Conjunctive Disjunctive Edge Graph Node Kernel

Dinh Tran Van[1], Alessandro Sperduti[1] and Fabrizio Costa[2]

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

2- Bioinformatics Group, Department of Computer Science, Freiburg University
Georges-Kohler-Allee 106, 79110 Freiburg, Germany

Abstract. The node proximity measure is one of the key points that determines the performance of the genetic graph based predictive systems. A considered number of graph node kernels have been proposed to define the proximity between nodes by exploiting graph properties. However, most kernels are based on the natural transitive property that cannot precisely capture the graph configuration. As a consequence, they do not show the high discriminative capacity. In this paper, we first propose a graph decomposition method using conjunctive and disjunctive edges to sparse graphs. Then we introduce a new kernel called Conjunctive Disjunctive Edge Graph Node Kernel, which is an instance of decomposition kernel, for node similarity measure. The empirical evaluation results show that our proposed kernel outperforms all the compared kernels.

1 Introduction

In genetic predictive systems, the definition of gene-gene similarity is one of the key aspects that determines the performance of the systems. A popular approach is to use a domain specific notion of relatedness between genes and materialize a graph where nodes are genes and edges are relationships that satisfy a stringent criterion (e.g. if the products of the two genes interact in a known pathway). Once the graph is materialized, one can define the node similarities by exploiting the graph properties.

Concerning graph node similarity measure, graph node kernels are known as the most effective paradigm to use. In the last decades, a number of kernels have been proposed. Most kernels are based on the natural transitive property (i.e connecting two nodes via a path). For instance, diffusion kernel takes into account all the paths connecting two nodes to compute the similarity. On the positive side, these approaches can be employed on networks that: (1) are dense, (2) have nodes with high maximal degrees, (3) have cliques. On the negative side, they do not have a high discriminative capacity since they (1) are based on considering in an additive and independent fashion. (2) employ a representation that is aware of the distance between nodes but cannot model the precise configuration of the context.

In order to increase the modelling discriminative capacity we propose to employ decomposition graph kernel (DGK) [1]. On one side, this approach is computationally efficient (quasi linear in the network size) and can adequately

represent contextual information. On the other side, it can be applied on networks that are sparse and have a small maximal node degree. In this paper, we first pose a new graph decomposition method whose edges are in either conjunctive or disjunctive form. Successively, we adjust the idea mentioned in [6] which has been originally proposed for graph similarities to introduce a new graph node kernel named Conjunctive Disjunctive Edge Graph Node Kenrel (CDGK) that can fully exploit the graph to measure the node similarity.

The paper is organized as follow: first we describe the state of the art concerning graph node kernels in Section 2. Second, we present the proposed graph decomposition method and the CDGK in the Section 3. Then, we evaluate the performance of different kernels and show the results in in Section 4 and 5, respectively. Finally, we make a conclusion in Section 6.

2 Related Work

In this section, we present the most used graph node kernels. These kernels are relied on the random walk paradigm, namely the similarity between two nodes is measured by taking into account the paths connecting them.

The most well-known graph node kernel named Diffusion kernel (DK) is proposed in [2]. This kernel is based on the heat diffusion phenomenon. First, a given amount of heat is put on each node and diffused through the edges of the graph in an arbitrary time interval. Then the similarity between the node couple (v_i, v_j) is measured as the amount of heat starting from v_i and reaching v_j within the given time. In DK, the similarity between two high degree nodes is generally higher compared to that between two degree ones. Intuitively, the more paths connect two vertices, the more heat can flow between them. This could be problematic since peripheral nodes have unbalanced similarities with respect to central nodes. In order to make the strength of individual vertices comparable, a modified version of DK is introduced in [3] called Markov exponential diffusion kernel (MED) in which it replaces the Laplacian matrix in DK by a Markov matrix. Another kernel called Markov diffusion kernel (MD) is introduced in [4]. It works by exploiting the idea of diffusion distance, which is a measure of how similar the pattern of heat diffusion is among a pair of initialized nodes. In other words, it expresses how much nodes influences each other in a similar fashion. If their diffusion ways are alike, the similarity will be high and, vice versa. The Regularized Laplacian kernel (RL) [5] represents a normalized version of the random walk with restart model. It measures the node similarity by counting the paths connecting two nodes with different lengths.

The above kernels face with the discrimination problem, especially in the graphs that have many disconnected components. To overcome that limitation, in this study, we first introduce a graph decomposition method which utilizes two forms of edges: Conjunctive and Disjunctive. Then we propose a new graph node kernel based on the Neighborhood Subgraph Pairwise Distance Kernel (NSPDK) [6]. The proposed kernel takes the decomposed graph achieved from decomposition process as its input. The detail of each method is described in

the next section.

