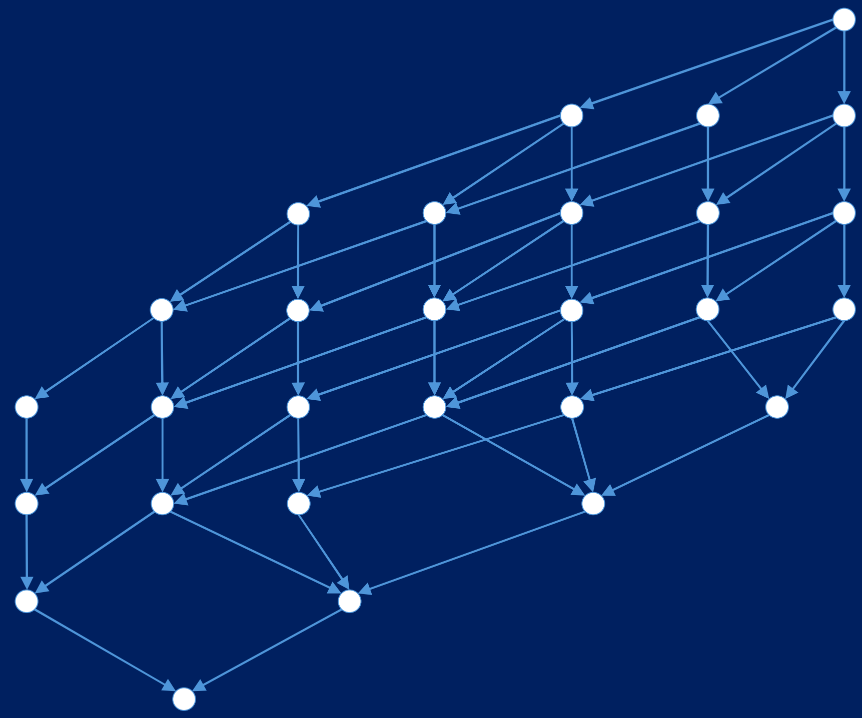
Graph Signal Processing

Bastian Seifert

Sommersemester 2025



Graph Neural Networks

TITEL	ZITIERT VON	JAHR
Semi-supervised classification with graph convolutional networks TN Kipf, M Welling International Conference on Learning Representations (ICLR)	44580	2016

TITEL	ZITIERT VON	JAHR
Graph Attention Networks P Veličković, G Cucurull, A Casanova, A Romero, P Liò, Y Bengio 6th International Conference on Learning Representations (ICLR 2018)	28235 *	2018

TITEL	ZITIERT VON	JAHR
Discrete signal processing on graphs A Sandryhaila, J Moura Signal Processing, IEEE Transactions on 61 (7), 1644-656	2100 *	2013

TITEL	ZITIERT VON	JAHR
The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains DI Shuman, SK Narang, P Frossard, A Ortega, P Vandergheynst IEEE Signal Processing Magazine 30 (3), 83-98	4974	2013

