Lecture 3: Graph Similarity

Part 2.2: Graph Embedding

Professor Dr. Petra Mutzel

Computational Analytics

Computer Science

University of Bonn

UNIVERSITÄT BONN

RHEINISCHE FRIEDRICH-WILHELMS-UNIVERSITÄT BONN

INSTITUT FÜR INFORMATIK DER UNIVERSITÄT BONN

Hausdorff School: Computational Combinatorial Optimization, September 12-16, 2022

Approaches to Graph Similarity

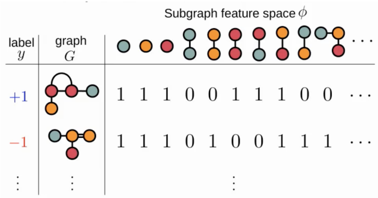
Structural

- isomorphism-based
- Frobenius distance
- graph edit distance

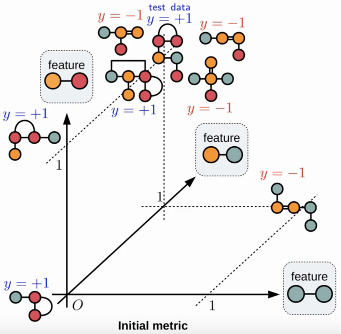
Graph embedding

- Weisfeiler-Leman
- fingerprint

Graph Embedding



feature vector

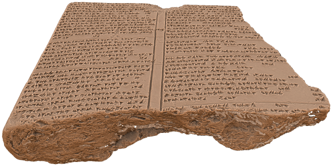


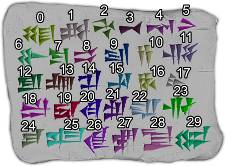
feature space

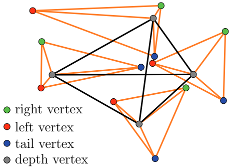
- Generate a feature vector for each graph in the given graph data set
- Use your favorite (data) classification/clustering method

Sources: https://www.kdd.org/kdd2019/accepted-papers/view/learning-interpretable-metric-between-graphs-convex-formulation-and-computa

Recognition of Cuneiform Characters



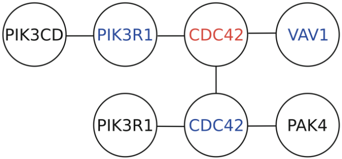




- 500.000 digitized cuneiform fragments (LS7, TU Dortmund)
- group of wedge signs corresponds to a character
- so far 1000 different characters known
- only a very small set of cuneiform characters have been classified
- Goal: Recognition of cuneiform characters for supporting classic Altphilologists

Kriege, Fey, Fisseler, Mutzel, Weichert: COST, Proc. MLR 2018
DFG SFB 876: Providing Information by Resource-Constrained Data Analysis

Protein Complex Similarity



- suggest similarity measures based on WL algorithm for protein complexes
- first run WL, then apply Jaccard similarity coefficient (def. for multisets: size of bag intersection divided by size of bag sum)
- evaluation on 500 000 simulated complexes of the human adhesome protein network
- → in agreement with graph edit similarity

Stöcker, Schäfer, Mutzel, Köster, Kriege, Rahmann: SISAP 2019

DFG SFB 876: Providing Information by Resource-Constrained Data Analysis

Outline

1. **Weisfeiler-Leman Algorithm and its Properties**
   - Original Weisfeiler-Leman Algorithm
   - k-dimensional WL Algorithm on Sets
   - Local k-WL on Sets
2. **Connections of WL to other fields**
   - Fractional Isomorphism
   - WL and Deep Learning
3. **WL for Data Analysis**
   - Classification with WL
4. **Outlook and Conclusion**

Literature: Overview

- M. Grohe, K. Kersting, M. Mladenov, P. Schweitzer: Color Refinement and its Applications, in: An Introduction to Lifted Probabilistic Inference, The MIT Press, 2021 (preprint via lics.rwth-aachen.de)
- K. Borgwardt, E. Ghisu, F. Llinares-López, L. OB́ray, B, Rieck: Graph Kernels: State-of-the-Art and Future Challenges, 2020, 978-1-68083-770-4 (ISBN), also arXiv:2011.03854
- M. Grohe: Descriptive Complexity, Canonisation, and Definable Graph Structure Theory, Lecture Notes in Logic, Band 47, 2017 (preprint via lics.rwth-aachen.de)

Literature

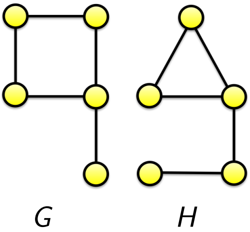
- C. Berkholz, P. Bonsma, M. Grohe: Tight Lower and Upper Bounds for the Complexity of Canonical Colour Refinement, Theory of Computing Systems 60(4), Springer, 2017
- C. Morris, G. Rattan, P. Mutzel: Weisfeiler and Leman go sparse: Towards scalable higher-order graph embeddings, Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020
- C. Morris, K. Kersting, and P. Mutzel: Glocalized Weisfeiler-Lehman Graph Kernels: Global-Local Feature Maps of Graphs, IEEE International Conference on Data Mining (ICDM 2017), New Orleans, LA, USA, pp. 327-336, IEEE Computer Society, 2017.

Weisfeiler-Leman Algorithm

- Color refinement algorithm with the goal of vertex classification

WL-Algorithm

- **Initial:** All vertices v have the same color.
- **Iteration:** Further separation of identically colored vertex sets based on color histograms of neighbors.



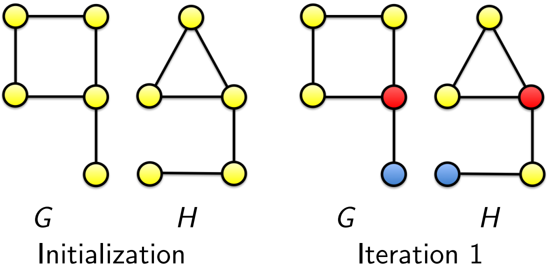
Initialization

Weisfeiler-Leman Algorithm

- Color refinement algorithm with the goal of vertex classification

WL-Algorithm

- **Initial:** All vertices v have the same color.
- **Iteration:** Further separation of identically colored vertex sets based on color histograms of neighbors.

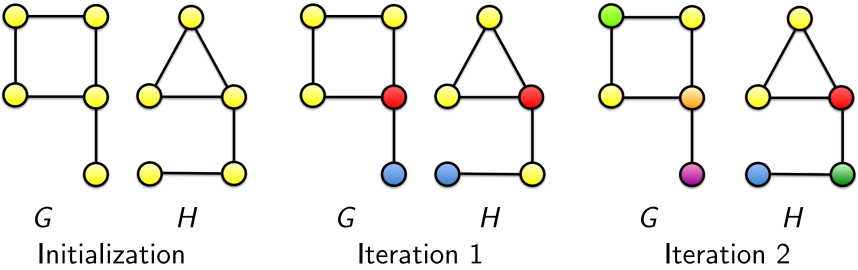


Weisfeiler-Leman Algorithm

- Color refinement algorithm with the goal of vertex classification

WL-Algorithm

- **Initial:** All vertices v have the same color.
- **Iteration:** Further separation of identically colored vertex sets based on color histograms of neighbors.

