The Conjunctive Disjunctive Graph Node Kernel

Dinh Tran Van, Alessandro Sperduti

Department of Mathematics, Padova University, Trieste, 63, 35121 Padova, Italy

Fabrizio Costa

Department of Computer Science, University of Exeter Exeter EX4 4QF, UK

Abstract

Gene-disease associations are inferred on the basis of similarities between the proteins encoded by genes. Biological relationships used to define similarities range from interacting proteins, proteins that participate in pathways and protein expression profiles. Though graph kernel methods have become a prominent approach for association prediction, most solutions are based on a notion of information diffusion that does not capture the specificity of different network parts. Here we propose a graph kernel method that explicitly models the configuration of each gene's context. An empirical evaluation on several biological databases shows that our proposal achieves state-of-the-art results.

Keywords: Graph node kernels, graph decomposition, disease gene prioritization

1. Introduction

Predictive systems for gene-disease associations are often based on a notion of similarity between genes. A common strategy is to encode relations between genes as a network and use graph based techniques to make useful inferences. In the last decades, a number of graph kernel methods have been proposed that directly exploit transitive properties in biological networks. The prototypical method is the Diffusion kernel (DK) [1] inspired by the heat diffusion phenomenon. The key idea is to allow a given amount of *heat* placed on nodes to *diffuse* through the edges. The similarity between two nodes v_i, v_j is then defined as the amount of heat starting from v_i and reaching v_j within a given time interval. In DK the heat flow is proportional to the number of paths connecting two nodes, which introduces an undesired bias that penalize peripheral nodes w.r.t. central ones. This problem is tackled by a modified version of DK called Markov exponential diffusion kernel (MED) [2] where a Markov matrix replaces the Laplacian matrix. Another kernel called Markov diffusion kernel (MD) [3], exploits instead the notion of *diffusion distances*, a measure of similarity between patterns of heat diffusion. The Regularized Laplacian kernel (RL) [4] represents instead a normalized version of the random walk with restart model and defines the node similarity as the number of paths connecting two nodes with different lengths. All these approaches can be applied to dense networks with high degree nodes. A drawback of these approaches is however their relatively low discriminative capacity. This is in part due to the fact that information is processed in an additive and independent fashion which prevents them from accurately modeling the configuration of each gene's context. To address this issue here we propose to employ a *decompositional* graph kernel (DGK) [5] technique in which the similarity function between graphs can be formed by decomposing each graph into subgraphs and by devising a valid local kernel between the subgraphs. To exploit its higher discriminative capacity we first decompose the network in a collection of connected sparse graphs and then we develop a suitable kernel, that we call Conjunctive Disjunctive Node Kernel (CDNK).

2. Material and methods

We start from the type of similarity notion computed by decomposition kernels between graph instances and adapt it to express the similarity between nodes in a single network. In this work we use three key ideas: 1) genes are described using their functional profile encoded as a vector of real values, 2) the network is marked to distinguish highly connected components from sparsely connected ones, and 3) we transform the neighborhood of each gene in a sparse high dimensional vector that can be easily processed by standard machine learning techniques such as SVMs.

Definitions. We represent a problem instance as a graph $G = (V, E)$ where V is the set of nodes and E is the set of links. We define the *distance* $\mathcal{D}(u, v)$ between two nodes u and v, as the number of edges on the shortest path between them. The *neighborhood* of a node u with radius r, $N_r(u) = \{v \mid d(u, v) \leq r\}$, is the set of nodes at distance no greater than r from u. The corresponding *neighborhood subgraph* $\mathcal{N}_r^u$ is the subgraph induced by the neighborhood (i.e. considering all the edges with endpoints in $N_r(u)$). The *degree* of a node u, $d(u) = |\mathcal{N}_1^u|$, is the cardinality of its neighborhood. The maximum node degree in the graph G is $d(G)$.

2.1. Gene Labeling

Gene-disease associations networks typically represent genes as nodes labeled with a gene identifier. Here we take a different approach and use the node labels to encode abstract information about the genes. In this way downstream machine learning algorithms can generalize from similar examples and allow the identification of overlooked but related genes. We experiment with two types of information: 1) topological information and 2) functional information.

