, the function $\phi_f^{\mathcal{A}} : G \to \mathbb{R}^k$ maps G to its* filtration histogram *which counts occurrences of f in each filtration graph of G.*

In the remainder of this work, we often omit the filtration function $\mathcal{A}$ in the notations if it is either irrelevant or clear from the context.

Using filtration histograms allows for the application of natural distance measures such as the Wasserstein distance. Intuitively speaking, the Wasserstein distance between such histograms describes the cost of shifting (accumulated) feature lifespans into another.

Definition 2 (Filtration Histogram Distance)**.** *Given graphs G, G', a filtration histogram mapping $\phi_f^{\mathcal{A}} : \mathcal{G} \to \mathbb{R}^k$ together with a distance function $d : \mathcal{A}_\alpha \times \mathcal{A}_\alpha \to \mathbb{R}$ on associated values $\mathcal{A}_\alpha = \{\alpha_1, \ldots, \alpha_k\}$, the* filtration histogram distance *is given by*

$$\mathcal{W}_d(\phi_f^{\mathcal{A}}(G), \phi_f^{\mathcal{A}}(G')) \quad . \tag{7}$$

The ground distance d of the Wasserstein distance defines how feature occurrences at different points in the filtrations are being compared to each other. Since values in $\mathcal{A}_\alpha$ can be viewed as points on the timeline $[\alpha_1, \alpha_k]$, a natural choice for this distance is the simple euclidean metric

$$d^1(\alpha_i, \alpha_j) = |\alpha_i - \alpha_j| \tag{8}$$

on elements $\alpha_i, \alpha_j \in \mathcal{A}_\alpha$. Furthermore, while the Wasserstein distance has cubic time complexity in the length of filtrations in general, this reduces to a *linear* time complexity when employing d^1 on the real line as ground distance (Peyré and Cuturi 2019, Rem. 2.30).

Graph Filtration Kernels

Recent results in optimal transport theory give rise to proper kernel functions using the above filtration histogram distances when equipped with a suitable ground distance function (Le et al. 2019). In fact, it can be shown that utilizing the Euclidean ground distance d^1 yields positive semi-definite kernels. These kernels serve as building blocks for our final filtration graph kernels. More precisely, we construct graph kernels by combining multiple base kernels k_f over a set of features $f \in \mathcal{F}$. Each such base kernel is concerned with a single feature and defines a similarity between graphs G and G' w.r.t. this particular feature. While the number of features may potentially be infinite, we can show that it suffices to consider only such features which appear in *both* graphs.

Since filtration histograms are not necessarily of equivalent mass, a normalization is necessary to fit the requirement of general Wasserstein distances. We denote such mass-normalized filtration histograms using function $\hat{\phi}_f$. The base kernel on graphs G, G' w.r.t. feature f (and parameter $\gamma \in \mathbb{R}_+$) is then defined over normalized histograms as follows

$$k_f^{\mathcal{A}}(G, G') = e^{-\gamma \mathcal{W}_{d^1}(\hat{\phi}_f^{\mathcal{A}}(G), \hat{\phi}_f^{\mathcal{A}}(G'))} \quad . \tag{9}$$

Filtration kernels can be constructed in form of linear combinations of base kernels. That is, they are sums of kernels k_f over features $f \in \mathcal{F}$. Note that the mass-normalization of filtration histograms results in a loss of information on the number of occurrences of features. This frequency information is often quite crucial. Thus, by introducing weights corresponding to the original masses of histograms, this information loss can be in part reverted. That

[2] Note that aggregating all feature occurrence intervals into a single histogram clearly loses information on the distribution of the individual lifespans. However, while a pairwise comparison of persistence intervals has been shown to lead to valid kernels in the context of persistent homology (Reininghaus et al. 2015), our approach relies on a single histogram representation, which we show leads to very powerful kernel functions, nevertheless.

is, we weigh each base kernel $k_f(G, G')$ using the original histogram masses of $\phi_f(G)$ and $\phi_f(G')$, i.e., $\|\phi_f(G)\|_1$ resp. $\|\phi_f(G')\|_1$.