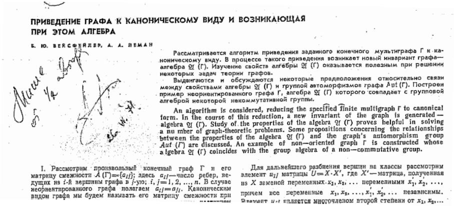


Historical Notes

- Vertex coloring algorithms have been re-invented several times
- First mentioned in paper by Boris Weisfeiler and Andrei Leman 1968 (in Russian)
- Andrei Leman spelled himself as Leman (mistake by Springer)

Historical Notes

- Vertex coloring algorithms have been re-invented several times
- First mentioned in paper by Boris Weisfeiler and Andrei Leman 1968 (in Russian)
- Andrei Leman spelled himself as Leman (mistake by Springer)



ПРИВЕДЕНИЕ ГРАФА К КАНОНИЧЕСКОМУ ВИДУ И ВОЗНИКАЮЩАЯ ПРИ ЭТОМ АЛГЕБРА

Б. Ю. ВЕЙСФЕЙЛЕР, А. А. ЛЕМАН

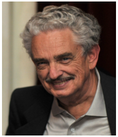
Рассматривается алгоритм приведения заданного конечного мультиграфа Г к каноническому виду. В процессе такого приведения возникает новый инвариант графа — алгебра $\mathfrak{A}$ (Г). Изучение свойств алгебры $\mathfrak{A}$ (Г) оказывается полезным при решении некоторых задач теории графов.

Выдвигаются и обсуждаются некоторые предположения относительно связи между свойствами алгебры $\mathfrak{A}$ (Г) и группой автоморфизмов графа Aut (Г). Построен пример неориентированного графа Г, алгебра $\mathfrak{A}$ (Г) которого совпадает с групповой алгеброй некоторой некоммутативной группы.

An algorithm is considered, reducing the specified finite multigraph Г to canonical form. In the course of this reduction, a new invariant of the graph is generated — algebra $\mathfrak{A}$ (Г). Study of the properties of the algebra $\mathfrak{A}$ (Г) proves helpful in solving a number of graph-theoretic problems. Some propositions concerning the relationships between the properties of the algebra $\mathfrak{A}$ (Г) and the graph's automorphism group *Aut* (Г) are discussed. An example of non—oriented graph Г is constructed whose algebra $\mathfrak{A}$ (Г) coincides with the group algebra of a non—commutative group.

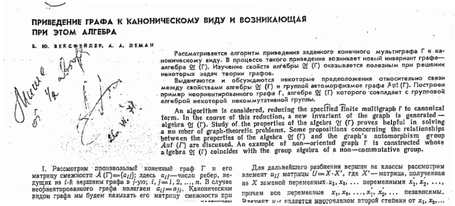
I. Рассмотрим произвольный конечный граф Г и его матрицу смежности $A(\Gamma)=\{a_{ij}\}$; здесь a_{ij} — число ребер, ведущих из i-й вершины графа в j-ую; $i, j=1, 2, \ldots, n$. В случае неориентированного графа полагаем $a_{ij}=a_{ji}$. Каноническим видом графа мы будем называть его матрицу смежности при …

Для дальнейшего разбиения вершин на классы рассмотрим элемент u_{ij} матрицы $U=X\cdot X'$, где X' — матрица, полученная из X заменой переменных $x_1, x_2, \ldots$ переменными $x_1', x_2', \ldots$, причем все переменные $x_1, x_2, \ldots, x_1', x_2', \ldots$ независимы. Элемент u_{ij} является многочленом второй степени от $x_1, x_2, \ldots$



Historical Notes

- Vertex coloring algorithms have been re-invented several times
- First mentioned in paper by Boris Weisfeiler and Andrei Leman 1968 (in Russian)
- Andrei Leman spelled himself as Leman (mistake by Springer)



ПРИВЕДЕНИЕ ГРАФА К КАНОНИЧЕСКОМУ ВИДУ И ВОЗНИКАЮЩАЯ ПРИ ЭТОМ АЛГЕБРА

Б. Ю. ВЕЙСФЕЙЛЕР, А. А. ЛЕМАН

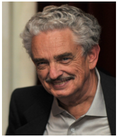
Рассматривается алгоритм приведения заданного конечного мультиграфа Г к каноническому виду. В процессе такого приведения возникает новый инвариант графа — алгебра $\mathfrak{A}$ (Г). Изучение свойств алгебры $\mathfrak{A}$ (Г) оказывается полезным при решении некоторых задач теории графов.

Выдвигаются и обсуждаются некоторые предположения относительно связи между свойствами алгебры $\mathfrak{A}$ (Г) и группой автоморфизмов графа Aut (Г). Построен пример неориентированного графа Г, алгебра $\mathfrak{A}$ (Г) которого совпадает с групповой алгеброй некоторой некоммутативной группы.

An algorithm is considered, reducing the specified finite multigraph Г to canonical form. In the course of this reduction, a new invariant of the graph is generated — algebra $\mathfrak{A}$ (Г). Study of the properties of the algebra $\mathfrak{A}$ (Г) proves helpful in solving a nu mber of graph-theoretic problems. Some propositions concerning the relationships between the properties of the algebra $\mathfrak{A}$ (Г) and the graph's automorphism group Aut (Г) are discussed. An example of non—oriented graph Г is constructed whose algebra $\mathfrak{A}$ (Г) coincides with the group algebra of a non—commutative group.

1. Рассмотрим произвольный конечный граф Г и его матрицу смежности $A(\Gamma)=\{a_{ij}\}$; здесь a_{ij} — число ребер, ведущих из i-й вершины графа в j-ую; $i, j=1, 2, \ldots, n$. В случае неориентированного графа полагаем $a_{ij}=a_{ji}$. Каноническим видом графа мы будем называть его матрицу смежности при

Для дальнейшего разбиения вершин на классы рассмотрим элемент u_{ij} матрицы $U=X\cdot X'$, где X' — матрица, полученная из X заменой переменных $x_1, x_2, \ldots$ переменными $x_1', x_2', \ldots$, причем все переменные $x_1, x_2, \ldots, x_1', x_2', \ldots$ независимы. Элемент u_{ij} является многочленом второй степени от $x_1, x_2, \ldots$



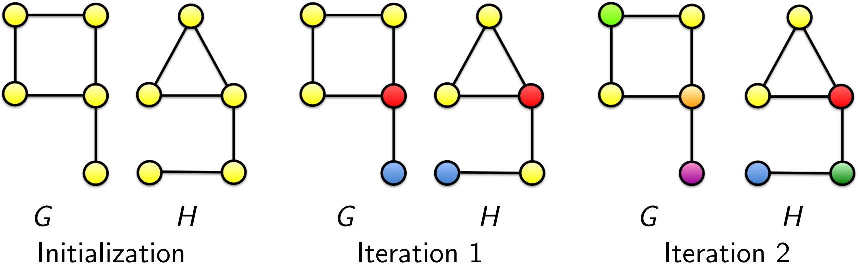
- Shervashidze, Borgwardt 2009: Weisfeiler-Leman based kernels for data analysis