3 Method

3.1 Notation and Definitions

We intently follow the notations used in [6]. A graph $G = (V, E)$ is a structure that consists of two sets: a node set V and an edge set E. The notation *V(G)* and *E(G)* are used to refer to the node set and edge set of G. The *distance* between two vertices u and v, notated as $\mathcal{D}(u, v)$, is the length of the shortest path between them. The *neighborhood* with radius r of a vertex v is the set of vertices at a distance no greater than r from v and denoted by $N_r(v)$. The *induced subgraph* $\mathcal{N}(W)$ is a graph induced from G with the node set W and the edge set containing every edge in G whose endpoints are in W. The *neighborhood subgraph* with radius r of vertex v is the subgraph induced by the neighborhood with radius r of v and is denoted by $\mathcal{N}_r^v$. The *degree* of a node v is the cardinality of its neighborhood set with radius 1 and is denoted as *d(v)*. A *clique* of the graph G is a fully connected subgraph of G.

3.2 The Neighborhood Subgraph Pairwise Distance Kernel

The NSPDK is an instance of convolution kernel which is designed for measuring the similarity between graphs.

Given a graph $G \in \mathcal{G}$ ($\mathcal{G}$ is the graph domain) and two rooted graphs A_u, B_v, the relation $R_{r,d}(A_u, B_v, G)$ is defined to be true iff A_u and B_v are in $\{\mathcal{N}_r^v : v \in V(G)\}$, where A_u (B_v) needs to be isomorphic with some $\mathcal{N}_r$ and $\mathcal{D}(u, v) = d$. We denote R^{-1} as the inverse relation that returns subgraphs of G, $R_{r,d}^{-1}(G) = \{A_u, B_v | R_{r,d}(A_u, B_v, G)\}$. The kernel $\kappa_{r,d}$ over $\mathcal{G} \times \mathcal{G}$ takes into account the number of identical neighboring graph pairs with radius r at distance d between two graph and is formulated as:

$$\kappa_{r,d}(G, G') = \sum_{\substack{A_v, B_u \in R_{r,d}^{-1}(G) \\ A'_{v'}, B'_{u'} \in R_{r,d}^{-1}(G')}} \sigma(A_v, A'_{v'})\sigma(B_u, B'_{u'}),$$

where $\sigma(x, y)$ is the *exact matching function* that returns 1 if x is isomorphic to y and 0 otherwise. In order to solve the graph isomorphism problem, an efficient approximate algorithm is also proposed in [6]. Finally, the NSPDK is defined as $K(G, G') = \sum_r \sum_d \kappa_{r,d}(G, G')$. For efficiency issue, we limit the values of r and d with the upper bounds r^* and d^*, respectively.

3.3 Graph Decomposition

In this section, we introduce a new method for graph decomposition. The method is based on the idea of kcore and clique decomposition discussed in [8], [9] respectively. Besides, we pose to use two forms of edge called *Conjunctive* and *Disjunctive*. In the method, when checking the node degree or clique,

we only consider the conjunctive. Given a graph G, a degree threshold D and a clique size threshold C, in the following we describe two steps of the method: kcore decomposition and clique decomposition.

Kcore Decomposition: Kcore decomposition intend to decompose a given graph to have a graph without any node degree greater than D. Kcore contains an iterative process in which each round consists of a procedure to alternately extract a high degree and a low degree subgraph from the input graph. The high degree subgraph (G_H) is the induced subgraph on the set of nodes with degree bigger than D, meanwhile the low degree subgraph (G_L) is the one on the set of nodes with degree smaller then D. At the first round, we takes G as the input. At the step i, the input graph is $G_{H_{i-1}}$ taken from the output of the step $(i-1)$. The iteration stops when the G_{H_i} of the output is empty. When the decomposition process is done, we form a decomposed graph by making the union of all G_{L_i} taken from all rounds. We then add the set of edges from G that are not present in any G_{L_i} as the *disjunctive* edges of the decomposed graph.

Clique Decomposition: The clique decomposition begins with finding the set of cliques L in G that have size no less than C. For each clique in L, first, we add a new node to G. We then connect the new node to clique's nodes with disjunctive edges and to all neighborhood of clique's nodes at distance 1 with conjunctive edges. Finally, we remove all conjunctive edges that have end points at one of the clique nodes.

3.4 Graph Node Labeling

We nominate a method to label for graph nodes (genes). In our method, the *gene ontology* [10] is employed to vectorize nodes. We consider the set of biological terms used in the database as the bag of terms. First, for each gene we construct a binary vector whose each element equals to 1 if the corresponding term relates to that gene and 0 otherwise. As a consequence, we have a list of vectors for a given gene list. Next, we cluster genes into a given number of clusters. Last, the cluster labels associated to genes are assigned to their corresponding nodes.

