Local Patterns and Supergraph for Chemical Graph Classification with Convolutional Networks

Évariste Daller, Sébastien Bougleux, Luc Brun, Olivier Lézoray

Local Patterns and Supergraph for Chemical Graph Classification with Convolutional Networks

Évariste Daller[1], Sébastien Bougleux[1], Luc Brun[2], and Olivier Lézoray[1]

[1] Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, Caen, France
[2] Normandie Univ, ENSICAEN, UNICAEN, CNRS, GREYC, Caen
{evariste.daller, bougleux, olivier.lezoray}@unicaen.fr
luc.brun@ensicaen.fr

Presented at IAPR Joint Workshops on Statistical Techniques in Pattern Recognition, and Structural and Syntactic Pattern Recognition
S+SSPR 2018, Beijing, China (http://ssspr2018.buaa.edu.cn/)

Abstract. Convolutional neural networks (CNN) have deeply impacted the field of machine learning. These networks, designed to process objects with a fixed topology, can readily be applied to images, videos and sounds but cannot be easily extended to structures with an arbitrary topology such as graphs. Examples of applications of machine learning to graphs include the prediction of the properties molecular graphs, or the classification of 3D meshes. Within the chemical graphs framework, we propose a method to extend networks based on a fixed topology to input graphs with an arbitrary topology. We also propose an enriched feature vector attached to each node of a chemical graph and a new layer interfacing graphs with arbitrary topologies with a full connected layer.

Keywords: Graph-CNNs, graph classification, graph edit distance.

1 Introduction

Convolutional neural networks(CNN) [13] have deeply impacted machine learning and related fields such as computer vision. These large breakthrough encouraged many researchers [5,10,9,4] to extend the CNN framework to unstructured data such as graphs, point clouds or manifolds. The main motivation for this new trend consists in extending the initial successes obtained in computer vision to other fields such as indexing of textual documents, genomics, computer chemistry or indexing of 3D models.

The initial convolution operation defined within CNN, uses explicitly the fact that objects (e.g. pixels) are embedded within a plane and on a regular grid. These hypothesis do not hold when dealing with convolution on graphs. A first approach related to the graph signal processing framework uses the link between convolution and Fourier transform as well as the strong similarities between the Fourier transform and the spectral decomposition of a graph. For

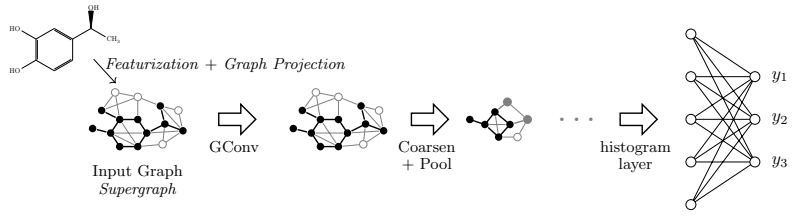
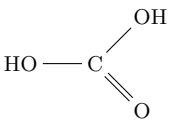


Fig. 1: Illustration of our propositions on a graph convolutional network

example, Bruna et al. [5] define the convolution operation from the Laplacian spectrum of the graph encoding the first layer of the neural network. However this approach requires a costly decomposition into singular Laplacian values during the creation of the convolution network as well as costly matrices multiplications during the test phase. These limitations are partially solved by Defferard et al. [9] who propose a fast implementation of the convolution based on Chebyshev polynomials (CGCNN). This implementation allows a recursive and efficient definition of the filtering operation while avoiding the explicit computation of the Laplacian. However, both methods are based on a fixed graph structure. Such networks can process different signals superimposed onto a fixed input layer but are unable to predict properties of graphs with variable topologies.