Convolutional Neural Networks

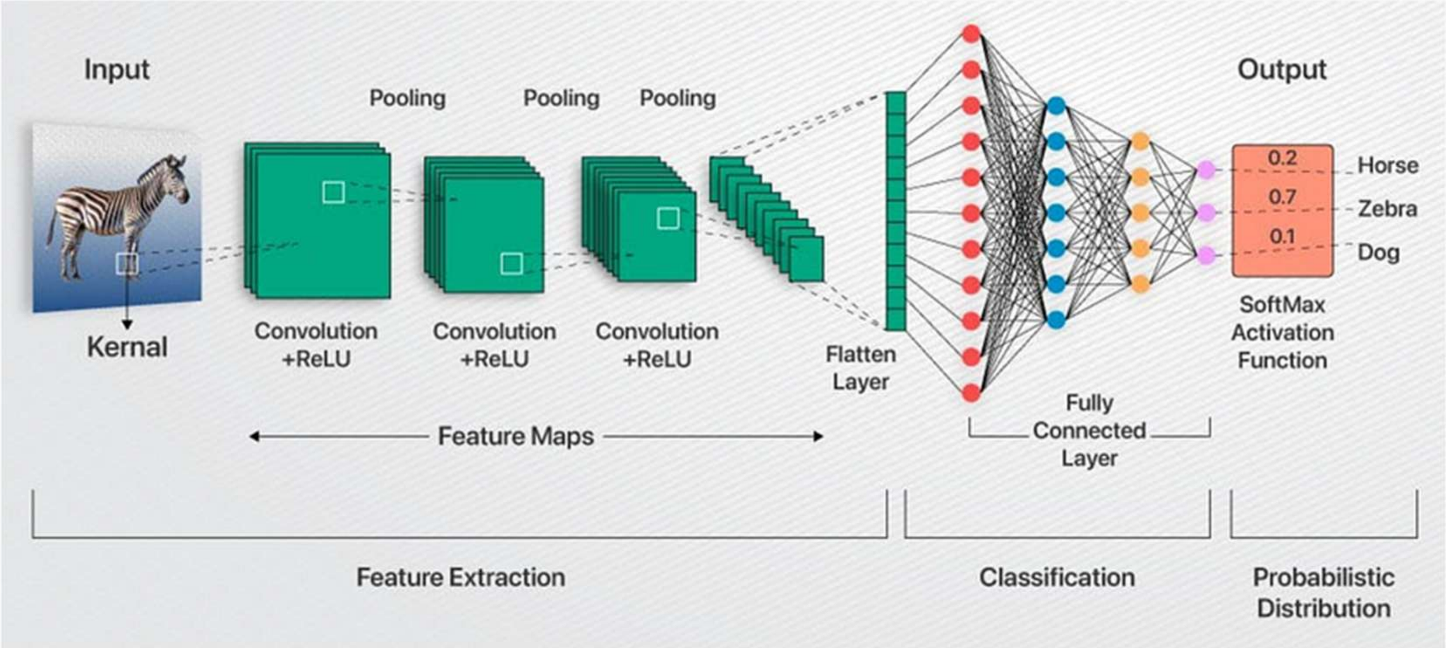
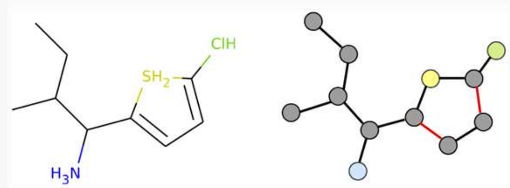


Figure: https://ravjot03.medium.com/decoding-cnns-a-beginners-guide-to-convolutional-neural-networks-and-their-applications-1a8806cbf536

Hidden layers: Convolution + Nonlinearity + Pooling

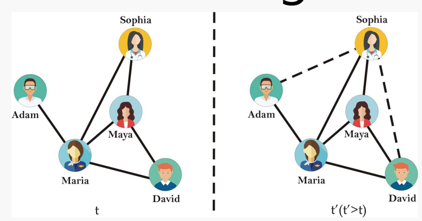
Types of graph learning task

- Graph-level tasks, try to predict properties of a whole graph
  Example: Classify molecules



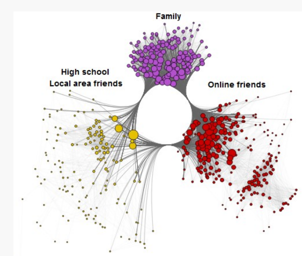
https://uvadlc-notebooks.readthedocs.io/en/latest/tutorial_notebooks/tutorial7/GNN_overview.html#PyTorch-Geometric

- Edge-level tasks, try to predict properties associated to edges of a graph
  Examples: Predicting links in social networks



https://media.springernature.com/full/springer-static/image/art%3A10.1038%2Fs41598-019-57304-y/MediaObjects/41598_2019_57304_Fig1_HTML.png

- **Node-level tasks:** try to predict properties associated to nodes of a graph
  Examples: Classify people in connection communities



https://i0.wp.com/neptune.ai/wp-content/uploads/2022/10/social-media-graph.png?ssl=1

