NSF CyberTraining Workshop
Data-Centric Network Science

Mahshid R. Naeini, PhD
Md Abul Hasnat (Ph.D. Student)
Md Jakir Hossain(Ph.D. Student)
Electrical Engineering, University of South Florida
mahshidr@usf.edu; hasnat@usf.edu; mdjakir@usf.edu

UNIVERSITY of SOUTH FLORIDA

Learning Objectives of Today's Training

- Understand what networks/graphs are
- Understand why networks are important
- Learn about the field of Network Science
- See examples of network science techniques, graph-based analyses and machine learning (ML) techniques on graphs
- See examples of how techniques from network science and ML on graphs can help in reliability and cyber security evaluation of smart grids

What do we mean by Network/Graph?

- A network, also called a **graph** in mathematical language
- A graph is structured data consisting of two components: vertices, and edges.
- Graph provides a mathematical structure to analyze the pair-wise relationship between entities.
  - Networks are abstraction of connections, relations, or interactions among entities

points	**lines**	
vertices	edges, arcs	math
nodes	links	computer science
sites	bonds	physics
actors	ties, relations	sociology

Why Graphs/Networks and Network Science are important?

- We are surrounded by networks!

- Structures, inter-relations, interactions among entities in systems and datasets can embed important information about the collective static or dynamic behavior of the system.

- Network science is a discipline that investigates the structure and dynamics of networks, aiming to better understand the behavior, function and properties of the underlying systems and data they represent.

Network Science and Related Fields

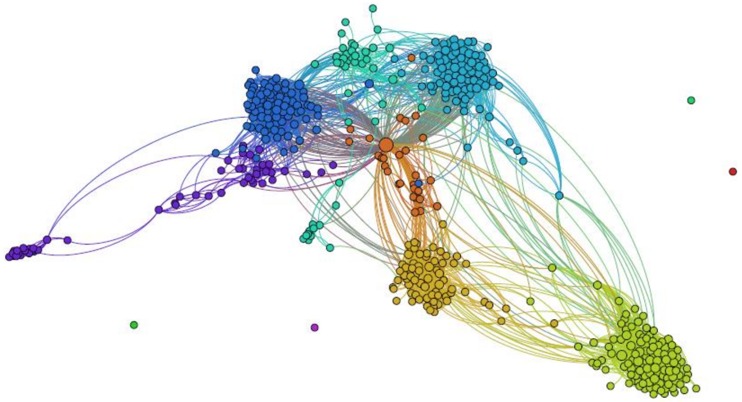
- Network Science extensively relies on:
  - **Graph theory**
    - Graph theory is mostly focused on static graphs
    - Graph theory does not scale well with the size of the real-world networks.
  - **Probability theory and statistics**
    - Understanding statistical features of networks!
    - For example, visualization of large networks are difficult but statistical methods help us to understand how the network looks like, when we can't actually look at it.
  - **Machine Learning on Graphs**
    - Studying the underlying graph structure by applying machine learning techniques to reveal insights on interaction of elements or use such interactions for descriptive and predictive analyses.
  - **Signal Processing on Graphs**
    - Defining signals on graph domain instead of Euclidean domain and use signal processing tools on graph signals.

Today's Training Schedule

- 9am-9:45am----Introduction and Short Course 1: Overview of Network Science
- 9:45am-10am ----Break
- 10am-10:45am----Short Course 2: Graph Signal Processing Basics with Applications in Smart Grids Cyber Security
- 10:45am-11am ----Break
- 11am-12pm----Short Course 3: Example of ML/Deep Learning on Graphs with Applications in Smart Grids
- 12pm-1pm----Lunch
- 1pm-2:15pm----Project 1
  - Security Evaluation of Smart Grids using Graph Signal Processing
- 2:15pm-2:30pm---Break
- 2:20pm-4pm----Project 2
  - Graph-CNN for Cyber Resilience of Smart Grids
- 4pm-4:15pm----Break
- 4-15pm-5pm----Conclusion and Discussion

Short Course 1:
Overview of Network Science

Examples of Networks: Social Networks

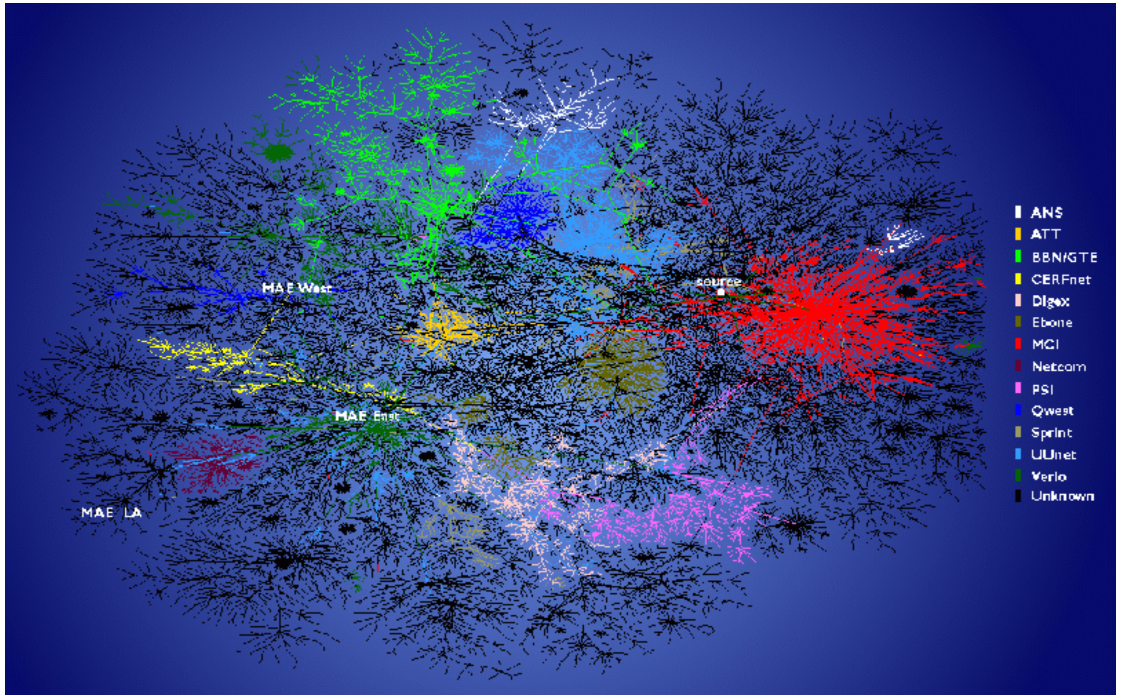


An individual's social network
(Network of friends in Facebook)
plotted using Gephi

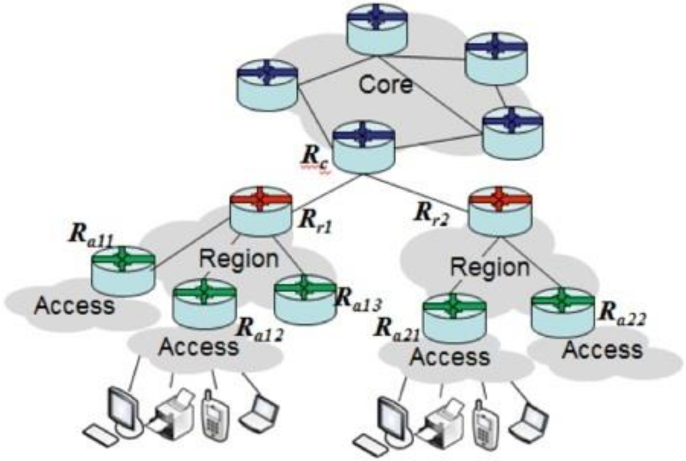
Imagine how large is the global social network

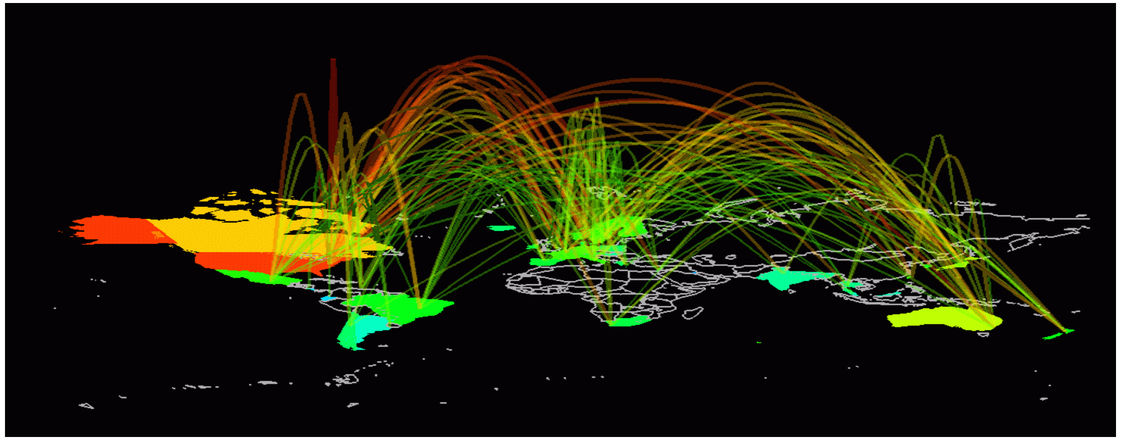
Keith Shepherd's "Sunday Best". http://baseballart.com/2010/07/shades-of-greatness-a-story-that-needed-to-be-told/

Examples of Networks: Internet network



Network map of Internet backbone network (without the physical location information)

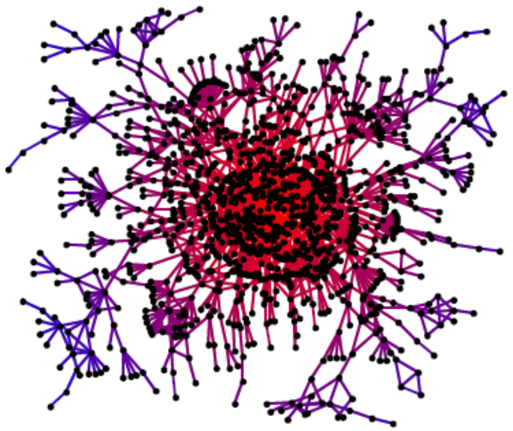




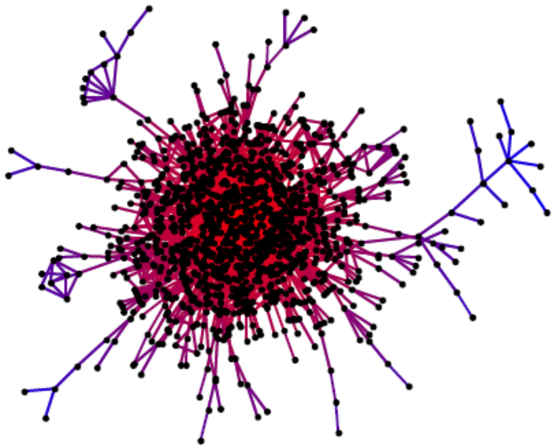
Network map of Internet backbone network with physical location information (an example of an infrastructure network)

Examples of Networks: World Wide Web

Network of personal homepages



Stanford

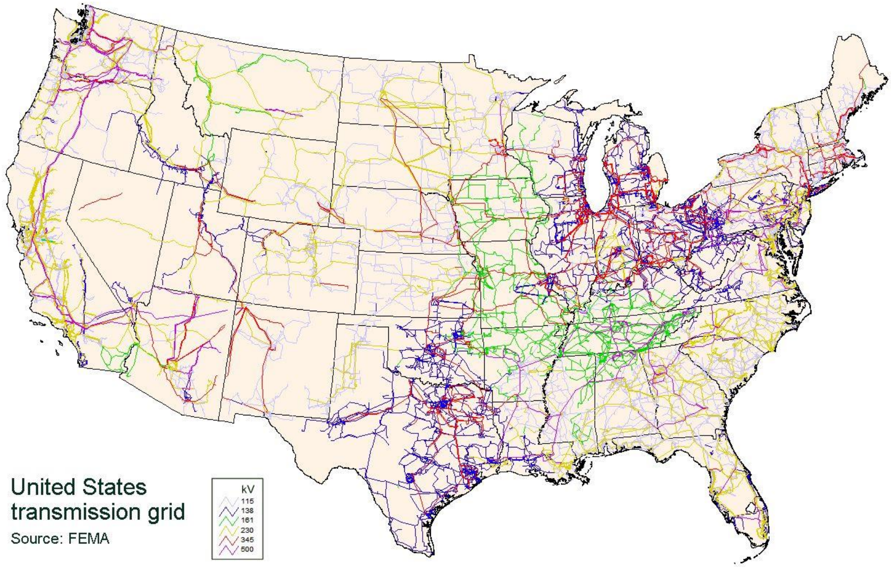


MIT

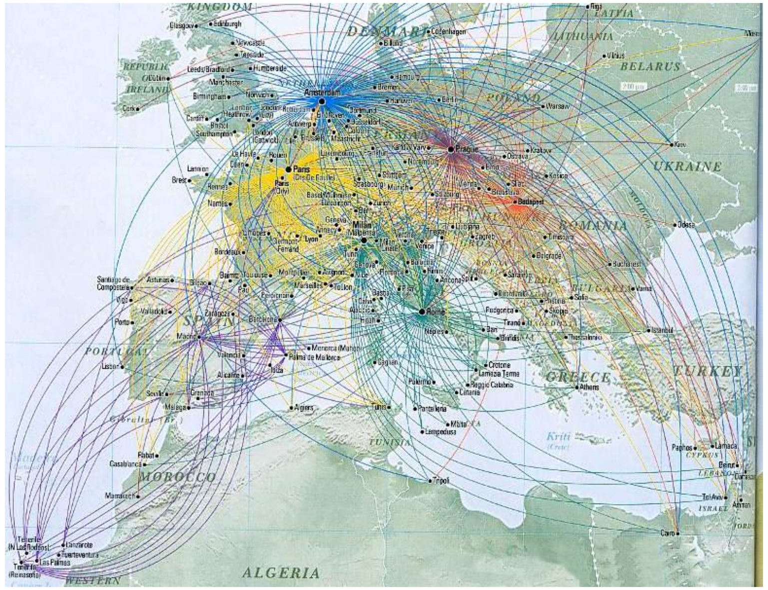
Source: Lada A. Adamic and Eytan Adar, 'Friends and neighbors on the web', *Social Networks*, 25(3):211-230, July 2003.

- WWW is an example of "**network of information**"
- Network of information are man made.
- Social networks can be considered as network of information.
- Nodes do not have physical locations

Examples of Networks: Critical Infrastructure Networks



Power grid



Airline map and transportation networks

Subway network and transportation networks

Other examples:

- Communication networks or Internet
- Phone and cellular network
- Water system
- Natural Gas Transportation system
- Railroad network
- Highways and roads

Examples of Networks: Business Tie Networks

Nodes: Companies

Investment

Pharma

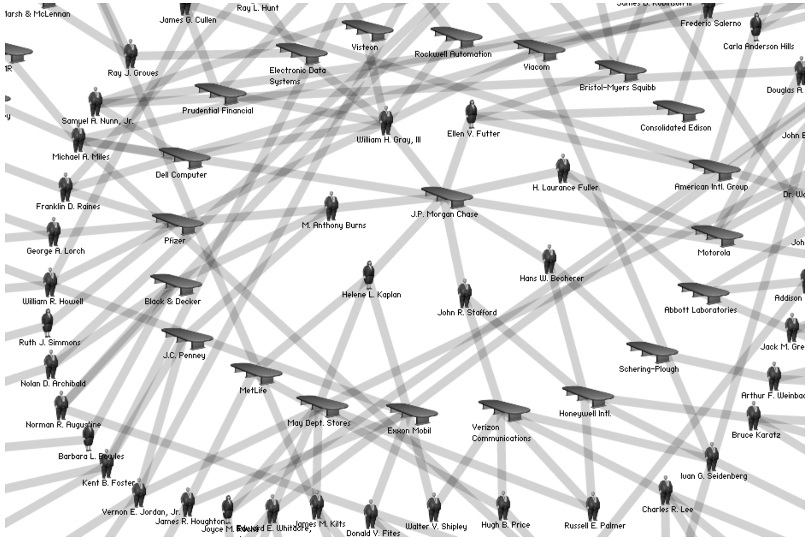
Research Labs

Public

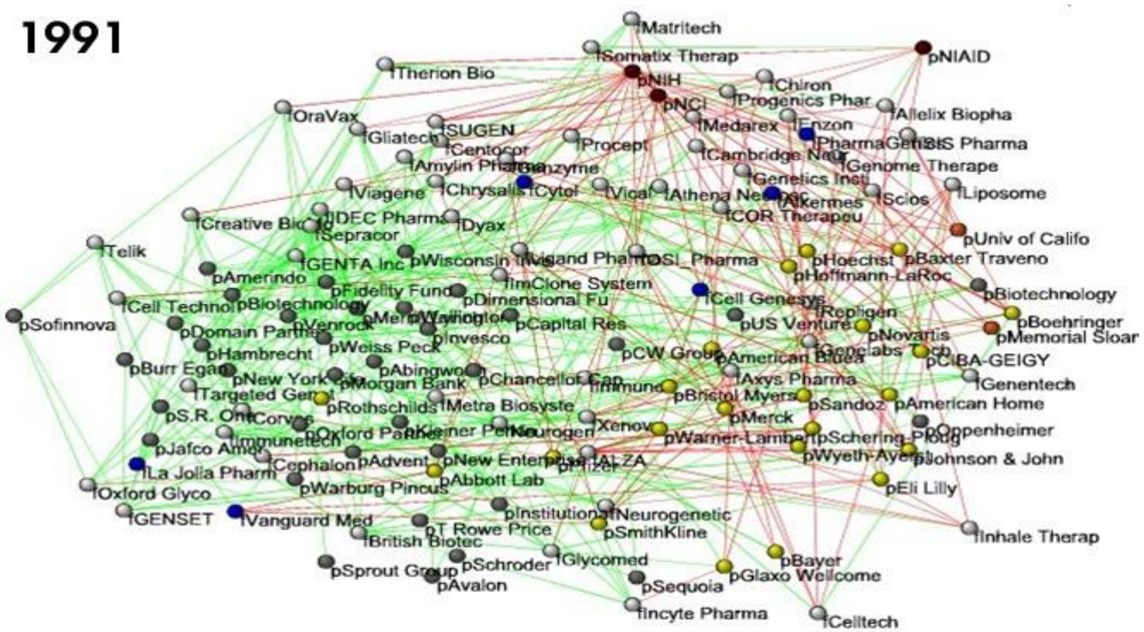
Biotechnology

Links: Collaborations

Financial



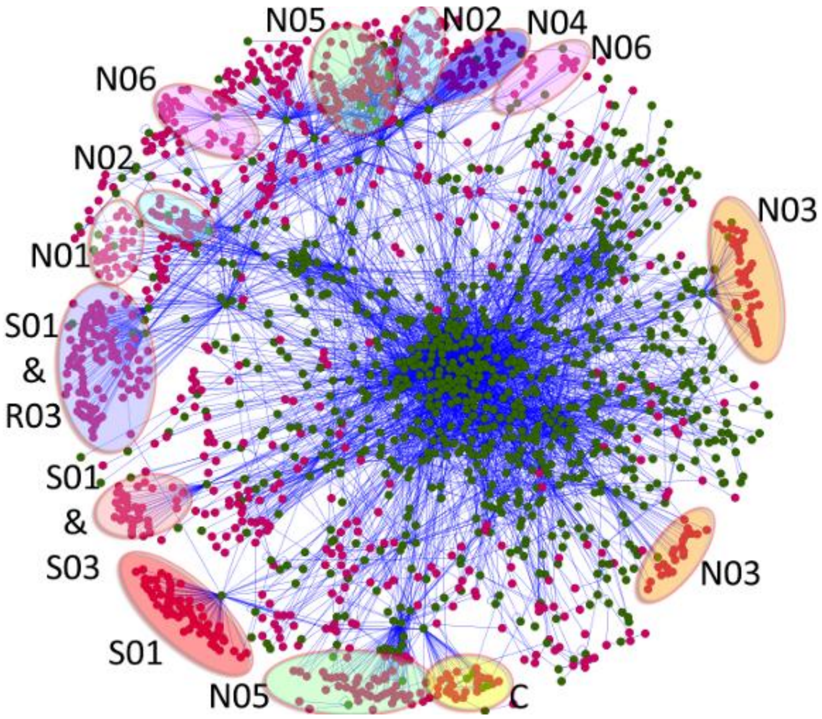
Source: http://theyrule.net



http://ecclectic.ss.uci.edu/~drwhite/Movie

Examples of Networks: Biological Networks

- Protein-protein interaction networks



http://bioinformatics.charite.de/synsysnet/

Other examples:

- Human gene network (human has 30,000 genes with complex interaction)
- Human diseases
- Human brain (10-100 billion Neurons connected by Synapses)

Other Examples of Networks

- Food web with species as nodes and predation as edges
- Postal network
- Citation network
- Financial network
- Email networks
- Peer-to-peer (P2P) networks, file sharing, Napster
- Recommender networks
- ….