Another family of methods is based on a spatial definition of the graph convolution operation. Kipf and Welling [12] proposed a model (CGN) which approximates the local spectral filters from [9]. Using this formulation, filters are no longer based on the Laplacian but on a weight associated to each component of the vertices' features for each filter. The learning process of such weights is independent of the graph topology. Therefore graph neural networks based on this convolution scheme can predict properties of graphs with various topologies. The model proposed by Duvenaud et al. [10] for fingerprint extraction is similar to [12], but considers a set of filters for each possible degree of vertices. These last two methods both weight each components of the vertices' feature vectors. Verma et al. [17] propose to attach a weight to edges through the learning of a parametric similarity measure between the features of adjacent vertices. Similarly, Simonovsky and Komodakis [15] learn a weight associated to each edge label. Finally, Atwood and Towsley [1] (with DCNN) remove the limitation of the convolution to the direct neighborhood of each vertex by considering powers of a transition matrix defined as a normalization of the adjacency matrix by vertices' degrees. A main drawback of this non-spectral approach is that there exist intrinsically no best way to match the learned convolution weights with the elements of the receptive field, hence this variety of recent models.

In this paper, we propose to unify both spatial and spectral approaches by using as input layer a super-graph deduced from a graph train set. In addition, we propose an enriched feature vector within the framework of chemical graphs. Finally, we propose a new bottleneck layer at the end of our neural network which



Pattern					
Frequency	2	1	1	2	1

Fig. 2: Frequency of patterns associated to the central node (C).

is able to cope with the variable size of the previous layer. These contributions are described in Section 2 and evaluated in Section 3 through several experiments.

2 Contributions

2.1 From symbolic to feature graphs for convolution

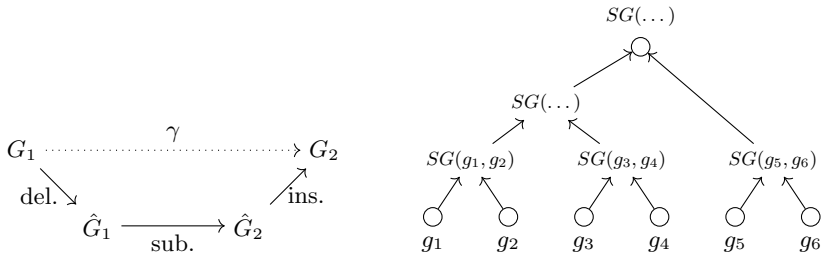
Convolution cannot be directly applied to symbolic graphs. So symbols are usually transformed into unit vectors of $\{0,1\}^{|\mathcal{L}|}$, where $\mathcal{L}$ is a set of symbols, as done in [1,10,15] to encode atom's type in chemical graphs. This encoding has a main drawback, the size of convolution kernels is usually much smaller than $|\mathcal{L}|$. Combined with the sparsity of vectors, this produces meaningless means for dimensionality reduction. Moreover, information attached to edges is usually unused.

Let us consider a graph $G = (V, E, \sigma, \phi)$, where V is a set of nodes, $E \subseteq V \times V$ a set of edges, and σ and ϕ functions labeling respectively G's nodes and edges. To avoid these drawbacks, we consider for each node u of V a vector representing the distribution of small subgraphs covering this node. Let $\mathcal{N}_u$ denotes its 1-hop neighbors. For any subset $S \subseteq \mathcal{N}_u$, the subgraph $M_u^S = (\{u\} \cup S, E \cap (\{u\} \cup S) \times (\{u\} \cup S), \sigma, \phi)$ is connected (through u) and defines a local pattern of u. The enumerations of all subsets of $\mathcal{N}_u$ provides all local patterns of u that can be organized as a feature vector counting the number of occurrences of each local pattern. Figure 2 illustrates the computation of such a feature vector. Note that the node's degree of chemical graphs is bounded and usually smaller than 4.

During the training phase, the patterns found for the nodes of the training graphs determine a dictionary as well as the dimension of the feature vector attached to each node. During the testing phase, we compute for each node of an input graph, the number of occurrences of its local patterns also present in the dictionary. A local pattern of the test set not present in the train set is thus discarded. In order to further enforce the compactness of our feature space, we apply a PCA on the whole set of feature vectors and project each vector onto a subspace containing 95% (fixed threshold) of the initial information.

2.2 Supergraph as input layer

As mentioned in section 1, methods based on spectral analysis [5,9] require a fixed input layer. Hence, these methods can only process functions defined on a



(a) Reordering of an edit path (b) Construction of a supergraph

Fig. 3: Construction of a supergraph (b) using common subgraphs induced by the Graph Edit Distance (a).

fixed graph topology (e.g. node's classification or regression tasks) and cannot be used to predict global properties of topologically variable graphs. We propose to remove this restriction by using as an input layer a supergraph deduced from graphs of a training set.