Isomorphism Test using Weisfeiler-Leman

One-Sided Isomorphism Test

- Apply WL to the two graphs G and H simultaneously
- If G and H get different colors $\Longrightarrow G \not\simeq H$

  $\Rightarrow$ WL distinguishes G and H
- Otherwise: we do not know whether G and H are isomorphic



Isomorphism Test using Weisfeiler-Leman

Properties of WL

- WL can identify all forests, i.e., non-isomorphic forests get different colors

Isomorphism Test using Weisfeiler-Leman

Properties of WL

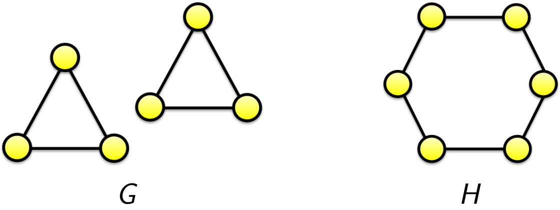
- WL can identify all forests, i.e., non-isomorphic forests get different colors
- random graphs G will be identified correctly with high probability

Isomorphism Test using Weisfeiler-Leman

Properties of WL

- WL can identify all forests, i.e., non-isomorphic forests get different colors
- random graphs G will be identified correctly with high probability
- cannot distinguish regular graphs (same degree) $\rightarrow$ same color

Isomorphism Test using Weisfeiler-Leman



Properties of WL

- WL can identify all forests, i.e., non-isomorphic forests get different colors
- random graphs G will be identified correctly with high probability
- running time: $O((|V| + |E|) \log |V|)$
- cannot distinguish regular graphs (same degree) $\rightarrow$ same color

Equitable Partitions

Definition

- A partition $\{C_1, C_2, \ldots, C_s\}$ of $V(G)$ is called equitable if for all i and j and for all $u, v \in C_i$ we have: $|N(u) \cap C_j| = |N(v) \cap C_j|$.
- We call the maximum element of the equitable partition lattice a coarsest equitable partition of G.

Equitable Partitions

Definition

- A partition $\{C_1, C_2, \dots, C_s\}$ of $V(G)$ is called equitable if for all i and j and for all $u, v \in C_i$ we have: $|N(u) \cap C_j| = |N(v) \cap C_j|$.
- We call the maximum element of the equitable partition lattice a coarsest equitable partition of G.

Lemma (Ramana, Scheinerman, Ullman 1994)

- *The Weisfeiler-Leman algorithm applied to graph G leads to the coarsest equitable partition.*

Equitable Partitions

Definition

- A partition $\{C_1, C_2, \ldots, C_s\}$ of $V(G)$ is called equitable if for all i and j and for all $u, v \in C_i$ we have: $|N(u) \cap C_j| = |N(v) \cap C_j|$.
- We call the maximum element of the equitable partition lattice a coarsest equitable partition of G.

Lemma (Ramana, Scheinerman, Ullman 1994)

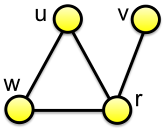
- *The Weisfeiler-Leman algorithm applied to graph G leads to the coarsest equitable partition.*
- *If the Weisfeiler-Leman algorithm is applied to graphs G and H, and the two stable partitions ρ_G and ρ_H are identical, then G and H have a common coarsest equitable partition.*

k-dimensional Weisfeiler-Leman Algorithm (Sets)

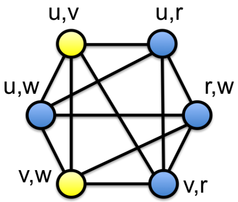
Two k-vertex sets are neighbors if they differ in only one element.

k-WL Algorithm for k-sets (simplified)

- **Initial:** k-sets U, W get the same color if $G[U] \simeq G[W]$.
- **Iteration:** Two identically colored k-sets U und W get different colors if there exists a color c so that U and W have a different cardinality set of neighbors of color c.



G



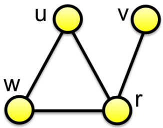
Initialization 2-WL

k-dimensional Weisfeiler-Leman Algorithm (Sets)

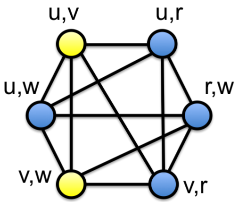
Two k-vertex sets are neighbors if they differ in only one element.

k-WL Algorithm for k-sets (simplified)

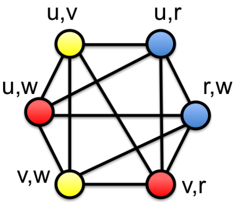
- **Initial:** k-sets U, W get the same color if $G[U] \simeq G[W]$.
- **Iteration:** Two identically colored k-sets U und W get different colors if there exists a color c so that U and W have a different cardinality set of neighbors of color c.



G



Initialization 2-WL

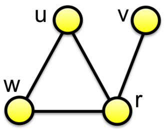


Iteration 1: 2-WL

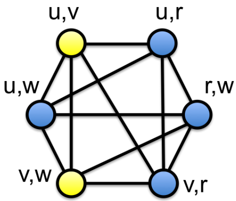
k-WL Algorithm

Properties of the k-WL

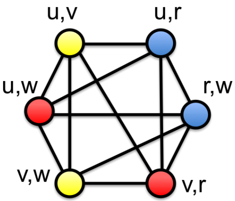
- k-WL ($k > 2$) is stronger than WL



G



Initialization 2-WL

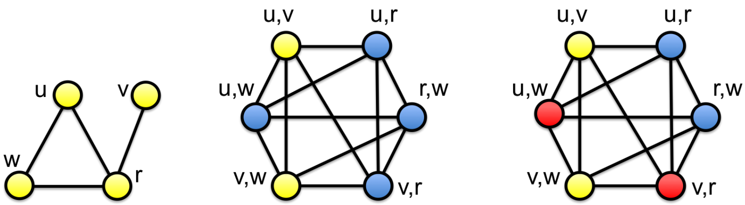


Iteration 1: 2-WL

k-WL Algorithm

Properties of the k-WL

- k-WL ($k > 2$) is stronger than WL
- exact for large enough k (identifies each graph)

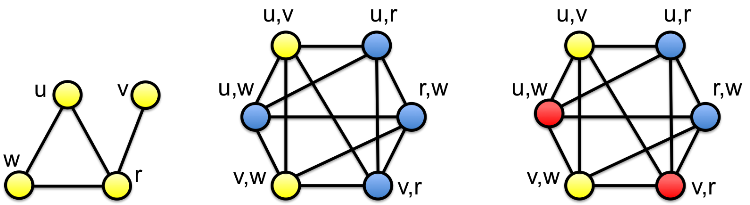


G Initialization 2-WL Iteration 1: 2-WL

k-WL Algorithm

Properties of the k-WL

- k-WL ($k > 2$) is stronger than WL
- exact for large enough k (identifies each graph)
- graph isomorphism approach by Babai uses $k = O(\log n)$