Network Analysis: Structure of Networks

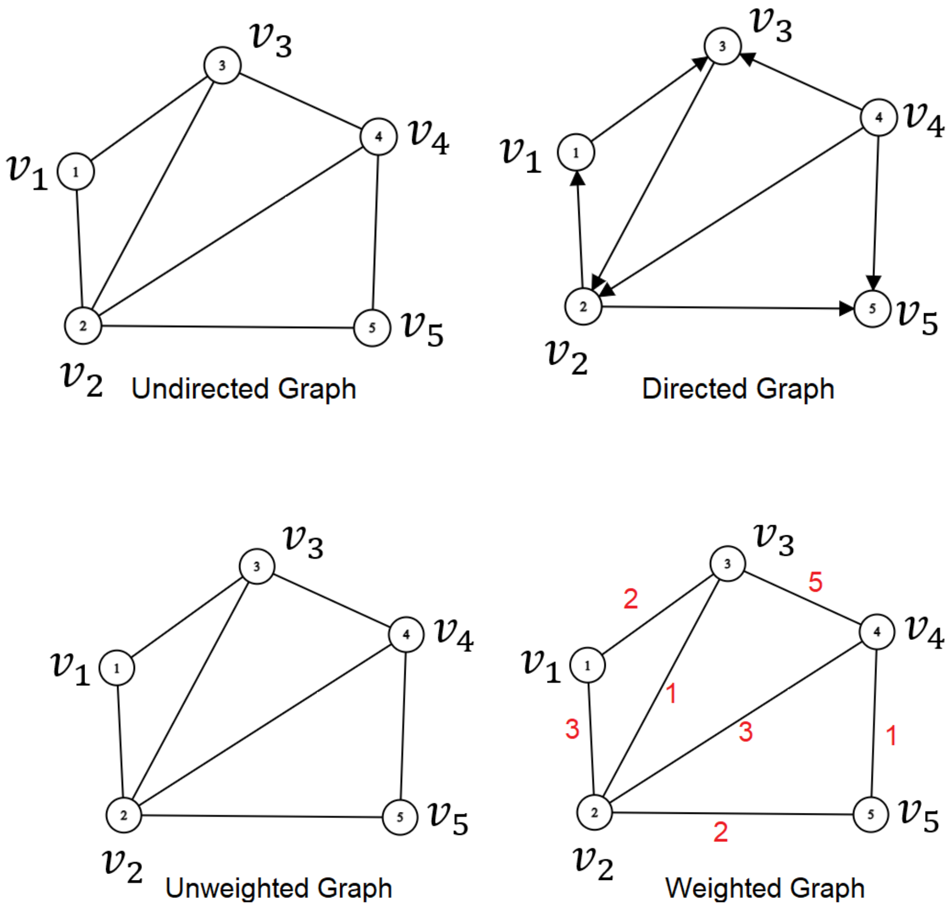
- Measure and metrics to evaluate network structures
- Degree of nodes
- Centrality measures
- Clustering and communities
- Hubs and authorities
- Shortest paths and graph diameter
- Network growth
  - random attachment: new node picks any existing node to attach to
  - preferential attachment: new node picks from existing nodes according to their degrees
- Network models
  - Erdos-Renyi random graph
  - Watts-Strogatz small world model
  - Barabasi-Albert scale-free networks

Network Analysis: Dynamics of networks

- Developing metrics and statistics to assess and identify change within and across networks.
- Developing and validating simulations to study network change, evolution, adaptation, decay.
- Node interactions
- Spread and Propagation Dynamics
  - Spread of disease
  - Opinion formation
  - Spread of computer viruses
  - Spread of failures in power systems
  - Gossip
- Control processes in networks

Review of basics of graphs!

- Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a set of vertices and set of edges.
- The Vertices are connected by the edges.



Review of basics of graphs!

Adjacency Matrix:

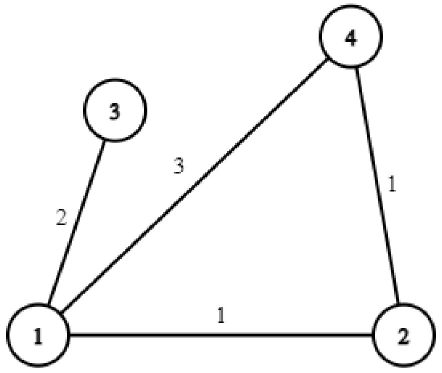
$$A_{ij} = \begin{cases} 1, & (i,j) \in \mathcal{E} \\ 0, & (i,j) \notin \mathcal{E} \end{cases}$$

Weighting Matrix:

$$W_{ij} = \begin{cases} w_{ij}, & (i,j) \in \mathcal{E} \\ 0, & (i,j) \notin \mathcal{E} \end{cases}$$

Degree Matrix:

$$D_{ij} = \begin{cases} \sum_{j=1}^{N} w_{ij}, & i = j \\ 0, & i \neq j \end{cases}$$



$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{W} = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 1 & 0 & 0 & 1 \\ 2 & 0 & 0 & 0 \\ 3 & 1 & 0 & 0 \end{bmatrix} \qquad \boldsymbol{D} = \begin{bmatrix} 6 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix}$$

Mathematical Presentations of Graphs

Weighting Matrix:

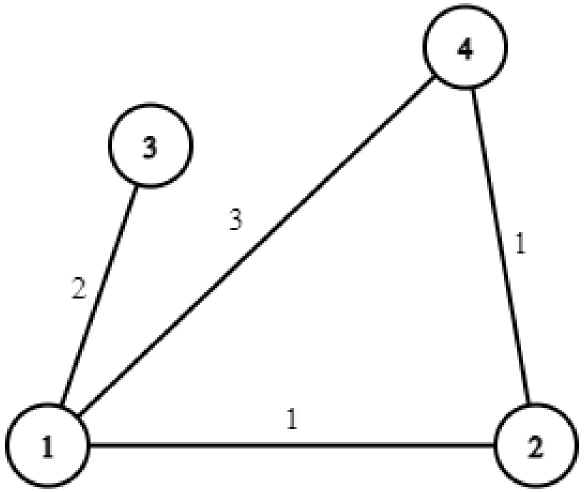
$$W_{ij} = \begin{cases} w_{ij}, & (i,j) \in \mathcal{E} \\ 0, & (i,j) \notin \mathcal{E} \end{cases}$$

Degree Matrix:

$$D_{ij} = \begin{cases} \sum_{j=1}^{N} w_{ij}\,, & i = j \\ 0, & i \neq j \end{cases}$$

Laplacian Matrix:

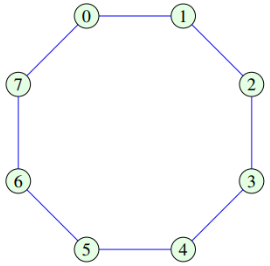
$$L_{ij} = \begin{cases} \sum_{j=1}^{N} w_{ij}\,, & i = j \\ -w_{ij}, & i \neq j \end{cases}$$



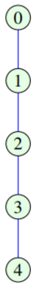
$$\boldsymbol{W} = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 1 & 0 & 0 & 1 \\ 2 & 0 & 0 & 0 \\ 3 & 1 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W} = \begin{bmatrix} 6 & -1 & -2 & -3 \\ -1 & 2 & 0 & -1 \\ -2 & 0 & 2 & 0 \\ -3 & -1 & 0 & 4 \end{bmatrix}$$

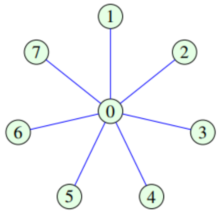
Network Models



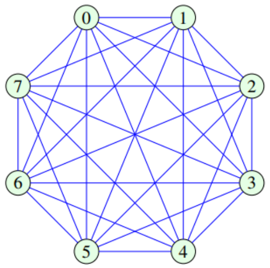
Circular Graph



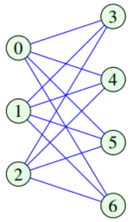
Path Graph



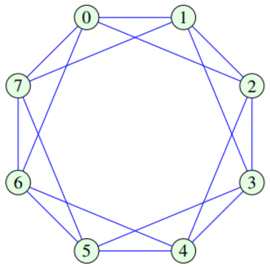
Star Graph



Complete Graph



Bipartite Graph

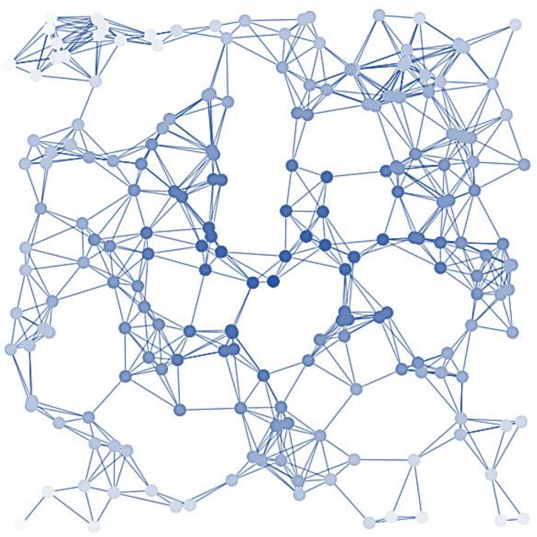


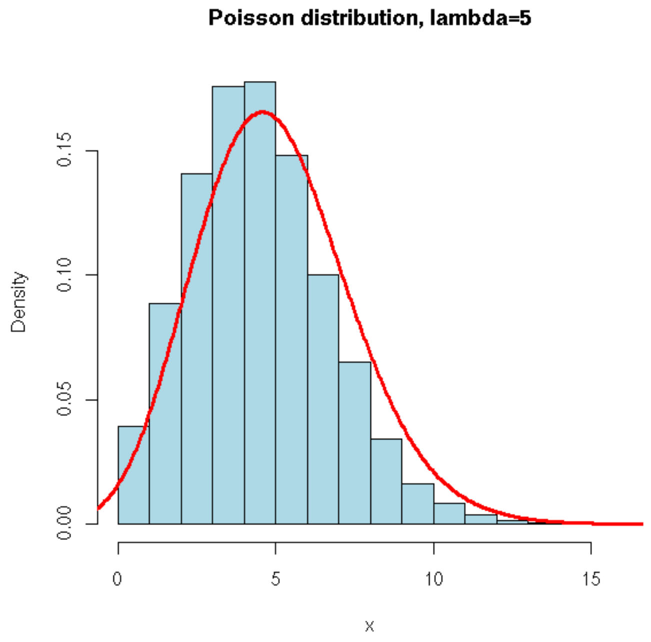
Regular Graph

Image Courtesy: Stankovic, L., Mandic, D., Dakovic, M., Brajovic, M., Scalzo, B., & Constantinides, T. (2019). Graph Signal Processing--Part I: Graphs, Graph Spectra, and Spectral Clustering. *arXiv preprint arXiv:1907.03467*.

Network Models: Random Networks

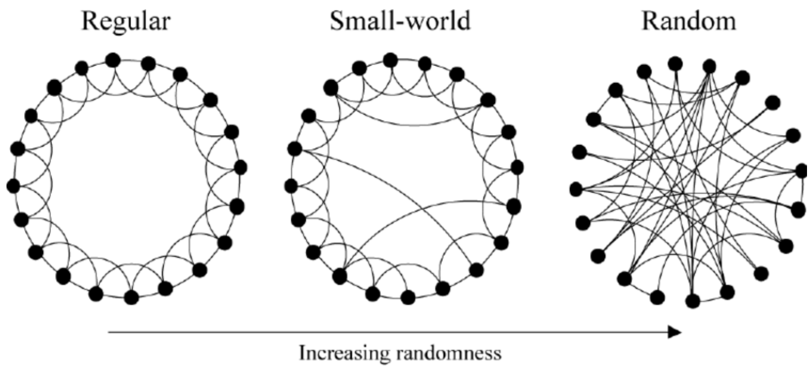
- Nodes connected at random
  - start with N nodes and connect every pair of them with probability p
- Number of edges incident on each node has Poisson distribution





Network Models: Small-World Network

- A famous experiments carried out by Stanley Milgram in the 1960s.
  - Asking participants to pass a letter to one of their acquaintances in an attempt to get it to an assigned target individual Outcome:
    - Most of the letters in the experiment were lost
    - A quarter reached the target and passed on average through the hands of only about six people in doing so.
  - The experiments probed the distribution of path lengths in an acquaintance.
    - Path length is defined by minimum number of edges needed to pass from first point to the other



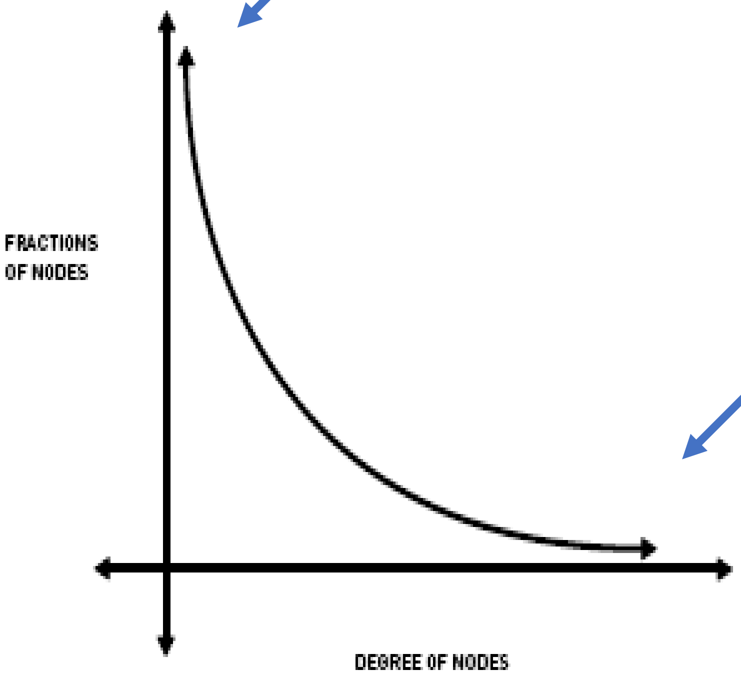
- This experiment was the origin of the popular concept of the “six degrees of separation“.
  - Only six hops separate any two people in the world
- Duncan Watts and Steven Strogatz
  - a few random links in an otherwise structured graph make the network a small world: the average shortest path is short

Slide’s reference: Albert-László Barabási and Roberta Sinatra at www.BarabasiLab.com

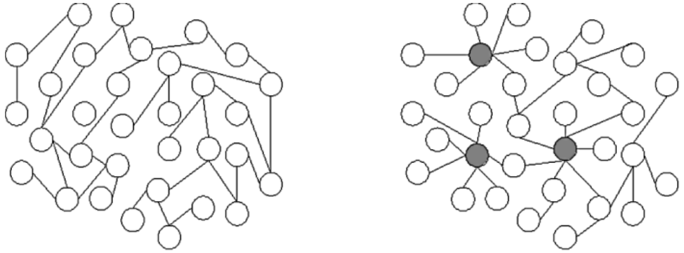
Network Models: Power Law Networks (Scale-free Networks)

- Many real-world networks contain hubs: highly connected nodes
- Usually, the distribution of edges is extremely skewed

many nodes with few edges



fat tail: a few nodes with a very large number of edges



(a) Random network (b) Scale-free network

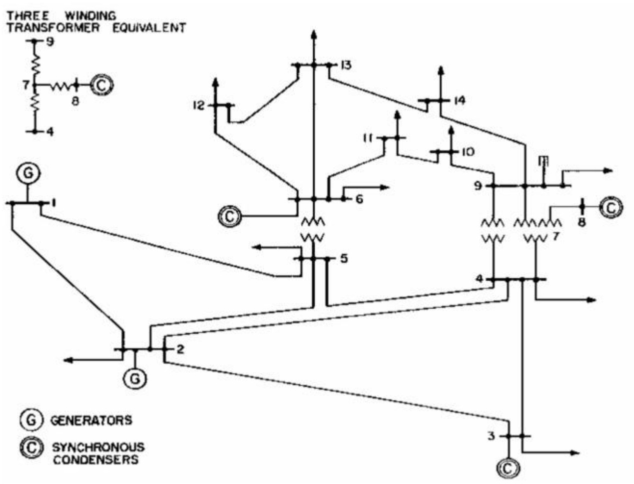
Where can network analysis be useful?

- Few example of useful network structure and dynamic analysis:
  - Reliability and resiliency analysis of infrastructures, smart grids and computer networks
  - Detection abnormal interactions
  - Descriptive or predictive analyses
  - Identifying vulnerable points of networks
  - Cyber security area: to model the spread of viruses
  - Cascading failures in power grids

Networks in Smart Grids

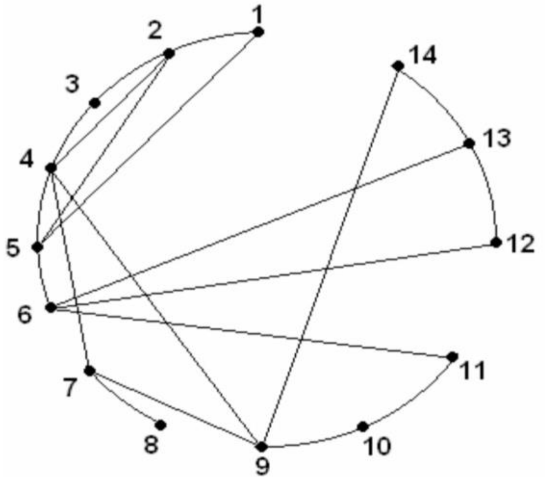
Physical Topology-based Graphs of Power Grids

- Power grids can be represented as unweighted/weighted graph $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$
  - Set of vertices $\boldsymbol{V}$ (substations - generators, transmission, and distribution)
  - Set of edges $\boldsymbol{E}$ (transmission lines)



IEEE 14 bus network

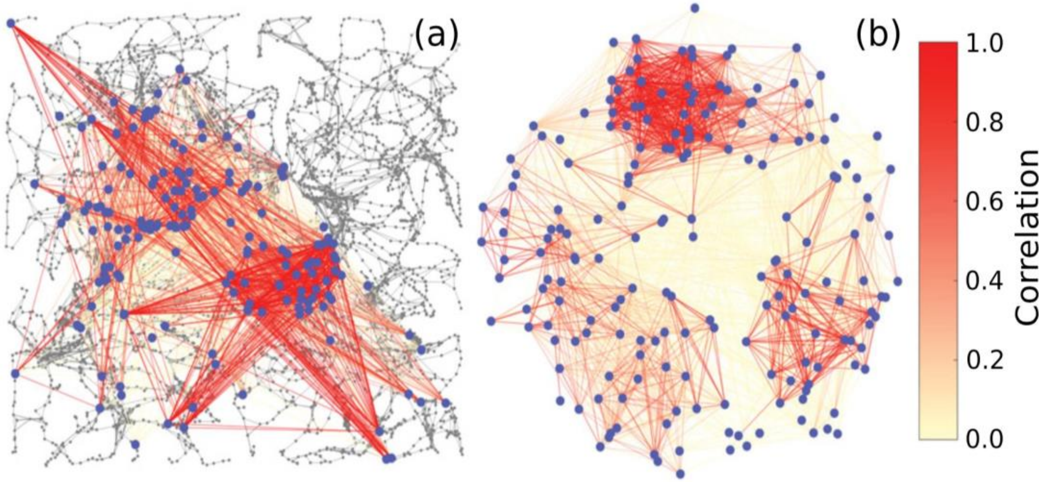
Representation of
IEEE 14 bus network
by graph G



Holmgren, Å.J., 2006. Using graph models to analyze the vulnerability of electric power networks. *Risk analysis*, *26*(4), pp.955-969.

Data-Driven Graphs: Correlation-based Interaction Graph

- Data-driven Interaction Graph: Correlation
  - Pairwise correlation between failures of components in cascades.
- Dependence between failures captured in correlation matrix.
  - i and j elements are Pearson correlation coefficient between failure statuses of components i and j

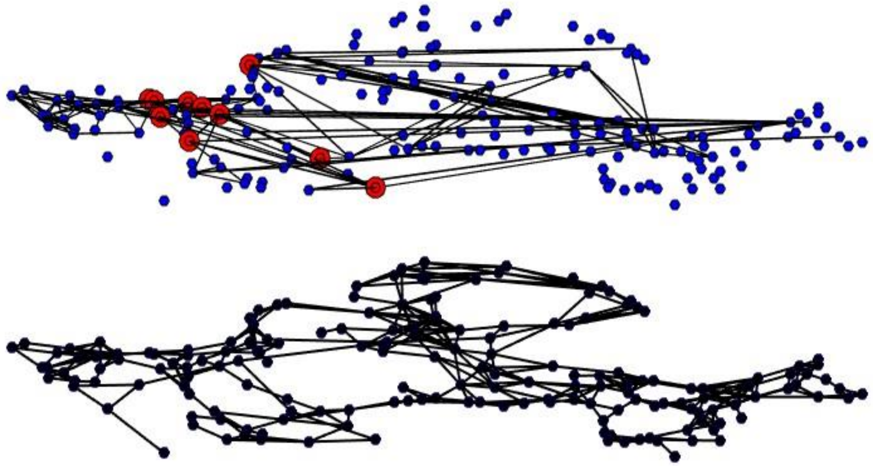


Correlation-based interactions in Texas power grid*

*Yang, Y., Nishikawa, T. and Motter, A.E., 2017. Vulnerability and cosusceptibility determine the size of network cascades. *Physical review letters*, *118*(4), p.048301.

Data-Driven Graphs: Influence-based Interaction Graphs

- Influence model
  - A networked Markov chain framework
  - Characterizes influences that a component j receives from all other components in a form of conditional probabilities

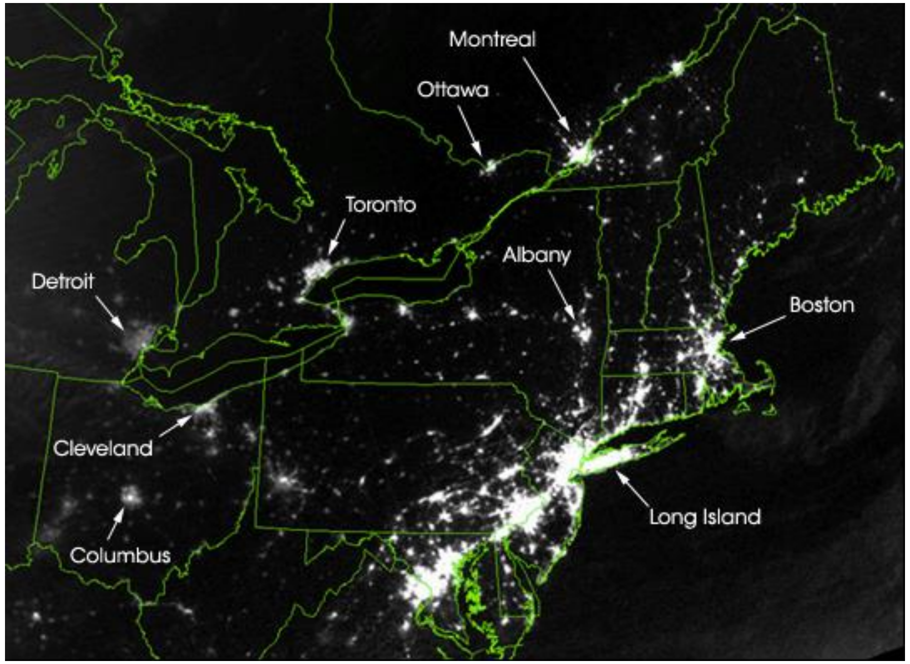


Influence-based topology atop transmission line graph of IEEE 118-bus system

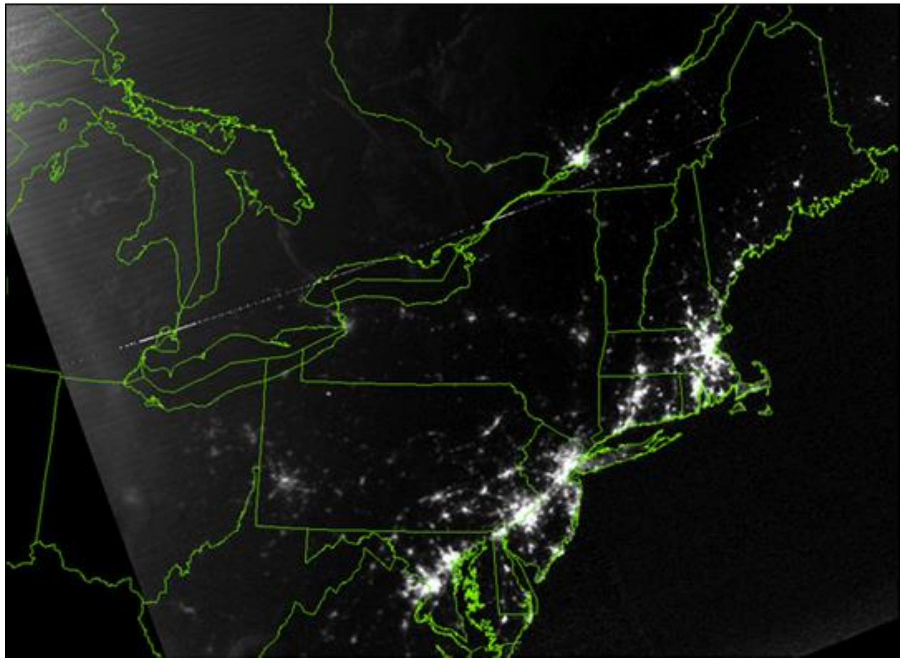
- A Review of various Interaction Graph Representation for Power Systems:
  - U. Nakarmi, M. Rahnamay Naeini, Md J. Hossain, Md A. Hasnat, Interaction Graphs for Cascading Failure Analysis in Power Grids: A Survey, Energies, vol. 13, no. 9, 2020.