A *common supergraph* of two graphs G_1 and G_2 is a graph S so that both G_1 and G_2 are isomorphic to a subgraph of S. More generally, a common supergraph of a set of graphs $\mathcal{G} = \{G_k = (V_k, E_k, \sigma_k, \phi_k)\}_{k=1}^{k=n}$ is a graph $S = (V_S, E_S, \sigma_S, \phi_S)$ so that any graph of $\mathcal{G}$ is isomorphic to a subgraph of S. So, given any two complementary subsets $\mathcal{G}_1, \mathcal{G}_2 \subseteq \mathcal{G}$, with $\mathcal{G}_1 \cup \mathcal{G}_2 = \mathcal{G}$, it holds that a supergraph of a supergraph of $\mathcal{G}_1$ and a supergraph of $\mathcal{G}_2$ is a supergraph of $\mathcal{G}$. The latter can thus be defined by applying this property recursively on the subsets. This describes a tree hierarchy of supergraphs, rooted at a supergraph of $\mathcal{G}$, with the graphs of $\mathcal{G}$ as leaves. We present a method to construct hierarchically a supergraph so that it is formed of a minimum number of elements.

A common supergraph S of two graphs, or more generally of $\mathcal{G}$, is a *minimum common supergraph* (MCS) if there is no other supergraph S' of $\mathcal{G}$ with $|V_{S'}| < |V_S|$ or $(|V_{S'}| = |V_S|) \wedge (|E_{S'}| < |E_S|)$. Constructing such a supergraph is difficult and can be linked to the following notion. A *maximum common subgraph* (mcs) of two graphs G_k and G_l is a graph $G_{k,l}$ that is isomorphic to a subgraph $\hat{G}_k$ of G_k and to a subgraph $\hat{G}_l$ of G_l, and so that there is no other common subgraph G' of both G_k and G_l with $|V_{G'}| > |V_{G_{k,l}}|$ or $(|V_{G'}| = |V_{G_{k,l}}|) \wedge (|E_{G'}| > |E_{G_{k,l}}|)$. Then, given a maximum common subgraph $G_{k,l}$, the graph S obtained from $G_{k,l}$ by adding the elements of G_k not in $\hat{G}_k$ and the elements of G_l not in $\hat{G}_l$ is a minimum common supergraph of G_k and G_l. This property shows that a minimum common supergraph can thus be constructed from a maximum common subgraph. These notions are both related to the notion of error-correcting graph matching and graph edit distance [6].

The *graph edit distance* (GED) captures the minimal amount of distortion needed to transform an attributed graph G_k into an attributed graph G_l by iteratively editing both the structure and the attributes of G_k, until G_l is ob-

tained. The resulting sequence of *edit operations* γ, called *edit path*, transforms G_k into G_l. Its cost (the strength of the global distortion) is measured by $L_c(\gamma) = \sum_{o \in \gamma} c(o)$, where $c(o)$ is the cost of the edit operation o. Among all edit paths from G_k to G_l, denoted by the set $\Gamma(G_k, G_l)$, a *minimal-cost edit path* is a path having a minimal cost. The GED from G_k to G_l is defined as the cost of a minimal-cost edit path: $d(G_k, G_l) = \min_{\gamma \in \Gamma(G_k, G_l)} L_c(\gamma)$.

Under mild constraints on the costs [3], an edit path can be organized into a succession of removals, followed by a sequence of substitutions and ended by a sequence of insertions. This reordered sequence allows to consider the subgraphs $\hat{G}_k$ of G_k and $\hat{G}_l$ of G_l. The subgraph $\hat{G}_k$ is deduced from G_k by a sequence of node and edge removals, and the subgraph $\hat{G}_l$ is deduced from $\hat{G}_k$ by a sequence of substitutions (Figure 3a). By construction, $\hat{G}_k$ and $\hat{G}_l$ are structurally isomorphic, and an *error-correcting graph matching* (ECGM) between G_k and G_l is a bijective function $f : \hat{V}_k \to \hat{V}_l$ matching the nodes of $\hat{G}_k$ onto the ones of $\hat{G}_l$ (correspondences between edges are induced by f).