Topological label. This information is simply based on the connectivity degree of the gene. The idea is that genes that have the same number of connections are more similar than genes with a different connectivity. The node labeling function ℓ assigns the degree for nodes u having degree less than or equal than a user defined threshold T ($T = 5$ in our experimental evaluation). However degree values larger than T are subsequently discretized into k levels. Formally, the labeling function is defined as:

$$\ell(u) = \begin{cases} d(u), & \text{if } d(u) \leq T \\ T + i, & \text{if } d(u) > T \end{cases},$$

where $i = \lceil \frac{d(u)-T}{bin} \rceil$, $bin = \frac{d(G)-T}{\lambda-T}$ and λ $(\lambda > T)$ is the maximum number of symbols used. The value of λ depend on the degree distribution and can be tuned as a hyperparameter of the approach.

Functional label. This type of information is based on the *Gene Ontology* [6] resource. We use the ontology to construct binary vectors representing a bag-of-words encoding for each gene (i.e. if one of the 26501 GO-terms is associated with the gene). The resulting vectors are then clustered using the k-means algorithm into a user defined number of classes K (tuned as a hyperparameter of the approach), so that genes with similar description profiles receive the same class identifier as label.

Real valued vector information encoding. In addition to encoding the functional information as a discrete label we add a richer description by computing the similarity vector w.r.t. to each cluster. In this way we can fully exploit the latent description of the genes in terms of the different functional groups captured by the clustering procedure. Formally, given a vector $v \in \mathbb{R}^{26501}$ we compute a similarity vector $S(v) = \{s_1, s_2, \dots s_K\}$ with entries $s_i = \frac{1}{1+d(v,c_i)}$ where $d(v, c_i)$ is the euclidean distance of v from the center of the ith cluster $c_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$ computed as the geometric mean of the elements in the cluster C_i.

2.2. Network Decomposition

In gene-disease associations networks it is not uncommon to find nodes with high degrees. Unfortunately these cases cannot be effectively processed by a neighborhood based decomposition kernel since these are based on the notion of exact matches and the probability of finding identical neighborhoods decreases exponentially as the degree increases. This means that in a finite network it quickly becomes impossible to find any match and hence learn or generalize at all. As an alternative, we propose a procedure to "sparsify" the network that is observed by the neighborhood kernel. In reality we do not alter the cardinality of the edge set, but rather mark the edges with special attributes that will inform the kernel computation. The result is a procedure that decomposes the network in a linked collection of sparse sub-networks where each node has a reduced

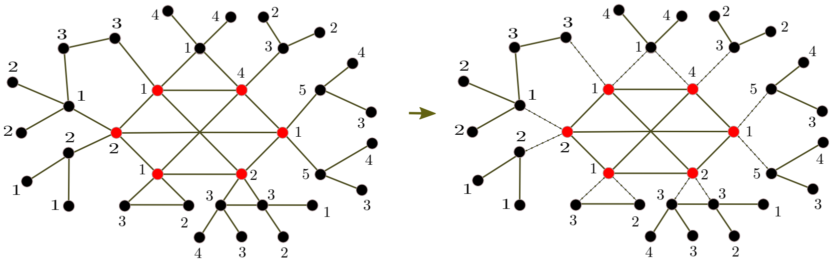

Figure 1: K-core decomposition

connectivity when considering the edges of a specific type. However the other edges are still available to connect the various sub-networks. We distinguish two types of edges: *conjunctive* and *disjunctive* edges. Nodes linked by conjunctive edges are going to be used jointly to define the notion of context and will be visible to the neighborhood graph kernel. Nodes linked by disjunctive edges are instead used to define features based only on the pairwise co-occurrence of the genes at the endpoints and are processed by our novel kernel.