Definition 3 (Filtration Kernel)**.** *Given graphs G, G', a filtration function $\mathcal{A}$, and a set of features $\mathcal{F}$ the* filtration kernel *is given by*

$$K_{Filt}^{\mathcal{F},\mathcal{A}}(G, G') = \sum_{f \in \mathcal{F}} k_f^{\mathcal{A}}(G, G') \ \left\|\phi_f^{\mathcal{A}}(G)\right\|_1 \left\|\phi_f^{\mathcal{A}}(G')\right\|_1 \tag{10}$$

Notice the special case that a feature f does not appear in the filtration sequence of G. Hence, the corresponding histogram has zero mass and the filtration histogram distance to some non-zero mass histogram is not properly defined. This issue can formally be solved by introducing a dummy histogram entry. We note that since the kernel $k_f(G, G')$ is multiplied by the histogram mass $\|\phi_f(G)\|_1$, which in this particular case amounts to 0, this is only a formal issue and becomes non-relevant in the kernel computation. From a similar argument it follows that only the features $f \in \mathcal{F}$ which appear in both graphs G and G' positively contribute to the similarity measure. The proofs for all following claims can be found in the extended article version (Schulz, Welke, and Wrobel 2021).

Theorem 1. *The Filtration Kernel $K_{Filt}^{\mathcal{F}}$ is positive semi-definite.*

The filtration kernel is closely related to the histogram kernel (c.f. Eq. 3). In fact, the histogram kernel is a special case of $K_{\text{Filt}}^{\mathcal{F}}$.

Proposition 1. *For filtrations of length $k = 1$ (i.e., filtrations consist of only the graph itself), the filtration kernel $K_{Filt}^{\mathcal{F}}$ reduces to the histogram kernel $K_H^{\mathcal{F}}$.*

Filtration kernels can alternatively be defined as *products* over base kernels. In fact, it can be shown that such kernels generalize the radial basis function kernel.

A Kernel Instance Using Weisfeiler-Lehman Subtree Features

We now discuss a concrete instance of the filtration kernel family which uses the well-known Weisfeiler-Lehman (WL) subtree features. We briefly recap the WL relabeling method, formulate the Weisfeiler-Lehman subtree filtration kernel and show that the kernel can in fact be computed in linear time. We furthermore show that considering WL labels over certain filtrations increases the expressiveness over the ordinary Weisfeiler-Lehman procedure in terms of distinguishing non-isomorphic graphs.

The Weisfeiler-Lehman Method

The key idea of the *Weisfeiler-Lehman* procedure is to iteratively aggregate neighborhoods by compressing each node's labels and that of its neighbors into a new label. This compression is done by first concatenating a node's label and its ordered (multi-)set of neighbor labels and subsequently hashing this string to a new label using a perfect hash function $f_\#$. Thus, with each iteration, a label incorporates increasingly large neighborhoods. More precisely, let $G = (V, E, \ell_0)$ be a graph with initial vertex label function $\ell_0 : V \to \Sigma_0$, where Σ_0 is the alphabet of original vertex labels. Assuming that there is a total order on alphabet Σ_i for all $i \geq 0$, the Weisfeiler-Lehman procedure recursively computes the new label of a node v in iteration $i + 1$ by

$$\ell_{i+1}(v) = f_\#(\ell_i(v), [\ell_i(u) : u \in \mathcal{N}(v)]) \in \Sigma_{i+1} \tag{11}$$

where the list of the neighbors' labels is sorted according to the total order on Σ_i.

Shervashidze et al. (2011) employed the Weisfeiler-Lehman method to define a family of graph kernels of which the *subtree kernel* is perhaps the most popular member. For two graphs G, G', the Weisfeiler-Lehman subtree kernel K_{WL} essentially counts all pairs of mutual node labels. This can be expressed as a histogram kernel on the combined label sets $\mathcal{F}_{WL} = \bigcup_{i \in [h]} \Sigma_i$, i.e.,

$$K_{WL}(G, G') = \sum_{l \in \mathcal{F}_{WL}} c(l(G))\, c(l(G')) \ , \tag{12}$$

where $c(l(G))$ is the number of appearences of label l in G and h is the depth parameter.

