Graph Kernels
A survey

Benedek András Rózemberczki

The University of Edinburgh

2018.10.18.

Overview

Motivation

Graph Kernels

Limitations of Graph Kernels

Motivation

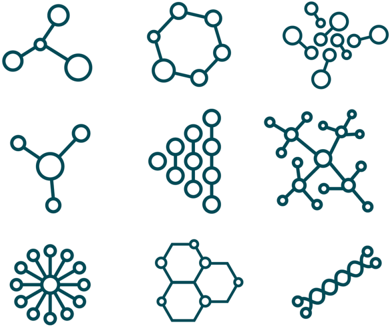
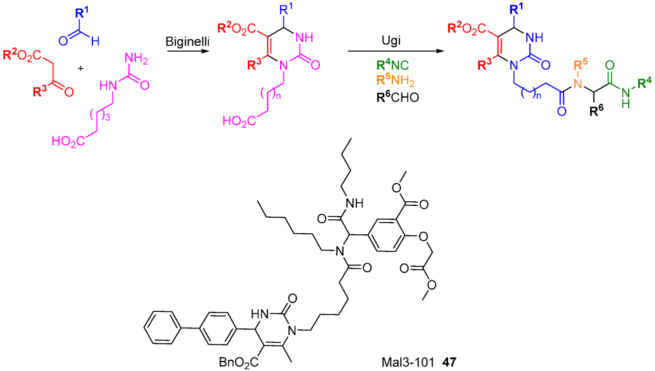


Figure 1: How can we measure the similarity of graphs? How can we capture structural similarity? Molecules might have different size, node features might be missing.

Application – Drug Discovery



Mal3-101 **47**

Figure 2: We could find drugs or just make inference about properties of molecules (Gärtner et al., 2003).

Application – Thread Classification

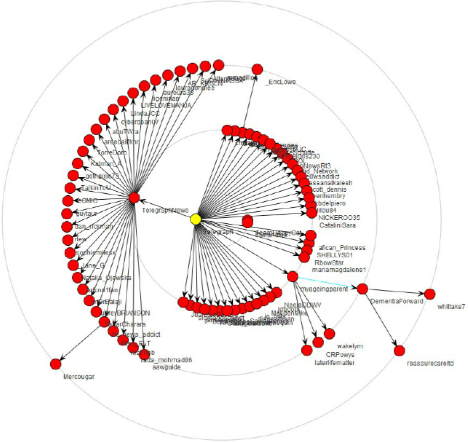


Figure 3: Potentially viral threads have structural properties which describe them (Yanardag, Vishwanathan, 2015).

Application – Fraud Detection

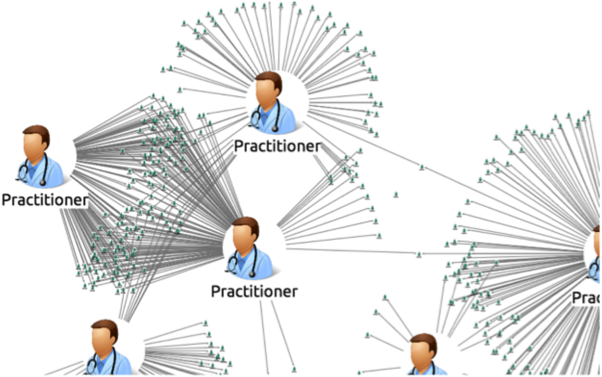


Figure 4: We could classify fraudulent transaction series (Kriege et al., 2016).

R-convolution and subgraph kernels

We have two composite objects $\mathcal{X}$ and $\mathcal{X}'$. The R-convolution kernel Haussler (1999) is defined as:

$$\mathcal{K}(\mathcal{X}, \mathcal{X}') = \sum_{x \subseteq \mathcal{X}} \sum_{x' \subseteq \mathcal{X}'} \mathcal{K}(S, S')$$

We essentially decompose the object into parts and compare everything with everything. Now we let as assume that we have two graphs:

$$G(V, E, L)$$
$$G'(V', E', L')$$

How do we measure similarity?

$$\mathcal{K}(G, G') = \sum_{S \subseteq G} \sum_{S' \subseteq G'} \mathcal{K}(S, S')$$

Calculating $\mathcal{K}(S, S')_{isomorphism}$ is NP-hard so is $\mathcal{K}(G, G')_{isomoprhism}$. We have to be smarter – manual extraction/definition of features.

Types of syntax driven graph kernels

We are interested in special types of subgraphs:

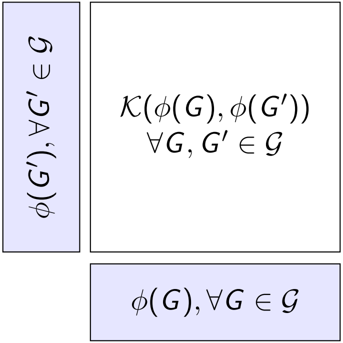
- Paths.
- Graphlets (motifs).
- Tree patterns.