Then ECGM, mcs and MCS are related as follows. For specific edit cost values [6] (not detailed here), if f corresponds to an optimal edit sequence, then $\hat{G}_k$ and $\hat{G}_l$ are mcs of G_k and G_l. Moreover, adding to a mcs of G_k and G_l the missing elements from G_k and G_l leads to an MCS of these two graphs. We use this property to build the global supergraph of a set of graphs.

Supergraph construction The proposed hierarchical construction of a common supergraph of a set of graphs $\mathcal{G} = \{G_i\}_i$ is illustrated by Fig. 3b. Each level k of the hierarchy contains N_k graphs. They are merged by pairs to produce $\lfloor N_k/2 \rfloor$ supergraphs. In order to restrain the size of the final supergraph, a natural heuristic consists in merging close graphs according to the graph edit distance. This can be formalized as the computation of a maximum matching $M^\star$, in the complete graph over the graphs of $\mathcal{G}$, minimizing:

$$M^\star = \arg\min_{M} \sum_{(g_i, g_j) \in M} d(g_i, g_j) \tag{1}$$

where $d(\cdot, \cdot)$ denotes the graph edit distance. An advantage of this kind of construction is that it is highly parallelizable. Nevertheless, computing the graph edit distance is NP-hard. Algorithms that solve the exact problem cannot be reasonably used here. So we considered a bipartite approximation of the GED [14] to compute $d(\cdot, \cdot)$ and solve (1), while supergraphs are computed using a more precise but more computationally expansive algorithm [7].

2.3 Projections as input data

The supergraph computed in the previous section can be used as an input layer of a graph convolutional neural network based on spectral graph theory [5,9] (Section 1). Indeed, the fixed input layer allows to consider convolution operations based on the Laplacian of the input layer. However, each input graph for which a property has to be predicted, must be transformed into a signal on the

supergraph. This last operation is allowed by the notion of projection, a side notion of the graph edit distance.

Definition 1 (Projection). *Let f be an ECGM between two graphs G and S and let $(\hat{V}_S, \hat{E}_S)$ be the subgraph of S defined by f (Fig. 3). A projection of $G = (V, E, \sigma, \phi)$ onto $S = (V_S, E_S, \sigma_S, \phi_S)$ is a graph $P_S^f(G) = (V_S, E_S, \sigma_P, \phi_P)$ where $\sigma_P(u) = (\sigma \circ f^{-1})(u)$ for any $u \in \hat{V}_S$ and 0 otherwise. Similarly, $\phi_P(\{u, v\}) = \phi(\{f^{-1}(u), f^{-1}(v)\})$ for any $\{u, v\}$ in $\hat{E}_S$ and 0 otherwise.*

Let $\{G_1, \ldots, G_n\}$ be a graph training set and S its the associated supergraph. The projection $P_S^f(G_i)$ of a graph G_i induces a signal on S associated to a value to be predicted. For each node of S belonging to the projection of G_i, this signal is equal to the feature vector of this node in G_i. This signal is null outside the projection of G_i. Moreover, if the edit distance between G_i and S can be computed through several edit paths with a same cost (i.e., several ECGM $f_1, \ldots, f_m$), the graph G_i will be associated to these projections $P_S^{f_1}(G_i), \ldots, P_S^{f_m}(G_i)$. Remark that a graph belonging to a test dataset may also have several projections. In this case, it is mapped onto the majority class among its projections. A natural data augmentation can thus be obtained by learning m equivalent representations of a same graph on the supergraph, associated to the same value to be predicted. Note that this data augmentation can also be increased by considering μm non-minimal ECGM, where μ is a parameter. To this end, we use [7] to compute a set of non-minimal ECGM between an input graph G_i and the supergraph S and we sort this set increasingly according to the cost of the associated edit paths.