3.5 Conjunctive Disjunctive Edge Graph Node Kernel

In this section, we describe our proposed kernel, CDGK, which is a modification of NSPDK to measure the node similarity in the graph which consists of conjunctive and disjunctive edges. In our kernel, we consider only conjunctive edges when computing the distance between nodes and extracting neighborhood subgraphs. We define two relations: the *conjunctive relation* $R^{\wedge}_{r,d}(A_u, B_v, G)$ to be true iff (i) A_u and B_v are in $\{\mathcal{N}^v_r : v \in V(G)\}$, where A_u (B_v) needs to be isomorphic with some $\mathcal{N}_r$, (ii) $\mathcal{D}(u,v) = d$; and the *disjunctive relation* $R^{\vee}_{r,d}(A_u, B_v, G)$ to be true iff (i) A_u and B_v are in $\{\mathcal{N}^v_r : v \in V(G)\}$, where A_u (B_v) needs to be isomorphic with some $\mathcal{N}_r$, (ii) there exists a vertex w such that $\mathcal{D}(u,w) = d$, (iii) (w,v) is a disjunctive edge.

We define $\kappa_{r,d}$, an instance of the DGK on the relation $R^{\wedge}_{r,d}$ and $R^{\vee}_{r,d}$ as

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

where $\mathbf{1}_{A \cong B}$ is the indicator function that returns 1 if A is isomorphic to B and 0 otherwise. $\kappa_{r,d}$ counts the number of identical pairs of neighboring graphs of radius r at a distance d between two vertices. The CDGK is defined as $K(u,v) = \sum_r \sum_d \kappa_{r,d}(u,v)$.

4 Evaluation

In this section, we aim at evaluating the performance of our proposed kernel and comparing it with four kernels described in Section 2.

4.1 Dataset

We employ two datasets used in [3] for the experiment.

BioGPS: A gene co-expression network is constructed from BioGPS dataset, which contains 79 tissues in duplicates, measured with the Affymetrix U133A array. Pairwise Pearson correlation coefficients (PCC) are calculated and a pair of genes are linked by an edge if the PCC value is larger than 0.5.

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

4.2 Evaluation Methods

We evaluate the performance of graph node kernels in the gene prioritization problem. Given a set of genes known to be associated to a given disease, gene prioritization is a task that aims to rank the candidate genes based on their probabilities to be related to that disease.

Similar to the evaluation process used in [3], we choose 12 diseases in which each one contains at least 30 confirmed genes. For each disease, we construct a positive set $\mathcal{P}$ and a negative set $\mathcal{N}$. The set $\mathcal{P}$ consists of all disease gene members. The set $\mathcal{N}$ is built by randomly picking genes from known disease genes, genes associated at least to one disease class, but not related to the current class, such that $|\mathcal{N}| = \frac{1}{2}|\mathcal{P}|$. After that, leave-one-out cross validation is used to evaluate the performance of the algorithm. Each turn one gene is out to be the test gene and the rest are used to train the model using SVM. Starting from the output scores, we compute a decision score q_i for the test gene g_i as the top percentage value of score s_i among all scores. $q_i = \frac{|\{j | s_i \geq s_j\}|}{N}, i = 1, 2, \ldots, N$, where s_i, s_j are scores, N is the length of gene list. We collect all decision scores for every gene in the training set of a disease to form a global decision score list. The performance of the algorithm is measured by using AUC calculated that list.

4.3 Parameter Selection

In order to select the optimal parameter values for each kernel, we use one disease gene set for parameter selection and use Kfold with k equals to 3. We set the values for diffusion parameter in DK and MED as $\{10^{-3}, 10^{-3}, 10^{-2}, 10^{-1}\}$, for time steps in MD as $\{1, 10, 100\}$ and for RL parameter as $\{1, 4, 7\}$. For CDGK, we set values for degree threshold in $\{10,\ 15,\ 20\}$, clique size threshold in $\{4,\ 5\}$, maximum radius in $\{1,\ 2\}$, maximum distance in $\{2,\ 3,\ 4\}$. Finally, the C of SVM is set as $\{10^{-5},\ 10^{-4},\ 10^{-3},\ 10^{-2},\ 10^{-1},\ 1,\ 10,\ 10^{2}\}$.