*C. Asavathiratham et al., "The influence model," IEEE Control Sys., vol. 21, no. 6, pp. 52–64, 2001.

Example Application: Vulnerability Analyses of Power Grids to Cascading Failures



August 14, 2003: 9:29pm EDT
20 hours before

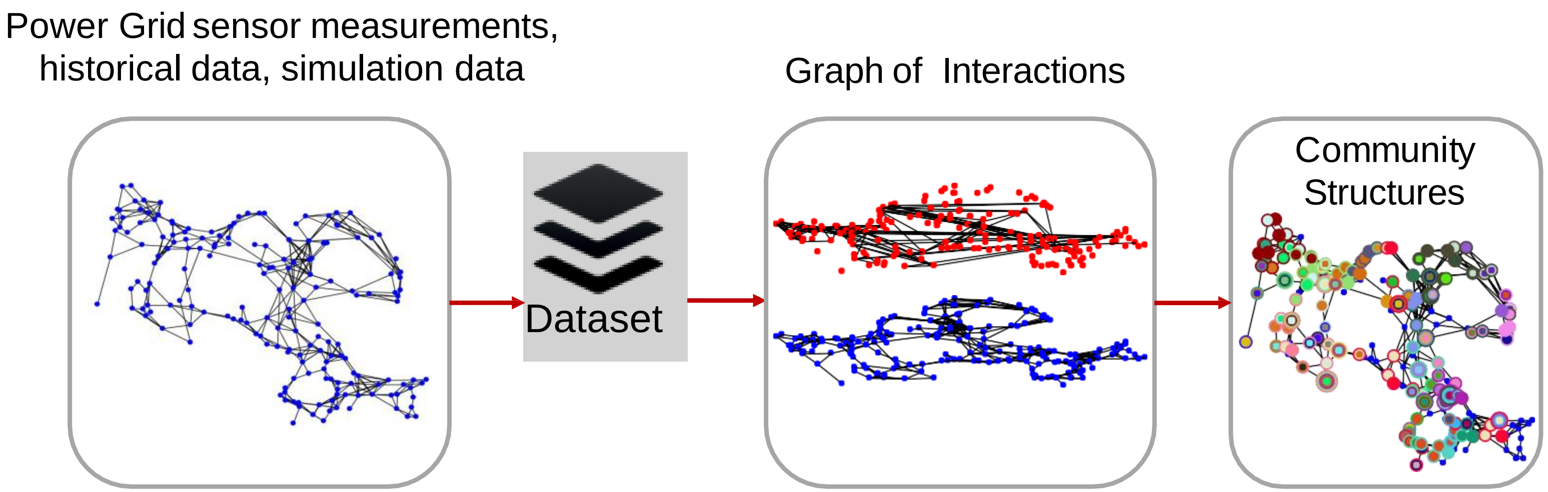


August 15, 2003: 9:14pm EDT
7 hours after

Cascading failures in power systems is an example of spreading phenomena on complex networks. We want to understand how

- network structure affects the robustness of the system.
- to develop quantitative tools to assess the interplay between network structure and the dynamical processes on the networks, and their impact on failures.
- to predict the spread of failures and blackouts using network science

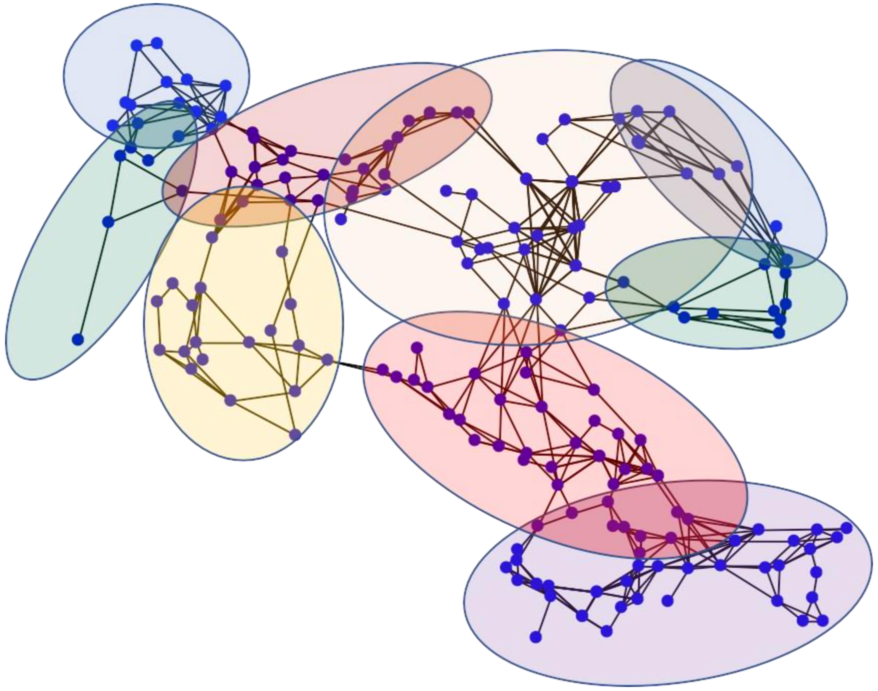
A Data-Driven Graph-based Approach to Analyze Vulnerabilities in Power Grids to Cascading Failures



*U. Nakarmi, M. Rahnamay-Naeini, and H. Khamfroush, **Critical Component Analysis in Cascading Failures for Power Grids using Community Structures in Interaction Graphs,** in IEEE Transactions on Network Science and Engineering, doi: 10.1109/TNSE.2019.2904008.

Communities in Graphs

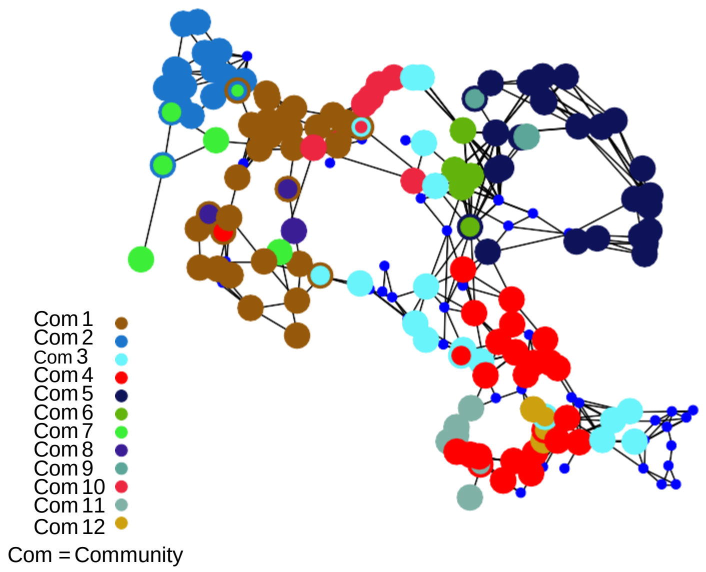
- Communities are densely connected groups of components with scarce connections to components of other groups.
- Communities are likely to trap failures during cascades.
  - strong internal influences/interactions
  - weak external interactions
- Capability to cripple global spread.
- Cascades may spread through links between communities.
- Community detection algorithms* can help identify various forms of communities in graphs (disjoint, overlapped)

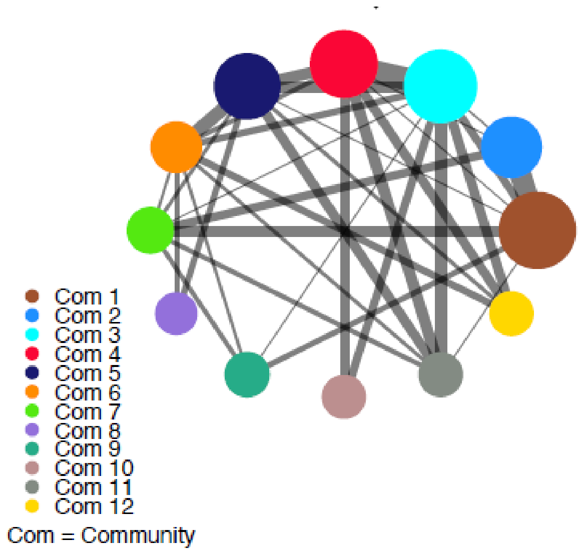


*Yang, Z., Algesheimer, R. & Tessone, C. A Comparative Analysis of Community Detection Algorithms on Artificial Networks. *Sci Rep* **6,** 30750 (2016).

Community Structures in Interaction Graphs

- Community membership of transmission lines of IEEE 118-bus transmission line graph





Infomap Overlap Community Structure

Conclusions

- Structure of the interactions reveal important information about the dynamics of the system (cascading failures in this example).
- Community structures affect the behavior of cascade processes by trapping failures within communities
  - The likelihood of failures spreading outside the community is lower.
  - Mechanisms can be designed based on community structures to prevent the spread of failures in the system (e.g., intentional islanding, enhanced capacity or added protection for central components in the cascade process).
- Community sizes can be used to estimate the cascade size when initiated.

U. Nakarmi, M. Rahnamay-Naeini, and H. Khamfroush, **Critical Component Analysis in Cascading Failures for Power Grids using Community Structures in Interaction Graphs,** in IEEE Transactions on Network Science and Engineering, doi: 10.1109/TNSE.2019.2904008.

Network Science Resources

- Books and Websites:
  - Albert-lászló Barabási is a scientist and professor at Northeastern University's Center for Complex Network Research (CCNR) a major contributor to the development of network science and the statistical physics of complex systems. (HTTPS://BARABASI.COM/)
    - Network Science, By A. Barabasi, [Available Online At http://networksciencebook.com/
  - Networks: An Introduction, By Mark Newman, Oxford University Press, 2010,   ISBN: 978-0199206650
- Network Analysis and Visualization Tools:
  - NetworkX: https://networkx.org/
  - Gephi: https://gephi.org/
  - Graphviz: http://www.graphviz.org/
- Datasets
  - Stanford Large Network Dataset Collection http://snap.stanford.edu/data/
  - KONET: http://konect.cc/
  - NetWiki: http://netwiki.amath.unc.edu/SharedData/SharedData

Questions?

Graph signal Processing Basics with Application in Smart Grid

Md Abul Hasnat (Ph.D. Student)

Supervised by

Dr. Mahshid Rahnamay Naeini

Electrical Engineering, University of South Florida

hasnat@usf.edu

Signal, Signal Processing, and Transforms

Signals

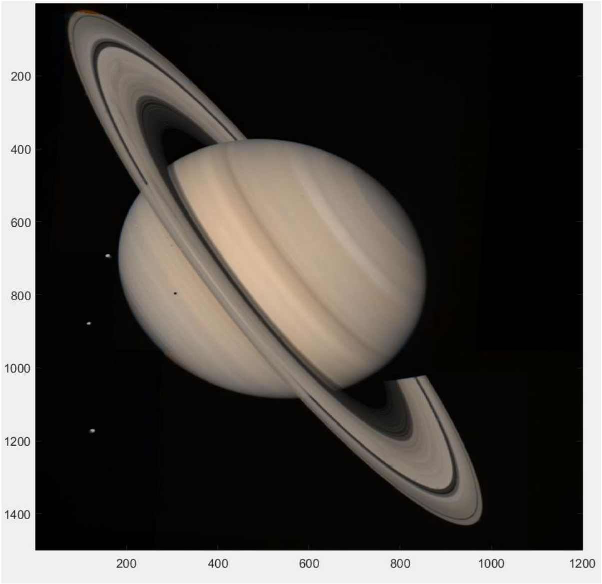
- Set of information representing any physical phenomenon.
- A mathematical function of one or more independent variables.

Dimension of a signal → number of independent variables

- 1-D Signals: Temperature of a point at time, t: $\theta(t)$, speech signal, Electro-cardiogram (ECG) signal.
- 2-D Signal: A gray-scale image, $I(x, y)$.
- 3-D Signal: A video, $V(x, y, t)$.

1D Signal: Electro-cardiogram (ECG) Signal

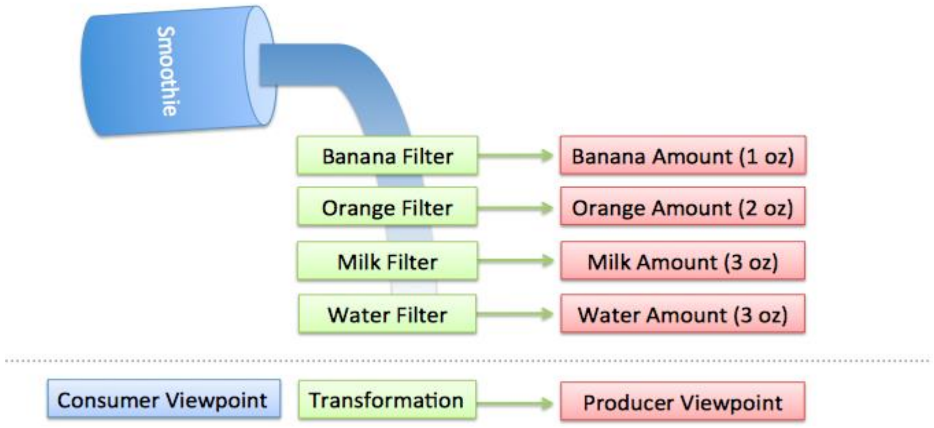
Courtesy: https://www.aedsuperstore.com/resources/ekg-quick-reference-guide/



2D Signal: Gray-Scale Image

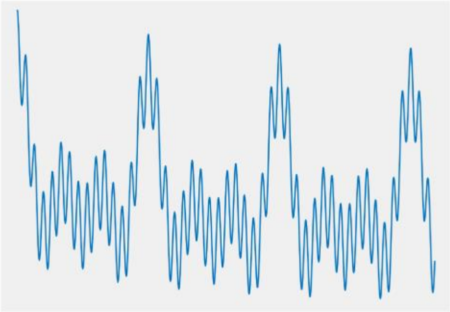
Transformation of Signals

Decomposing Smoothie

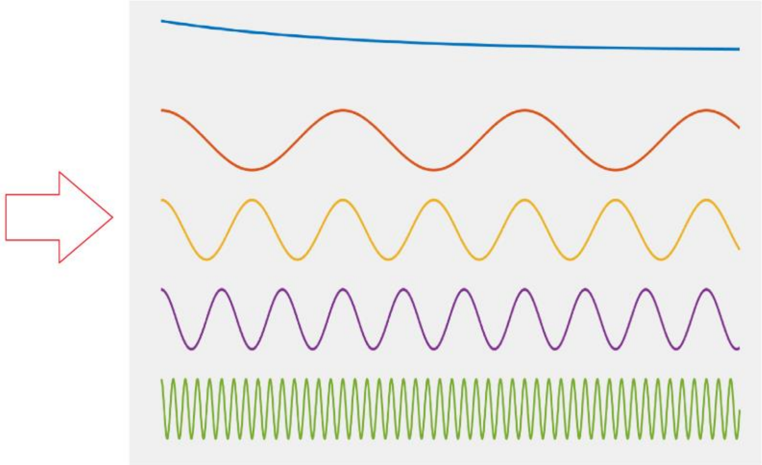


Courtesy: https://betterexplained.com/articles/an-interactive-guide-to-the-fourier-transform/

Decomposing A Signal



Original Signal



Basis Signals (Ingredients)

Why we need transforms?

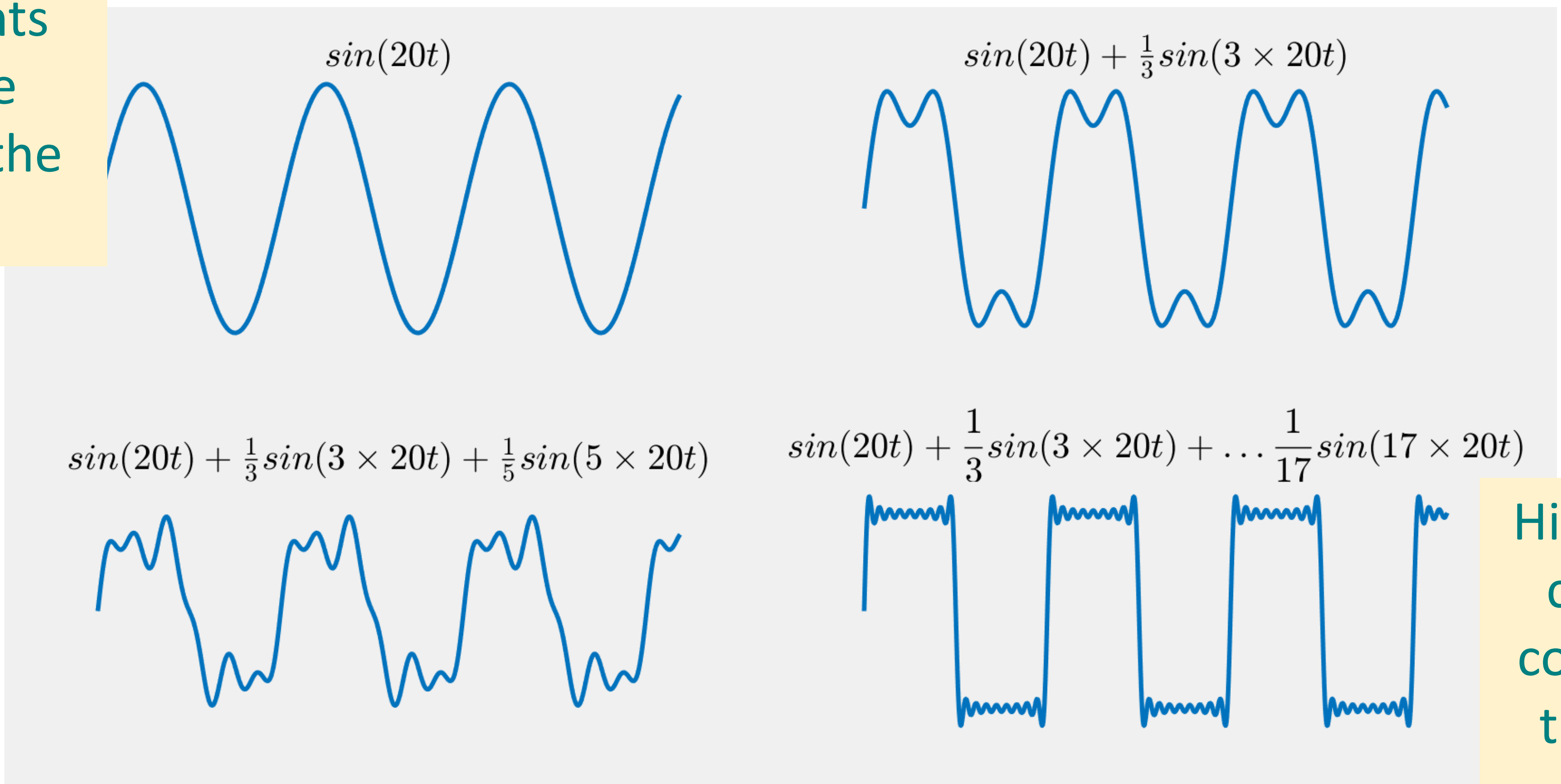
- Provide another perspective of looking into signal
- Make processing signals easier

Examples: Fourier Transform, Laplace Transform, Wavelet Transform.

- Signals breaks down into basis functions.
- Basis functions add up to the signal. (Linear Combination)
- Different transforms involves different set of basis functions.

Decomposing signal into sinusoids

Low- frequency components forms the 'shape' of the signal

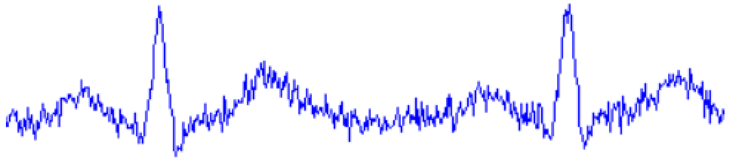


High-frequency components corresponds to the details of the signal

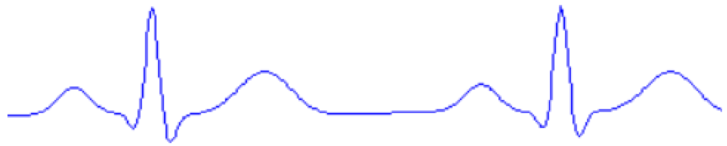
Processing of Signals

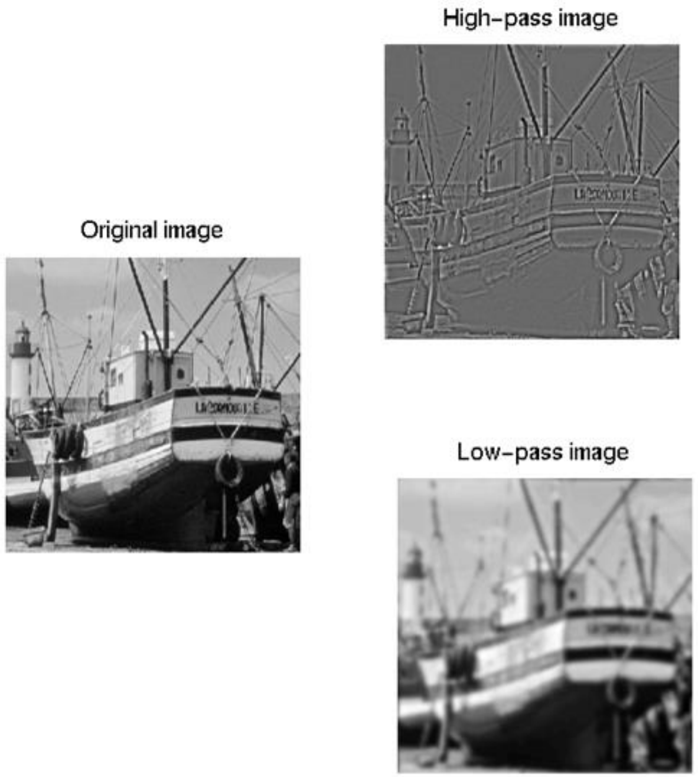
- Processing of signal means changing the signal for any specific purpose.
- Filtering, sampling, etc. are common processing tasks on signals.