Iterative k-core decomposition [7]: The node set is partitioned in two groups on the basis of the degree of each node w.r.t. a threshold degree D, the first part contains all nodes with degree smaller or equal than D and the second part the remaining ones. The node partition is used to induce the ”conjunctive’ vs ”disjunctive’ notion for the edge partition: edges that have both endpoints in the same part (be it the first or the second) are marked as conjunctive, otherwise they are marked as disjunctive. We apply the k-core decomposition iteratively, where at each iteration we consider only the graph induced by the conjunctive edges. We stop iterating the decomposition after a user defined number of steps. Note that this decomposition does not alter the cardinality of the edge set, it is simply a procedure to mark each edge with the attribute conjunctive or disjunctive.

Clique decomposition [8]: To model the notion that nodes in a clique are tightly related, we summarize the whole clique with a new ’representative’ node. All the cliques (completely connected subgraphs) with a number of nodes greater than a threshold size C are identified. The endpoints of all edges incident on

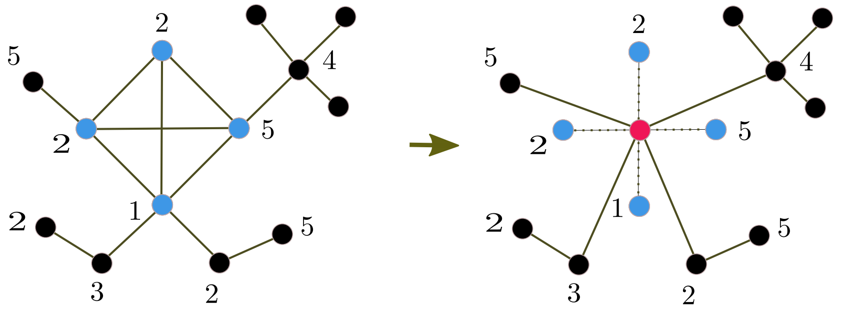

Figure 2: Clique decomposition

the clique's nodes are moved to the representative node. Disjunctive edges are introduced to connect each node in the clique to the representative. Finally all edges with both endpoints in the clique are removed.

In our work a network is transformed by applying first the iterative k-core decomposition and then the clique decomposition.

2.3. Node Graph Kernels

We start from the Neighborhood Subgraph Pairwise Distance Kernel (NSPDK) [9] and adapt it to express the similarity between nodes in a single network. The key idea in NSPDK is to decompose graphs in small fragments and count how many pairs of fragments are shared between two instances. We introduce two improvements: 1) we partition the features according to the individual node's neighborhood, and 2) we introduce a distinction between "disjunctive" and "conjunctive" edges.

2.3.1. The Neighborhood Subgraph Pairwise Distance Kernel

The NSPDK is an instance of convolution kernel [5] where given a graph $G \in \mathcal{G}$ and two rooted graphs A_u, B_v, the relation $R_{r,d}(A_u, B_v, G)$ is true *iff* $A_u \cong \mathcal{N}_r^u$ is (up to isomorphism $\cong$) a neighborhood subgraph of radius r of G and so is $B_v \cong \mathcal{N}_r^v$, with roots at distance $\mathcal{D}(u, v) = d$. We denote R^{-1} as the inverse relation that returns all pairs of neighborhoods of radius r at distance d in G, $R_{r,d}^{-1}(G) = \{A_u, B_v | R_{r,d}(A_u, B_v, G) = true\}$. The kernel $\kappa_{r,d}$ over $\mathcal{G} \times \mathcal{G}$, counts the number of such fragments in common in two input graphs:

$$\kappa_{r,d}(G, G^{'}) = \sum_{\substack{A_u, B_v \;\in\; R^{-1}_{r,d}(G) \\ A'_{u'}, B'_{v'} \;\in\; R^{-1}_{r,d}(G')}} \mathbf{1}_{A_u \cong A'_{u'}} \cdot \mathbf{1}_{B_v \cong B'_{v'}},$$

where $\mathbf{1}_{A \cong B}$ is the *exact matching function* that returns 1 if A is isomorphic to B and 0 otherwise. Finally, the NSPDK is defined as $K(G, G') = \sum_r \sum_d \kappa_{r,d}(G, G')$, where for efficiency reasons, the values of r and d are upper bounded to a given maximal r^* and d^*, respectively.

