Kernel Machines on Strings & Graphs

CS6140

Predrag Radivojac
Khoury College of Computer Sciences
Northeastern University

Fall 2024

Perceptron

Algorithm:

```
    $\mathbf{w} \leftarrow \mathbf{0}$
    repeat until convergence
        pick an example $\mathbf{x}$ from $\mathcal{D}$
        if $\mathbf{x}$ is incorrectly classified
            $\mathbf{w} \leftarrow \mathbf{w} + \eta y \mathbf{x}$
        else
            do nothing
        end
    end
```

$\eta \in (0, 1]$ = parameter

Solution:

$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$

Prediction:

given a new example $\mathbf{x}$

evaluate $\mathbf{w}^T\mathbf{x}$

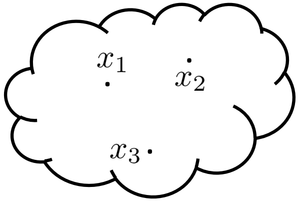
$$\begin{aligned} \mathbf{w}^T\mathbf{x} &= \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i^T \mathbf{x} \\ &= \sum_{i=1}^{n} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) \end{aligned}$$

$$\Big\downarrow$$

$$k(\mathbf{x}_i, \mathbf{x}) = \boldsymbol{\phi}^T(\mathbf{x}_i)\boldsymbol{\phi}(\mathbf{x})$$

String Kernels

Given: $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$. $\mathcal{X} \neq \mathbb{R}^d$.



$$x_1 = \text{'airplane'}$$
$$x_2 = \text{'aeroplane'}$$
$$x_3 = \text{'Flugzeug'}$$

$$\mathbf{K} = \begin{bmatrix} 1 & .7 & .1 \\ .7 & 1 & .1 \\ .1 & .1 & 1 \end{bmatrix}$$

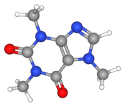
1) k-mer representation
2) sequence similarity (need to ensure positive semi-definite **K**)

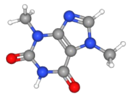
1) Kernel machines allow us to directly work with structured objects
2) Mapping $\phi(\cdot)$ to vector spaces need not be known, we only need $k(\cdot, \cdot)$

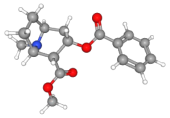
Graph Classification

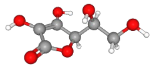
Safe

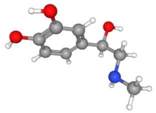
Unsafe

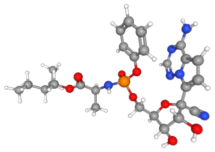

caffeine = $C_8H_{10}N_4O_2$


theobromine = $C_7H_8N_4O_2$


cocaine = $C_{17}H_{21}NO_4$


vitamin C = $C_6H_8O_6$


adrenaline = $C_9H_{13}NO_3$

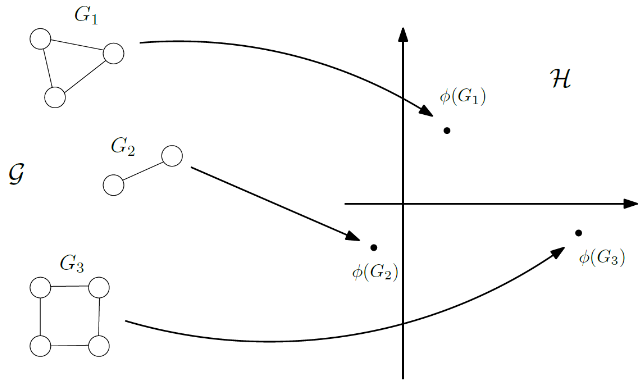

remdesivir = $C_{27}H_{35}N_6O_8P$

Each molecule i is a graph $G_i = (V_i, E_i, \Sigma, \Xi)$.

Nodes and edges can be of different types. We can think of vertex and edge alphabets.