Original ECG Signal:



ECG Signal after low-pass filtering:





Courtesy:https://eeweb.engineering.nyu.edu/iselesni/sass/SASS_toolbox/html/Example2.html

The Fourier Transform

Any Signal can be expressed as a linear combination of finite or infinite number of complex sinusoids.

$$\underbrace{x(t)}_{\text{Signal}} = \underbrace{\int_{\omega=-\infty}^{\infty}}_{\text{Linear Combination}} \underbrace{\frac{1}{2\pi}X(\omega)}_{\text{Fourier Coefficient}} \underbrace{e^{j\omega t}}_{\text{Basis Functions}} d\omega$$

Signal

Linear Combination

Fourier Coefficient

($\frac{1}{2\pi}$ is for normalization)

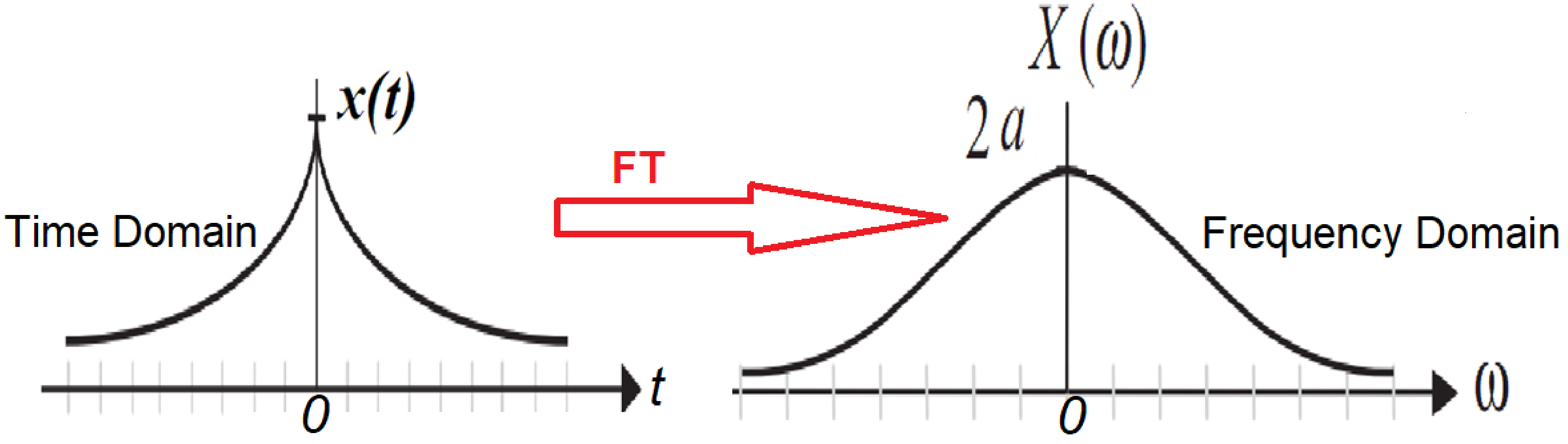
Basis Functions: Complex Sinusoids of different frequencies,ω

$$e^{j\omega t} = \cos(\omega t) + j\sin(\omega t)$$

Fourier Transform of $x(t)$:

$$X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t}dt$$

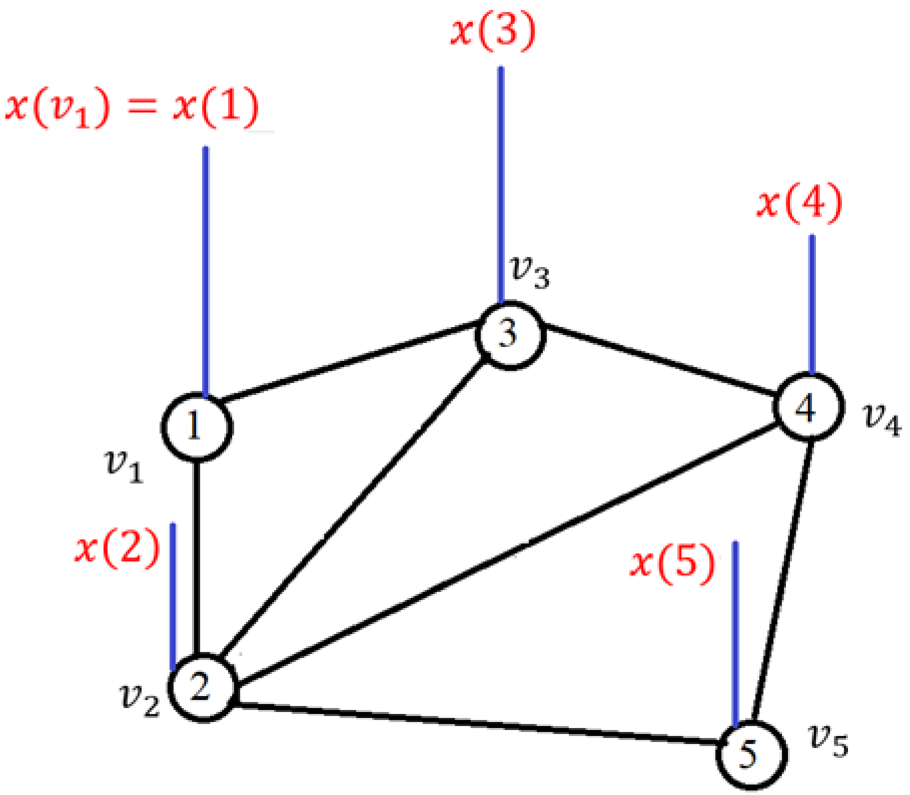
The time and the frequency domain



- Both $x(t)$ and $\boldsymbol{X(\omega)}$ presents the same physical signal.
- $x(t)$ represents the variation of amplitude with t (time).
- $\boldsymbol{X(\omega)}$ represents the contribution of each frequency component (complex sinusoids, $\boldsymbol{e^{j\omega t}}$) in the signal.
- Analyzing the signal in terms of $x(t)$ is called *time domain* analysis while analyzing it in term of $\boldsymbol{X(\omega)}$ is called *frequency domain* analysis.

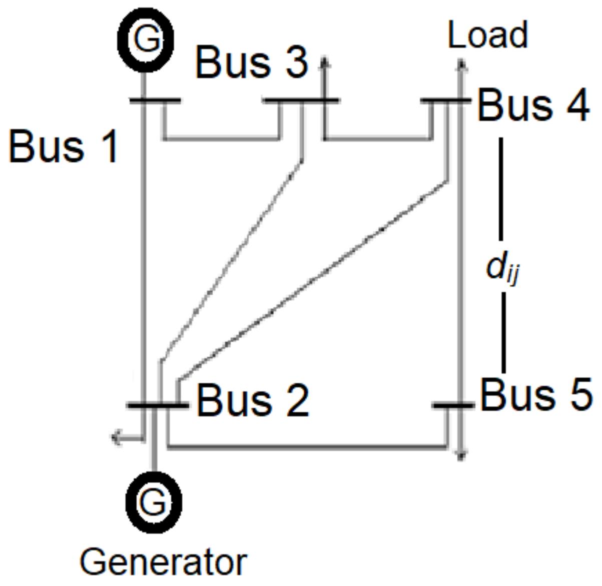
Graph signal and smart grid graph signals

Signals on Graphs

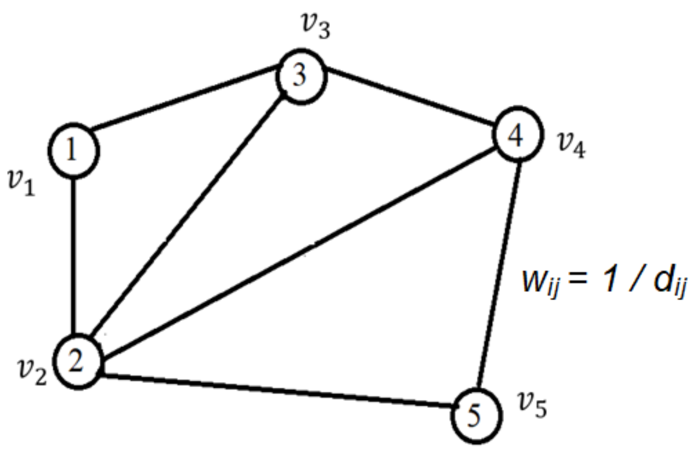


- Graph Signal: Signal defined on a graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.
- Signals reside in the non-Euclidean graph domain.
- The independent variable is the graph vertices.
- $x: \mathcal{V} \rightarrow \mathbb{R}$.
- $x(v_n)$ is the value of the graph signal at vertex (v_n). For simplicity, we use $x(n)$.

Example: Smart grid graph signals: Grid as a Graph



5 Bus Power System



Corresponding Graph

- Weighted Undirected graph,
  $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$.
- Buses are Vertices ($\mathcal{V}$)
- Transmission lines are edges ($\mathcal{E}$)

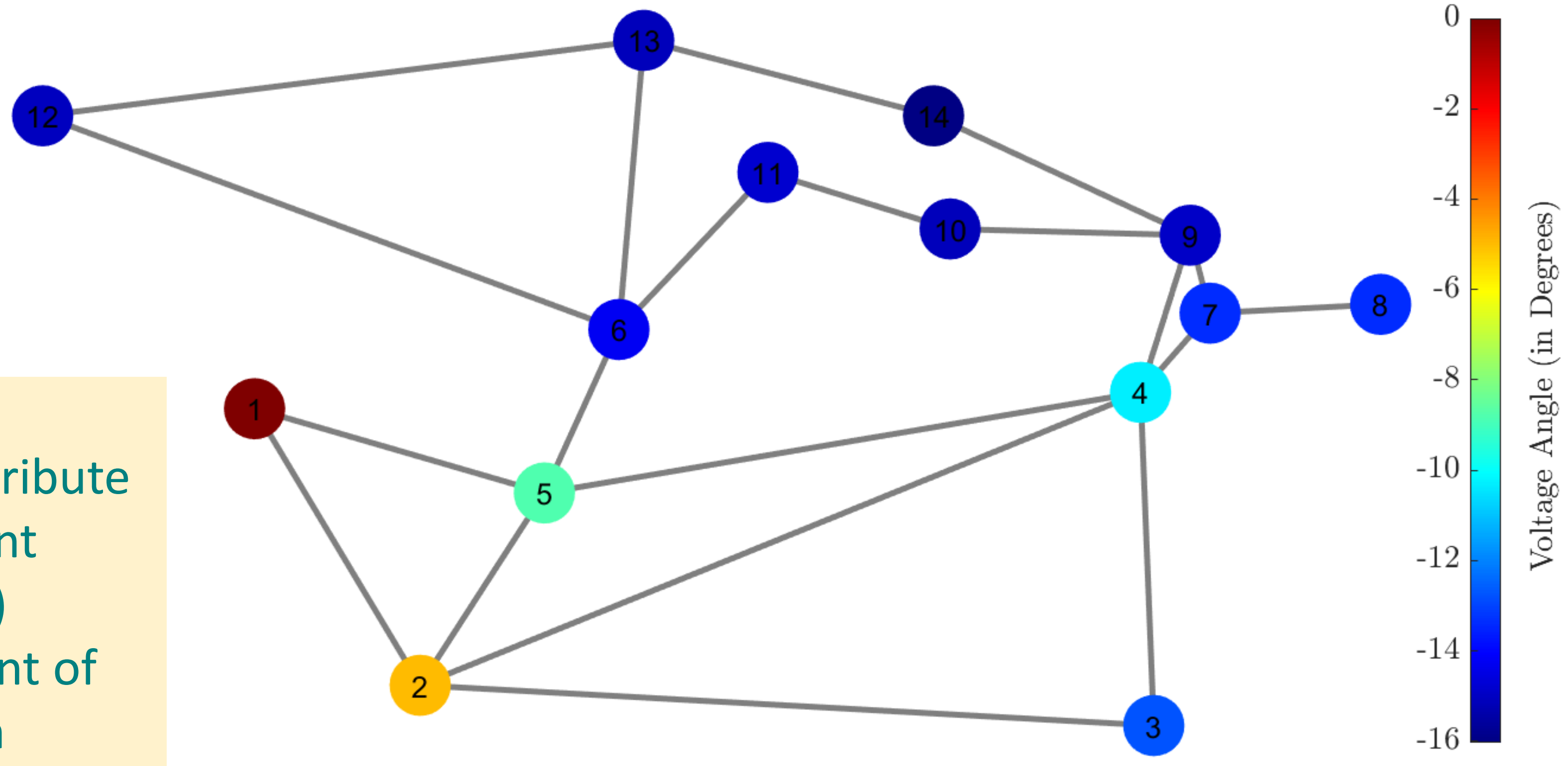
Different Graph definitions:

- Weights: $w_{ij} = \frac{1}{\text{Geographical Distance between bus } i \text{ and } j}$.
- Considering the bus admittance matrix, Y as the graph Laplacian.
- Many other options.

Example: Smart grid graph signals

- Measurement of any electrical attribute (Voltage magnitude, phase, Current magnitude, phase, frequency etc.) associated with a bus at one instant of time can be considered as a graph signal.

- Measurement can be obtained from PMU measurements or state estimation.



The voltage angle Graph Signal on IEEE 14 bus system

Graph Fourier Transform (GFT)

The Graph Spectral Domain – Graph Fourier Transform (GFT)

Classical Fourier Transform:

$$X(\omega) = \int_{t=-\infty}^{\infty} x(t)(e^{j\omega t})^* dt$$

Basis

Graph Fourier Transform:

$$X(\lambda_k) = \sum_{n=1}^{N} x(n)\, v_k{}^*(n)$$

In Matrix Form: $\boldsymbol{X} = \boldsymbol{V}^{-1}\boldsymbol{x}$

Laplacian Matrix of a Graph:

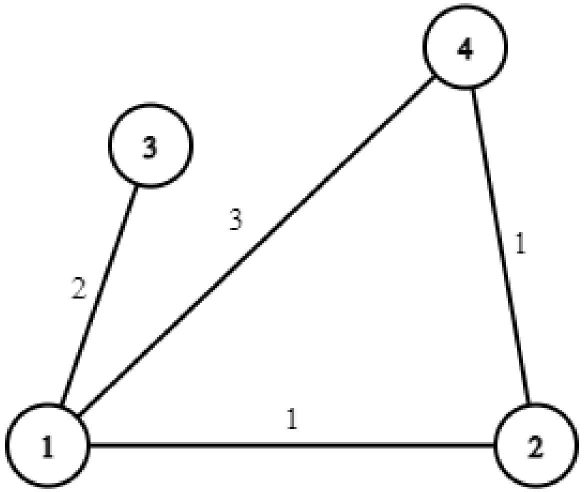
$$L_{ij} = \begin{cases} \sum_{j=1}^{N} w_{ij}\,, & i = j \\ -w_{ij}, & i \neq j \end{cases}$$

$$\boldsymbol{L v_k = \lambda_k v_k}$$

Eigenvectors of $\boldsymbol{L}$

Eigenvalues of $\boldsymbol{L}$

Example: Graph Fourier Transform (GFT)



$$\boldsymbol{L} = \begin{bmatrix} 6 & -1 & -2 & -3 \\ -1 & 2 & 0 & -1 \\ -2 & 0 & 2 & 0 \\ -3 & -1 & 0 & 4 \end{bmatrix} \quad \boldsymbol{\Lambda} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix}$$

$$\boldsymbol{V} = \begin{bmatrix} 0.5 & 0 & 0 & -0.866 \\ 0.5 & -0.4082 & -0.7071 & 0.2887 \\ 0.5 & 0.8165 & 0 & 0.2887 \\ 0.5 & -0.4082 & 0.7071 & 0.2887 \end{bmatrix}$$

Basis Functions (vectors) of GFT

Eigenvalues:

$$\lambda_1 = 0, \lambda_2 = 1, \lambda_3 = 3, \lambda_4 = 4$$

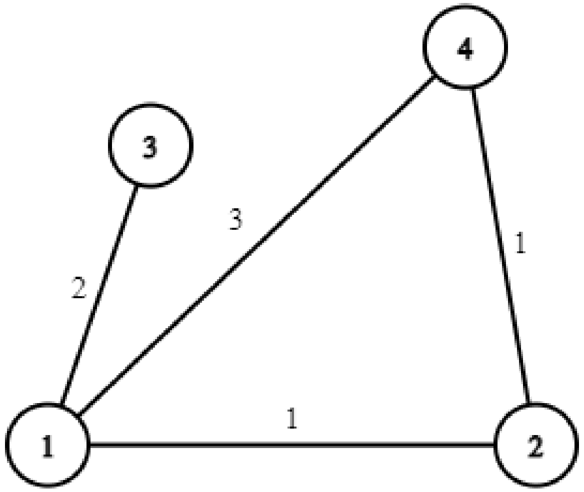
Graph Frequencies:

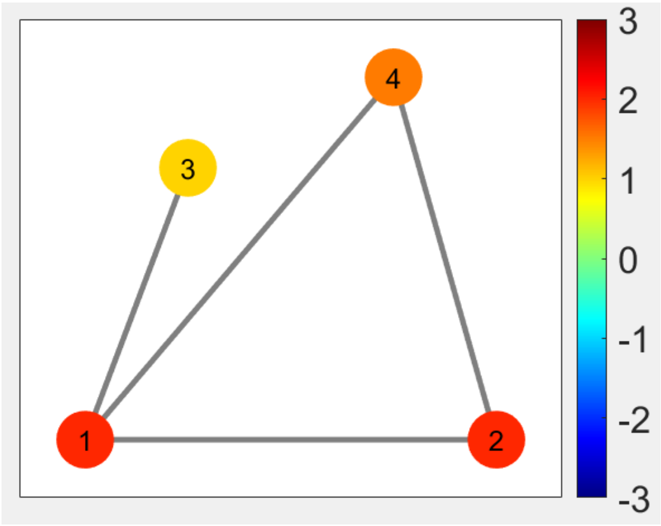
$$\lambda_k: \{0, 1, 3, 4\}$$

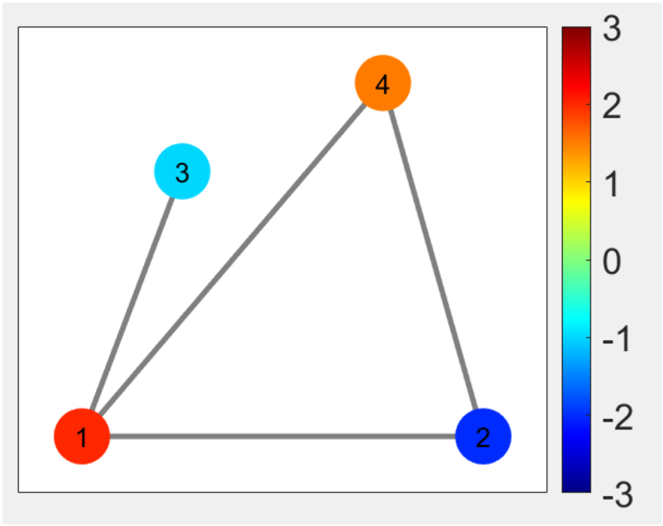
Normalized Graph Frequencies:

$$\hat{\lambda}_k: \{0, 0.25, 0.75, 1\}$$

Example: Graph Fourier Transform (GFT)