Using a feature vector $\Phi(\cdot)$ of a graph we want to describe the structural properties of the graph. We use a kernel on pairs of graph feature vectors:

$$\mathcal{K}(G, G') = \mathcal{K}(\Phi(G), \Phi(G'))$$

If we compare the similarity of each $G, G' \in \mathcal{G}$ we can define a kernel matrix.

The graph kernel matrix



$\phi(G'), \forall G' \in \mathcal{G}$

$\mathcal{K}(\phi(G), \phi(G'))$
$\forall G, G' \in \mathcal{G}$

$\phi(G), \forall G \in \mathcal{G}$

Figure 5: A graph kernel describes the similarity of two graphs $G, G' \in \mathcal{G}$ using the features of these graphs $\phi(G)$, $\phi(G')$.

Random Walk Kernels

Instead of subgraphs we enumerate all paths (not shortest paths) up to order w. These labeled sequences are in multisets $\mathcal{H}$ and $\mathcal{H}$'. The random walk kernel (Ramon, Gärtner, 2003) between G and G' is calculated as:

$$\mathcal{K}_{RW}(G, G') = \sum_{h \subseteq H} \sum_{h' \subseteq H'} \mathcal{K}(h, h')$$

The marginalized variant (Kashima et al., 2003):

$$\mathcal{K}_{RW}(G, G') = \sum_{h \subseteq H} \sum_{h' \subseteq H'} \mathcal{K}(h, h') P(h|G) P(h'|G')$$

These can be calculated in $O(|V|^3)$ assuming that $|V| > |V'|$.

Limitations of Random Walk Kernels

1. Tottering behavior.
2. Non-expressiveness.
3. Polynomial feature space growth.

Graphlet Kernels (Shervashidze et al., 2009)

Idea: Count graphlet patterns in each graph up to a fixed order w – use the count vectors as feature vectors. The runtime is $O(|V|^w)$.

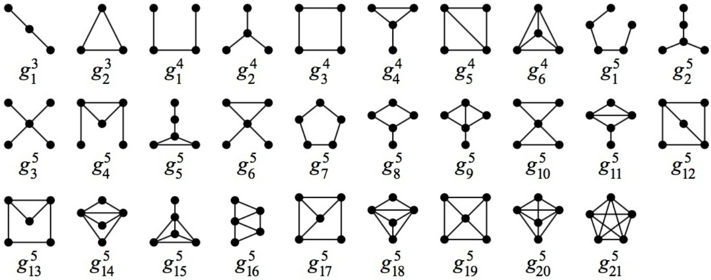


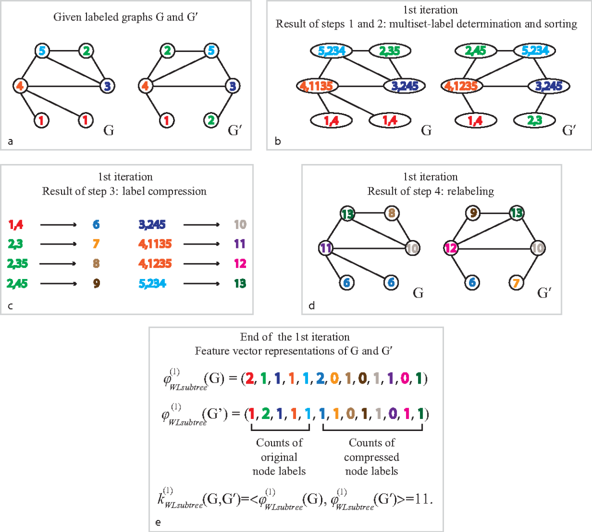
Figure 6: All connected graphlets of size 3,4 and 5.

Subtree Kernels – Weisfeiler-Lehman Kernel (Shervashidze et al., 2011)

The WL algorithm:

1. Multiset creation for each node.
2. Sorting.
3. Compression of labels – hashing.
4. Relabeling.

Weisfeiler-Lehman Kernel


Given labeled graphs G and G′
5
2
4
3
1
1
G
2
5
4
3
1
2
G′
a
1st iteration
Result of steps 1 and 2: multiset-label determination and sorting
5,234
2,35
4,1135
3,245
1,4
1,4
G
2,45
5,234
4,1235
3,245
1,4
2,3
G′
b
1st iteration
Result of step 3: label compression
c
1st iteration
Result of step 4: relabeling
13
8
11
10
6
6
G
9
13
12
10
6
7
G′
d
End of the 1st iteration
Feature vector representations of G and G′
Counts of original node labels
Counts of compressed node labels
e

1,4 → 6		3,245 → 10	
2,3 → 7		4,1135 → 11	
2,35 → 8		4,1235 → 12	
2,45 → 9		5,234 → 13	

$$\varphi^{(1)}_{WLsubtree}(G) = (2,1,1,1,1,2,0,1,0,1,1,0,1)$$

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

$$k^{(1)}_{WLsubtree}(G,G')=<\varphi^{(1)}_{WLsubtree}(G),\ \varphi^{(1)}_{WLsubtree}(G')>=11.$$