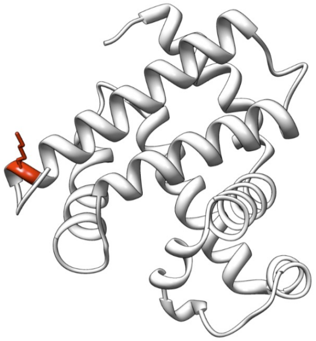
https://pubchem.ncbi.nlm.nih.gov

Graph Classification

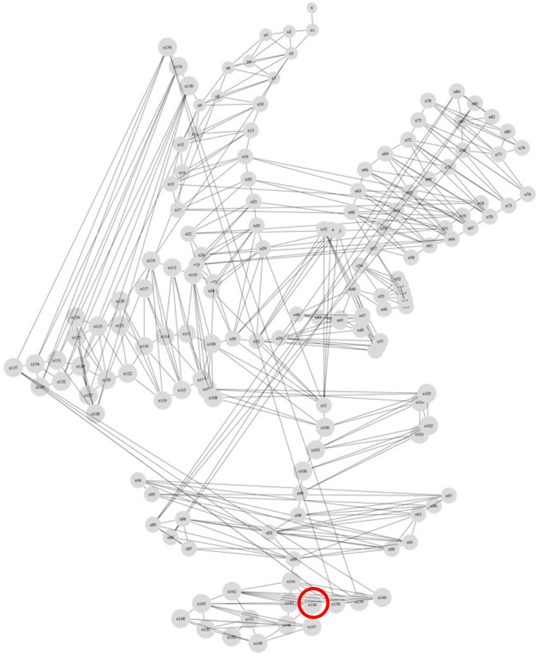


Nikolentzos et al. arXiv 1904.12218 2019

VERTEX CLASSIFICATION



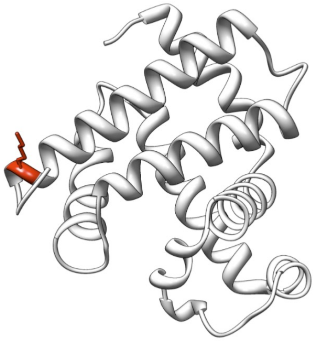
$$\xrightarrow{C_\alpha - C_\alpha \leq 6\text{Å}}$$



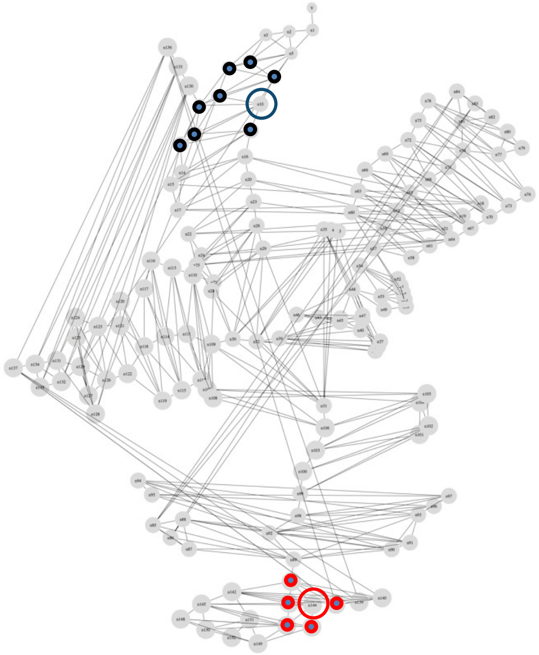
We have a single graph $G = (V, E, \Sigma, \Xi)$.

Nodes and edges can be of different types. We can think of vertex and edge alphabets.

Vertex Classification



$$\xrightarrow{C_\alpha - C_\alpha \leq 6\text{Å}}$$



We have a single graph $G = (V, E, \Sigma, \Xi)$.

Nodes and edges can be of different types. We can think of vertex and edge alphabets.

KERNEL FUNCTION

$\mathcal{X}$ = input space

$\mathcal{H}$ = a Hilbert space

$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$

- symmetric
- positive (semi-)definite

Guarantees a map $\boldsymbol{\phi} : \mathcal{X} \to \mathcal{H}$ s.t. $k(x', x'') = \boldsymbol{\phi}^T(x')\boldsymbol{\phi}(x'')$ for all $x', x'' \in \mathcal{X}$.

Questions:

- What graph or vertex similarity functions satisfy kernel property?
  - We will often need to prove this.
- Can we compute the kernel function efficiently?
  - Good news and bad news.
- Can the kernel function lead to accurate learning?
  - Empirical evaluation.

Properties of Kernels

Kernels are closed under the following operations:

1) Addition: $k_1(x_i, x_j) + k_2(x_i, x_j)$	$\boldsymbol{\phi}(x) = (\boldsymbol{\phi}_1(x), \boldsymbol{\phi}_2(x))$
2) Scaling: $c \cdot k(x_i, x_j)$, $c > 0$	$\boldsymbol{\phi}(x) = \sqrt{c} \cdot \boldsymbol{\phi}(x)$
3) Multiplication: $k_1(x_i, x_j) \cdot k_2(x_i, x_j)$	$\phi(x)_{ij} = \phi_{1i}(x)\phi_{2j}(x)$

Random Walks for Graph Classification

Given: Set of graphs $G_i = (V_i, E_i, \Sigma, \Xi)$, $i = 1, 2, \dots$

Objective: Design a kernel function

Idea: count the number of matching random walks

w' = walk on graph G'

w'' = walk on graph G''