The Weisfeiler-Lehman Isomorphism Test The Weisfeiler-Lehman method was originally designed to decide isomorphism between graphs. Two graphs G, G' are not isomorphic if the corresponding multisets $\{\!\{\ell_i(v) : v \in V(G)\}\!\}$ and $\{\!\{\ell_i(v') : v' \in V(G')\}\!\}$ differ for some $i \in \mathbb{N}$; otherwise they may or may not be isomorphic. However, G and G' are isomorphic with high probability if the multisets are equal (Babai and Kucera 1979).

The Weisfeiler-Lehman Subtree Filtration Kernel

Recall that filtrations describe an order in which edges are successively added until the final graph is obtained. During this sequence, neighborhoods of vertices evolve and with them their Weisfeiler-Lehman labels change. By also considering WL labels appearing only in filtration graphs, such labels allow to consider *partial* neighborhoods which contribute to a *finer* similarity measure. Fig. 1 depicts an example of a filtration where appearances of a specific WL feature are being tracked. Plugging the WL features $\mathcal{F}_{WL}$ into Eq. 10 yields the *Weisfeiler-Lehman filtration kernel* $K_{\text{Filt}}^{\mathcal{F}_{WL}}$.

Theorem 2. *The Weisfeiler-Lehman filtration kernel $K_{Filt}^{\mathcal{F}_{WL}}(G, G')$ on graphs G, G' can be computed in time $O(hkm)$, where h is the WL depth parameter, k is the filtration length and m denotes the number of edges.*

Thus, the kernel increases the complexity of the ordinary Weisfeiler-Lehman subtree kernel merely by the factor k, i.e., the length of the filtration.

On the Expressive Power of Weisfeiler-Lehman Filtration Kernels

We now show that tracking WL features over filtrations yields powerful methods in terms of expressiveness. The *expressive power* of a method describes its ability to distinguish non-isomorphic graphs; i.e., method A is said to be "more expressive" or "more powerful" than method B

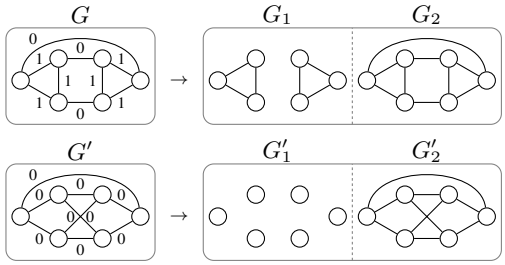

Figure 2: Consider the unlabeled 3-regular graphs G, G' which cannot be distinguished using the WL isomorphism test. By adding edge weights corresponding to the number of triangles that each edge is part of yields filtration graphs G_1, G'_1, which can now be distinguished by the WL isomorphism test. Note that the same filtration is generated using the edge weight function w_w^2 (see experiments).

if $A(G) = A(G') \Rightarrow B(G) = B(G')$ and there exist non-isomorphic graphs G, G' such that $A(G) \neq A(G')$ and $B(G) = B(G')$. Recall that the (1-dimensional) WL test for isomorphism is known to be inexact even after n iterations (Cai, Fürer, and Immerman 1992). However, considering *WL labels over certain filtrations* strictly increases the expressive power.

Theorem 3. *There exists a filtration function $\mathcal{A}$ such that $\phi_f^{\mathcal{A}}(G) = \phi_f^{\mathcal{A}}(G')$ for all $f \in \mathcal{F}_{WL}$ if and only if G and G' are isomorphic.*

In fact, it holds that already for depth-1 WL labels (i.e. $h = 1$), there are filtrations that *correctly* decide isomorphism for *all* graphs. As a first implication, the Weisfeiler-Lehman subtree filtration kernel is – given a suitable filtration – strictly more expressive than the ordinary WL subtree kernel on the original graphs.

Corollary 1. *There exists a filtration function $\mathcal{A}$ such that the kernel $K_{Filt}^{\mathcal{F}_{WL},\mathcal{A}}$ is* complete*.*

A kernel is called *complete* if its feature map φ satisfies $\varphi(G) = \varphi(G') \Leftrightarrow G, G'$ are isomorphic (Gärtner, Flach, and Wrobel 2003). While such a filtration is not known to be efficiently computable, there are efficiently computable filtrations that result in strictly more expressive (but incomplete) WL filtration kernels when compared to the ordinary WL subtree kernel. As an example of such an efficiently computable filtration, consider the function that annotates edges by the number of triangles they belong to. This measure can be computed in polynomial time. Figure 2 shows two 3-regular graphs that can be distinguished using this filtration.[3] This concept can be extended to larger (or multiple) subgraphs, which allows for more expressive filtrations, similar to the approach of Barceló et al. (2021).

