Graph Neural Networks

Lecture 10

Danila Biktimirov
Applied Computer Science
Neapolis University Pafos

9 May 2025

Overview

1. Graph Signal Processing
2. Probabilistic Graphical Models
3. Graph Isomorphism
4. Conclusion

Motivation

- Extend convolutions to graphs
- Graph signals: values assigned to nodes
- Use graph structure to guide transformations

Graph Fourier Transform

Let f and h be two functions. We can define the general continuous convolution operation $\star$ as

$$(f \star h)(\mathbf{x}) = \int_{\mathbb{R}^d} f(\mathbf{y}) h(\mathbf{x} - \mathbf{y})\, d\mathbf{y}.$$

Fourier Transform and Convolution

Fourier transforms of the two functions:

$$(f \star h)(\mathbf{x}) = \mathcal{F}^{-1}\left(\mathcal{F}(f(\mathbf{x})) \circ \mathcal{F}(h(\mathbf{x}))\right),$$

where

$$\mathcal{F}(f(\mathbf{x})) = \hat{f}(\mathbf{s}) = \int_{\mathbb{R}^d} f(\mathbf{x}) e^{-2\pi \mathbf{x}^\top \mathbf{s} i}\, d\mathbf{x}$$

$$\mathcal{F}^{-1}(\hat{f}(\mathbf{s})) = \int_{\mathbb{R}^d} \hat{f}(\mathbf{s}) e^{2\pi \mathbf{x}^\top \mathbf{s} i}\, d\mathbf{s}.$$

Discrete Circular Convolution and DFT

For discrete data over $t \in \{0, \ldots, N-1\}$, we define:

$$(f \star_N h)(t) = \sum_{\tau=0}^{N-1} f(\tau) h((t-\tau)_{\text{mod } N})$$

The discrete Fourier transform (DFT):

$$s_k = \frac{1}{\sqrt{N}} \sum_{t=0}^{N-1} f(x_t) e^{-i \frac{2\pi}{N} kt}$$

$$= \frac{1}{\sqrt{N}} \sum_{t=0}^{N-1} f(x_t) \left(\cos\left(\frac{2\pi}{N} kt\right) - i \sin\left(\frac{2\pi}{N} kt\right) \right)$$

Translation Equivariance of Convolution

Convolution acts as a filtering operation and is translation (shift) equivariant:

$$f(t+a) \star g(t) = f(t) \star g(t+a) = (f \star g)(t+a)$$

Graph Representation of Time-Varying Signals

For a chain graph, the circulant adjacency matrix $\mathbf{A}_c$ is:

$$\mathbf{A}_c[i,j] = \begin{cases} 1 & \text{if } j = (i+1)_{\text{mod } N} \\ 0 & \text{otherwise} \end{cases}$$

The unnormalized Laplacian:

$$\mathbf{L}_c = \mathbf{I} - \mathbf{A}_c$$

Matrix Representation of Convolution

Convolution as matrix multiplication:

$$(f \star h)(t) = \sum_{\tau=0}^{N-1} f(\tau)h(\tau - t) = \mathbf{Q}_h \mathbf{f},$$

Equivariance and Polynomial Filters

Translation equivariance requires:

$$\mathbf{A}_c\mathbf{Q}_h = \mathbf{Q}_h\mathbf{A}_c$$

Difference operator equivariance:

$$\mathbf{L}_c\mathbf{Q}_h = \mathbf{Q}_h\mathbf{L}_c$$

These are satisfied if:

$$\mathbf{Q}_h = p_N(\mathbf{A}_c) = \sum_{i=0}^{N-1} \alpha_i \mathbf{A}_c^i$$

General Polynomial Filter on Graphs

Convolutional filters can be written as:

$$\mathbf{Q}_h = \alpha_0 \mathbf{I} + \alpha_1 \mathbf{A} + \alpha_2 \mathbf{A}^2 + \cdots + \alpha_N \mathbf{A}^N$$

Applied to a node feature vector $\mathbf{x} \in \mathbb{R}^{|V|}$:

$$\mathbf{Q}_h \mathbf{x} = \alpha_0 \mathbf{I}\mathbf{x} + \alpha_1 \mathbf{A}\mathbf{x} + \alpha_2 \mathbf{A}^2\mathbf{x} + \cdots + \alpha_N \mathbf{A}^N\mathbf{x}$$

Graph Convolutions and Message Passing

Basic GNN layer applies a simple convolutional filter:

$$\mathbf{Q}_h = \mathbf{I} + \mathbf{A}$$

Fourier Transform and the Laplace Operator

The Laplace operator for smooth functions $f : \mathbb{R}^d \to \mathbb{R}$:

$$\Delta f(\mathbf{x}) = \nabla^2 f(\mathbf{x}) = \sum_{i=1}^{n} \frac{\partial^2 f}{\partial x^2}$$

The Graph Fourier Transform

Graph Fourier transform is defined via the eigendecomposition of the Laplacian:

$$\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$$

- $\mathbf{U}$: graph Fourier modes (eigenvectors),
- $\mathbf{\Lambda}$: eigenvalues, representing graph frequencies.