$k(w', w'')$ = similarity function between two walks (compare attribute values of nodes and edges)

$$k(G', G'') = \textstyle\sum_{w'} \sum_{w''} k(w', w'')$$

Random Walks for Graph Classification

Given: Set of graphs $G_i = (V_i, E_i, \Sigma, \Xi)$, $i = 1, 2, \ldots$

Objective: Design a kernel function

Idea: Pair label space kernel

$\ell_i, \ell_j \in \Sigma$

$\mathcal{W}_n =$ all walks in G of length n

$l_k(w) =$ label of the k-th stop in walk w

$\lambda_n \geq 0 =$ user-defined, for convergence

$L =$ an $|\Sigma| \times |V|$ matrix of vertex labels

Note: LEL^T gives the number of edges btw vertices labeled as ℓ_i and ℓ_j

$$\phi_{\ell_i,\ell_j}(G) = \sum_{n=1}^{\infty} \lambda_n \left|\{w \in \mathcal{W}_n(G) : l_1(w) = \ell_i \wedge l_{n+1}(w) = \ell_j\}\right|$$

Then,

$$\phi(G) = L \left(\sum_{i=0}^{\infty} \lambda_i E^i \right) L^T$$

Gärtner et al. COLT 2003

Random Walks for Graph Classification

Given: Set of graphs $G_i = (V_i, E_i, \Sigma, \Xi)$, $i = 1, 2, ...$

Objective: Design a kernel function

Idea: Labeled sequence space kernel

$\mathcal{S}_n =$ all labeled sequences for walks of length n

$\mathcal{W}_n =$ all walks in G of length n

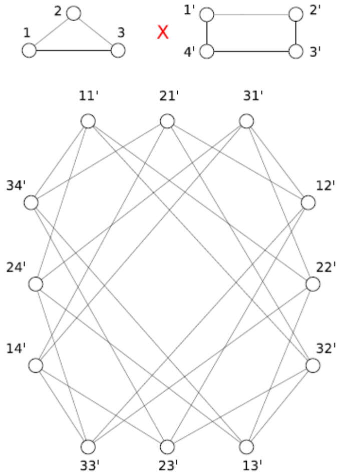
$l_k(w) =$ label of the k-th stop in walk w

$\lambda_n \geq 0 =$ user-defined, for convergence

$$\phi_s(G) = \sqrt{\lambda_n}\,|\{w \in W_n(G), \forall i : s_i = l_i(w)\}|$$

Direct Product Graphs

Given: Graphs $G' = (V', E', \Sigma, \Xi)$ and $G'' = (V'', E'', \Sigma, \Xi)$.



$$V_\times = V(G' \times G'') = \{(v', v'') \in V' \times V'' : label(v') = label(v'')\}$$

$$E_\times = E(G' \times G'') = \{((u', u''), (v', v'')) \in V^2(G' \times G'') : \\ (u', v') \in E' \wedge (u'', v'') \in E'' \wedge label(u', v') = label(u'', v'')\}$$

Proposition w/o proof:

$$|\{w \in \mathcal{W}_n(G' \times G''), \forall i : s_i = l_i(w)\}| = |\{w \in \mathcal{W}_n(G'), \forall i : s_i = l_i(w)\}| \cdot |\{w \in \mathcal{W}_n(G''), \forall i : s_i = l_i(w)\}|$$

Vishwanathan et al. *J Mach Learn Res* 2010

Random Walks for Graph Classification

Random walk kernel:

$$k_\times(G', G'') = \sum_{i,j=1}^{|V_\times|} \left[\sum_{n=0}^{\infty} \lambda_n E_\times^n \right]_{ij} \quad = 1^T (I - \lambda E_\times)^{-1} 1$$

Proposition w/o proof:

$$k_\times(G', G'') = \boldsymbol{\phi}^T(G')\boldsymbol{\phi}(G'')$$

Efficiently computing geometric series:

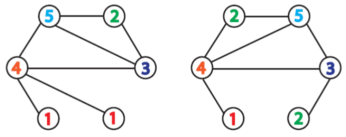
$$\lim_{n \to \infty} \sum_{i=0}^{n} \gamma^i E^i = (I - \gamma E)^{-1}$$

$\forall \gamma < \frac{1}{a}$, where $a = \min\{\Delta^-(G), \Delta^+(G)\}$

Weisfeiler-Lehman Graph Kernels

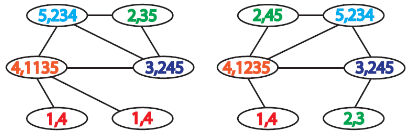
- based on Weisfeiler-Lehman isomorphism test

0)

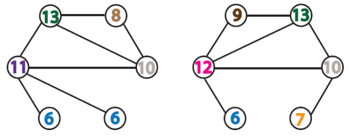


$G' = G'_0$ $\quad$ $G'' = G''_0$

Create new labels



1)