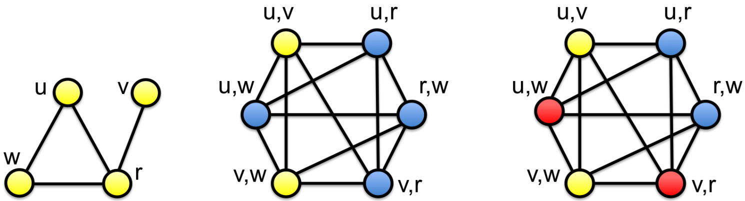


G Initialization 2-WL Iteration 1: 2-WL

k-WL Algorithm

Properties of the k-WL

- k-WL ($k > 2$) is stronger than WL
- exact for large enough k (identifies each graph)
- graph isomorphism approach by Babai uses $k = O(\log n)$
- there exist graph classes for which $k = \theta(n)$ is necessary

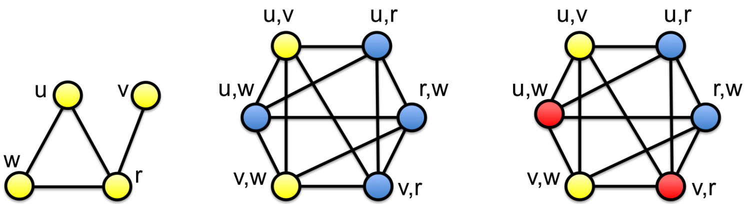


G Initialization 2-WL Iteration 1: 2-WL

k-WL Algorithm

Properties of the k-WL

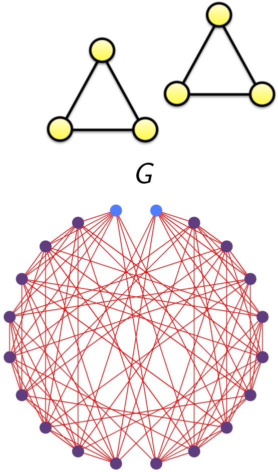
- k-WL ($k > 2$) is stronger than WL
- exact for large enough k (identifies each graph)
- graph isomorphism approach by Babai uses $k = O(\log n)$
- there exist graph classes for which $k = \theta(n)$ is necessary
- running time: $O(k^2|V|^{k+1} \log |V|)$



G — Initialization 2-WL — Iteration 1: 2-WL

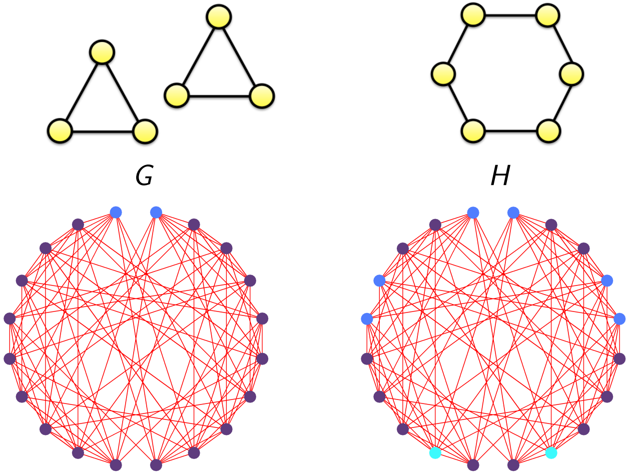
k-WL Algorithm

k-WL can distinguish regular graphs: $k = 3$



k-WL Algorithm

k-WL can distinguish regular graphs: $k = 3$

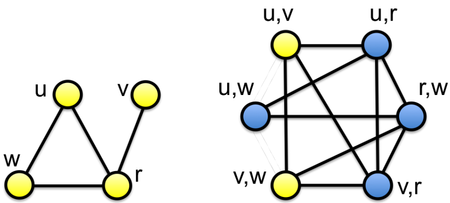


$\rightarrow$ strong for higher k, but also very slow

Local k-WL Algorithm (k-LWL)

Idea: Define neighbors of the k-sets depending on graph structure

- Two k-sets U und W are neighbors, if they differ in only one element and for the exchanged vertices $s \in U, t \in W$ there exists an edge from s to a vertex in W and an edge from t to a vertex in U.
- $\rightarrow$ takes sparsity of the original graph into account
- $\rightarrow$ considers local and global graph properties

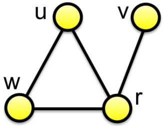


G Initialization 2-LWL

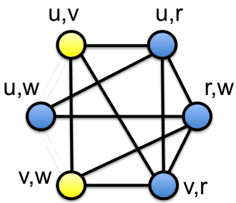
Local k-WL Algorithm (k-LWL)

Idea: Define neighbors of the k-sets depending on graph structure

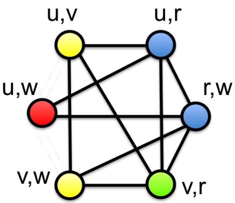
- Two k-sets U und W are neighbors, if they differ in only one element and for the exchanged vertices $s \in U, t \in W$ there exists an edge from s to a vertex in W and an edge from t to a vertex in U.
- $\rightarrow$ takes sparsity of the original graph into account
- $\rightarrow$ considers local and global graph properties



G



Initialization 2-LWL

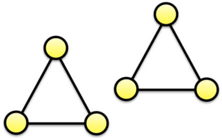


1. Iteration 2-LWL

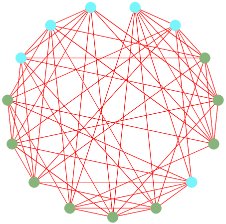
Comparison: Local k-LWL vs. k-WL

Running time

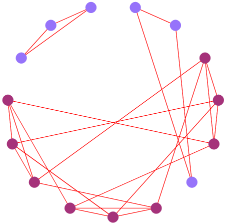
- k-sets of the k-LWL have much less neighbors
- much faster than k-WL



G



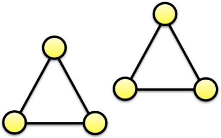
2-WL

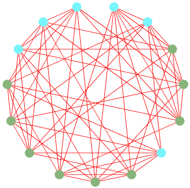


2-LWL

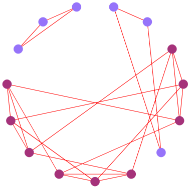
Morris, Kersting, Mutzel 2017

Comparison of Distinction Power

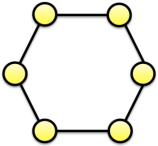


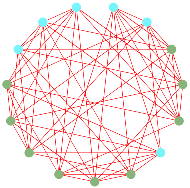


2-WL: 2 color classes

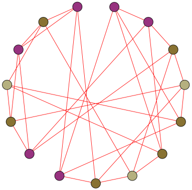


2-LWL: 2 color classes





2-WL: 2 color classes



2-LWL: 3 color classes

$\rightarrow$2-WL cannot distinguish graphs, but 2-LWL

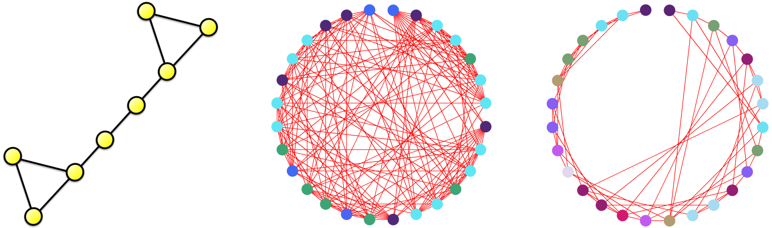
Comparison: Local k-LWL vs. k-WL

Separation Strength (for connected G)

Comparison: Local k-LWL vs. k-WL

Separation Strength (for connected G)

- Local k-LWL refines at least as much as k-WL.