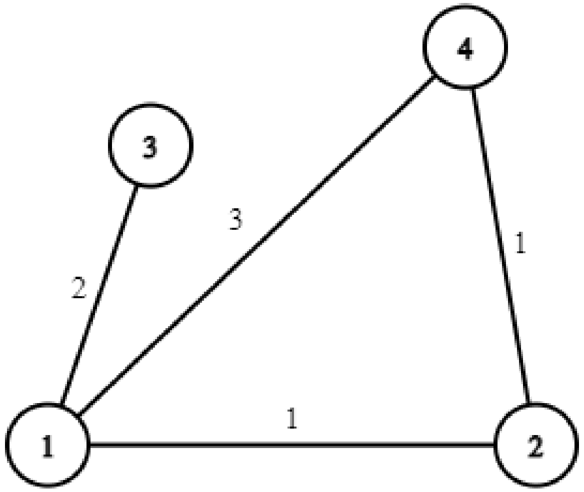
$$x(n) = \{2, 2, 1, 1.5\}$$

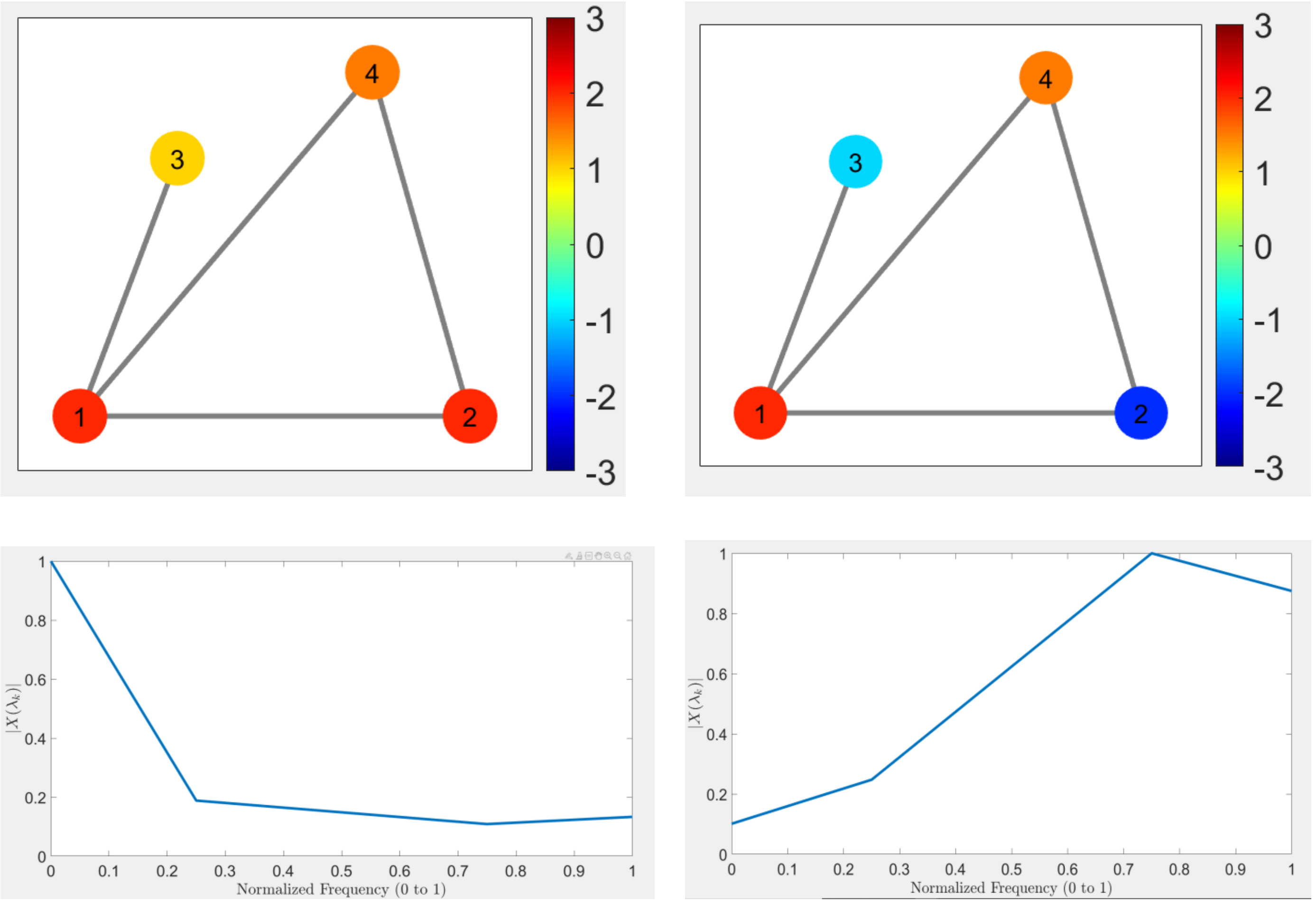
$$X(\lambda_k) = \boldsymbol{V}^{-1} \begin{bmatrix} 2 \\ 2 \\ 1 \\ 1.5 \end{bmatrix} = \begin{bmatrix} 3.2500 \\ -0.6124 \\ -0.3536 \\ -0.4330 \end{bmatrix}$$

$$\mathrm{w}(n) = \{2, -2, -1, 1.5\}$$

$$W(\lambda_k) = \boldsymbol{V}^{-1} \begin{bmatrix} 2 \\ -2 \\ -1 \\ 1.5 \end{bmatrix} = \begin{bmatrix} 0.2500 \\ -0.6124 \\ 2.4749 \\ -2.1651 \end{bmatrix}$$

Example: Graph Fourier Transform (GFT)





Smooth Signals are consisted of low graph-frequency components. Rapid changes correspond to high-graph frequency components.

Graph Filters

- Graph filters are functions of graph-frequency components, λ_k.
- A low/high pass-filter *emphasizes* the low/high-frequency components of a graph signal and *de-emphasizes* the high/low-frequency components.

Ideal Low-Pass Graph Filter:

$$H_L(\lambda_k) = \begin{cases} 0, & \lambda_k \leq \lambda_{cut-off} \\ 1, & \lambda_k > \lambda_{cut-off} \end{cases}$$

$$X(\lambda_k) = \begin{bmatrix} 3.2500 \\ -0.6124 \\ -0.3536 \\ -0.4330 \end{bmatrix}, \ H_L(\lambda_k) = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

Low-Pass Graph Filtering:

$$Y(\lambda_k) = X(\lambda_k)\, H_L(\lambda_k) = \begin{bmatrix} 3.2500 \\ -0.6124 \\ 0 \\ 0 \end{bmatrix}$$

Ideal High-Pass Graph Filter:

$$H_H(\lambda_k) = \begin{cases} 1, & \lambda_k \leq \lambda_{cut-off} \\ 0, & \lambda_k > \lambda_{cut-off} \end{cases}$$

$$X(\lambda_k) = \begin{bmatrix} 3.2500 \\ -0.6124 \\ -0.3536 \\ -0.4330 \end{bmatrix}, \ H_H(\lambda_k) = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

High-Pass Graph Filtering:

$$Y(\lambda_k) = X(\lambda_k)\, H_L(\lambda_k) = \begin{bmatrix} 0 \\ 0 \\ -0.3536 \\ -0.4330 \end{bmatrix}$$

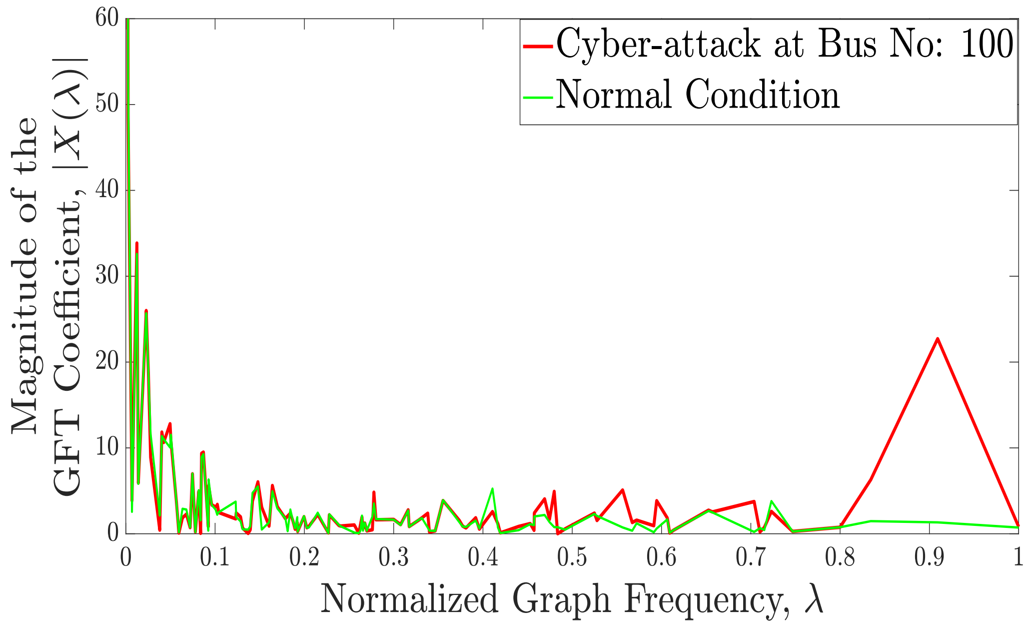
Application of GSP in Power System

Application of GSP in Power System

Problem: Detecting Anomalous or False Data in Power Grid [8, 9]

- Graph Signal: Bus Voltage Angles in the grid.
- Assumption: Bus voltage angles graph signals are smooth in normal condition → Mostly lie in the low-frequency region → Absence of high graph-frequency component.
- Extract high graph-frequency components (beyond $\lambda_{cut-off}$) of the graph signal by a graph filter, $H_H(\lambda_k)$.
- Calculate amount of High graph-frequency component:

$$\gamma(t) = \sum_{k=1}^{N} |X(\lambda_k)\, H_H(\lambda_k)|$$



$\gamma(t) < Threshold \rightarrow Attack\ Declared.$

References

1. A. Ortega, P. Frossard, J. Kovaevi, J. M. F. Moura and P. Vandergheynst, ”Graph Signal Processing: Overview, Challenges, and Applications,” in Proceedings of the IEEE, vol. 106, no. 5, pp. 808-828, May 2018.

2. D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega and P. Vandergheynst, ”The emerging field of signal processing on graphs: Extending highdimensional data analysis to networks and other irregular domains,” in IEEE Signal Processing Magazine, vol. 30, no. 3, pp. 83-98, May 2013.

3. D. I. Shuman, B. Ricaud and P. Vandergheynst, ”Vertex-frequency analysis on graphs,” Applied and Computational Harmonic Analysis 40, no. 2 : 260-291, 2016.

4. L. Stankovic, M. Dakovic and E. Sejdic, ”Vertex-Frequency Analysis: A Way to Localize Graph Spectral Components [Lecture Notes],” in IEEE Signal Processing Magazine, vol. 34, no. 4, pp. 176-182, July 2017.

5. L. T. Thanh, N. Linh-Trung, N. V. Dung and K. Abed-Meraim, ”A new windowed graph Fourier transform,” 4th NAFOSTED Conference on Information and Computer Science, Hanoi, 2017, pp. 150-155, 2017.

References

6. D. K. Hammond, P. Vandergheynst and R. Gribonval, ”Wavelets on graphs via spectral graph theory” Applied and Computational Harmonic Analysis, 30(2), 129-150, 2011.

7. L. Stankovic, M. Dakovic, and T. Thayaparan, ”Time-frequency signal analysis with applications,” Artech house, 2014. [15] L. Stankovi, E. Sejdi and M. Dakovi, ”Vertex-Frequency Energy Distributions,” in IEEE Signal Processing Letters, vol. 25, no. 3, pp. 358-362, March 2018.

8. E. Drayer and T. Routtenberg, "Detection of False Data Injection Attacks in Smart Grids Based on Graph Signal Processing," in *IEEE Systems Journal*, vol. 14, no. 2, pp. 1886-1896, June 2020

9. R. Ramakrishna and A. Scaglione, "Detection of False Data Injection Attack Using Graph Signal Processing for the Power Grid," *2019 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2019, pp. 1-5.

10. R. Ramakrishna and A. Scaglione, "Grid-Graph Signal Processing (Grid-GSP): A Graph Signal Processing Framework for the Power Grid," in *IEEE Transactions on Signal Processing*, vol. 69, pp. 2725-2739, 2021

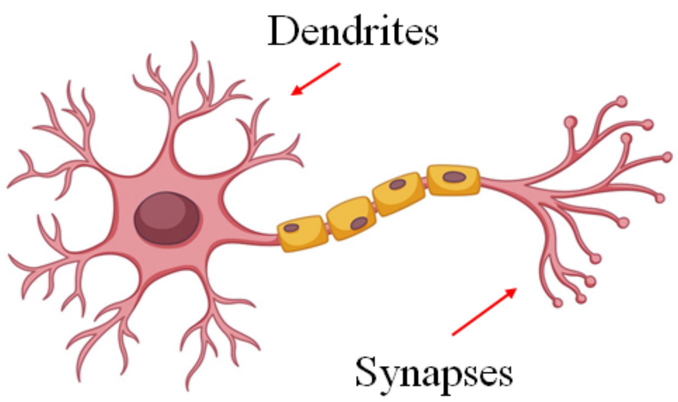
Deep Learning on Graphs: Examples from Smart Grid

Md Jakir Hossain (Ph.D. Student)
Supervised by
Dr. Mahshid Rahnamay Naeini
Electrical Engineering, University of South Florida
mdjakir@usf.edu

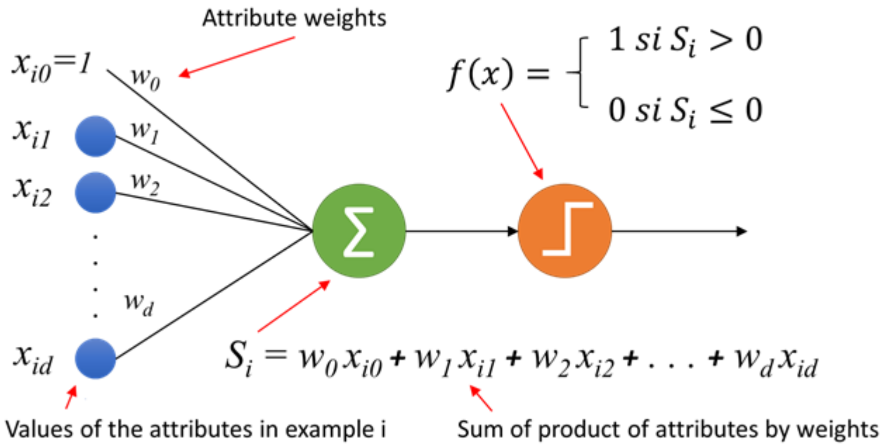
Concepts will be discussed….

- General overview of **Deep Learning**
- Convolution and **Convolutional Neural Networks**
- Why do we need separate deep learning approach for graph?
- What can we do with **deep learning on graphs**?
- **CNN vs GCNN**: similarity, differences, and analogy.
- What are the implications of **graph convolution**?
- A **case study** from cyber-physical **smart grids**
- Recent developments……..

Artificial Neural Network (ANN) on Regular Domain



NEURON



PERCEPTRON

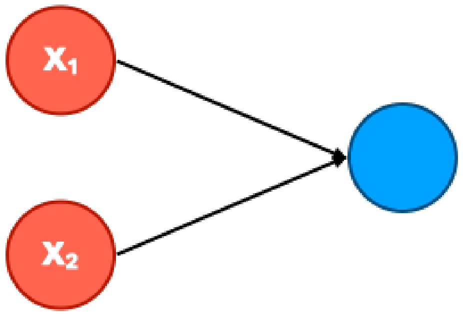
x_i= Attribute/feature
w_i= Weight of feature x_i
$f(x)$ = Activation function

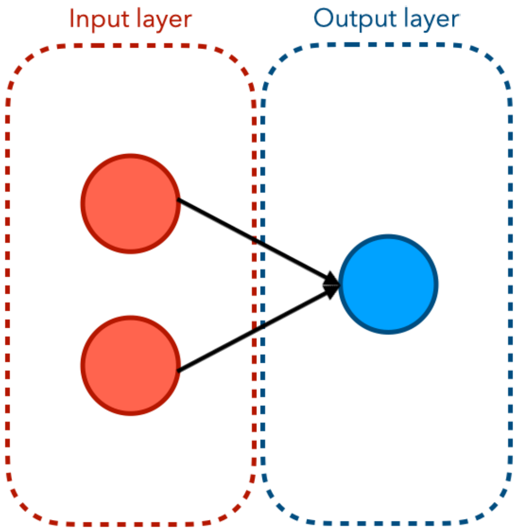
A single perceptron and its analogy to a biological neuron….

- Perceptron receives the value of the attributes of an example, just as dendrites do in a neuron.
- Each attribute has a weight that measures its contribution to the result, which is the sum of the multiplications of the value of each attribute by its corresponding weight. If the sum is greater than zero Perceptron returns a value of 1, otherwise it yields 0.

Artificial Neural Network (ANN) on Regular Domain

Going deeper into ANN…….

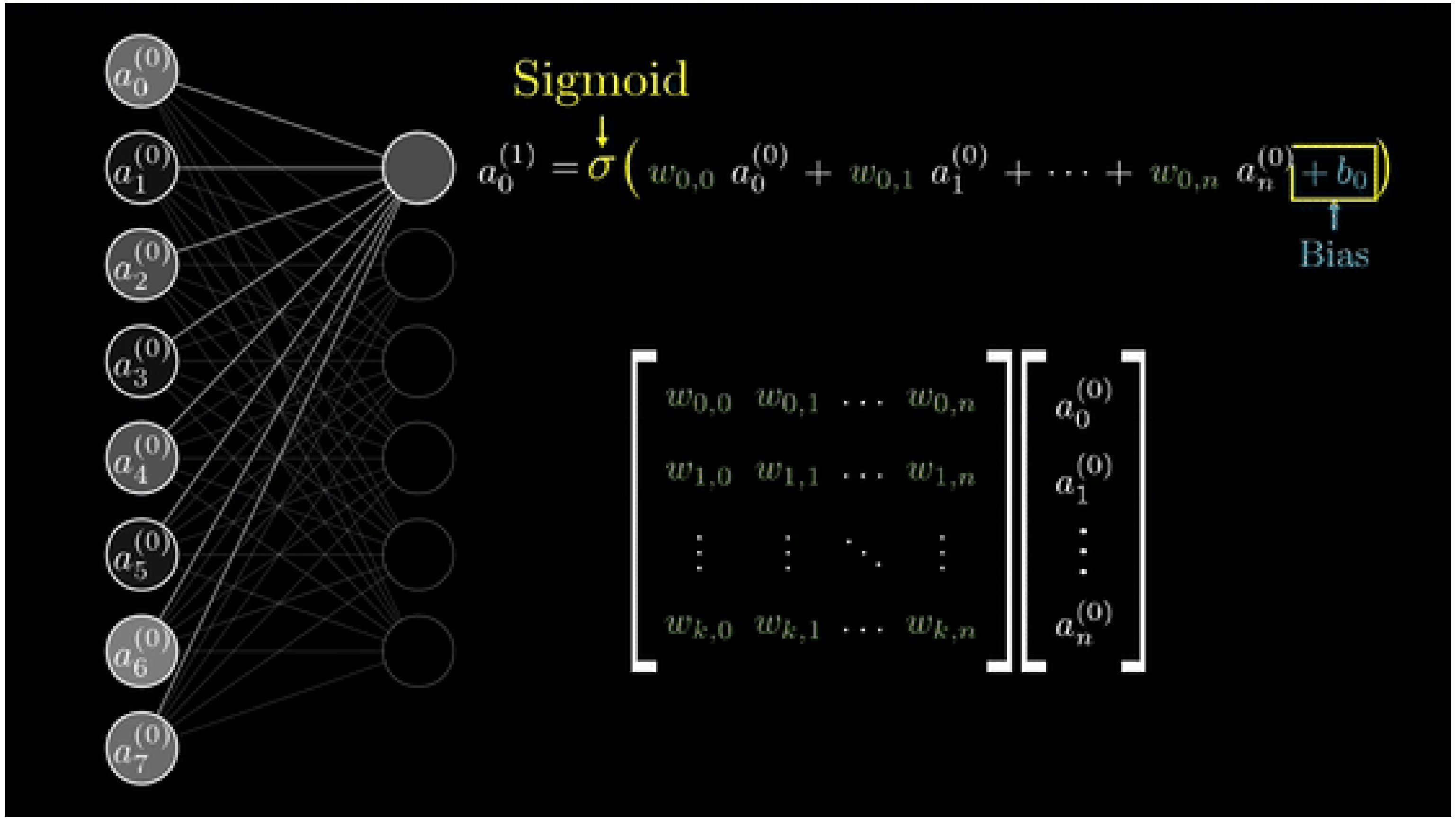




Stack several layers of perceptron, now we have deep neural network (DNN)

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

Artificial Neural Network (ANN) on Regular Domain

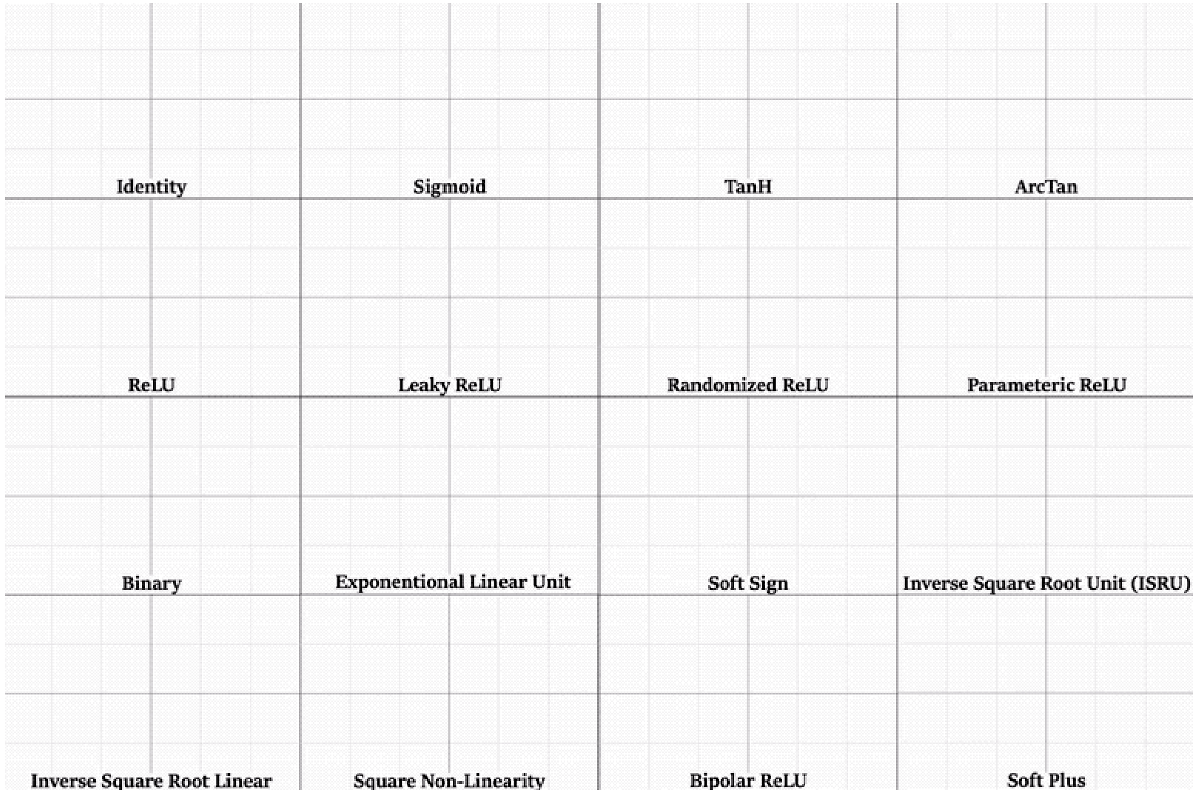


As we can see, without activation functions neural network is just a linear combination of weights and features comparable to linear regression.