Recently, there has been a large body of work that relates the expressive power of certain graph neural networks to that of the Weisfeiler-Lehman isomorphism test (Xu et al. 2019; Morris et al. 2019). While this is orthogonal to our work, Theorem 3 implies that a (neural) ensemble of graph neural networks over a suitably chosen filtration is strictly more powerful than a graph neural network on the original graph(s) alone:

Corollary 2. *There exists a filtration function $\mathcal{A}$ and GNN (Xu et al. 2019) $\mathcal{N}$ such that $\mathcal{N}$ can distinguish any two non-isomorphic graphs when provided with the filtration graphs corresponding to $\mathcal{A}$.*

Related Work

Graph Kernels Over the past twenty years there has been a multitude of works constructing kernels for graph structured data, with many well known examples within the $\mathcal{R}$-convolution framework (Haussler 1999; Gärtner, Flach, and Wrobel 2003; Borgwardt and Kriegel 2005; Shervashidze et al. 2009, 2011; Horváth, Gärtner, and Wrobel 2004). We refer to Borgwardt et al. (2020); Kriege, Johansson, and Morris (2020) for recent surveys. The seminal work by Shervashidze et al. (2011) introduce Weisfeiler-Lehman labels as a means to define graph kernels. Kriege, Giscard, and Wilson (2016) propose a discrete optimal assignment graph kernel based on vertex kernels obtained from the hierarchy of their WL labels. Togninalli et al. (2019) extend this idea and allow fractional assignments using Wasserstein distances. Rieck, Bock, and Borgwardt (2019) introduce a method which individually weighs WL label occurences by their persistent homology information. The difference to our approach is that we consider arbitrary graph features. Furthermore, in the particular case of Weisfeiler-Lehman features, we allow additional WL features appearing only in filtration graphs. Nikolentzos et al. (2018) compute k-core decompositions and compare graphs using the associated hierarchy of subgraphs. While conceptually similar to our approach, the method is limited to nested subgraph chains generated by k-cores. Moreover, the authors revert to applying existing graph kernels to the graphs' k-cores while we introduce a family of novel graph kernels. In contemporaneous work, which we became aware of after submission, O'Bray, Rieck, and Borgwardt (2021) consider filtration curves as graph representations and compare them via random forests. Filtration curves describe the averaged curves of feature counts over graph filtration sequences. In particular, the authors considered node labels and connected components as feature types for labeled, resp. unlabeled, graphs.

Expressiveness In pioneer work, Gärtner, Flach, and Wrobel (2003) have shown that *complete* graph kernels, i.e., kernels that map isomorphism classes to distinct points in some Hilbert space, are at least as hard to compute as graph isomorphism. While not complete, the Weisfeiler-Lehman subtree kernel maps graphs that can be distinguished by the WL isomorphism test to different feature vectors and is hence equally expressive. Recently, Xu et al. (2019) and Morris et al. (2019); Morris, Rattan, and Mutzel (2020); Morris, Fey, and Kriege (2021) have investigated the connection between the expressive power of graph neural networks and the WL isomorphism test. Briefly speaking, if the

[3] It is obvious that using the proposed weights as edge labels allows the Weisfeiler-Lehman isomorphism test to distinguish these two edge labeled graphs. However, it can be shown that it suffices to consider the WL labels on the unlabeled filtration graphs.

(k-dimensional) Weisfeiler-Lehman isomorphism test cannot distinguish two graphs then (higher order) graph neural networks cannot distinguish these graphs, either, and vice versa. Recently, however, there has been significant work in extending the expressive power of graph neural networks by adding auxiliary information such as subgraph counts (Barceló et al. 2021) or distance encodings (Li et al. 2020).

Experimental Evaluation

In this section, we evaluate the predictive performance of our Weisfeiler-Lehman filtration kernel[4] introduced above and compare it to several state-of-the-art graph kernels. We demonstrate that our method significantly outperforms its competitors on several real-world benchmark datasets.