Limitations of Graph Kernels

1. Diagonal dominance.
2. Distribution of features counts.
3. Growth of feature space.

Diagonal Dominance

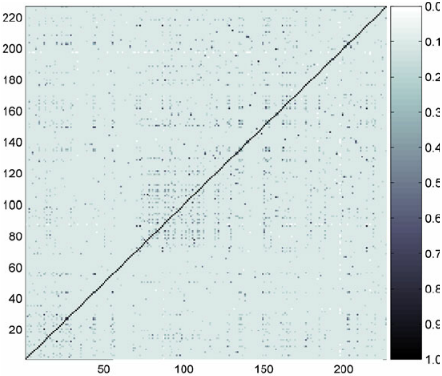
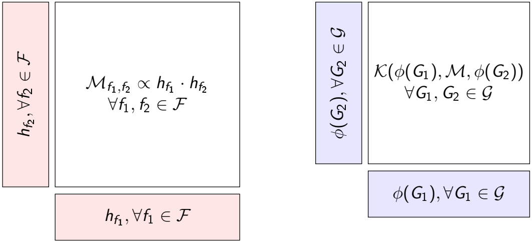


Figure 7: Weisfeiler-Lehman graph kernel matrix for the MUTAG dataset (Shervashidze et al., 2011).

Solutions to the Problems

1. Downsampling the noisy features.
2. Nested hashing of features (Li et al., 2012).
3. Deep graph kernels (Yanardag, Vishwanathan, 2015).
4. Embedding the graphs in a space (Narayanan et al., 2016).

Solution – Deep Graph Kernels (Yanardag, Vishwanathan, 2015)



$h_{f_2}, \forall f_2 \in \mathcal{F}$

$\mathcal{M}_{f_1,f_2} \propto h_{f_1} \cdot h_{f_2}$
$\forall f_1, f_2 \in \mathcal{F}$

$h_{f_1}, \forall f_1 \in \mathcal{F}$

$\phi(G_2), \forall G_2 \in \mathcal{G}$

$\mathcal{K}(\phi(G_1), \mathcal{M}, \phi(G_2))$
$\forall G_1, G_2 \in \mathcal{G}$

$\phi(G_1), \forall G_1 \in \mathcal{G}$

Figure 8: Deep Graph kernel first approximate a similarity matrix $\mathcal{M}$ for $\forall f \in \mathcal{F}$. This $\mathcal{M}$ matrix is approximated and factorized into a d dimensional embedding space with an implicit factorization machine so that the similarity can be calculated on demand while storing $\mathcal{M}$ takes $\mathcal{O}(|\mathcal{F}|{\cdot}d)$ memory instead of $\mathcal{O}(|\mathcal{F}|^2)$. Using $\mathcal{M}$ and the graph features an improved graph kernel matrix $\mathcal{K}$ can be calculated.

Solution – Embedding Graphs (Narayanan et al., 2016)

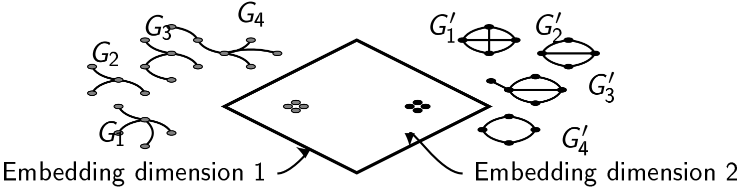


Figure 9: Graph embedding algorithms create low dimensional representations of whole graphs in an embedding space.

Thank You for the kind attention!

Bibliography I

Thomas Gärtner, Peter Flach, Stefan Wrobel. On graph kernels: Hardness results and efficient alternatives. In *Learning Theory and Kernel Machines*, pages 129–143. Springer, 2003.

David Haussler. Convolution kernels on discrete structures. Technical report, Technical report, Department of Computer Science, University of California at Santa Cruz, 1999.

Hisashi Kashima, Koji Tsuda, Akihiro Inokuchi. Marginalized kernels between labeled graphs. In *Proceedings of the 20th international conference on machine learning (ICML-03)*, pages 321–328, 2003.

Nils M Kriege, Pierre-Louis Giscard, Richard Wilson. On valid optimal assignment kernels and applications to graph classification. In *Advances in Neural Information Processing Systems*, pages 1623–1631, 2016.

Bibliography II

Bin Li, Xingquan Zhu, Lianhua Chi, Chengqi Zhang. Nested subtree hash kernels for large-scale graph classification over streams. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 399–408. IEEE, 2012.

Annamalai Narayanan, Mahinthan Chandramohan, Lihui Chen, Yang Liu, Santhoshkumar Saminathan. subgraph2vec: Learning distributed representations of rooted sub-graphs from large graphs. *arXiv preprint arXiv:1606.08928*, 2016.

Jan Ramon, Thomas Gärtner. Expressivity versus efficiency of graph kernels. In *First international workshop on mining graphs, trees and sequences*, pages 65–74. Citeseer, 2003.

Nino Shervashidze, SVN Vishwanathan, Tobias Petri, Kurt Mehlhorn, Karsten Borgwardt. Efficient graphlet kernels for large graph comparison. In *Artificial Intelligence and Statistics*, pages 488–495, 2009.

Bibliography III

Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, Karsten M Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(Sep):2539–2561, 2011.

Pinar Yanardag, SVN Vishwanathan. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1365–1374. ACM, 2015.