2.4 Bottleneck layer with variable input size

A multilayer perceptron (MLP), commonly used in the last part of multilayer networks, requires that the previous layer has a fixed size and topology. Without the notion of supergraph, this last condition is usually not satisfied. Indeed, the size and topology of intermediate layers are determined by those of the input graphs, which generally vary. Most of graph neural networks avoid this drawback by performing a global pooling step through a bottleneck layer. This usually consists in averaging the components of the feature vectors across the nodes of the current graph, the so-called global average pooling (GAP). If for each node $v \in V$ of the previous layer, the feature vector $h(v) \in \mathbb{R}^D$ has a dimension D, GAP produces a mean vector $(\frac{1}{|V|} \sum_{v \in V} h_c(v))_{c=1,\ldots,D}$ describing the graph globally in the feature space.

We propose to improve the pooling step by considering the distribution of feature activations across the graph. A simple histogram can not be used here, due to its non-differentiability, differentiability being necessary for backpropagation. To guarantee this property holds, we propose to interpolate the histogram by using averages of Gaussian activations. For each component c of a given a feature vector $h(v)$, the height of a bin k of this pseudo-histogram is computed as follows:

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

The size of the layer is equal to $D \times K$, where K is the number of bins defined for each component.

In this work, the parameters μ_{ck} and σ_{ck} are fixed and not learned by the network. To choose them properly, the model is trained with a GAP layer for few iterations (10 in our experiments), then it is replaced by the proposed layer. The weights of the network are preserved, and the parameters μ_{ck} are uniformly spread between the minimum and the maximum values of $h_c(v)$. The parameters σ_{ck} are fixed to $\sigma_{ck} = \delta_\mu/3$ with $\delta_\mu = \mu_{ci+1} - \mu_{ci},\ \forall 1 \leq i < K$, to ensure an overlap of the Gaussian activations.

Since this layer has no learnable parameters, the weights $\alpha_c(i)$ of the previous layer h are adjusted during the backpropagation for every node $i \in V$, according to the partial derivatives of the loss function L: $\frac{\partial L}{\partial \alpha_c(i)} = \frac{\partial L}{\partial b_{ck}(h)} \frac{\partial b_{ck}(h)}{\partial h_c(i)} \frac{\partial h_c(i)}{\partial \alpha_c(i)}$. The derivative of the bottleneck layer w.r.t. its input is given by:

$$\forall i \in V, \quad \frac{\partial b_{ck}(h)}{\partial h_c(i)} = \frac{-2(h_c(i) - \mu_{ck})}{|V|\sigma_{ck}^2} \exp\left(\frac{-(h_c(i) - \mu_{ck})^2}{\sigma_{ck}^2}\right). \tag{3}$$

It lies between $-\frac{\sqrt{2}}{|V|\sigma_{ck}} e^{-1/2}$ and $\frac{\sqrt{2}}{|V|\sigma_{ck}} e^{-1/2}$.

3 Experiments

We compared the behavior of several graph convolutional networks, with and without the layers presented in the previous section, for the classification of chemical data encoded by graphs. The following datasets were used: NCI1, MUTAG, ENZYMES, PTC, and PAH. Table 1 summarizes their main characteristics. NCI1 [18] contains 4110 chemical compounds, labeled according to their capacity to inhibit the growth of certain cancerous cells. MUTAG [8] contains 188 aromatic and heteroaromatic nitrocompounds, the mutagenicity of which has to be predicted. ENZYMES [2] contains 600 proteins divided into 6 classes of enzymes (100 per class). PTC [16] contains 344 compounds labeled as carcinogenic or not for rats and mice. PAH[3] contains non-labeled cyclic carcinogenic and non-carcinogenic molecules.