Experimental Setup & Datasets We compare our method to a variety of state-of-the-art graph kernels and include a simple baseline method to put the results into perspective. The experiments are conducted on the well-established molecular datasets DHFR, NCI1 and PTC-MR (obtained from Morris et al. 2020) as well as the large network benchmark datasets IMDB-BINARY (obtained from Morris et al. 2020) and EGO-1 to EGO-4.[4] This selection contains only such datasets on which a simple histogram baseline kernel was outperformed by more sophisticated graph kernels. We measure the accuracies obtained by support vector machines (SVM) using a 10-fold stratified cross-validation. A grid search over sets of kernel specific parameters is used for optimal training. We perform 10 such cross-validations and report the mean and standard deviation.

Filtration Variants

A unique parameter of our kernels is the graph filtration. If such a filtration is not provided through expert knowledge or via available edge weights, one can be generated based on the data at hand. Recall that a filtration is induced by the value sequence $\alpha_1 \geq \ldots \geq \alpha_k$ (c.f. Eq. 6). While there exist infinitely many such sequences, in the following experiments, we govern only its length (i.e. k) and generate the α_is according to some fixed process. In this work, we implement this process by first ordering and subsequently partitioning the (multi)-set of edge weights that appear in a dataset using a simple k-means clustering. We choose α_i as the minimum within the i-th cluster. Thus, the i-th filtration graph G_i contains edges with edge weight at least α_i.

As the above datasets are not explicitly equipped with edge weights, we consider two exemplary edge weight functions w_d and w_w^λ that each assign weights to edges according to the edges' structural relevance w.r.t. a specific property. The first function w_d assigns an edge $e = \{u, v\}$ a weight equal to the maximum degree of its incident vertices, i.e., $w_d(e) = max(deg(u), deg(v))$. Thus, edges with incident vertices of high degree appear early in the graph filtrations. The second function w_w^λ considers the number of walks between adjacent vertices. That is, for edge $e = \{u, v\}$, $w_w^\lambda(e)$ is the number of walks of length at most λ from u to v.

[4]Available at https://github.com/mlai-bonn/wl-filtration-kernel

In the following, the Weisfeiler-Lehman filtration kernel applied on graphs with edge weights calculated according to w_d and w_w^λ is referred to as FWL-D, resp. FWL-W.

Real-World Benchmarks

Figure 3 compares the predictive performances of the filtration kernel to various state-of-the-art graph kernels on a range of real-world benchmark datasets. On datasets NHFR, NCI1, PTC-MR and IMDB-BINARY, there are only small discrepancies between our method and the best performing competitor kernels. In fact, with PTC-MR being an exception, the results of all kernels using Weisfeiler-Lehmann subtree features are virtually indistinguishable. This changes for the EGO datasets. While on EGO-1, our FWL-D variant is second only to the shortest-path kernel, it significantly outperforms all tested kernels on EGO-2, EGO-3 and EGO-4, amounting to a roughly 10% accuracy increase for the case of the latter dataset. It becomes apparent from the results of the FWL-W variant that our approach is particularly reliant on the provided edge weights. However, the evaluation also suggests that in case the data at hand is not equipped by meaningful edge weights, very simple edge weight functions already suffice to significantly increase the predictive performance over state-of-the-art graph kernels.

The Influence of the Filtration Length k

As central aspect of our approach, the filtration clearly plays a critical role in practical applications. In order to investigate the influence of filtrations, we provide experiments for varying choices of k, i.e., the filtration length. Figure 4 shows results on the EGO datasets for values $k \in \{1, \ldots, 10\}$ using the FWL-D variant. Recall that the case $k = 1$ is equivalent to the ordinary Weisfeiler-Lehman approach. Note that already for $k = 2$ the predictive performance significantly improves over $k = 1$ in all cases and that all datasets reach their accuracy peak at only $k = 2$ or $k = 3$. Since the value k linearly influences the kernel's complexity, our approach improves the predictive performance over the ordinary WL subtree kernel at only a small additional computational cost.