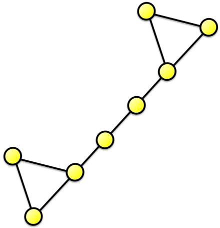
2-WL: 5 color classes
$12, 6, 6, 3, 1$

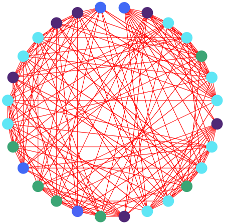
2-LWL: 10 color classes
$4, 4, 4, 4, 2, 4, 2, 2, 1, 1$

Comparison: Local k-LWL vs. k-WL

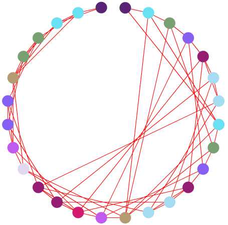
Separation Strength (for connected G)

- Local k-LWL refines at least as much as k-WL.
- If k-WL can distinguish two graphs, then also k-LWL.





2-WL: 5 color classes
$12, 6, 6, 3, 1$

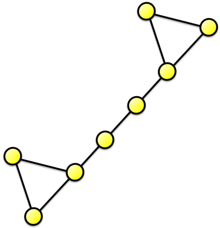


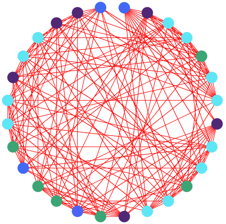
2-LWL: 10 color classes
$4, 4, 4, 4, 2, 4, 2, 2, 1, 1$

Comparison: Local k-LWL vs. k-WL

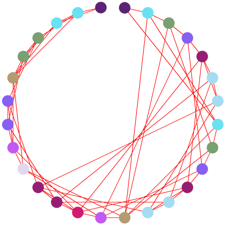
Separation Strength (for connected G)

- Local k-LWL refines at least as much as k-WL.
- If k-WL can distinguish two graphs, then also k-LWL.
- Local k-LWL is stronger than k-WL.





2-WL: 5 color classes
$12, 6, 6, 3, 1$

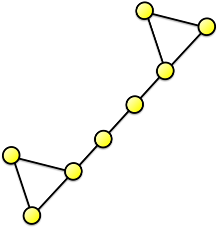


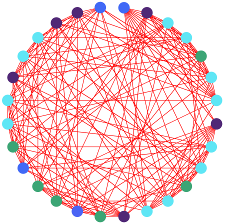
2-LWL: 10 color classes
$4, 4, 4, 4, 2, 4, 2, 2, 1, 1$

Comparison: Local k-LWL vs. k-WL

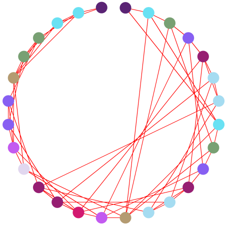
Separation Strength (for connected G)

- Local k-LWL refines at least as much as k-WL.
- If k-WL can distinguish two graphs, then also k-LWL.
- Local k-LWL is stronger than k-WL.
- Sherali-Adams Relaxation of k-LWL is stronger than that of k-WL.



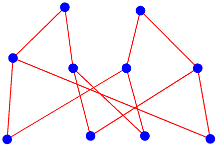


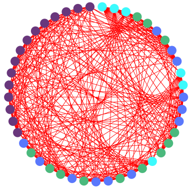
2-WL: 5 color classes
$12, 6, 6, 3, 1$



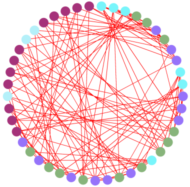
2-LWL: 10 color classes
$4, 4, 4, 4, 2, 4, 2, 2, 1, 1$

Cai, Fürer, Immerman - Graphs for lower bound

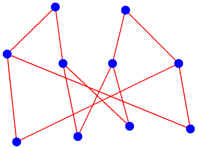


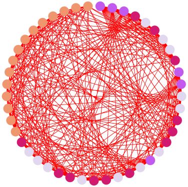


2-WL: 4 color classes

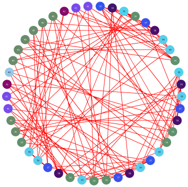


2-LWL: 5 color classes





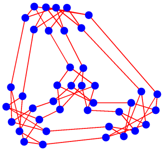
2-WL: 4 color classes

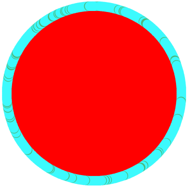


2-LWL: 11 color classes

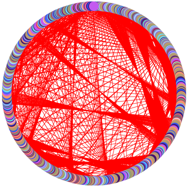
$\rightarrow$2-WL cannot distinguish graphs, but 2-LWL

Immerman, Grohe - Graphs for lower bound



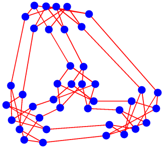


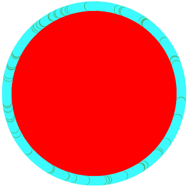
2-WL: 2 color classes



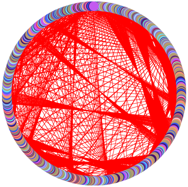
2-LWL: 15 color classes

Immerman, Grohe - Graphs for lower bound

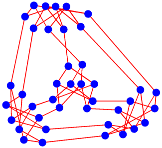


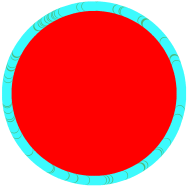


2-WL: 2 color classes

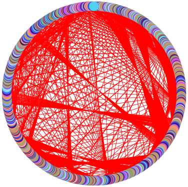


2-LWL: 15 color classes





2-WL: 2 color classes

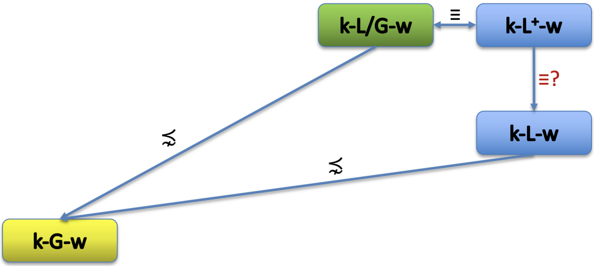


2-LWL: 15 color classes

$\rightarrow$ 2-LWL refines more but unfortunately does not distinguish

Discriminatory Power of k-WL versions (tupel)