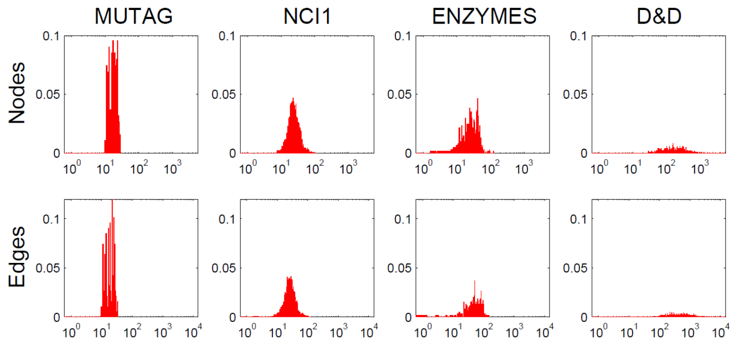
G'_1 $\quad$ G''_1

$$\phi^{(1)}_{\mathrm{WL}}(G') = (2, 1, 1, 1, 1, 2, 0, 1, 0, 1, 1, 0, 1)$$

$$\phi^{(1)}_{\mathrm{WL}}(G'') = (\underbrace{1, 2, 1, 1, 1}_{\text{Counts of original node labels}}, \underbrace{1, 1, 0, 1, 1, 0, 1, 1}_{\text{Counts of compressed node labels}})$$

$$k^{(h)}_{\mathrm{WL}}(G', G'') = k(G'_0, G''_0) + k(G'_1, G''_1) + \ldots + k(G'_h, G''_h)$$

Empirical Evaluation



Method/Data Set	MUTAG	NCI1	NCI109	ENZYMES	D & D
WL subtree	82.05 (±0.36)	82.19 (± 0.18)	82.46 (±0.24)	52.22 (±1.26)	79.78 (±0.36)
WL edge	81.06 (±1.95)	84.37 (±0.30)	84.49 (±0.20)	53.17 (±2.04)	77.95 (±0.70)
WL shortest path	83.78 (±1.46)	84.55 (±0.36)	83.53 (±0.30)	59.05 (±1.05)	79.43 (±0.55)
Ramon & Gärtner	85.72 (±0.49)	61.86 (±0.27)	61.67 (±0.21)	13.35 (±0.87)	57.27 (±0.07)
p-random walk	79.19 (±1.09)	58.66 (±0.28)	58.36 (±0.94)	27.67 (±0.95)	66.64 (±0.83)
Random walk	80.72 (±0.38)	64.34 (±0.27)	63.51 (± 0.18)	21.68 (±0.94)	71.70 (±0.47)
Graphlet count	75.61 (±0.49)	66.00 (±0.07)	66.59 (±0.08)	32.70 (±1.20)	78.59 (±0.12)
Shortest path	87.28 (±0.55)	73.47 (±0.11)	73.07 (±0.11)	41.68 (±1.79)	78.45 (±0.26)

Table 1: Prediction accuracy (± standard deviation) on graph classification benchmark data sets

Summary: Graph Classification

Types of graph kernels:

- based on random walks
- based on small subgraphs (graphlets)

Take home:

- "complete" graph kernels are NP-hard
- useful efficiently computable kernels exist
- domain knowledge needed for specific problems

Graph reconstruction conjecture:

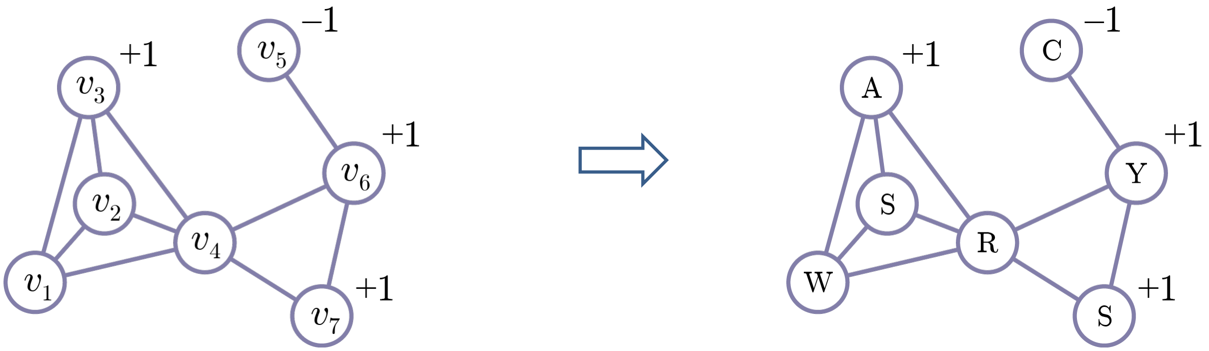
- a graph with n nodes can be reconstructed from all of its subgraphs up to size $n - 1$