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

Artificial Neural Network (ANN) on Regular Domain

Various Activation Functions



Wikipedia has a nice article on Activation Functions

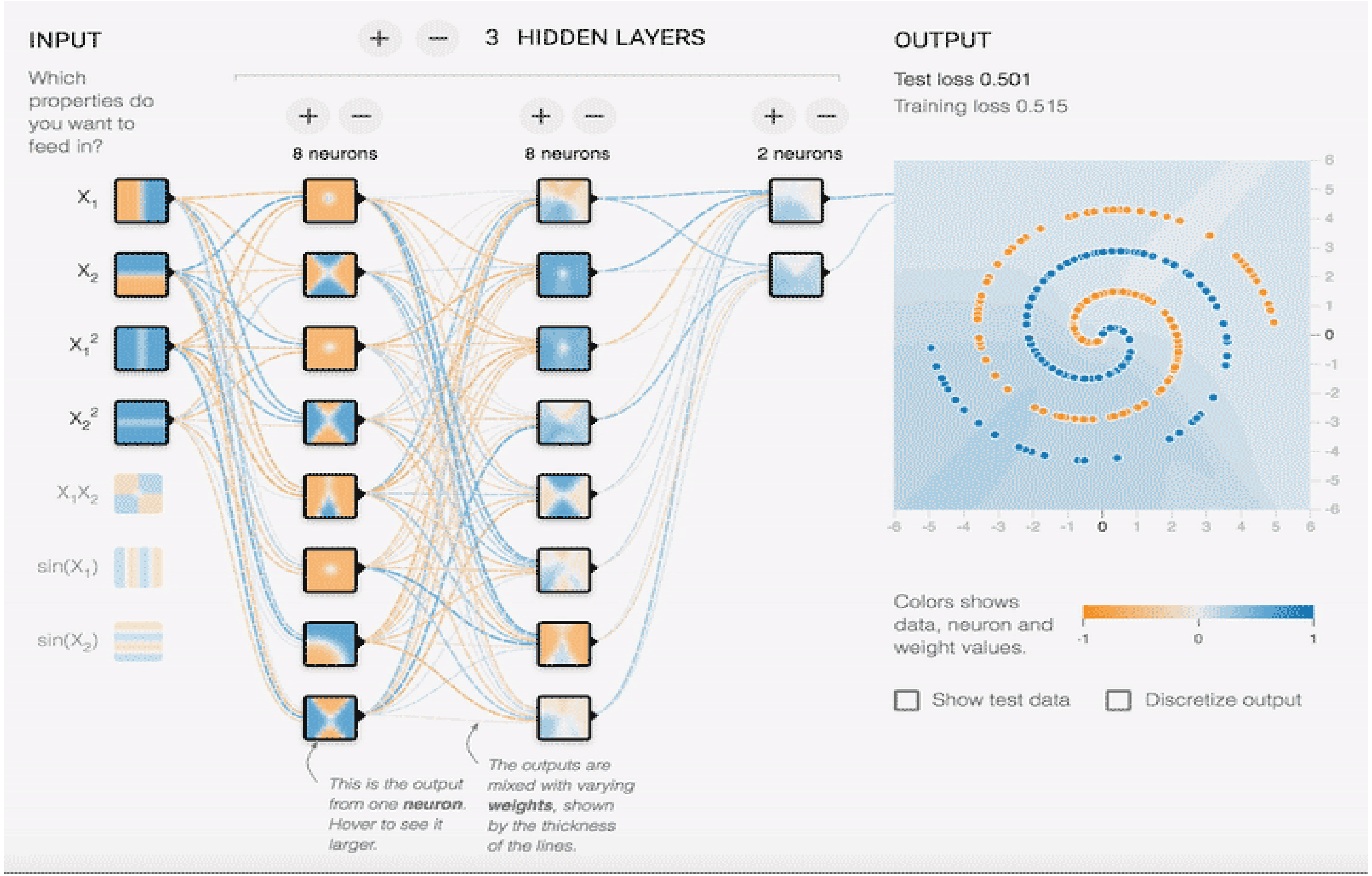
- Activation functions are mathematical equations that determine the output of a neural network model.
- Activation function also helps to normalize the output of any input in the range between 1 to -1 or 0 to 1.
- Activation functions introduce nonlinearity and come in all shape and sizes.

Properties of activation functions

- Non-Linearity
- Continuously differentiable
- Range
- Monotonic
- Approximates identity near the origin

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

Artificial Neural Network (ANN) on Regular Domain



Tensorflow Playground:

- A fun tool to learn how neural network works.
- Thickness of line means importance of the weights.

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

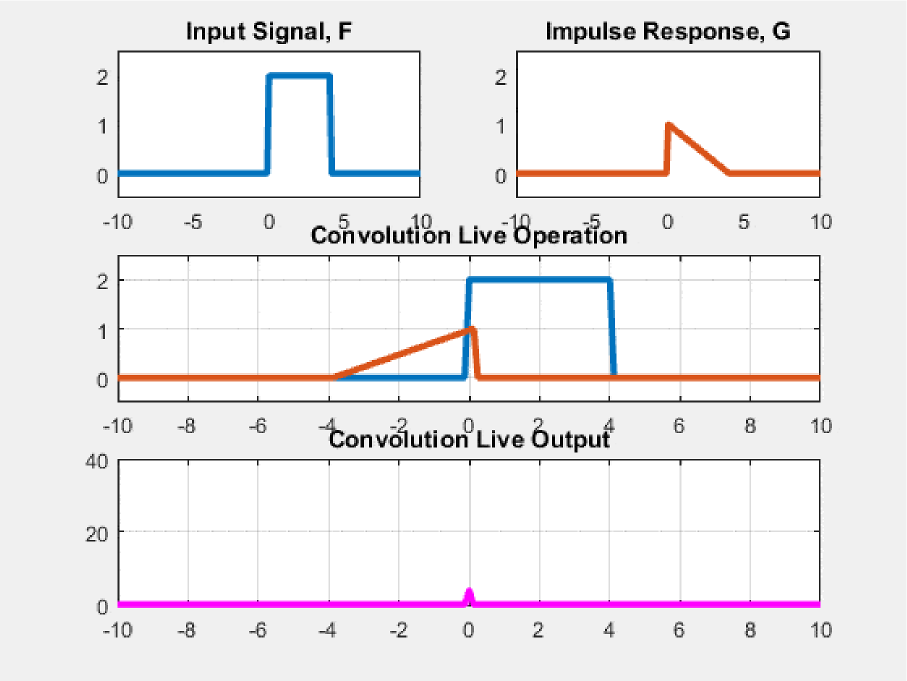
Artificial Neural Network (ANN) on Regular Domain

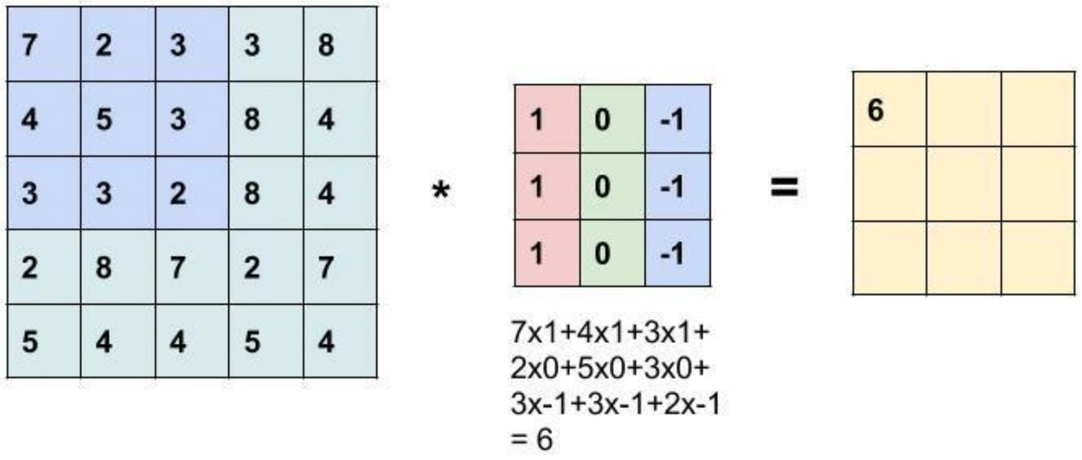
Architecture of neural deep neural networks varies depending on the type of applications. Here are some Popular deep learning techniques:

- **Convolutional Neural Networks (CNNs)**
- Long Short-Term Memory Networks (LSTMs)
- Recurrent Neural Networks (RNNs)
- Generative Adversarial Networks (GANs)
- Radial Basis Function Networks (RBFNs)
- Multilayer Perceptron's (MLPs)
- Self Organizing Maps (SOMs)
- Deep Belief Networks (DBNs)
- Restricted Boltzmann Machines(RBMs)

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

What is Convolution?





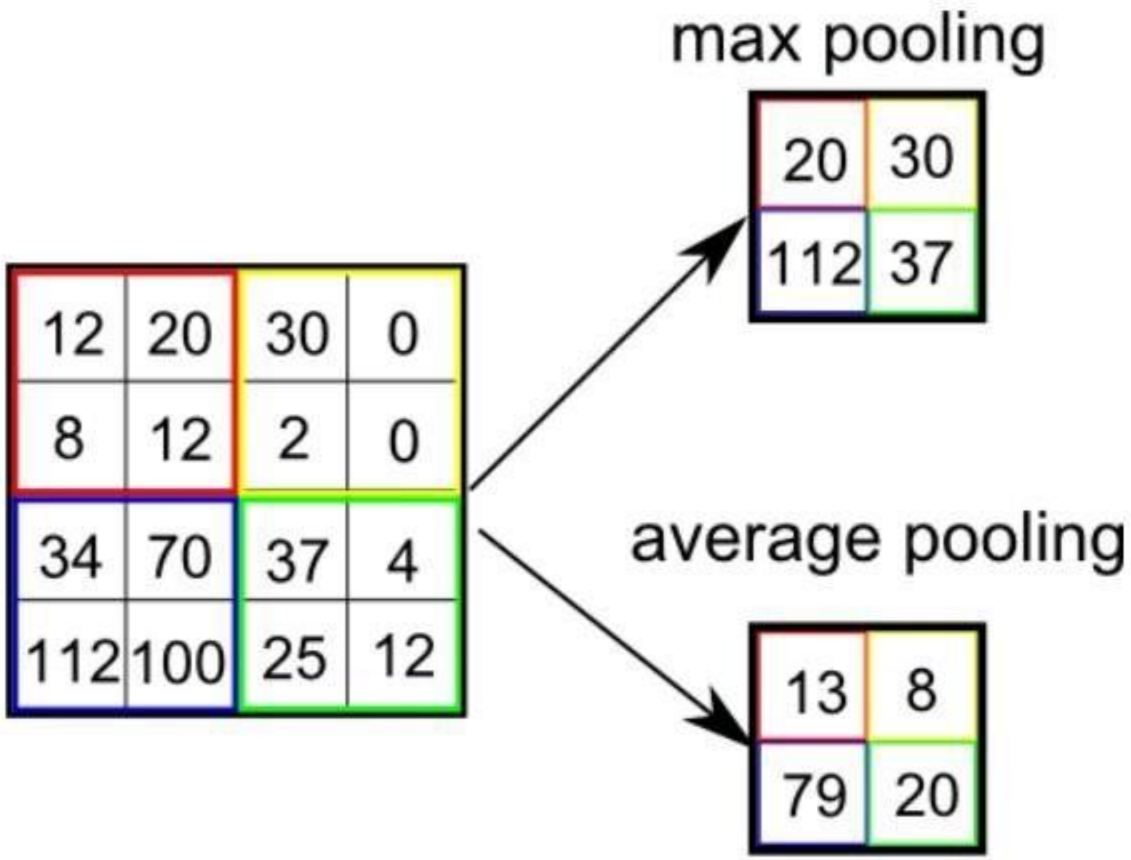
Convolution: convolution is a mathematical operation defined by

$$(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau) g(t-\tau)$$

Convolution operation in two-dimensional data (e.g., Image). The base of convolutional neural network (CNN).

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

What is Pooling?



- Pooling layers provide an approach to down sampling feature maps by summarizing the presence of features in patches of the feature map.

- Two common pooling methods are average pooling and max pooling that summarize the average presence of a feature and the most activated presence of a feature respectively.

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

How Do Convolutional Neural Network (CNN) Work?

CNN's, also known as ConvNets, consist of multiple layers and are mainly used for image processing and object detection. CNN's multiple layers process and extract features from data:

Convolution Layer

- CNN has a convolution layer that has several filters to perform the convolution operation.

Rectified Linear Unit (ReLU)

- CNN's have a ReLU layer to perform operations on elements. The output is a rectified feature map.

Pooling Layer

- The rectified feature map next feeds into a pooling layer. Pooling is a down-sampling operation that reduces the dimensions of the feature map.
- The pooling layer then converts the resulting two-dimensional arrays from the pooled feature map into a single, long, continuous, linear vector by flattening it.

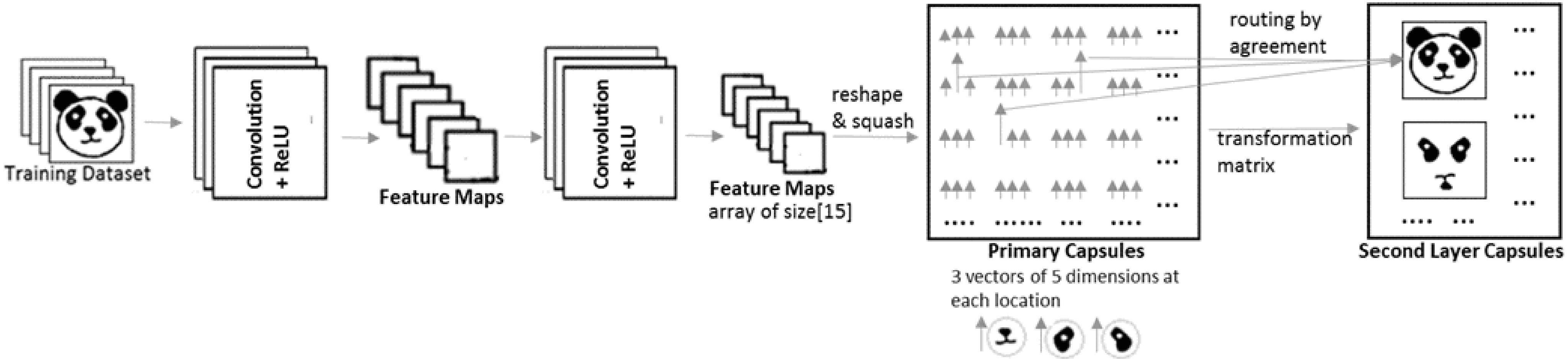
Fully Connected Layer

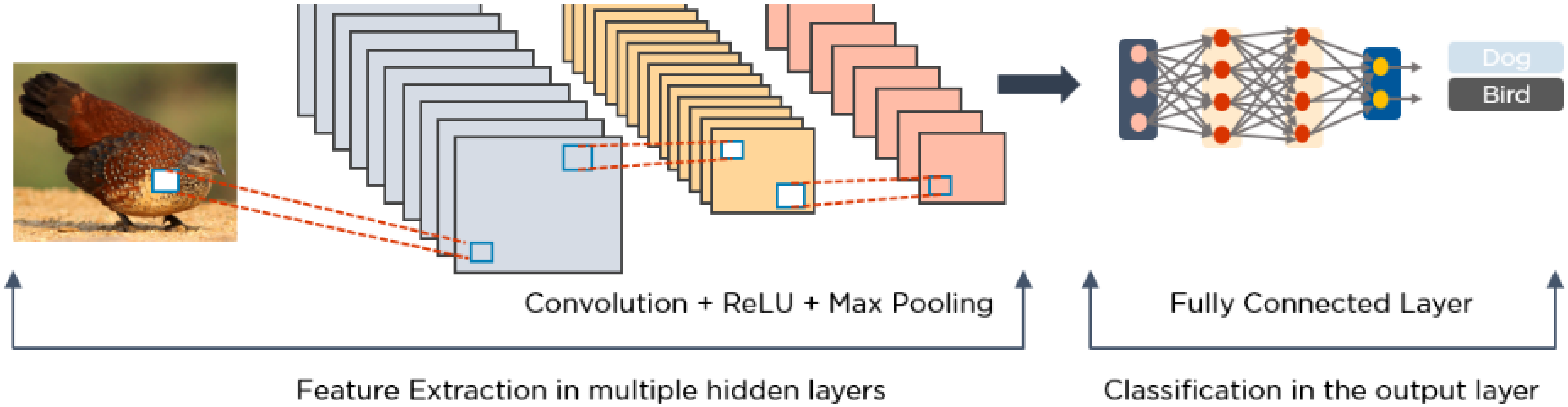
- A fully connected layer forms when the flattened matrix from the pooling layer is fed as an input, which classifies and identifies the images.

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

Convolutional Neural Network (CNN) on Regular Domain

Basic steps of how Convolutional neural network learns the data



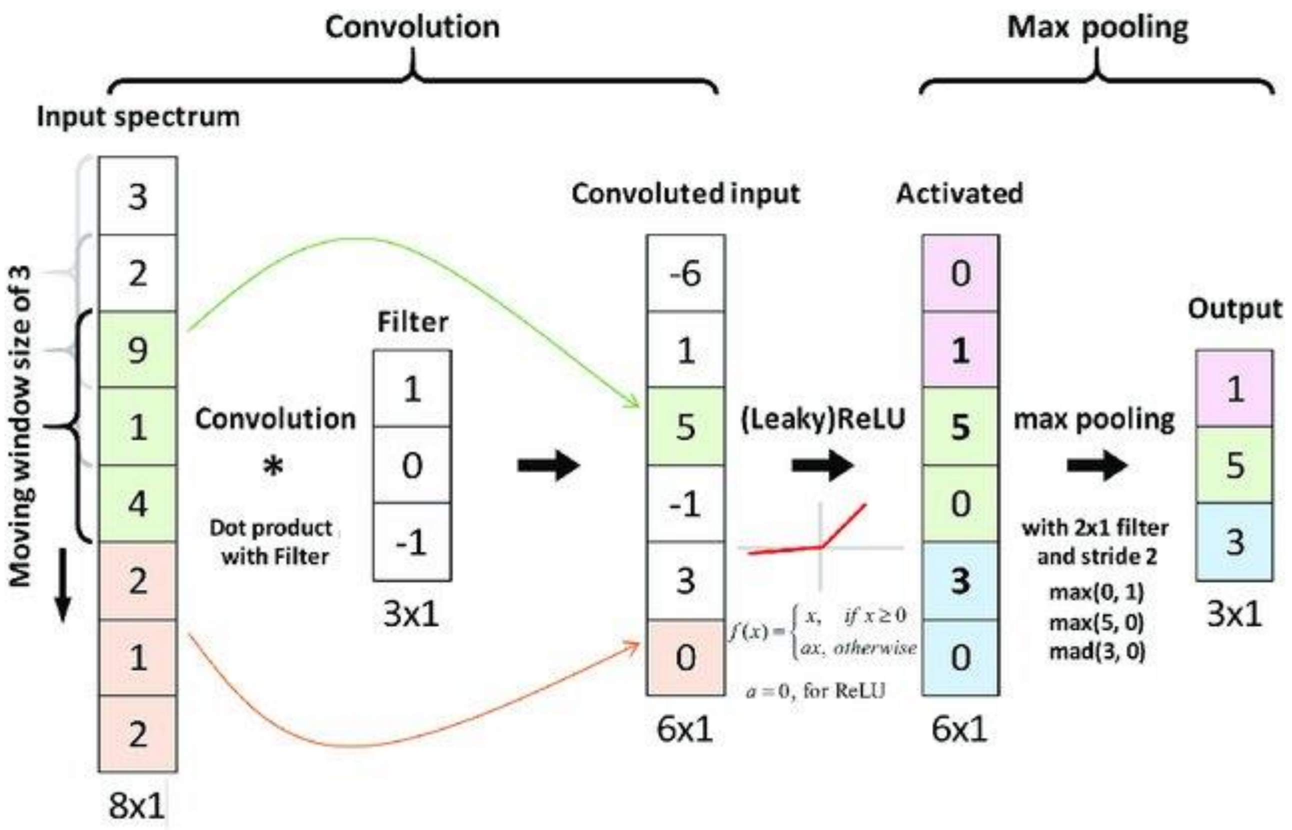


CNN's are widely used to identify satellite images, process medical images, forecast time series, and detect anomalies.

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

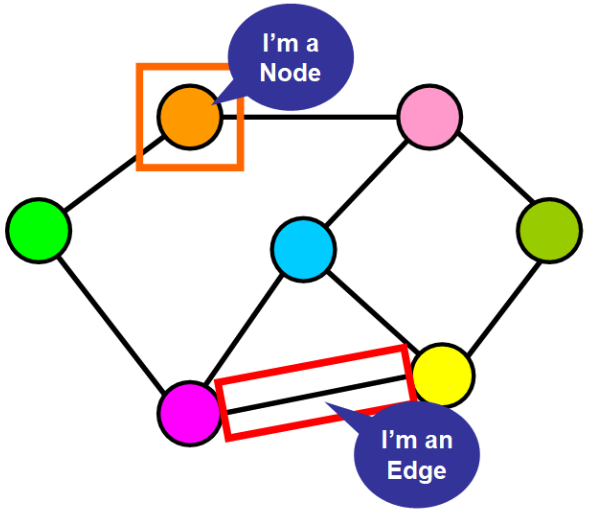
Convolutional Neural Network (CNN) on Regular Domain

In case if anyone wondering how all this done numerically? Here is the process on one-dimensional sequential data……

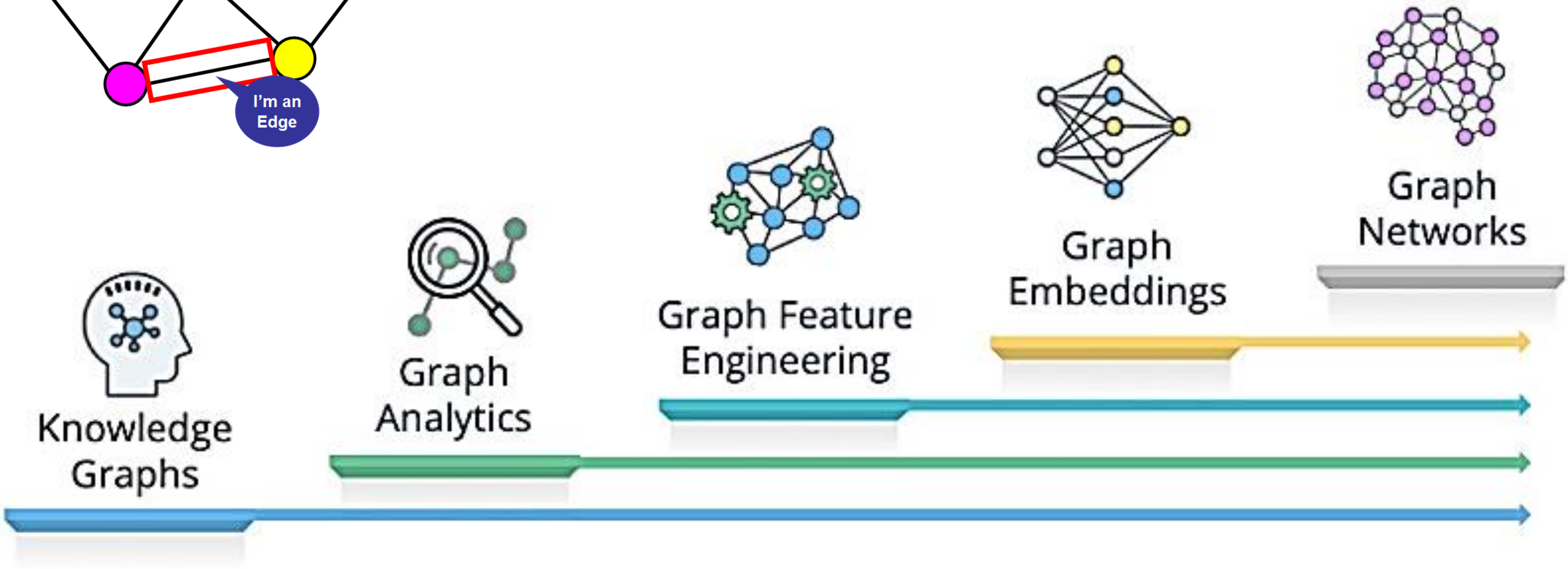


A schematic diagram showing the numeric operation in various layers of CNN for one-dimensional data.