So far: k-sets, but k-tupels are stronger

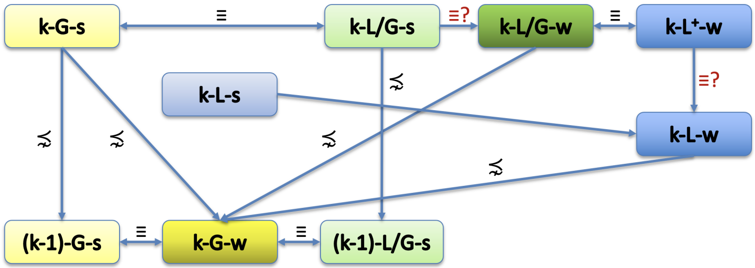


G: global neighborhood, L/G: both neighborhoods, L: local

$A \equiv B$: A is at least as strong as B, $A \precnsim B$: stronger

Morris, Rattan, Mutzel: NeurIPs 2020

Discriminatory Power of k-WL versions



G: global neighborhood, L/G: both neighborhoods, L: local

$A \equiv B$: A is at least as strong as B, $A \precnsim B$: stronger

Two different definitions of the k-WL: w: weak, s: strong

Morris, Rattan, Mutzel: NeurIPs 2020

Connections of WL to other fields

Connections to

- descriptive complexity
- k-pebble counting games
- counting homomorphisms
- Gröbner basis
- Sherali-Adams relaxation of the natural ILP
- graph neural networks (deep learning)

for more information: ask Martin Grohe

Integer Linear Program for Graph Isomorphism

Observation

Let A and B be the adjacency matrices for graphs G and H. G and H are isomorphic if there is a permutation matrix P so that $AP = PB$.

Binary variables with $X_{ab} = 1$ iff a-th vertex in G will be mapped to b-th vertex in H for $a, b \in [n]$

$$
(ISO) \quad
\begin{array}{rcll}
\sum_{a' \in [n]} A_{aa'} X_{a'b} & = & \sum_{b' \in [n]} X_{ab'} B_{b'b} & \forall\, a, b \in [n] \\
\sum_{b' \in [n]} X_{ab'} & = & 1 & \forall\, a \in [n] \\
\sum_{a' \in [n]} X_{a'b} & = & 1 & \forall\, b \in [n] \\
X_{ab} & \geq & 0 & \forall\, a, b \in [n] \\
X_{ab} & \in & \{0, 1\} & \forall\, a, b \in [n]
\end{array}
$$

Relaxing the integer requirement leads to doubly stochastic matrix.

Integer Linear Program for Graph Isomorphism

Definition

- Relaxing the integer requirement leads to a doubly stochastic matrix.
- G is fractionally isomorphic to H if there exists a doubly stochastic matrix S so that $AS = SB$.
- In other words: G is fractionally isomorphic to H if the polytope defined by $(rISO)$ is non-empty.

$$(rISO) \quad \begin{array}{rcll} \sum_{a' \in [n]} A_{aa'} X_{a'b} & = & \sum_{b' \in [n]} X_{ab'} B_{b'b} & \forall\, a, b \in [n] \\ \sum_{b' \in [n]} X_{ab'} & = & 1 & \forall\, a \in [n] \\ \sum_{a' \in [n]} X_{a'b} & = & 1 & \forall\, b \in [n] \\ X_{ab} & \geq & 0 & \forall\, a, b \in [n] \end{array}$$

Tinhofers Theorem

Lemma (Tinhofer 1986)

- *G and H are fractionally isomorphic if and only if they have a common coarsest equitable partition.*
- *In other words: If the WL-algorithm cannot distinguish G and H, then $(rISO)$ has a feasible solution, and vice versa.*

Tinhofers Theorem

Lemma (Tinhofer 1986)

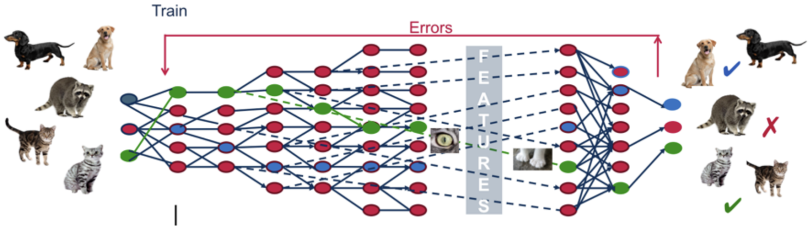
- *G and H are fractionally isomorphic if and only if they have a common coarsest equitable partition.*
- *In other words: If the WL-algorithm cannot distinguish G and H, then (rISO) has a feasible solution, and vice versa.*

When are two fractionally isomorphic graphs isomorphic to each other?

Lemma (Tinhofer 1986, Ramana et al. 1994)

Let F and G be fractionally isomorphic graphs and let F be a forest. Then F is isomorphic to G.

WL and Deep Learning



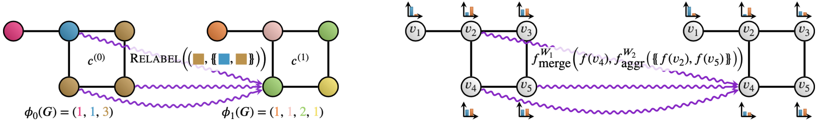
State-of-the-Art Graph Neural Networks (GNNs)

- Graph Convolutional Networks (Kipf, Welling 2017)
- GraphSAGE (Hamilton et al. 2017)
- Graph Isomorphism Networks (Xu et al. 2019)
- Neural Message Parsing (Gilmer et al. 2017), and many others

They only differ in their neighborhood aggregation.

Source: https://www.thegioimaychu.vn/blog/thuat-ngu/deep-learning/

Relations between WL and GNN



General form of Graph Neural Networks

$$\mathbf{H}_v^t = f_{merge}(\mathbf{H}_v^{t-1}, f_{aggr}(\{\!\{\mathbf{H}_w^{t-1} \mid w \in N(v)\}\!\}))$$

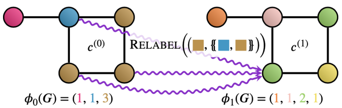
Functions f_{merge} and f_{aggr} can be arbitrary differentiable, permutation-invariant functions. Both methods aggregate features of their neighbors.

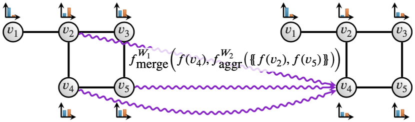
WL Algorithmus

$$\mathbf{C}_v^t = enc(\mathbf{C}_v^{t-1}, \{\!\{\mathbf{C}_w^{t-1} \mid w \in N(v)\}\!\})$$

Source: Morris, Fey, Kriege, 2021

Relations between WL and GNN





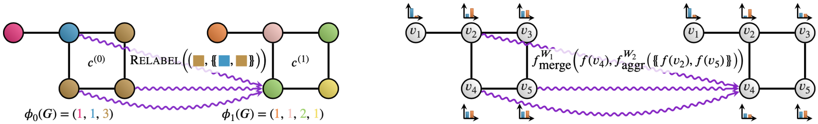
Discriminatory power of GNNs

- If a GNN distinguishes two graphs, so does the WL.
- There are functions f_{merge} und f_{aggr} so that both methods are equally strong with respect to their discriminatory power.