2.3.2. The Conjunctive Disjunctive Node Kernel

We extend NSPDK and define a node kernel $K(G_u, G_{u'})$ between two copies of the same network G where we distinguish the nodes u and u' respectively. The idea is to define the features of a node u as the subset of NSPDK features that always have the node u as one of the roots. In addition we distinguish between two types of edges, called *conjunctive* and *disjunctive* edges. When computing distances to induce neighborhood subgraphs, only conjunctive edges are considered. When choosing the pair of neighborhoods to form a single feature, we additionally consider roots u and v that are not at distance d but such that u is connected to w via a disjunctive edge and such that w is at distance d from v. In this way disjunctive edges can still allow an *information flow* even if their endpoints are only considered in a pairwise fashion and not jointly.

Formally, we define two relations: the *conjunctive relation* $R^{\wedge}_{r,d}(A_u, B_v, G_u)$ identical to the NSPDK relation $R_{r,d}(A_u, B_v, G)$, and (ii) $\mathcal{D}(u, v) = d$; the *disjunctive relation* $R^{\vee}_{r,d}(A_u, B_v, G_u)$ is true *iff* (i) $A_u \cong \mathcal{N}^u_r$ and $B_v \cong \mathcal{N}^u_r$ are true, (ii) $\exists w$ s.t. $\mathcal{D}(w, v) = d$, and (iii) (u, w) is a disjunctive edge. We define $\kappa_{r,d}$ on the inverse relations ${R^{\wedge}_{r,d}}^{-1}$ and ${R^{\vee}_{r,d}}^{-1}$

$$\kappa_{r,d}(G_u, G_{u'}) = \sum_{\substack{A_u, B_v \in {R^{\wedge}_{r,d}}^{-1}(G_u) \\ A'_{u'}, B'_{v'} \in {R^{\wedge}_{r,d}}^{-1}(G_{u'})}} \mathbf{1}_{A_u \cong A'_{u'}} \cdot \mathbf{1}_{B_v \cong B'_{v'}} + \sum_{\substack{A_u, B_v \in {R^{\vee}_{r,d}}^{-1}(G_u) \\ A'_{u'}, B'_{v'} \in \; {R^{\vee}_{r,d}}^{-1}(G_{u'})}} \mathbf{1}_{A_u \cong A'_{u'}} \cdot \mathbf{1}_{B_v \cong B'_{v'}}.$$

The CDNK is finally defined as $K(G_u, G_v) = \sum_r \sum_d \kappa_{r,d}(G_u, G_v)$, where once again for efficiency reasons, the values of r and d are upper bounded to a given maximal r^* and d^*.

2.4. Real valued node information

In order to process real vectors we do a convolution of the vector with the node sparse vector.

3. Experimental

We perform an empirical evaluation of the predictive performance of several kernel based methods on two of the databases used in [2].

BioGPS: A gene co-expression network is constructed from BioGPS dataset, which contains 79 tissues, measured with the Affymetrix U133A array. Edges are inserted when the pairwise Pearson correlation coefficient (PCC) between genes is larger than 0.5.

Pathways: Pathway information is retrieved from KEGG, Reactome, PharmGKB and the Pathway Interaction Database. If a couple of proteins co-participate in any pathway, the two corresponding genes are linked.

To evaluate the performance of graph node kernels we analyze the *gene prioritization*, i.e. given a set of genes known to be associated to a given disease, gene prioritization is the task to rank the candidate genes based on their probabilities to be related to that disease. Similar to the evaluation process used in [2], we choose 12 diseases [10] with at least 30 confirmed genes. For each disease, we construct a positive set $\mathcal{P}$ with all confirmed disease genes, and a negative set $\mathcal{N}$ which contains random genes associated at least to one disease class which is not related to the class that is defining the positive set. In [2] the ratio between the dataset sizes is chosen as $|\mathcal{N}| = \frac{1}{2}|\mathcal{P}|$. The predictive performance of each method is evaluated via a leave-one-out cross validation: one gene is kept out in turn and the rest are used to train an SVM model. We compute a decision score q_i for the test gene g_i as the top percentage value of score s_i among all candidate gene scores. We collect all decision scores for every gene in the training set to form a global decision score list on which we compute the AUC ROC.