Bondy & Hemminger *J Graph Theory* 1977

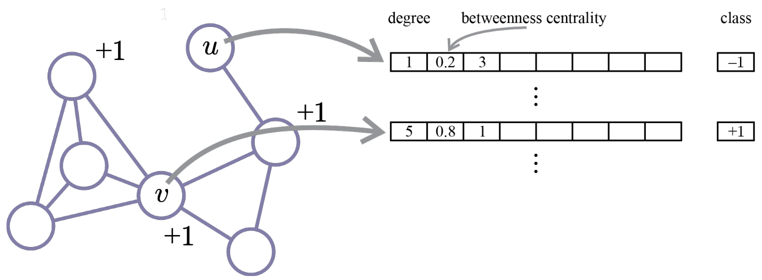
Vertex Classification

$G = (V, E)$
$t : V \to \{-1, +1\}$

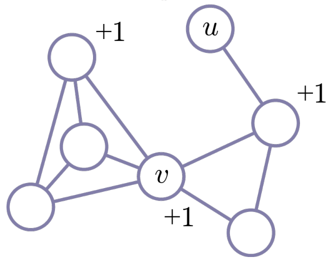
$\Sigma = \{\text{A, C,} \ldots\text{, W, Y}\}$
$label : V \to \Sigma$



Approaches to Vertex Classification



Vector space model



$$w_u = (u, u_1, u_2, \ldots)$$

$$w_v = (v, v_1, v_2, \ldots)$$

$$k(u,v) = \sum_{w_u} \sum_{w_v} k(w_u, w_v)$$

Kernel-based approach

Kernels for Vertex Classification

Given: Graph $G = (V, E)$.

Objective: Design a kernel function

Idea: Diffusion kernel

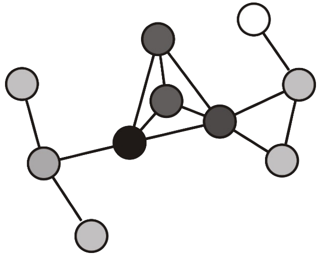
D = diagonal matrix of vertex degrees

E = adjacency matrix

L = Laplacian matrix; $L = D - E$

β = parameter

$$K = e^{-\beta L}$$



Continuous time limit for lazy random walks.

Kondor & Lafferty ICML 2002

FUNCTIONAL FLOW

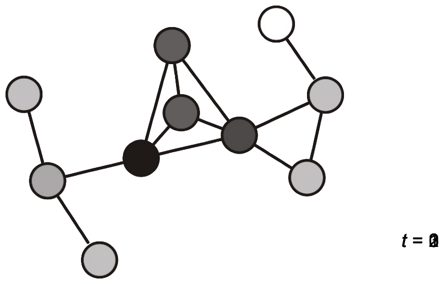
Given: Graph $G = (V, E)$.

Objective: Design a kernel function

Idea 1: simulate flow of liquid

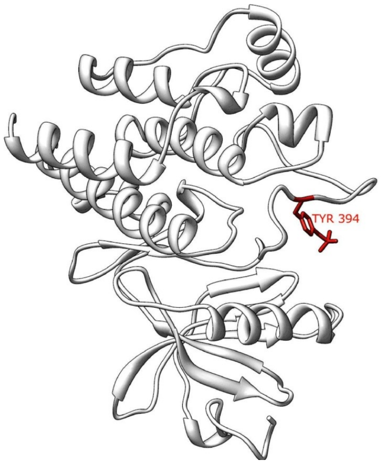
- labeled nodes have a full reservoire
- flow goes from nodes with more to less liquid
- flow that reaches a node in a fixed number of steps is used as prediction

