Graph Neural Networks

E. Daller, S. Bougleux, and L. Brun

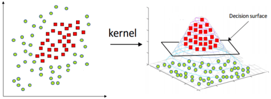
April 19, 2018

Different Levels/Steps of Recognition

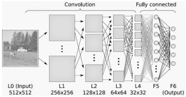
Level 0: Hand made classification (Expert Systems)

If $x > 0.3$ and $y < 1.5$ then CANCEROUS

Level 1: Design of feature vectors/(di)similarity measures. Automatic Classification

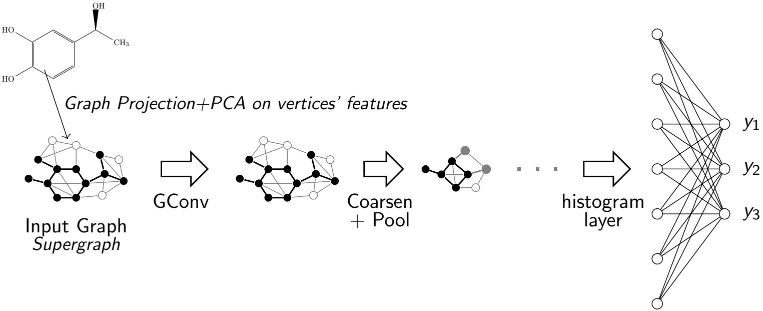


Level 2: Automatic design of pertinent features / metric from huge amount of examples.



Chemoinformatic is mainly at level 1, Image / Computer vision at level 2.

Graph Neural Network



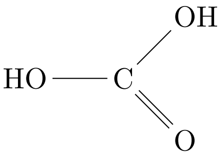
Input Data

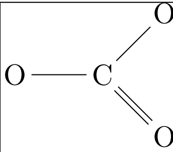
- A usual encoding associate to an atom C the vector:

N	O	C	...
0	0	1	...

☹: NonInformative and sparse vectors not convenient for convolutions.

- First Idea : Adapt the notion of treelets



Pattern	**C**–O	**C**=O	O–**C**–O	O–**C**=O	
Frequency	2	1	1	2	1

☺ Each vertex encodes its local configuration
☹ High dimensional vectors

- We do a PCA.

Input Graphs

- Some Graph Neural Networks learn weight without taking account the structure of the graph.
- Some others take the structure of the graph into account but are limited to fixed graph structures.

How to remove this limitation ?

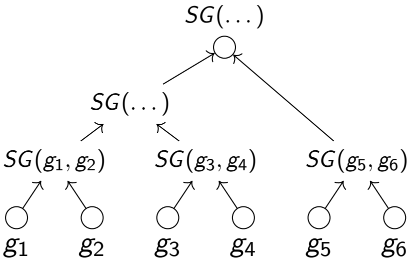
- We compute a super-graph by using the GED.

Computation of the super-graph

- Given a trainning set $\{g_1, \dots, g_n\}$ we compute a set of pairs (a maximal matching) minimizing:

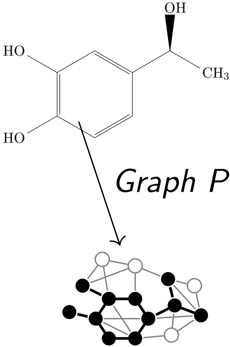
$$M^\star = \arg\min_M \sum_{(g_i, g_j) \in M} d(g_i, g_j)$$

- We then compute the super graph of each pair and so on up to the appex:



Processing of input Graphs using the Super-Graph

- Each graph of the trainning set is a subgraph of the super-graph.
- It may be considered as one (or several) signal(s) on the super-graph.



Graph Projection+PCA on vertices' features

Input Graph
Supergraph

Last layer of a Graph convolutional network

- The final classification / regression stage requires a layer with a fixed geometry. Pb: Graphs does not have a fixed geometry (unless we use the super-graph)
- Usual solution : a GAP (Global average pooling). If vertices' features have dimension D it creates a vector H where:

  $$\forall c \in \{1, \dots, D\} \; H(i) = \frac{1}{|V|} \sum_{v \in V} h_c(v)$$

  where $h(v)$ is the feature vector of $v \in V$.

  ☹ Very rough estimate.
- We propose to compute instead a $D \times K$ pseudo-histogram where K is the number of bins per component. The height of a *bin* k of this pseudo-histogram is computed as follows:

  $$b_{ck}(h) = \frac{1}{|V|} \sum_{v \in V} e^{\frac{-(h_c(v)-\mu_{ck})^2}{\sigma_{ck}^2}} \tag{1}$$

Experiments: The Datasets

- Datasets

	NCI1	MUTAG	ENZYMES	PTC	PAH
#graphs	4110	188	600	344	94
mean $\|V\|$, mean $\|E\|$	(29.9, 32.3)	(17.9, 19.8)	(32.6, 62.1)	(14.3, 14.7)	(20.7, 24,4)
#labels, #patterns	(37, 424)	(7, 84)	(3, 240)	(19, 269)	(1, 4)
#classes	2	2	6	2	2
#pos., #neg.	(2057, 2053)	(125, 63)	–	(152, 192)	(59, 35)

- Results :

GConv	feat.	s-g	gpool	NCI1	MUTAG	ENZYMES	PTC	PAH
DCNN	–	–	GAP	62.61	66.98	18.10	56.60	57.18
	⋆	–	GAP	67.81	81.74	31.25	59.04	54.70
	⋆	–	hist	71.47	82.22	38.55	60.43	66.90
	⋆	⋆	hist		83.57			71.35
GCN	–	–	GAP	55.44	70.79	16.60	52.17	63.12
	⋆	–	GAP	66.39	82.22	32.36	58.43	57.80
	⋆	–	hist	74.76	82.86	37.90	62.78	72.80
	⋆	⋆	hist		80.44		61.60	71.50
CGCNN	⋆	⋆	–					

Conclusion and future Works

- Our improvment of the first and last layers seems effective.
- We should investigate why GCN does not like the super-graph.
- Next Steps:
  - Replace the PCA in order to take into account the objective function.
  - Define a better convolution
  - Define a better coarsening.