A short note on Graph Convolution Networks

Kunal Sharma

Abstract

I felt motivated to write this after reading the wonderful paper by Bruna et al [1] extending CNN to graphs. There are a few ways to build Graph Neural Networks (GNN) and Graph Convolution Networks (GCN) are one type of such networks. However, even in GCN there are spectral and non-spectral apporaches. In this this note though we will focus on the spectral approach and elucidate on the definition of convolution over graph given in [Br].

Motivation & Background

GCN is part of the larger program of 'Geometric deep learning' where the aim is to extend Deep learning to non-Euclidean domains. Recall that CNNs have been successful primarily for image data and 1-D sequences, both of which are Euclidean spaces, $\mathbb{R}^2$ and $\mathbb{R}$ respectively. The next logical step is to extend it to more general spaces like graphs and manifolds. Graphs are much more tractable than manifolds and as real world data is both discrete and well represented in graphs in many cases. So exentding CNNs to graphs is the next logical step. This allows appropriate frameworks to study Knowledge graphs, social networks etc. Some of the typical tasks include node classification, link prediction, embeddings and graph representations.

Recall that the distinguishing features of CNN that make them so effective include *locality, translation invariance,* and *subsampling.* In order to extend CNN to graph its imperative to find analogues of this in case of graph. The below section focuses on the spectral approach whereas the spatial approach based on agglomerative clustering is not so well defined and robust.

Convolution on graphs

Let f, g be two functions defined on $\mathbb{Z}$. Then in this discrete case convolution is given as

$$f * g(t) \coloneqq \sum_{-\infty}^{\infty} f(x)g(t-x), \forall t \in \mathbb{Z}$$

Here the sequences are infinite but in practice the functions f, g are compactly supported so the sum is well-defined. This raises the following:

How to extend the above to graphs?

As an analyst, one of the first things that comes to my mind is

Theorem. *(Plancherel identity) Let $f, g \in \mathcal{S}(\mathbb{R}^n)$ then*

$$\mathcal{F}(f * g) = \mathcal{F}(f)\mathcal{F}(g)$$

Here $\mathcal{S}$ is the Schwartz space consisting of functions that are 'rapidly decreasing' and $\mathcal{F}$ stands for Fourier transform.

So, as a consequence of the above identity, if f and g are such that $\mathcal{F}(f)$ and $\mathcal{F}(g)$ exists then we can define convolution as

$$f * g(x) \coloneqq \mathcal{F}^{-1}(\mathcal{F}(f(x))\mathcal{F}(g(x)))$$

In case of graphs, the Fourier transform of f, g has a nice representation using spectral theory via projection on eigenvectors of Laplacian matrix.

Let $\mathcal{G} = (V, E)$ be a graph with V and E being the vertices and edges and let $|V| = N$. Recall that Fourier transform decomposes a function into eignefunctions of translations, in particular the 'plane waves'. More precisely,

$$\triangle e^{2\pi i \xi t} = -4\pi^2 |\xi|^2 e^{2\pi i \xi t}$$

Here W is the weight adjacency matrix while $\triangle$ is the 'combinatorial graph Laplacian',

$$\triangle f(x) = \sum_{f \in \mathcal{N}_x} W_{x,y} |f(x) - f(y)|$$

where $\mathcal{N}_x$ is the neighborhood of x i.e. the set of vertices y connected to x by an edge. If L is the matrix representation of $\triangle$ then L is a real symmetric matirx and so it has a complete set of orthonormal eigenvectors say, $U = \{u_i\}_{i=0,\dots,N-1}$. with corresponding eigenvalues $\{\lambda_i\}_{i=0,\dots,N-1}$. Then the *graph Fourier transform* is given as

$$\mathcal{F}(f(\lambda_k)) \coloneqq \langle f, u_k \rangle = \sum_{i=1}^{N} f(i) u_k^*(i)$$

where we replaced the complex exponentials with the graph Laplacian's eigenvectors. The inverse Fourier transform can be defined analogously.

Localaization of spectral filters

Now that graph Fourier transform is defined, a definition of convoution can be presented that respects localization in the spirit of CNN. Borrowing signal-processing terminology from [1] and [3], let's think of our functions x as a signal on $\mathcal{G}$ and by abuse of notation let L be the normalization of L now (see [2]). Then define the convolution of x be the operation of applying a filter g_θ to x as

$$g_\theta \star x \coloneqq U g_\theta U^T x,$$

where U^T is the Fourier transform of x. Typically, g_θ is some non-linear function and conventional choice is $g_\theta = diag(\theta)$ where $\theta \in \mathbb{R}^N$ is a vector of Fourier coefficients.

The diagonal filter however is *non-local* and being dependent on N, the *computational complexity* is $\mathcal{O}(n)$. A solutiion to this is to consider polynomial filter that is 'K-localized' i.e.

$$g_\theta(\Lambda) = \sum_{i=0}^{K-1} \theta_i \Lambda^i,$$

To see why this is localized, it's required to recall a result from Graph theory. So if $v_1, v_2 \in V$and $d_{\mathcal{G}}(v_1, v_2)$ denotes the shortest path distance between v_1 and v_2 which is the min. number of edges connecting the two vertices. Then for each

$$\forall s > 0, (L^s)_{i,j} = \text{number of paths of length } s \text{ connecting } i \text{ and } j$$

As a consequence, if $d_{\mathcal{G}}(v_1, v_2) > s$ then $(L^s)_{i,j} = 0$. Therefore, g_θ as defined, a K^{th} degree polynomial in the Laplacian, depends only in the K^{th}-*order neighborhood*(set of nodes that are maximum 'K-hops' away) from the node where the filter is centered.

References

[1] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. *Spectral networks and locally connected networks on graphs.* In International Conference on Learning Representations (ICLR), 2014.

[2] Chung, Fan. *Spectral Graph Theory.* American Mathematical Society (1997) [1992].

[3] Michael Defferrard, Xavier Bresson, and Pierre Vandergheynst. *Convolutional neural networks on graphs with fast localized spectral filtering.* In Advances in neural information processing systems (NIPS), 2016.