Idea 2: do not look at steady state

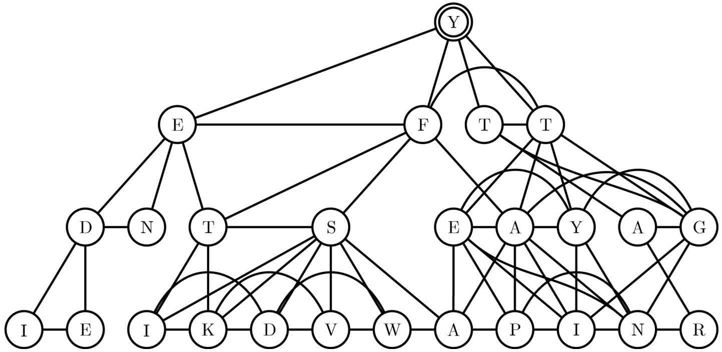


Nabieva et al. *Bioinformatics* 2005

Local Vertex Neighborhood



Y394 of human lymphocyte kinase

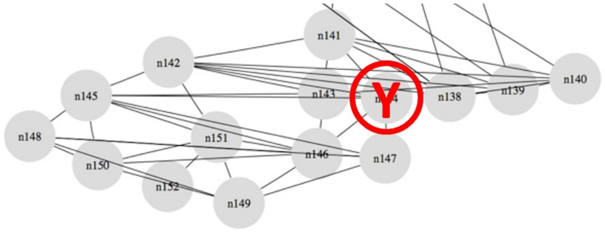


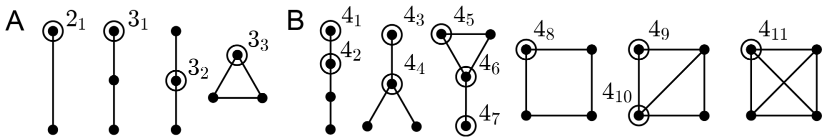
Depth-3 graph neighborhood for Y394

Local Vertex Neighborhood

Idea:

- count small graphs rooted at the vertex of interest;
  i.e., *graphlets*
- create a similarity measure between the counts for two vertices;
  i.e., a *kernel function*
- use kernel functions for SVM classification



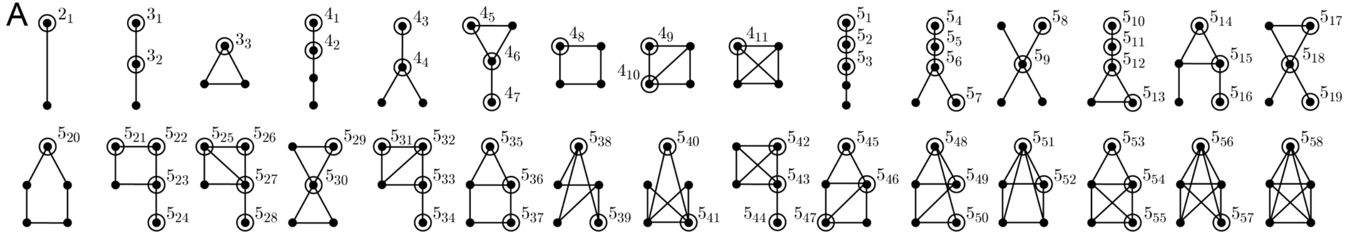


Vacic *et al. J Comput Biol* 2010

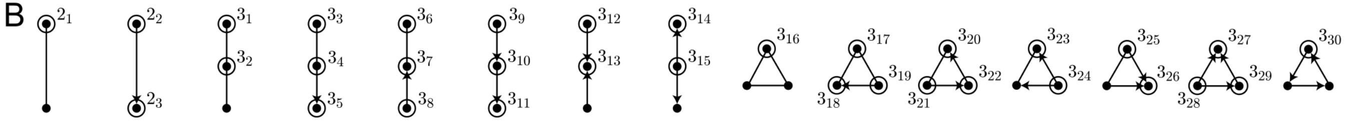
Graphlets

Graphlets: simple small (typically of order 5 or less) connected rooted graphs.

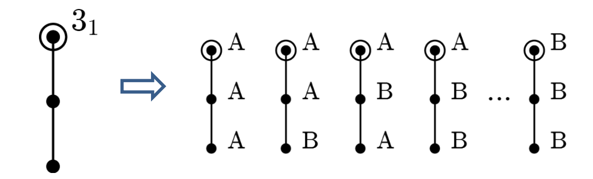
Undirected:



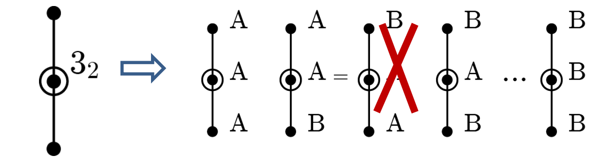
Directed:



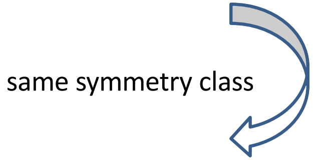
Labeled Graphlets

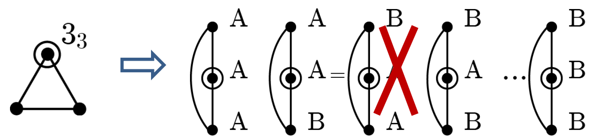


$$|\Sigma|^n = 2^3 = 8$$



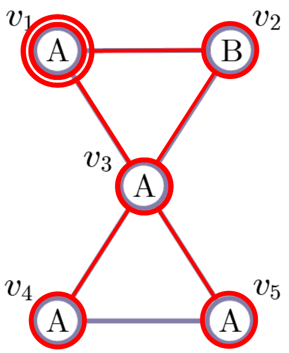
$$|\Sigma| \cdot \binom{|\Sigma|+1}{|\Sigma|-1} = 6$$