Synthetic Benchmarks: Circular Skip Link Graphs

In this section, we investigate the discriminate power of graph filtration kernels. In particular, we consider a benchmark setup as discussed in Murphy et al. (2019) which classifies *circular skip link* (CSL) graphs. A CSL graph $G_{n,s}$ is a 4-regular graph where the n nodes form a cycle and all pairs of nodes with path distance s on that cycle are furthermore connected by an edge. Examples for CSL graphs are depicted in Fig. 5. Following Murphy et al. (2019), for $n = 41$ and $s \in \{2, 3, 4, 5, 6, 9, 11, 12, 13, 16\}$, graphs are pairwise non-isomorphic. We construct a dataset containing 10 permuted copies of each graph. The classification task is then to assign graphs to their skip link value s which measures a model's capability to distinguish non-isomorphic graphs.

In this experiment, we consider the FWL-W variant and do not limit the number of filtrations k but allow as many filtrations as there are distinct edge weights in the dataset graphs. Thus, the number of filtration graphs is directly governed by λ, i.e., the parameter of w_w^λ which denotes the

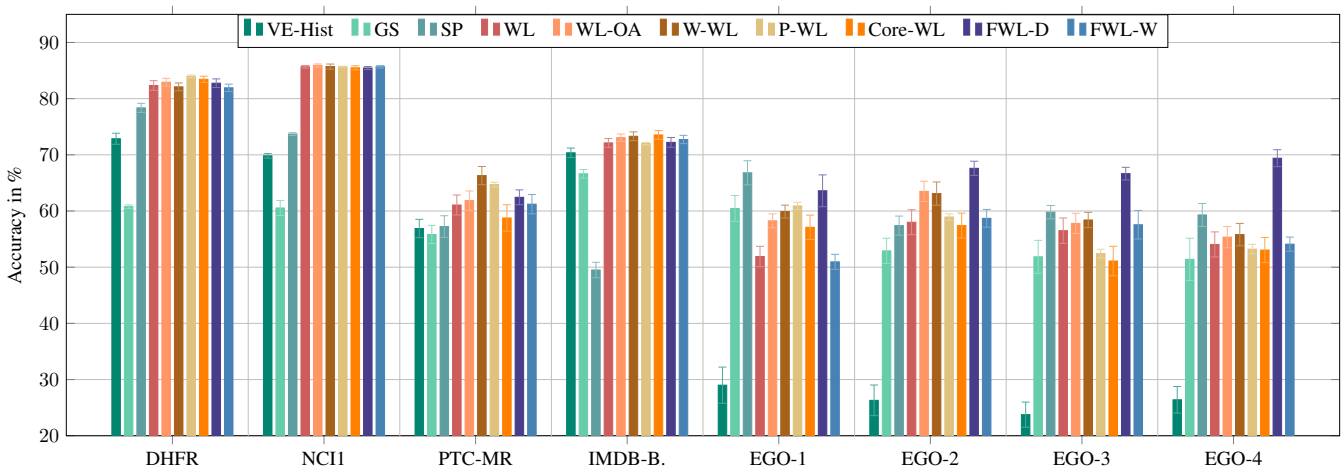

Figure 3: Classification accuracies and standard deviations on real-world benchmark datasets. We compare our approach to a simple baseline kernel (VE-Hist), the graphlet sampling (GS) kernel (Shervashidze et al. 2009), the shortest-path (SP) kernel (Borgwardt and Kriegel 2005), the ordinary Weisfeiler-Lehman (WL) subtree kernel (Shervashidze et al. 2011), the Weisfeiler-Lehman optimal assignment (WL-OA) kernel (Kriege, Giscard, and Wilson 2016), the Wasserstein Weisfeiler-Lehman (W-WL) kernel (Togninalli et al. 2019), the persistent Weisfeiler-Lehman (P-WL) method (Rieck, Bock, and Borgwardt 2019), and the core variant of the Weisfeiler-Lehman (Core-WL) kernel (Nikolentzos et al. 2018). We employ our WL filtration kernel (FWL-D and FWL-W) using two different edge weight functions on graphs (see Sect. *Filtration Variants*).