3.1 Baseline for classification

We considered three kinds of graph convolutional networks. They differ by the definition of their convolutional layer. CGCNN [9] is a deep network based on a pyramid of reduced graphs. Each reduced graph corresponds to a layer of the network. The convolution is realized by spectral analysis and requires the computation of the Laplacian of each reduced graph. The last reduced graph is followed by a fully connected layer. GCN [12] and DCNN [1] networks do not use spectral analysis and are referred to as spatial networks. GCN can be seen as an approximation of [9]. Each convolutional layer is based on F filtering operations

[3] PAH is available at: `https://iapr-tc15.greyc.fr/links.html`

Table 1: Characteristics of datasets. V and E denotes resp. nodes and edges sets of the datasets' graphs, while V_S and E_S denotes nodes and edges sets of the datasets' supergraphs

	NCI1	MUTAG	ENZYMES	PTC	PAH
#graphs	4110	188	600	344	94
mean $\|V\|$, mean $\|E\|$	$(29.9, 32.3)$	$(17.9, 19.8)$	$(32.6, 62.1)$	$(14.3, 14.7)$	$(20.7, 24{,}4)$
mean $\|V_S\|$	192.8	42.6	177.1	102.6	26.8
mean $\|E_S\|$	4665	146	1404	377	79
#labels, #patterns	$(37, 424)$	$(7, 84)$	$(3, 240)$	$(19, 269)$	$(1, 4)$
#classes	2	2	6	2	2
#positive, #negative	$(2057, 2053)$	$(125, 63)$	–	$(152, 192)$	$(59, 35)$

associating a weight to each component of the feature vectors attached to nodes. These weighted vectors are then combined through a local averaging. DCNN [1] is a nonlocal model in which a weight on each feature is associated to a hop $h < H$ and hence to a distance to a central node (H is thus the radius of a ball centered on this central node). The averaging of the weighted feature vectors is then performed on several hops for each node.

To measure the effects of our contributions when added to the two spatial networks (DCNN and GCN), we considered several versions obtained as follows (Table 2). We used two types of characteristics attached to the nodes of the graphs (input layer): characteristics based on the canonical vectors of $\{0, 1\}^{|\mathcal{L}|}$ as in [1,10,15], and those based on the patterns proposed in Section 1. Note that PAH has few different patterns (Table 1), PCA was therefore not applied to this data to reduce the size of features. Since spatial networks can handle arbitrary topology graphs, the use of a supergraph is not necessary. However, since some nodes have a null feature in a supergraph (Definition 1), a convolution performed on a graph gives results different from those obtained by a similar convolution performed on the projection of the graph on a supergraph. We hence decided to test spatial networks with a supergraph. For the other network (CGCNN), we used the features based on patterns and a supergraph.

For the architecture of spatial networks, we followed the one proposed by [1], with a single convolutional layer. For CGCNN we used two convolutional layers to take advantage of the coarsening as it is part of this method. For DCNN, $H = 4$. For CGCNN and GCN, $F = 32$ filters were used. The optimization was achieved by Adam [11], with at most 500 epochs and early stopping. The experiments were done in 10 fold cross-validation which required to compute the supergraphs of all training graphs. Datasets were augmented by 20% of non-minimal cost projections with the method described in Section 2.3.

3.2 Discussion

As illustrated in Table 2, the features proposed in Section 2.1 improve the classification rate in most cases. For some datasets, the gain is higher than 10 per-

Table 2: Mean accuracy (10-fold cross validation) of graph classification by three networks (GConv), with the features proposed in Section 2.1 (*feat.*) and the supergraph (*s-g*). Global pooling (gpool) is done using global average pooling (GAP) or with histogram bottleneck layer (hist).

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	**73.95**	**83.57**	**40.83**	56.04	**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	73.02	80.44	**46.23**	61.60	71.50
CGCNN	✓	✓	–	68.36	75.87	33.27	60.78	63.73

centage points. The behavior of the two spatial models (DCNN and GCN) is also improved, for every dataset, by replacing global average pooling by the histogram bottleneck layer described in Section 2.4. These observations point out the importance of the global pooling step for these kind of networks

Using a supergraph as an input layer (column s-g) opens the field of action of spectral graph convolutional networks to graphs with different topologies, which is an interesting result in itself. Results are comparable to the ones obtained with the other methods (improve the baseline models with no histogram layer), but this is a first result for these networks for the classification of graphs. The sizes of supergraphs reported in Table 1 remain reasonable regarding the number of graphs and the maximum size in each dataset. Nevertheless, this strategy only enlarge each data up to the supergraph size.