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

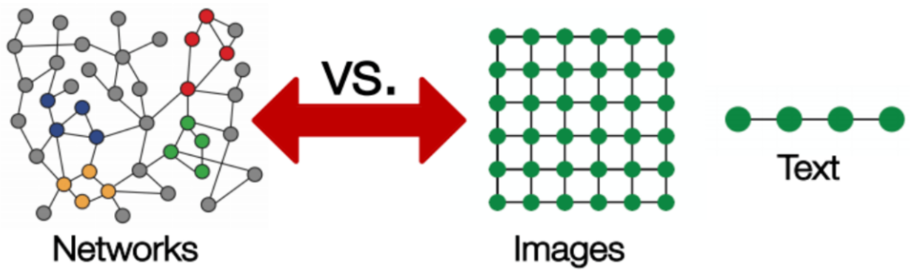


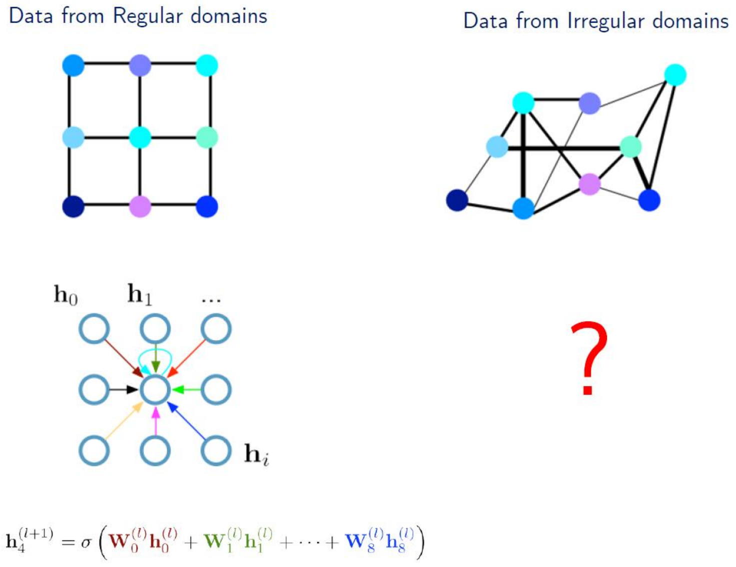
Deep Learning on Graphs



Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

Why do we need deep learning on Graphs, separately?





Graphs are far more complex! The reason is that conventional Machine Learning and Deep Learning tools are specialized in Euclidian data type.

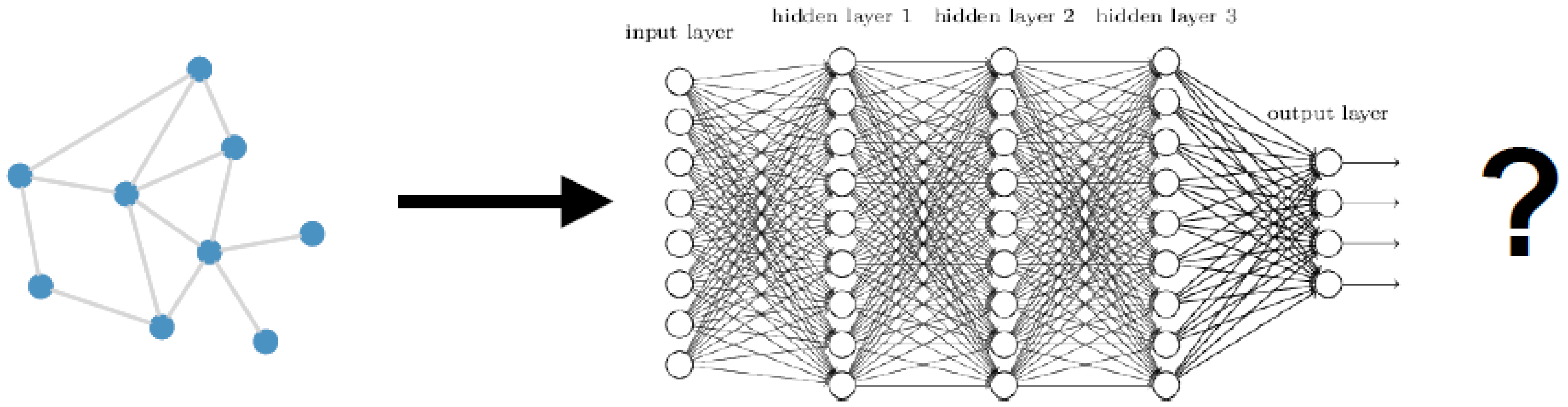
- Arbitrary size and complex topological structure (i.e., no spatial locality like grids)
- No fixed node ordering or reference point
- Often dynamic and have multimodal features

Challenges:

- How to extend Convolution Nets to graph structured data?
- Assumption: Non-Euclidean data are locally stationary and manifest hierarchical structures.
- How to define compositionality on graphs? (convolution and pooling on graphs)
- How to make them fast? (linear complexity)

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

How do we do deep learning on graphs?



Approach-I : A naïve approach

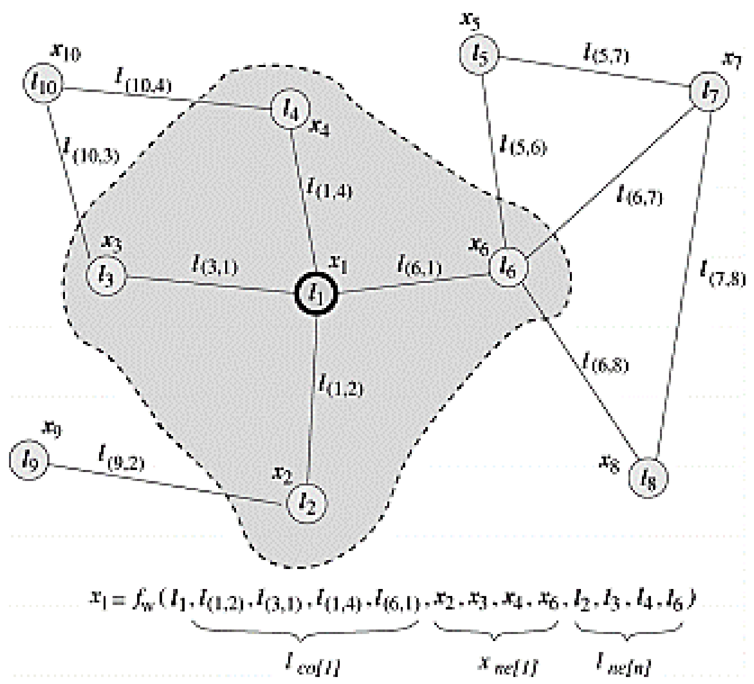
- Take adjacency matrix A and feature matrix $\mathrm{X} \epsilon \mathbb{R}^{N \times E}$
- Concatenate them $X_{in} = [X, A] \in \mathbb{R}^{N \times (N+E)}$
- Feed them into DNN (fully connected)

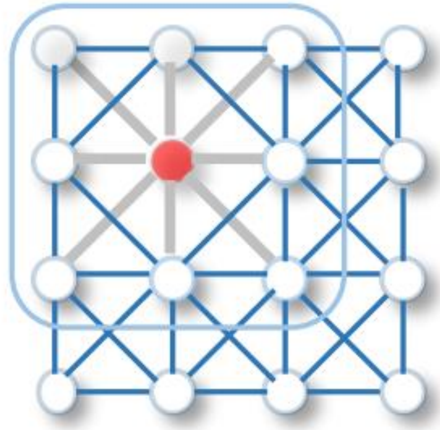
Problems:

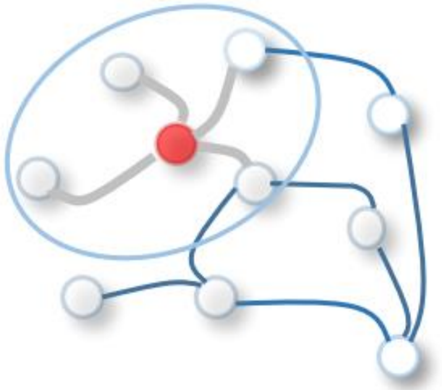
- Huge number of parameters to train $\mathrm{O}(N)$
- Needs to be re-trained if number of nodes changes
- Does not generalize across graphs

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

Graph Neural Network (GNN)?







Approach-II : graph embedding, message passing based on adjacency of nodes.

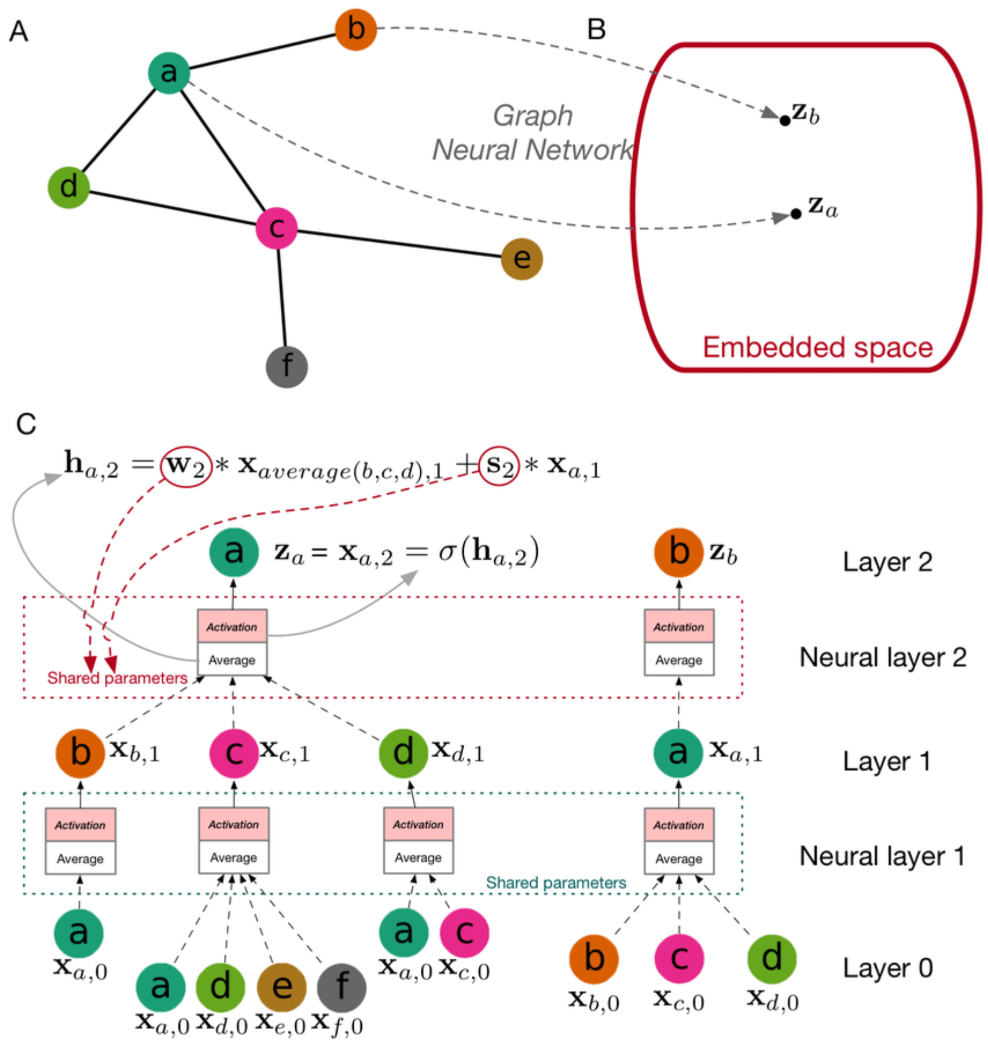
- Neighborhood aggregation
- In the Convolution layer, we use the size of the convolution kernel to indicate the size of the neighborhood (how many pixels will contribute to the resulting value).
- Similarly, the Graph Convolution layer uses neighbors of a particular graph node to define a convolutional operation in it.

Problems:

- First, it is computationally inefficient to update the hidden states of nodes iteratively to get the fixed point.
- Second, vanilla GNN uses the same parameters in the iteration while most popular neural networks use different parameters in different layers, which serves as a hierarchical feature extraction method.

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

How Neighborhood Aggregation Works?



$$h_v^0 = x_v$$

Embedding of the 0-th layer is equal to the features

Non-linear activation function (e.g. RELU)

Weights of neighboring nodes

Embedding of node v in previous layer

$$h_v^k = \sigma\left(W_k \sum_{u \in N(v)} \frac{h_v^{k-1}}{deg(v)} + B_v h_v^{k-1} \right)$$

Embedding of node v in the k-th layer

Average over embeddings of neighboring nodes

Weights of node v

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

Graph Convolutional Network Spatial Method

Graph Convolutional Network (GCN): Graph convolutional networks are slight variation on the neighborhood aggregation idea [3]. Empirically, they found this configuration to give the best results.

- More parameter sharing.
- Down-weights high degree neighbors.
- Can be efficiently implemented using sparse batch operations:

$$H^{(k+1)} = \sigma(D^{-\frac{1}{2}}\tilde{A}D^{-\frac{1}{2}}H^{(k)}W_k)$$
$$\tilde{A} = A + I$$

- $O(|E|)$ time complexity overall.

Normal neighbourhood aggregation

$$h_v^k = \sigma\left(W_k \sum_{u\in N(v)} \frac{h_v^{k-1}}{deg(v)} + B_v h_v^{k-1} \right)$$

Different Normalization

Same weights

GCN neighbourhood aggregation

$$h_v^k = \sigma\left(W_k \sum_{u\in N(v)\cup v} \frac{h_v^{k-1}}{\sqrt{deg(v)+deg(u)}} \right)$$

[3]. Thomas N. Kipf, Max Welling, ''Semi-Supervised Classification with Graph Convolutional Networks", arXiv:1609.02907, Published as a conference paper at ICLR 2017.

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

Graph Convolutional Network Spectral Method

Spectral convolution: spectral convolution of $f * g \in L^2(v)$ can be defined by analogy

$$f * g = \sum_{k\geq 1} \langle f, \phi_k \rangle_{L^2(v)} \langle g, \phi_k \rangle_{L^2(v)} \phi_k$$

Product in the Fourier domain

Inverse Fourier transform

In matrix-vector notation

$$f * g = \underbrace{\Phi diag(\hat{g}_1, \dots \dots, \hat{g}_n)\Phi^T}_{G} f$$

Not shift-invariant! (G has no circular structure)
Filter co-efficients depends on basis $(\phi_1, \dots \dots, \phi_n)$

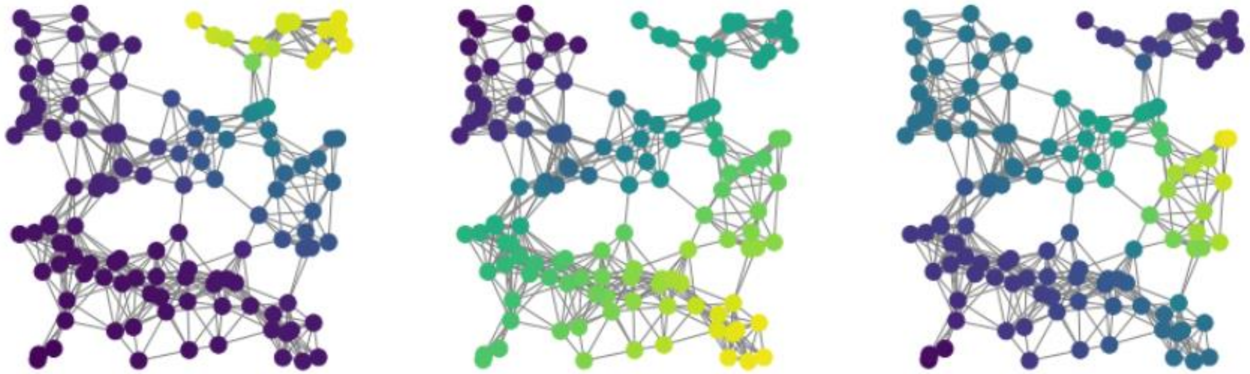


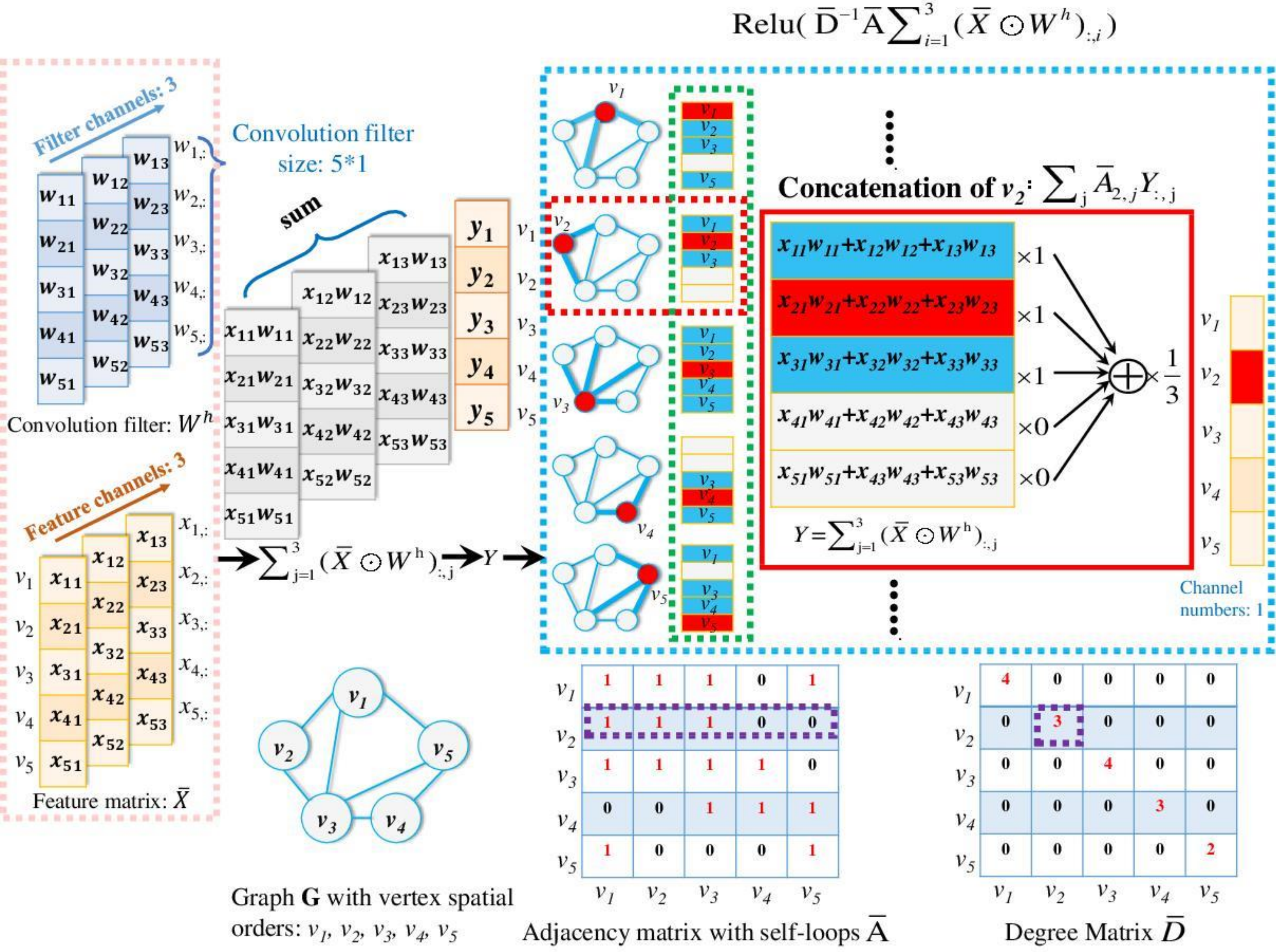
Fig: Graph Fourier Transform (GFT), some graph Fourier modes on a random sensor graph. From left to right: first non constant eigenvector (Fiedler vector)u_1, second and third eigenvectors (u_2 and u_3). Colormap is: positive values in yellow, negative ones in blue [4].

[4]. Benjamin Ricaud, Pierre Borgnat, Nicolas Tremblay, Paulo Gonçalves, Pierre Vandergheynst,"Fourier could be a data scientist: From graph Fourier transform to signal processing on graphs", Comptes Rendus Physique, Volume 20, Issue 5, 2019.

Convolution Operation on Graph?