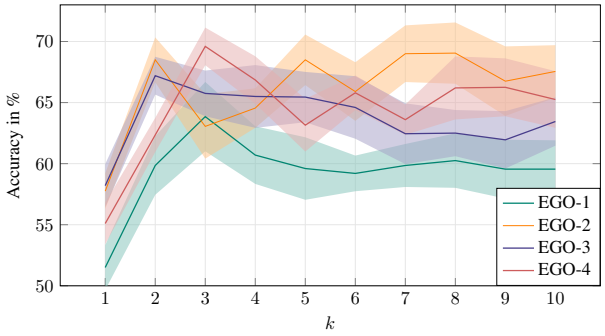

Figure 4: Classification accuracies and standard deviations of the FWL-D variant for different filtration lengths k.

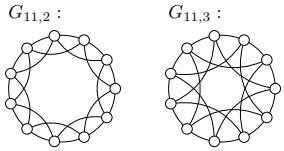

Figure 5: Circular skip link graphs.

$\lambda : k$	1 : 1	2 : 3	3 : 4	4 : 7	5 : 9	6 : 14	7 : 18	8 : 19
Acc.	10%	20%	30%	50%	70%	90%	100%	100%

Figure 6: Accuracies of FWL-W on CSL graphs.

maximal walk length. For example, for $\lambda = 4$, there exist seven distinct edge weights. We, thus, generate $k = 7$ filtration graphs; one for each edge weight. Table 6 shows the predictive performances for different choices of λ (and thus k). The case $\lambda = 1$ implies $k = 1$ and therefore corresponds to the ordinary Weisfeiler-Lehman kernel. Since the Weisfeiler-Lehman method falls short of distinguishing regular graphs, the predictive performance corresponds to that of a random classifier. However, for increasing values of λ, the number of filtrations k grows as well which results in the kernel's ability to distinguish more and more CSL graphs. Finally, for the case $\lambda = 7$, all non-isomorphic graphs can be told apart. It is noteworthy, that these results are not entirely surprising as the considered edge weights provide the filtration kernel with increasing degrees of cyclic information. Nonetheless, the experiments highlight the power of filtration kernels and practically support Corollary 1.

Conclusion

We introduced filtration kernels, a family of graph kernels which compare graphs at different levels of resolution by tracking existence intervals of features over graph filtrations. We showed that a particular member of this family enriches the expressive power of the WL subtree kernel. The implications of this result extend beyond graph kernels, and allow to construct more powerful graph neural networks, as the expressivity of such networks is limited by the WL method.

Empirically, we have demonstrated that our proposed kernel shows comparable or improved predictive performances with respect to other state-of-the-art methods on real-world benchmark datasets. Given the theoretical insights mentioned above, a particularly interesting research question is concerned with practical performances of filtration-aware graph neural networks.

Acknowledgements

This material was produced within the Competence Center for Machine Learning Rhine-Ruhr (**ML2R**) which is funded by the Federal Ministry of Education and Research of Germany (grant no. 01—S18038C). The authors gratefully acknowledge this support.

References

Aktas, M. E.; Akbas, E.; and Fatmaoui, A. E. 2019. Persistence homology of networks: methods and applications. *Appl. Netw. Sci.*, 4(1): 61:1–61:28.

Babai, L.; and Kucera, L. 1979. Graph canonization in linear average time. In *IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, 39–46.

Barceló, P.; Geerts, F.; Reutter, J.; and Ryschkov, M. 2021. Graph Neural Networks with Local Graph Parameters. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Borgwardt, K. M.; Ghisu, M. E.; Llinares-López, F.; O'Bray, L.; and Rieck, B. 2020. Graph Kernels: State-of-the-Art and Future Challenges. *Foundations and Trends in Machine Learning*, 13(5-6).

Borgwardt, K. M.; and Kriegel, H. 2005. Shortest-Path Kernels on Graphs. In *IEEE International Conference on Data Mining (ICDM)*, 74–81.

Cai, J.; Fürer, M.; and Immerman, N. 1992. An optimal lower bound on the number of variables for graph identifications. *Combinatorica*, 12(4): 389–410.

Edelsbrunner, H.; and Harer, J. 2010. *Computational Topology - an Introduction*. American Mathematical Society. ISBN 978-0-8218-4925-5.

Gärtner, T.; Flach, P. A.; and Wrobel, S. 2003. On Graph Kernels: Hardness Results and Efficient Alternatives. In *Computational Learning Theory and Kernel Machines (COLT/Kernel)*, 129–143.