Model Selection: The hyper parameters of the various methods are set using a k-fold on a dataset set that is then never used in the predictive performance estimate. We try the values for diffusion parameter in DK and MED in $\{10^{-3}, 10^{-3}, 10^{-2}, 10^{-1}\}$, time steps in MD in $\{1, 10, 100\}$ and RL parameter in $\{1, 4, 7\}$. For CDNK, we try for the degree threshold values in $\{10,\ 15,\ 20\}$, clique size threshold in $\{4,\ 5\}$, maximum radius in $\{1,\ 2\}$, maximum distance in $\{2,\ 3,\ 4\}$. Finally, the regularization trade off parameter C for the SVM is searched in $\{10^{-5},\ 10^{-4},\ 10^{-3},\ 10^{-2},\ 10^{-1}, 1,\ 10,\ 10^{2}\}$.

4. Results and discussion

Table 1 and 2 show the AUC performance of the models trained by using different graph node kernels on 11 gene-disease association problems using the BioGPS and Pathways datasets to materialize the gene relation network. We note that CDNK ranks first in 7 out of 11 cases using both networks. CDNK is the top performant kernel also when considering the average AUC ROC and the average rank with 73.3/2.0, 76.5/1.8, with a difference w.r.t. the second best of 5.5% and 1% on BioGPS and Pathways, respectively. MED and RL show similar and moderate results with small gap between them. DK and MD exhibit modest performance on average and are ranked last in many cases: 7 times out of 11 for MD in BioGPS and 10 out of 11 for DK in Pathways. Low performance cases include networks that have high *average* node degree which is likely to yield very sparse and fragmented network decompositions.

5. Conclusions

We have shown how decomposing a network in a set of connected sparse graphs allows us to take advantage of the discriminative power of CDNK, a novel decomposition kernel, to achieve state-of-the-art results. In future work we will investigate how to 1) decompose networks in a data driven way and 2) extend the CDNK approach to gene-disease association problems exploiting multiple heterogeneous information sources in a joint way.

	BioGPS							
Disease	DK	MD	MED	RL	CDNK1	CDNK2	CDNK3	CDNK4
1	51.9/8	57.4/7	59.0/6	59.2/5	65.1/4	69.5/2	69.3/3	**70.3/1**
2	81.7/5	78.5/6	75.2/7	75.0/8	88.3/2	**88.8/1**	85.1/4	86.8/3
3	64.3/7	59.6/8	71.6/3	71.8/2	65.5/5	**72.5/1**	64.7/6	66.4/4
4	65.3/7	58.2/8	67.8/6	67.8/5	71.9/4	**78.7/1**	73.9/3	77.0/2
5	64.0/8	64.1/7	66.5/5	66.2/6	75.9/4	76.2/3	76.9/2	**77.4/1**
6	74.6/5	70.2/8	71.0/7	71.2/6	79.3/2	**83.7/1**	76.7/4	79.0/3
7	73.0/5	66.7/8	75.4/3	75.6/2	68.8/6	73.9/4	67.3/7	**76.9/1**
8	74.4/8	76.8/3	76.2/5	76.4/4	74.7/7	**77.7/1**	76.0/6	76.8/2
9	71.5/2	65.6/8	67.7/5	69.9/3	66.8/7	**71.7/1**	67.1/6	68.1/4
10	54.0/6	50.3/8	56.1/5	51.1/7	77.6/4	**82.7/1**	80.5/2	80.0/3
11	58.2/7	51.3/8	59.3/6	59.3/5	71.8/4	**80.2/1**	75.3/3	77.1/2
$\overline{AUC}$	66.6	63.5	67.8	67.6	73.2	**77.8**	73.9	76.0
$\overline{Rank}$	6.18	7.18	5.27	4.82	4.45	**1.55**	4.18	2.36

Table 1: *Predictive performance on 11 gene-disease associations in percentage using network induced by the BioGPS. Best results are in bold. We report the AUC ROC and the rank for each kernel method in which CDNK1 uses ontology for discrete labels, CDNK2 uses ontology for both discrete and continous labels, CDNK3 uses node degree for discrete labels and CDNK4 uses degree and ontology for both discrete and continous labels, respectively.*