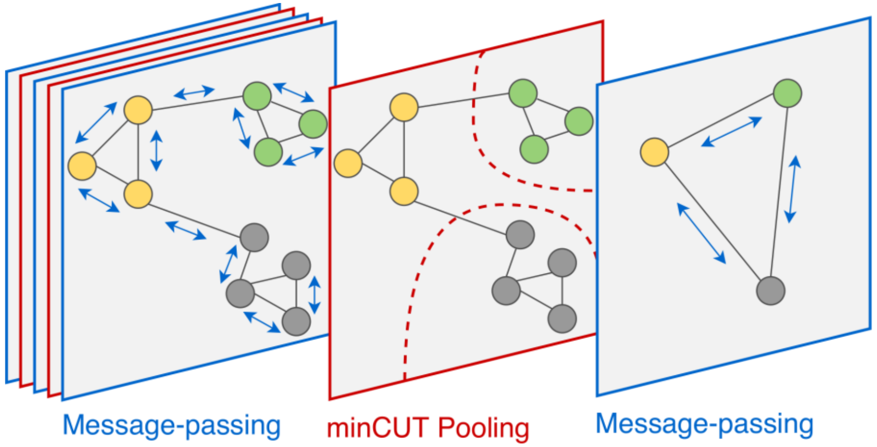
In case, some of you are wondering how convolution is done on graph data….

Internal mechanism of graph convolution



Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

How Pooling Works on GCN?



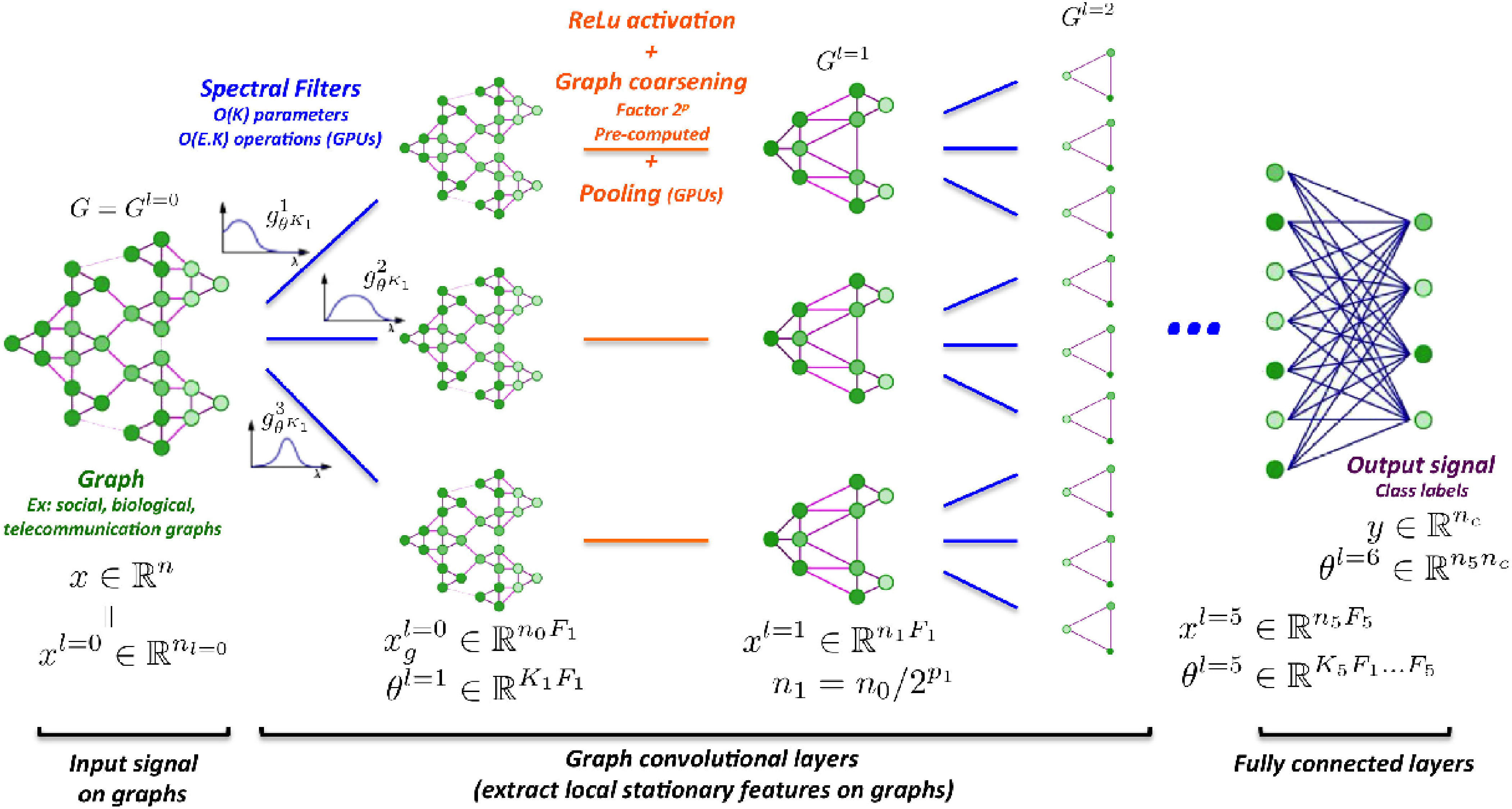
In graph theory, a minimum cut or min-cut of a graph is a cut (a partition of the vertices of a graph into two disjoint subsets) that is minimal in some metric.

-Wiki

Let $G(V, E)$ be a (not necessarily simple) undirected edge-weighted graph with nonnegative weights. A cut C of G is any nontrivial subset of V, and the weight of the cut is the sum of weights of edges crossing the cut. A **mincut** is then defined as **a cut of G of minimum weight.**

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

Graph Convolutional Network Spectral Method



Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

Applications of Graph Neural Networks

- Graph clustering
  - Community detection
  - Network Partitioning
- Topic detection
  - Citation Tracking
  - Plagiarism Detection
- Recommender systems
  - Content suggestions in online retails
  - Movie recommendation
- Graph-based classification
  - DNA sequence detection
  - Molecule identification
- Link prediction
  - Topology estimation
  - Failure detection
- Graph alignment
  - Object tracking

Case Study from Smart Grid

Fault Location in Power Distribution Systems via Deep Graph Convolutional Networks

Kunjin Chen, Jun Hu, *Member, IEEE*, Yu Zhang, *Member, IEEE*,
Zhanqing Yu, *Member, IEEE*, and Jinliang He, *Fellow, IEEE*

Problem Formulation:

- Distribution systems are constantly under the threat of short-circuit faults that would cause power outages.
- In order to enhance the operation quality and reliability of distribution systems, system operators have to deal with outages in a timely manner.
- Thus, it is of paramount importance to accurately locate and quickly clear faults immediately after the occurrence, so that quick restoration can be achieved.

Methodology:

- a GCN model is proposed for fault location in distribution systems.
- Unlike existing machine learning models used for fault location tasks, the architecture of the proposed model preserves the spatial correlations of the buses and learns to integrate information from multiple measurement units.
- Features are extracted and composited in a layer-by layer manner to facilitate the faulty bus classification task.

Case Study from Smart Grid

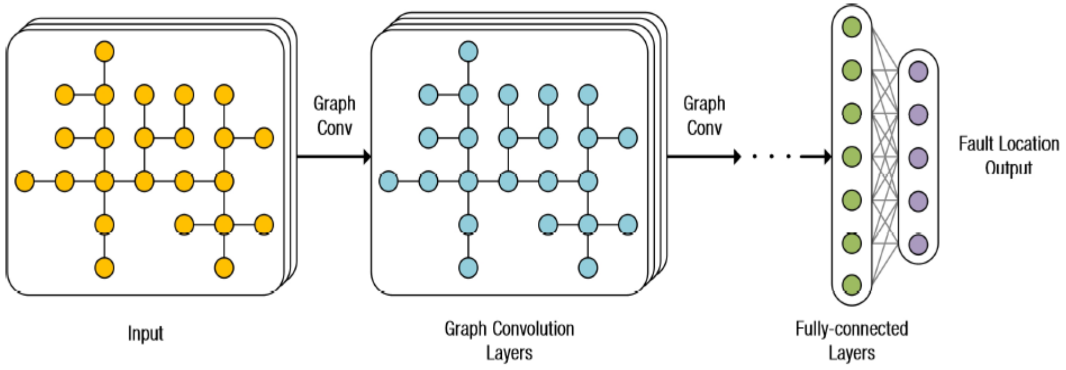


Fig. 1. The structure of the GCN model. Several graph convolution layers are followed by two fully-connected layers.

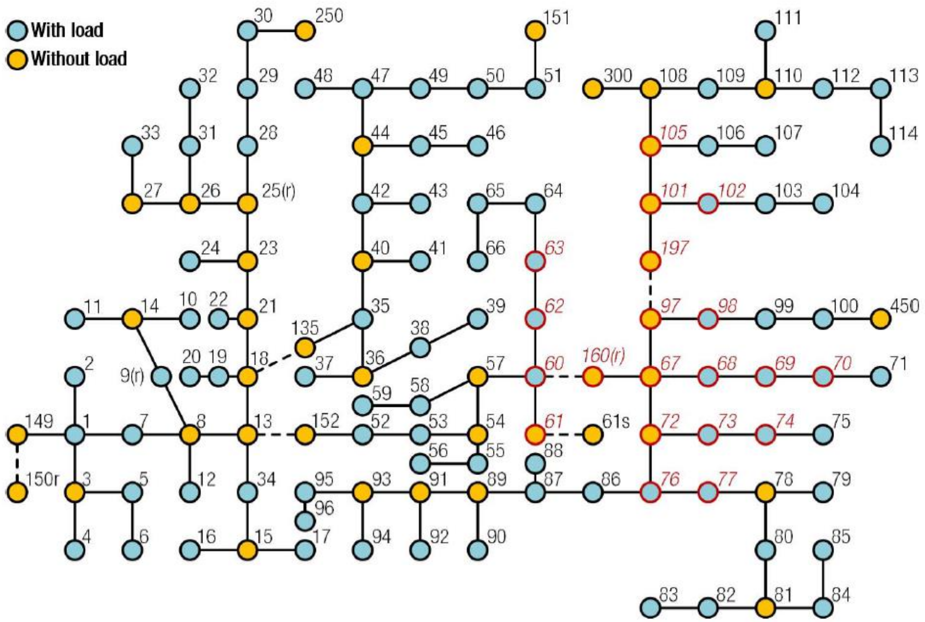


Fig. 2. An illustration of the IEEE 123 bus system. It is assumed that voltage and current phasors of PQ buses (connected to loads) are measured. As an example, the 20 buses that are closest (in distance) to bus 67 are highlighted with red color and italic numbers. Normally closed switches are represented by dashed lines.

Modeling:

- **Buses** are defined as **nodes** and transmission **lines** as **edges**.
- For a given measured bus in a distribution system, its three-phase **voltage** and **current** phasors are used as **node features**.
- Values corresponding to unmeasured phases are set to zero.
- Then formulated the fault location task as a **node classification problem**.
- **Spectral Graph Convolution** is applied to classify/ locate the faulty nodes

Detection **Accuracy**:

- CNN: 85.38~96.20
- **GCN: 97.65~99.77**

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

Case Study from Smart Grid

sensors

MDPI

Article

End-to-End Deep Graph Convolutional Neural Network Approach for Intentional Islanding in Power Systems Considering Load-Generation Balance

Zhonglin Sun [1], Yannis Spyridis [1], Thomas Lagkas [2,3,*], Achilleas Sesis [1], Georgios Efstathopoulos [1] and Panagiotis Sarigiannidis [2]

Intentional islanding is a corrective procedure that aims to protect the stability of the power system during an emergency, by dividing the grid into several partitions and isolating the elements that would cause cascading failures

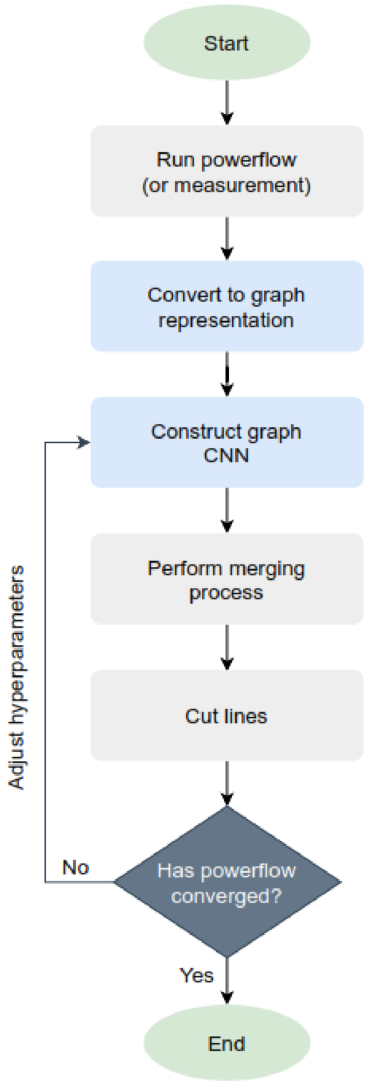
A **cascading failure** is a process in a system of interconnected parts in which the failure of one or few parts can trigger the failure of other parts and so on.

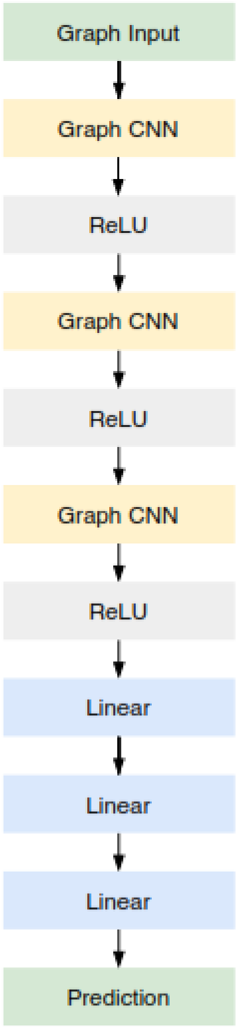
This is a **graph/network partitioning** problem

Methodology:

- a GCN model is proposed for Intentional Islanding problem.
- Two types of loss functions are examined for the graph partitioning task, and a loss function is added on the deep learning model, aiming to minimize the load-generation imbalance in the formed islands.
- the proposed method is dynamic, relying on real-time system conditions to calculate the result.

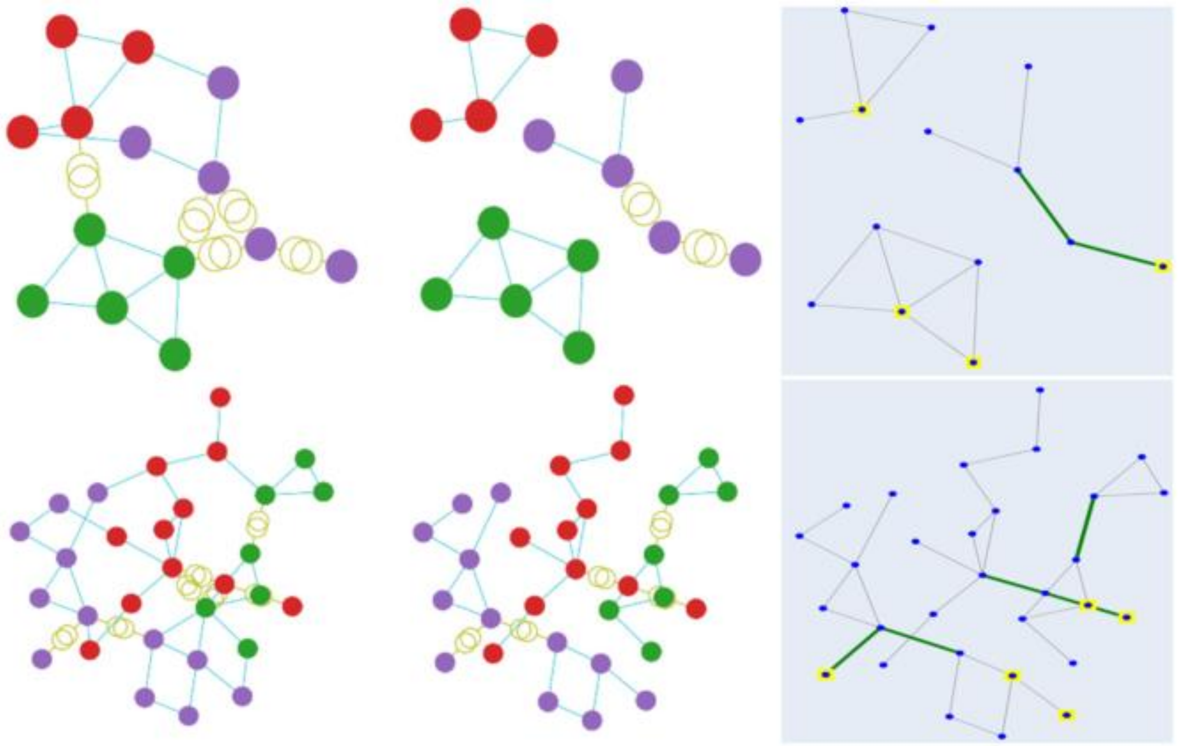
Case Study from Smart Grid





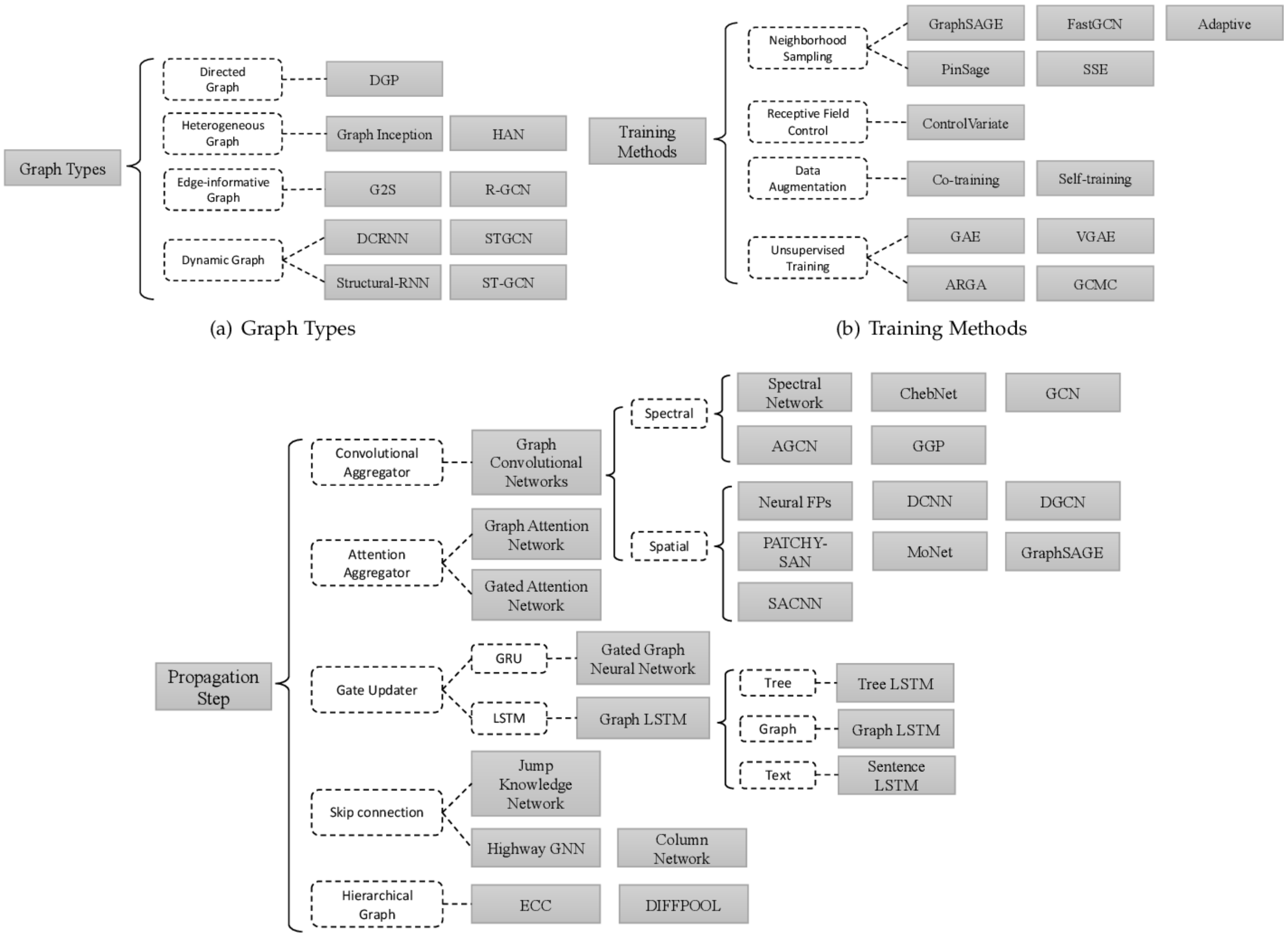
Modeling:

- **Buses** are defined as **nodes** and transmission **lines** as **edges**.
- **Adjacency matrix** and **power flow** used as weights.
- Used the **Generalizable Approximate Partitioning** (GAP) framework and the **normalized min-cut problem** for graph partitioning.



Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

Recent development on GNN



(a) Graph Types

(b) Training Methods

(c) Propagation Steps

An overview of variants of graph neural networks.

- Spectral Method:
  - ChebNet
  - GCN
- Non-Spectral Method:
  - DCNN
  - GraphSAGE
  - Neural FPs
- Graph Attention Network (GAT)
- Graph LSTM

[5]. Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, Maosong Sun, "Graph neural networks: A review of methods and applications", AI Open, Volume 1, 2020.

Note: Original source of all graphical illustrations have been embedded into each image. Click on the image to view the source.

Interested? Here are more references….

[6]. A. Ajit, K. Acharya and A. Samanta, "A Review of Convolutional Neural Networks," 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), 2020 .

[7]. S. J. Plathottam, H. Salehfar and P. Ranganathan, "Convolutional Neural Networks (CNNs) for power system big data analysis," 2017 North American Power Symposium (NAPS), 2017.

[8]. Z. Zhang, P. Cui and W. Zhu, "Deep Learning on Graphs: A Survey," in *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[9]. Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang and P. S. Yu, "A Comprehensive Survey on Graph Neural Networks," in IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 1, pp. 4-24, Jan. 2021.

[10]. Liao, Wenlong & Bak-Jensen, Birgitte & Pillai, Jayakrishnan & Wang, Yuelong & Wang, Yusen,"A Review of Graph Neural Networks and Their Applications in Power Systems", Preprint, 2021.

[11]. Ronald J. Brachman, Jie Zhou, "Introduction to Graph Neural Network" Book,2020.

[12]. Zhang, S., Tong, H., Xu, J. et al., "Graph convolutional networks: a comprehensive review", Comput Soc Netw, 2019.

Python libraries on Graph Neural Network………………

[13]. Deep Graph Library (DGL), Easy Deep Learning on Graphs.

[14]. Spektral

[15]. Gnn

[16]. PYTORCH GEOMETRIC

And many more……………………

Thank You