4 Conclusions

We proposed features based on patterns to improve the performances of graph neural networks on chemical graphs. We also proposed to use a supergraph as input layer in order to extend graph neural networks based on spectral theory to the prediction of graph properties for arbitrary topology graphs. The supergraph can be combined with any graph neural network, and for some datasets the performances of graph neural networks not based on spectral theory were improved. Finally, we proposed an alternative to the global average pooling commonly used as bottleneck layer in the final part of these networks.

References

1. Atwood, J., Towsley, D.: Diffusion-convolutional neural networks. In: Advances in Neural Information Processing Systems 29. pp. 2001–2009 (2016)

2. Borgwardt, K.M., Ong, C.S., Schönauer, S., Vishwanathan, S.V.N., Smola, A.J., Kriegel, H.P.: Protein function prediction via graph kernels. Bioinformatics **21**(suppl 1), i47–i56 (2005). https://doi.org/10.1093/bioinformatics/bti1007
3. Bougleux, S., Brun, L., Carletti, V., Foggia, P., Gazre, B., Vento, M.: Graph edit distance as a quadratic assignment problem. Pattern Recognition Letters **87**, 38–46 (2017). https://doi.org/10.1016/j.patrec.2016.10.001
4. Bronstein, M.M., Bruna, J., LeCun, Y., Szlam, A., Vandergheynst, P.: Geometric deep learning: Going beyond euclidean data. IEEE Signal Processing Magazine **34**(4), 18–42 (2017). https://doi.org/10.1109/MSP.2017.2693418
5. Bruna, J., Zaremba, W., Szlam, A., Lecun, Y.: Spectral networks and deep locally connected networks on graphs. Tech. rep. (2014), arXiv:1312.6203v2 [cs.LG]
6. Bunke, H., Jiang, X., Kandel, A.: On the minimum common supergraph of two graphs. Computing **65**(1), 13–25 (2000). https://doi.org/10.1007/PL00021410
7. Daller, É., Bougleux, S., Gaüzère, B., Brun, L.: Approximate Graph Edit Distance by Several Local Searches in Parallel. In: Proceedings of ICPRAM'18 (2018). https://doi.org/10.5220/0006599901490158
8. Debnath, A., L. Lopez de Compadre, R., Debnath, G., Shusterman, A., Hansch, C.: Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. Journal of Medicinal Chemistry **34**, 786–797 (1991). https://doi.org/10.1021/jm00106a046
9. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. In: Advances in Neural Information Processing Systems 29. pp. 3844–3852 (2016)
10. Duvenaud, D., Maclaurin, D., Aguilera-Iparraguirre, J., Gómez-Bombarelli, R., Hirzel, T., Aspuru-Guzik, A., Adams, R.P.: Convolutional networks on graphs for learning molecular fingerprints. In: Advances in Neural Information Processing Systems 28. pp. 2224–2232 (2015)
11. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. CoRR **abs/1412.6980** (2014)
12. Kipf, T., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (2017)
13. LeCun, Y., Bengio, Y.: The handbook of brain theory and neural networks. chap. Convolutional Networks for Images, Speech, and Time Series, pp. 255–258 (1998)
14. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. Image and Vision Computing **27** (2009). https://doi.org/10.1016/j.imavis.2008.04.004
15. Simonovsky, M., Komodakis, N.: Dynamic Edge-Conditioned Filters in Convolutional Neural Networks on Graphs. In: IEEE Conference on Computer Vision and Pattern Recognition (2017). https://doi.org/10.1109/cvpr.2017.11
16. Toivonen, H., Srinivasan, A., King, R., Kramer, S., Helma, C.: Statistical evaluation of the predictive toxicology challenge 2000-2001. Bioinformatics **19**, 1179–1182 (2003). https://doi.org/10.1093/bioinformatics/btg130
17. Verma, N., Boyer, E., Verbeek, J.: FeaStNet: Feature-Steered Graph Convolutions for 3D Shape Analysis. In: IEEE Conference on Computer Vision and Pattern Recognition (2018)
18. Wale, N., Watson, I.A., Karypis, G.: Comparison of descriptor spaces for chemical compound retrieval and classification. Knowledge and Information Systems **14**(3), 347–375 (2008). https://doi.org/10.1109/icdm.2006.39