Graph Fourier Transform and Spectral Convolution

Fourier transform on a graph:

$$\mathbf{s} = \mathbf{U}^\top \mathbf{f}$$

Inverse transform:

$$\mathbf{f} = \mathbf{U}\mathbf{s}$$

Graph convolution in the spectral domain:

$$\mathbf{f} \star_{\mathcal{G}} \mathbf{h} = \mathbf{U}\left(\mathbf{U}^\top \mathbf{f} \circ \mathbf{U}^\top \mathbf{h}\right)$$

Spectral Graph Convolutions

Non-parametric spectral convolution:

$$\mathbf{f} \star_{\mathcal{G}} \mathbf{h} = \mathbf{U}(\mathbf{U}^\top \mathbf{f} \circ \theta_h) = \left(\mathbf{U}\,\mathrm{diag}(\theta_h)\,\mathbf{U}^\top\right)\mathbf{f}$$

Parametrization using polynomial of eigenvalues:

$$\mathbf{f} \star_{\mathcal{G}} \mathbf{h} = \left(p_N(\mathbf{\Lambda})\mathbf{U}^\top\right)\mathbf{f} = p_N(\mathbf{L})\mathbf{f}$$

Graph Convolutional Networks (GCNs)

A basic GCN layer (Kipf and Welling, 2016a):

$$\mathbf{H}^{(k)} = \sigma\left(\tilde{\mathbf{A}}\mathbf{H}^{(k-1)}\mathbf{W}^{(k)}\right)$$

where $\tilde{\mathbf{A}} = (\mathbf{D}+\mathbf{I})^{-\frac{1}{2}}(\mathbf{I}+\mathbf{A})(\mathbf{D}+\mathbf{I})^{-\frac{1}{2}}$

Basic GNN interpretation:

$$\mathbf{H}^{(k)} = \sigma\left(\mathbf{A}\mathbf{H}^{(k-1)}\mathbf{W}^{(k)}_{\text{neigh}} + \mathbf{H}^{(k-1)}\mathbf{W}^{(k)}_{\text{self}}\right)$$

GNNs without Message Passing

Simplified GNN model:

$$\mathbf{Z} = \mathrm{MLP}_\theta \left(f(\mathbf{A}) \, \mathrm{MLP}_\phi(\mathbf{X}) \right)$$

Example from Wu et al. (2019):

$$f(\mathbf{A}) = \tilde{\mathbf{A}}^k$$

Example from Klicpera et al. (2019), based on personalized PageRank:

$$f(\mathbf{A}) = \alpha(\mathbf{I} - (1-\alpha)\tilde{\mathbf{A}})^{-1} = \alpha \sum_{k=0}^{\infty} (\mathbf{I} - \alpha\tilde{\mathbf{A}})^k$$

MRF and Joint Distribution

$$p(\{x_v\}, \{z_v\}) \propto \prod_{v \in V} \phi(x_v, z_v) \prod_{(u,v) \in E} \psi(z_u, z_v)$$

- ϕ node potential, ψ edge potential

Mean-Field Approximation

$$p(\{z_v\}|\{x_v\}) \approx q(\{z_v\}) = \prod_{v \in V} q_v(z_v)$$

- Variational inference via factorized distributions

Mean-Field Update Rule

$$\log q_v(z_v) = c + \log \phi(x_v, z_v) + \sum_{u \in N(v)} \int q_u(z_u) \log \psi(z_u, z_v) dz_u$$

Embedding Mean Field in Hilbert Space

$$\mu_v = \int q_v(z_v)\phi(z_v)dz_v$$

- View μ_v as feature vector

Message Passing Rule (GNN)

$$\mu_v^{(t)} = \sigma\left(W_{self}^{(t)} x_v + W_{neigh}^{(t)} \sum_{u \in N(v)} \mu_u^{(t-1)}\right)$$

- Iterative updates = mean-field inference steps

Graph Isomorphism Definition

$$PA_1P^T = A_2, \quad PX_1 = X_2$$

- Permutation of adjacency and features

Weisfeiler-Lehman Test

$$l_v^{(k)} = \text{HASH}\left(l_v^{(k-1)}, \{l_u^{(k-1)} : u \in N(v)\}\right)$$

- Node labels updated iteratively

WL-GNN Connection

$$h_v^{(k+1)} = \text{UPDATE}(h_v^{(k)}, \text{AGGREGATE}\{h_u^{(k)} : u \in N(v)\})$$

Expressive Power of GNNs

- GNNs with injective AGGREGATE and UPDATE match WL
- Otherwise, less expressive

Limitations of WL

- Cannot distinguish some non-isomorphic graphs
- E.g., regular graphs with same degree sequence

Relational Pooling (RP-GNN)

$$\text{RP-GNN}(A, X) = \sum_{P \in \mathcal{P}} \text{GNN}(PAP^T, PX)$$

- Permutation invariant by averaging over permutations

Higher-Order GNNs

- Based on k-WL test
- Consider tuples of nodes (k-tuples)
- More expressive, but less scalable

Summary

- Graph convolutions from spectral theory
- GNNs as variational mean-field inference
- GNN expressiveness linked to WL test

Future Directions

- Beyond message passing
- Higher-order and relational methods
- Efficient and expressive GNN design