5 Results and Discussion

	BioGPS					**Pathways**				
Disease	K1	K2	K3	K4	K5	K1	K2	K3	K4	K5
1	52/5	57/4	59/3	59/2	**65/1**	75/5	76/4	79/3	79/2	**80/1**
2	82/2	79/3	75/4	75/5	**88/1**	55/5	65/4	77/3	77/2	**81/1**
3	64/4	60/5	72/2	72/1	66/3	55/5	63/4	64/3	66/2	67/1
4	65/4	58/5	68/3	68/2	**72/1**	54/5	65/4	**74/1**	74/2	66/3
5	64/5	64/4	67/2	66/3	**76/1**	53/5	56/4	63/2	63/3	**68/1**
6	75/2	70/5	71/4	71/3	**79/1**	83/5	93/4	97/2	**97/1**	93/3
7	73/3	67/5	75/2	**76/1**	69/4	85/5	88/4	89/2	**90/1**	89/3
8	74/5	**77/1**	76/3	76/2	75/4	54/5	66/4	72/3	72/2	**73/1**
9	**72/1**	66/5	68/3	70/2	67/4	53/5	65/2	64/4	64/3	**81/1**
10	54/3	50/5	56/2	51/4	**78/1**	69/3	65/5	74/2	**74/1**	67/4
11	58/4	51/5	59/3	59/2	**72/1**	54/5	69/4	75/2	74/3	**77/1**
$\overline{AUC}$	66.6	63.5	67.8	67.5	**73.3**	62.7	70.1	75.3	75.5	**76.5**
$\overline{Rank}$	3.5	4.3	2.8	2.5	**2.0**	4.8	3.9	2.5	2.0	**1.8**

Table 1: *The performance of kernels on different genetic diseases using BioGPS and Pathway dataset. Each element in the table shows the AUC in percentage and the order of kernel comparing to the rest (AUC/Rank). K1 = DK, K2 = MD, K3 = MED, K4 = RL, K5 = CDGK.*

Table 1 shows the AUC performance of the models trained by using different graph node kernels on 11 genetic diseases using BioGPS and Pathways datasets. In the table, the best result on each disease is marked in bold. By observing the results, we note that the kernel CDGK perform the best results comparing with other considered kernels. Particularly, the CDGK is ranked at the first order in seven out of 11 diseases on both datasets. It also illustrates the highest results in average AUC and rank with 73.3/2.0, 76.5/1.8, and the AUC difference with the second best ones are 5.5% and 1% on BioGPS and Pathways, respectively. The MED and RL show similar and moderate results with small gap between them. Last, DK and MD demonstrate modest performance in average comparing with

other adopted kernels. They are ranked in the last position in many diseases, especially 7 times out of 11 for MD in BioGPS and 10 out of 11 for DK in Pathways. While DK shows better performance than MD in BioGPS, it presents worse in Pathways. In conclusion, CDGK outperforms all employed graph node kernels in term of both average rank and AUC measure.

The CDGK shows the state of the art results. However, in the case that the input graph has high average node degree and they are uniformly distributed, the decomposed graph is too sparse and it can lead our kernel to the poor performance.

6 Conclusions

In this paper, we first introduce an algorithm for graph decomposition in which we propose the use of conjunctive and disjunctive edges. Second we presented a new graph node kernel that is able to efficiently exploit the graph properties to measure node similarities in the graph. The evaluation results demonstrate that our graph node kernel outperforms all kernels used in the experiments.

For the coming work of our research, we plan to apply the CDGK for the disease gene identification problem which is relied on data fusion.

References

[1] Haussler, David. Convolution kernels on discrete structures. Vol. 646. Technical report, Department of Computer Science, University of California at Santa Cruz, 1999.

[2] Kondor, Risi Imre, and John Lafferty, Diffusion kernels on graphs and other discrete input spaces. *ICML*. Vol. 2. 2002.

[3] Chen. B, et al. Disease gene identification by using graph kernels and Markov random fields. Science China Life Sciences 57.11 (2014): 1054-1063.

[4] Fouss F, et al. An experimental investigation of graph kernels on a collaborative recommendation task. Proceedings of the 6th international conference on data mining 2006. ICDM 2006, 863-868.

[5] Chebotarev P and Shamis E. The matrix forest therem and measuring relations in small social groups. Automation and Remote Control 1997, 58(9):1505-1514.

[6] C. Fabrizio, and K. De Grave, Fast neighborhood subgraph pairwise distance kernel. Proceedings of the 26th, International Conference on Machine Learning. Omnipress, 2010.

[7] Goh KI et al, The human disease network. Proceedings of the National Academy of Sciences 104.21 (2007): 8685-8690.

[8] A. Hamelin, et al, K-core decomposition: A tool for the visualization of large scale networks. arXiv preprint cs/0504107 (2005).

[9] R. E. Tarjan, Decomposition by clique separators, Discrete Math, vol. 55, no. 2, pp. 221-232, July. 1985.

[10] Gene Ontology Consortium. The Gene Ontology (GO) database and informatics resource. Nucleic acids research,(2004) 32(suppl 1), D258-D261.