This result also holds for k-WL and k-GNNs.

Morris, Ritzert, Fey, Hamilton, Lenssen, Rattan, Grohe 2019, Xu, Hu, Leskovec, and Jegelka 2019, Maron, Ben-Hamu, Serviansky, Lipman 2019
Images: Morris, Fey, Kriege, 2021

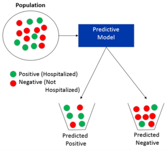
Relations between WL and GNN

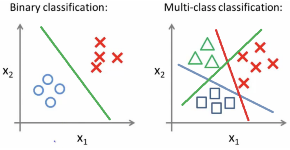


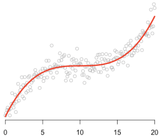
- WL-variants can be transferred to GNN variants
- GNNs better adapt to the learning task because they learn the weights
- However: high resource consumption

Image source: Morris, Fey, Kriege, 2021

Classification (Supervised Learning)



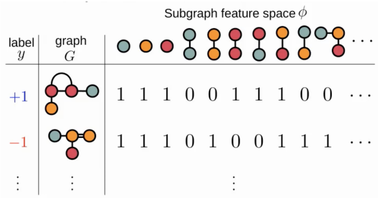




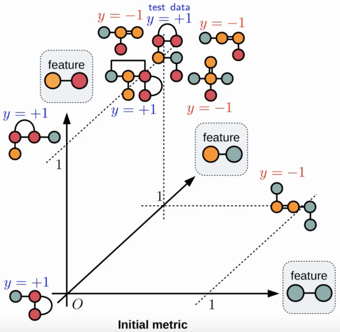
Given: Training set with labelled items (classes).
Goal: Train a classifier so that a new item will be assigned to its correct class.

Sources: towardsdatascience.com/, www.ritchieng.com/logistic-regression/, medium.freecodecamp.org/

Graph (Dataset) Classification



feature vector

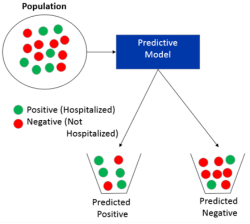


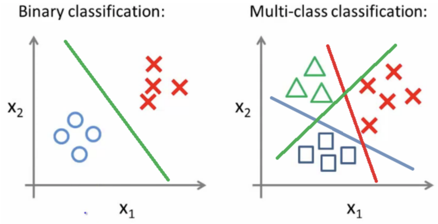
feature space

- Generate a feature vector for each graph in the graph data set
- Use your favorite (data) classification method

Sources: https://www.kdd.org/kdd2019/accepted-papers/view/learning-interpretable-metric-between-graphs-convex-formulation-and-computa

Classification with Distance-based Approaches





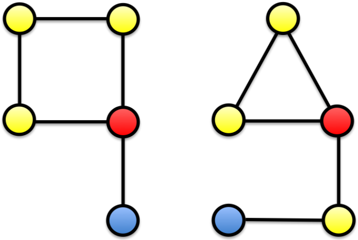
Idea

- compute the distances (e.g. scalar products of the feature vectors) for each pair of graphs in the data set
- compute one or more separating hyperplanes in a high dimensional space (SVM)

Classification with WL, k-WL, k-LWL

Idea: Combination of WL with similarity measures

- Construct feature vector for each graph
- e.g. after each round: sort the vertices according to colors, vector gets information of number of vertices with color c
- append these (possibly weighted) vectors to each other

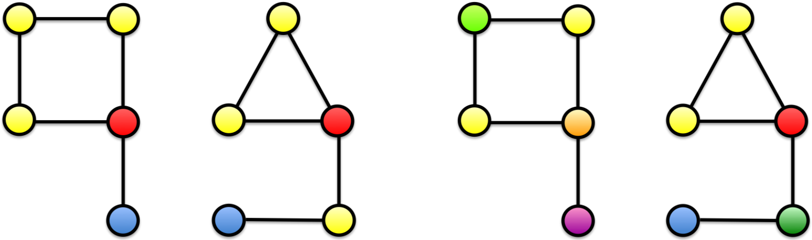


$\Phi(G) = (3, 1, 1)$ $\Phi(H) = (3, 1, 1)$

Classification with WL, k-WL, k-LWL

Idea: Combination of WL with similarity measures

- Construct feature vector for each graph
- e.g. after each round: sort the vertices according to colors, vector gets information of number of vertices with color c
- append these (possibly weighted) vectors to each other

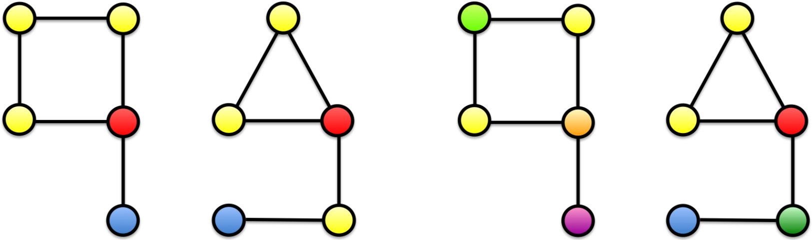


$\Phi(G) = (3, 1, 1)$ $\Phi(H) = (3, 1, 1)$ $\Phi(G) = (3, 1, 1, 2, 0, 0, 1, 1, 1, 0)$ $\Phi(H) = (3, 1, 1, 2, 1, 1, 0, 0, 0, 1)$

Classification with WL, k-WL, k-LWL

Idea: Combination of WL with similarity measures

- Construct feature vector for each graph
- e.g. after each round: sort the vertices according to colors, vector gets information of number of vertices with color c
- append these (possibly weighted) vectors to each other



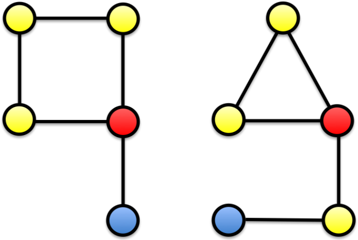
$\Phi(G) = (3, 1, 1)$ $\Phi(H) = (3, 1, 1)$ $\Phi(G) = (3, 1, 1, 2, 0, 0, 1, 1, 1, 0)$ $\Phi(H) = (3, 1, 1, 2, 1, 1, 0, 0, 0, 1)$

$\rightarrow$ Similarity measure based on graph kernel, Jaccard-Coefficient, ...