Haussler, D. 1999. Convolution Kernels on Discrete Structures. Technical Report UCSC-CRL-99-10, Univerisity of California - Santa Cruz.

Horváth, T.; Gärtner, T.; and Wrobel, S. 2004. Cyclic pattern kernels for predictive graph mining. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 158–167.

Kriege, N. M.; Giscard, P.; and Wilson, R. C. 2016. On Valid Optimal Assignment Kernels and Applications to Graph Classification. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1615–1623.

Kriege, N. M.; Johansson, F. D.; and Morris, C. 2020. A survey on graph kernels. *Applied Network Science*, 5(1): 6.

Le, T.; Yamada, M.; Fukumizu, K.; and Cuturi, M. 2019. Tree-Sliced Variants of Wasserstein Distances. In *Advances in Neural Information Processing Systems (NeurIPS)*, 12283–12294.

Li, P.; Wang, Y.; Wang, H.; and Leskovec, J. 2020. Distance Encoding: Design Provably More Powerful Neural Networks for Graph Representation Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Morris, C.; Fey, M.; and Kriege, N. M. 2021. The Power of the Weisfeiler-Leman Algorithm for Machine Learning with Graphs. *CoRR*, abs/2105.05911.

Morris, C.; Kriege, N. M.; Bause, F.; Kersting, K.; Mutzel, P.; and Neumann, M. 2020. TUDataset: A collection of benchmark datasets for learning with graphs. *CoRR*, abs/2007.08663.

Morris, C.; Rattan, G.; and Mutzel, P. 2020. Weisfeiler and Leman go sparse: Towards scalable higher-order graph embeddings. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Morris, C.; Ritzert, M.; Fey, M.; Hamilton, W. L.; Lenssen, J. E.; Rattan, G.; and Grohe, M. 2019. Weisfeiler and Leman Go Neural: Higher-Order Graph Neural Networks. In *AAAI Conference on Artificial Intelligence, (AAAI)*, 4602–4609.

Murphy, R. L.; Srinivasan, B.; Rao, V. A.; and Ribeiro, B. 2019. Relational Pooling for Graph Representations. In *International Conference on Machine Learning (ICML)*, 4663–4673.

Nikolentzos, G.; Meladianos, P.; Limnios, S.; and Vazirgiannis, M. 2018. A Degeneracy Framework for Graph Similarity. In Lang, J., ed., *International Joint Conference on Artificial Intelligence, (IJCAI)*, 2595–2601.

O'Bray, L.; Rieck, B.; and Borgwardt, K. 2021. Filtration Curves for Graph Representation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 1267–1275. New York, NY, USA.

Peyré, G.; and Cuturi, M. 2019. Computational Optimal Transport. *Foundations and Trends in Machine Learning*, 11(5-6): 355–607.

Reininghaus, J.; Huber, S.; Bauer, U.; and Kwitt, R. 2015. A stable multi-scale kernel for topological machine learning. In *IEEE Conference on Computer Vision and Pattern Recognition, (CVPR)*, 4741–4748.

Rieck, B.; Bock, C.; and Borgwardt, K. M. 2019. A Persistent Weisfeiler-Lehman Procedure for Graph Classification. In *International Conference on Machine Learning (ICML)*, 5448–5458.

Schulz, T. H.; Welke, P.; and Wrobel, S. 2021. Graph Filtration Kernels. *CoRR*, abs/2110.11862.

Shervashidze, N.; Schweitzer, P.; van Leeuwen, E. J.; Mehlhorn, K.; and Borgwardt, K. M. 2011. Weisfeiler-Lehman Graph Kernels. *Journal of Machine Learning Research*, 12: 2539–2561.

Shervashidze, N.; Vishwanathan, S. V. N.; Petri, T.; Mehlhorn, K.; and Borgwardt, K. M. 2009. Efficient graphlet kernels for large graph comparison. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 488–495.

Togninalli, M.; Ghisu, M. E.; Llinares-López, F.; Rieck, B.; and Borgwardt, K. M. 2019. Wasserstein Weisfeiler-Lehman Graph Kernels. In *Advances in Neural Information Processing Systems (NeurIPS)*, 6436–6446.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019. How Powerful are Graph Neural Networks? In *International Conference on Learning Representations (ICLR)*.