$$|\Sigma| \cdot \binom{|\Sigma|+1}{|\Sigma|-1} = 6$$

Example



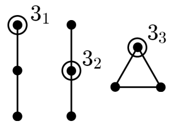
$$V = \{v_1, v_2, v_3, v_4, v_5\} \qquad \Sigma = \{\mathrm{A},\ \mathrm{B}\}$$

$$label : V \to \Sigma$$

$$label\,(v_1) = \mathrm{A}$$

$$label\,(v_2) = \mathrm{B}$$

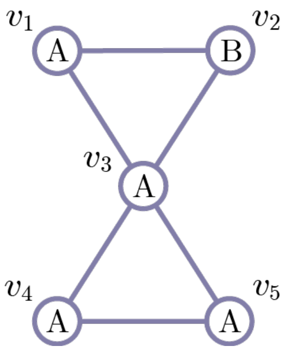
$$\vdots$$



	3_1								3_2						3_3					
	AAA	AAB	ABA	ABB	BAA	BAB	BBA	BBB	AAA	AAB	ABB	BAA	BAB	BBB	AAA	AAB	ABB	BAA	BAB	BBB
$\phi_3(v_1)$	2															1				
$\phi_3(v_5)$	1	1													1					

$$k_3\,(v_1, v_5) = \phi_3^T(v_1)\phi_3(v_5) = 2$$

MORE DETAILS



$$k_n(u,v) = \boldsymbol{\phi}_n^T(u)\boldsymbol{\phi}_n(v)$$

where

$$\boldsymbol{\phi}_n(v) = (\varphi_{n_1}(v), \varphi_{n_2}(v), \ldots, \varphi_{n_{\kappa(n,\Sigma)}}(v))$$

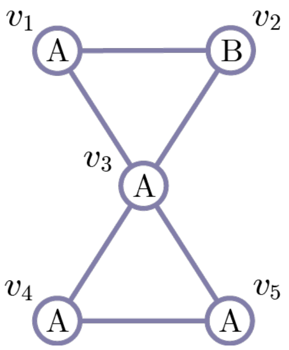
$$k(u,v) = \sum_{n=1}^{N} k_n(u,v)$$

Graphlet kernel

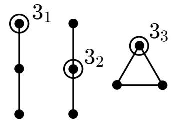
$$k'(u,v) = \frac{k(u,v)}{\sqrt{k(u,u)\,k(v,v)}}$$

Normalized graphlet kernel

Label Mismatch Kernel



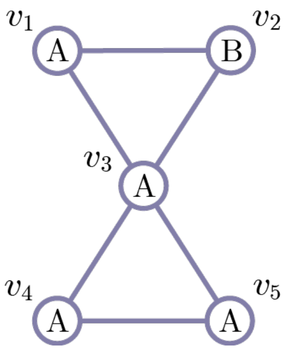
IDEA: Allow approximate matching;
i.e., allow mismatch in labels



3_1								3_2						3_3					
AAA	AAB	ABA	ABB	BAA	BAB	BBA	BBB	AAA	AAB	ABB	BAA	BAB	BBB	AAA	AAB	ABB	BAA	BAB	BBB
2	2	2		2										1	1	1		1	
2	2	1	1	1	1									1	1		1		

$$k^{l}_{(3,1)}(v_1, v_5) = \phi^{T}_{(3,1)}(v_1)\phi_{(3,1)}(v_5) = 14$$

Edge Mismatch Kernel



AGAIN: Allow approximate matching addition or removal of edges

Generalization: graph edit distance!

3_1								3_2						3_3					
AAA	AAB	ABA	ABB	BAA	BAB	BBA	BBB	AAA	AAB	ABB	BAA	BAB	BBB	AAA	AAB	ABB	BAA	BAB	BBB
2	1	1							1					2	1				
3	1							1						2	1				