	Pathways							
Disease	DK	MD	MED	RL	CDNK1	CDNK2	CDNK3	CDNK4
1	74.7/8	76.4/7	78.7/6	78.8/5	80.2/3	**82.4/1**	80.6/2	79.4/4
2	55.1/8	64.9/7	76.6/6	76.6/5	81.1/2	80.3/3	79.8/4	**82.2/1**
3	55.0/8	62.7/7	64.1/5	65.6/4	67.1/3	63.6/6	68.5/2	**71.5/1**
4	54.3/8	65.2/7	**73.7/1**	73.7/2	66.1/5	68.1/3	67.5/4	65.6/6
5	52.9/8	55.7/7	62.7/5	62.7/6	68.3/2	**69.6/1**	66.3/4	66.8/3
6	83.4/8	92.7/7	96.5/2	**96.5/1**	93.0/6	94.1/5	94.4/4	95.5/3
7	84.5/8	88.3/7	89.4/3	89.5/2	88.5/6	88.5/5	88.7/4	**90.5/1**
8	53.7/8	65.6/7	72.0/6	72.3/5	72.5/3	72.5/2	72.3/4	**76.5/1**
9	52.5/8	64.9/5	64.2/7	64.2/6	**81.3/1**	81.0/2	79.6/3	78.8/4
10	68.8/6	65.4/8	74.4/5	74.4/4	66.9/7	76.8/2	76.1/3	**79.5/1**
11	53.7/8	69.2/7	74.6/5	74.1/6	77.0/2	**78.7/1**	75.4/4	77.0/3
$\overline{AUC}$	62.6	70.1	75.2	75.3	76.5	77.8	77.2	**78.5**
$\overline{Rank}$	7.82	6.91	4.64	4.18	3.64	2.82	3.45	**2.55**

Table 2: *Predictive performance on 11 gene-disease associations in percentage using network induced by the Pathways. Best results are in bold. We report the AUC ROC and the rank for each kernel method in which CDNK1 uses ontology for discrete labels, CDNK2 uses ontology for both discrete and continous labels, CDNK3 uses node degree for discrete labels and CDNK4 uses degree and ontology for discrete and continous labels, respectively.*

References

[1] R. I. Kondor, J. Lafferty, Diffusion kernels on graphs and other discrete input spaces, in: ICML, Vol. 2, 2002, pp. 315–322.

[2] B. Chen, M. Li, J. Wang, F.-X. Wu, Disease gene identification by using graph kernels and markov random fields, Science China. Life Sciences 57 (11) (2014) 1054.

[3] F. Fouss, L. Yen, A. Pirotte, M. Saerens, An experimental investigation of graph kernels on a collaborative recommendation task, in: Data Mining, 2006. ICDM'06. Sixth International Conference on, IEEE, 2006, pp. 863–868.

[4] P. Chebotarev, E. Shamis, The matrix-forest theorem and measuring relations in small social groups, arXiv preprint math/0602070.

[5] D. Haussler, Convolution kernels on discrete structures, Tech. rep., Technical report, Department of Computer Science, University of California at Santa Cruz (1999).

[6] G. O. Consortium, et al., The gene ontology (go) database and informatics resource, Nucleic acids research 32 (suppl 1) (2004) D258–D261.

[7] J. I. Alvarez-Hamelin, L. Dall'Asta, A. Barrat, A. Vespignani, k-core decomposition: A tool for the visualization of large scale networks, arXiv preprint cs/0504107.

[8] R. E. Tarjan, Decomposition by clique separators, Discrete mathematics 55 (2) (1985) 221–232.

[9] F. Costa, K. De Grave, Fast neighborhood subgraph pairwise distance kernel, in: Proceedings of the 26th International Conference on Machine Learning, Omnipress, 2010, pp. 255–262.

[10] K.-I. Goh, M. E. Cusick, D. Valle, B. Childs, M. Vidal, A.-L. Barabási, The human disease network, Proceedings of the National Academy of Sciences 104 (21) (2007) 8685–8690.