Graph Convolutional Network

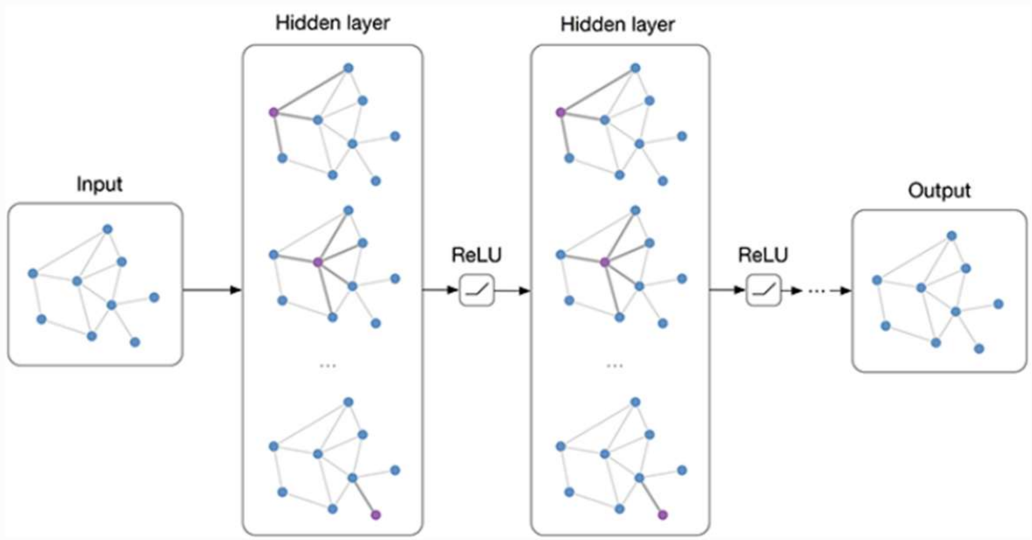


Figure: https://uvadlc-notebooks.readthedocs.io/en/latest/tutorial_notebooks/tutorial7/GNN_overview.html#PyTorch-Geometric

GCN = Learn filter coefficients + nonlinearities

Goal

Learn signal on a graph from feature matrix

$$X \in \mathbb{R}^{N \times D}$$

(i.e., D signals on the graph).

$A = U \Lambda U^{-1} \Rightarrow A^k = U \Lambda^k U^{-1}$

Spectral convolution

Spectral convolution is multiplication of a signal with a filter parameterized in graph Fourier domain

$$g_\theta x = U g_\theta(\Lambda) U^{-1} x,\ \theta \in \mathbb{R}^N$$

Evaluation of spectral convolution is computationally expensive

$$O(N^2)$$

(and for large graphs finding the eigenvectors even more so)

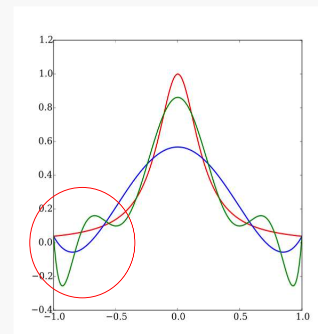
Chebyshev polynomials

Chebyshev polynomials are recursively defined as

$$T_{n+1}(x) = 2x T_n(x) - T_{n-1}(x)$$

$$T_0(x) = 1, T_1(x) = x, T_2(x) = 2x^2 - 1$$

they minimize approximation error and avoid Runge's phenomenon



Approximation

$$g_\theta(\Lambda) \approx \sum_{k=0}^{K} \theta_k T_k(\widetilde{\Lambda})$$

with rescaled $\widetilde{\Lambda} = \frac{2}{\lambda_{\max}} \Lambda - I$

Appproximation is a K-hop operator

Linear approximation

Using linear approximation and noting that we can approximate $\lambda_{\max} \approx 2$

as neural networks will adapt to this change, one arrives at

$$g_\theta x \approx \theta_0 x + \theta_1 (L - I) x = \theta_0 x - \theta_1 D^{-1/2} A D^{-1/2} x$$

In practical applications it's better to constrain parameter even more

$$\theta = \theta_0 = -\theta_1$$

leading to

$$g_\theta x \approx \theta (I + D^{-1/2} A D^{-1/2}) x$$

To avoid numercial instabilities and exploding gradients, one uses renormalization trick

$$I + D^{-1/2} A D^{-1/2} \rightarrow \widetilde{D}^{-1/2} \widetilde{A} \widetilde{D}^{-1/2}, \; \widetilde{A} = A + I$$

Degree von $\widetilde{A}$

Neural Network Layer

Every layer of a neural network is a non-linear function such that

$$H^{(\ell+1)} = f(H^{(\ell)}, A)$$

specific models only differ in how the non-linear function is choosen and parameterized.