$$k^e_{(3,1)}(v_1, v_5) = \phi^T_{(3,1)}(v_1)\phi_{(3,1)}(v_5) = 12$$

Empirical Evaluation

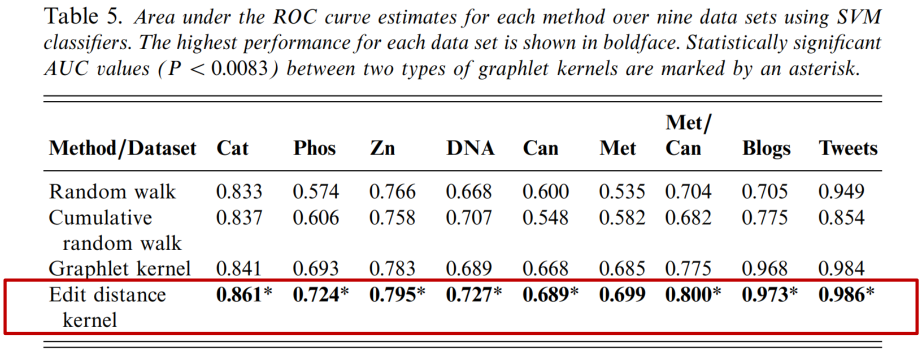


Table 5. *Area under the ROC curve estimates for each method over nine data sets using SVM classifiers. The highest performance for each data set is shown in boldface. Statistically significant AUC values ($P < 0.0083$) between two types of graphlet kernels are marked by an asterisk.*

Method/Dataset	**Cat**	**Phos**	**Zn**	**DNA**	**Can**	**Met**	**Met/ Can**	**Blogs**	**Tweets**
Random walk	0.833	0.574	0.766	0.668	0.600	0.535	0.704	0.705	0.949
Cumulative random walk	0.837	0.606	0.758	0.707	0.548	0.582	0.682	0.775	0.854
Graphlet kernel	0.841	0.693	0.783	0.689	0.668	0.685	0.775	0.968	0.984
Edit distance kernel	**0.861***	**0.724***	**0.795***	**0.727***	**0.689***	**0.699**	**0.800***	**0.973***	**0.986***

Mapping to Hypergraphs

Consider three problems:

- vertex classification
- edge classification
- link prediction

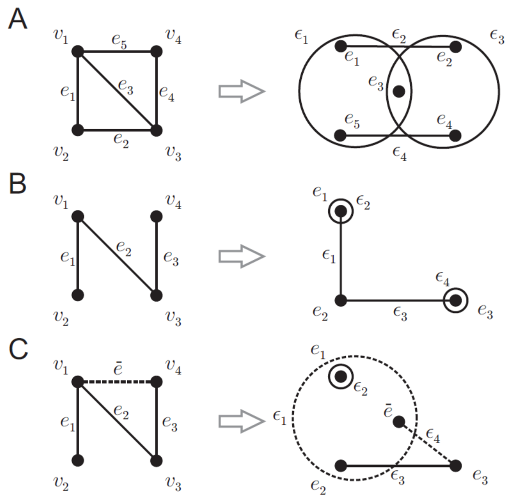
Hypergraph:

$G = (V, E)$

V = a set of vertices

E = a family of non-empty subsets of V

Hypergraph duality:



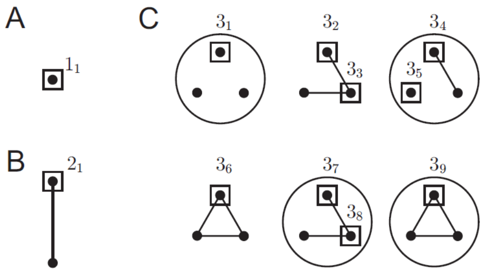
Lugo-Martinez et al. *Bioinformatics* 2020

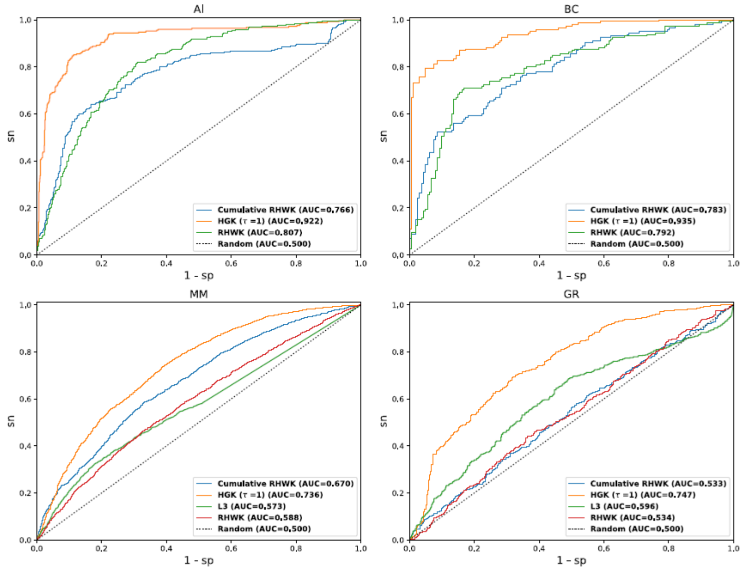
Edit-Distance Hypergraphlet Kernels

Idea: enumerate small hypergraphs

- ◦ section hypergraphs
- ◦ subhypergraphs

Hypergraphlets:





Lugo-Martinez et al. *Bioinformatics* 2020

Thank you