Classification with WL, k-WL, k-LWL

Similarity score between each pair of graphs

- Let $\Phi_t^i(G_i)$ be the feature vector for each graph G_i of round t
- we define: $\Phi_t(G_i) = [\Phi_0(G_i), \ldots, \Phi_t(G_i)]$
- Weisfeiler-Leman subtree graph kernel for t rounds:
  $k_t(G_i, G_j) = \langle \Phi_t(G_i), \Phi_t(G_j) \rangle$
- $\Rightarrow$ leads to a (normalized) gram matrix
- use this as input to a SVM (kernel trick)

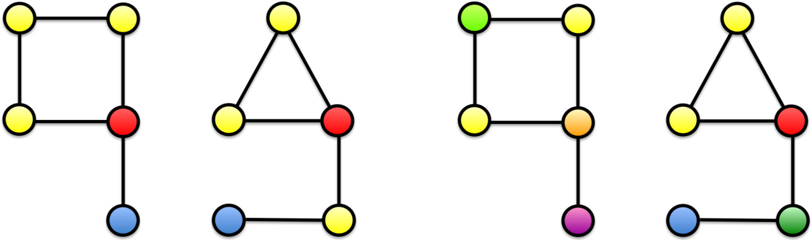


$\Phi_1(G) = (3, 1, 1)$ $\Phi_1(H) = (3, 1, 1)$

Classification with WL, k-WL, k-LWL

Similarity score between each pair of graphs

- Let $\Phi_t^i(G_i)$ be the feature vector for each graph G_i of round t
- we define: $\Phi_t(G_i) = [\Phi_0(G_i), \ldots, \Phi_t(G_i)]$
- Weisfeiler-Leman subtree graph kernel for t rounds:
  $k_t(G_i, G_j) = \langle \Phi_t(G_i), \Phi_t(G_j) \rangle$
- $\Rightarrow$ leads to a (normalized) gram matrix
- use this as input to a SVM (kernel trick)



$\Phi_1(G) = (3, 1, 1)$ $\Phi_1(H) = (3, 1, 1)$ $\Phi_2(G) = (2, 0, 0, 1, 1, 1, 0)$ $\Phi_2(H) = (2, 1, 1, 0, 0, 0, 1)$

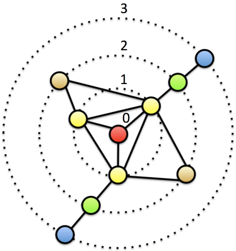
Scalability: sampling for local k-LWL Algorithm

Idea: Increase scalability by sampling

- Sample a subset of the k-sets
- Explore the t-neighborhood around these sets
- Run the local k-LWL on each of the t-neighborhoods

Lemma (Morris, Kersting, M. 2017)

These k-sets get the same color as the k-LWL after t rounds.



Approximation Result

Theorem (Morris, Kersting, M. 2017)

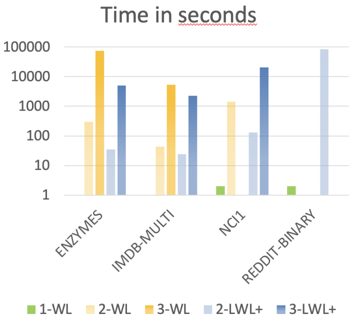
Let G be a d-bounded degree graph and $\epsilon \in (0, 1]$.

- *Then for every number of iterations t of the local k-LWL, there exists an adaptive sampling algorithm which approximates the normalized feature vector of the local k-LWL on t iterations up to ϵ with probability $(1 - \delta)$ for $\delta \in (0, 1)$.*
- *The running time only depends on d, k, δ, ϵ and t (not on the graph size V).*

Such a result is not possible for the k-WL.

Evaluation of the k-LWL (graph kernels, SVM)

- Protein interaction networks ENZYMES
- social networks IMDB-MULTI, REDDIT-BINARY
- cancer dataset NCI1



2-6 classes, 600-4000 graphs, 13-430 vertices, partially vertex labels

Morris, Rattan, Mutzel: NeurIPs 2020

Comparison of WL with other graph kernels

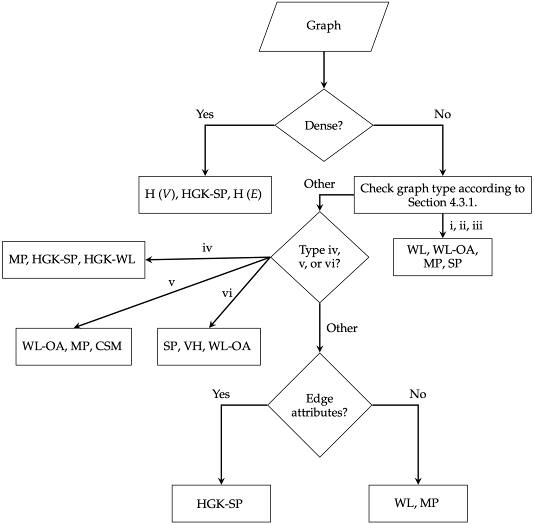
Experimental comparison of 13 different graph kernels

- 4 WL variants, one based on message passing
- 4 variants with shortest paths and random walks
- multiscale Laplacian
- subgraph matching, graphlet
- pure histograms (vertices, edges)

41 benchmark data sets from the TU Dataset: Social networks, molecule graphs, bioinformatics, computer vision

Borgwardt et al.: Graph Kernels: State-of-the-Art and future Challenges, Nov. 2020

Selection of Graph Kernels [Borgwardt et al. 2020]



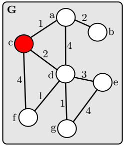
Source: Borgwardt et al.: Graph Kernels: State-of-the-Art and future Challenges, S. 124, Nov. 2020

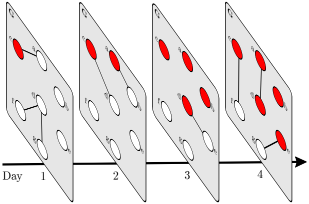
Graph Similarity via Graph Kernels: Playground

Source: farmeramania.de

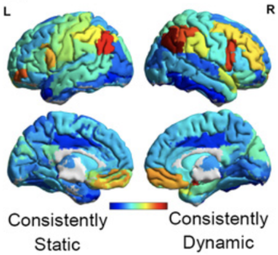
Outlook

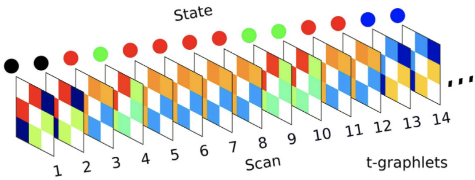
- expressiveness vs. generalizability
- integration of expert knowledge
- integration of uncertainty
- integration to temporal graphs





Outlook: Temporal Graphs from fMRI Data





Student project with Dr. Xenia Kobeleva (Universitätsklinikum Bonn, DZNE) and Lutz Oettershagen

Aim

- analysis of temporal graphs constructed from fMRI data
- dynamic processes vs. static graphs with time windows

Source: Zalesky et al.: PNAS 2014; Thompson, Fransson: Scientific Reports 2016

- motivation to get interested in the area of Graph Similarity
- plenty of opportunities for new theoretical results as well as practical impact

Source: www.whatsnext.com/what-is-motivation