Choose

$$f(H^{(\ell)}, A) = \sigma(\widetilde{D}^{-1/2} \widetilde{A} \widetilde{D}^{-1/2} H^{(\ell)} W^{(\ell)})$$

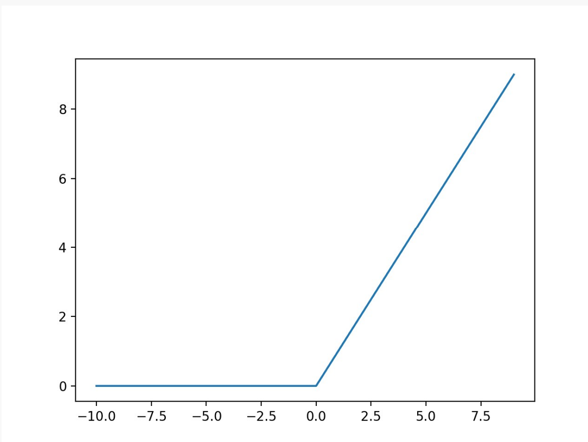
with some activation function and input layer

$$H^{(0)} = X$$

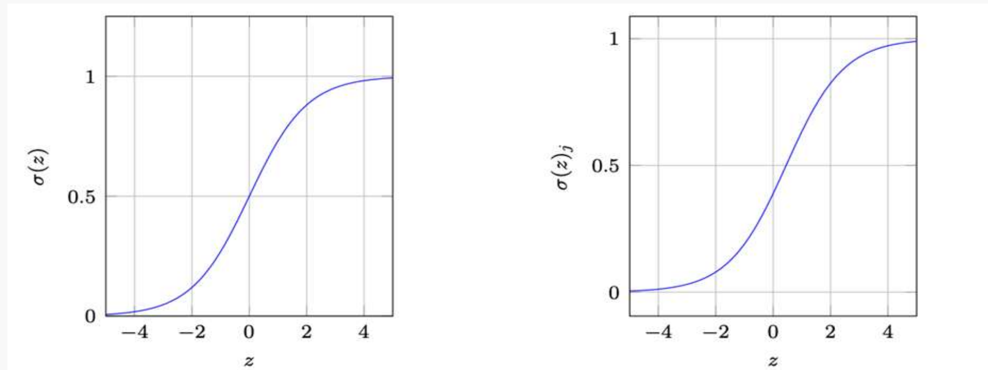
as well as output layer

$$H^{(L)} = H^{(L-1)}$$

Activation functions



ReLu



(a) Sigmoid activation function. (b) Softmax activation function.

https://databasecamp.de/wp-content/uploads/image-23-e1667663718858.png

Activation function controlls how well the model learns training data (but it's an art)

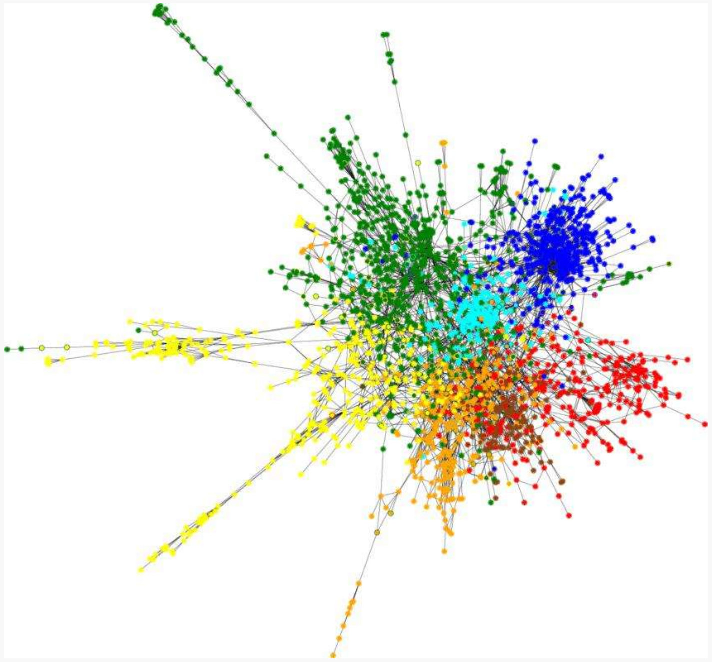
Application example: Classification of papers in citation network

Cora dataset

- 2708 scientific publications
- 5429 links (edges)
- 7 classes

Features/signal on nodes:

- 1433 words
- 0-1 vector indicating if word is present or not



https://production-media.paperswithcode.com/datasets/Cora-0000000700-ce1c5ec7_LD7pZnT.jpg

Activation function controlls how well the model learns training data (but it's an art)