Graph Neural Networks

GNN

@avkashchauhan

Tutorial Agenda

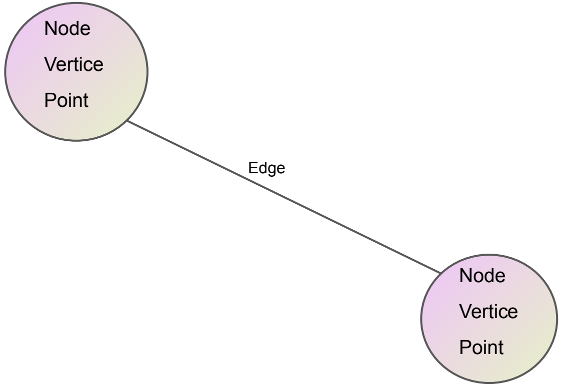
PART 1

- Fundamentals of Graph
- Mathematics of Graph
- Introduction to NetworkX Python Package
- Graph Programming with NetworkX
- Introduction to GNN
- Relationship between GNN and CNN
- Introduction to PyG (pytorch_geometric)
- Graph Visualization Tools
  - Gephi
  - yEd
- Various Graph Data Manipulation

PART 2

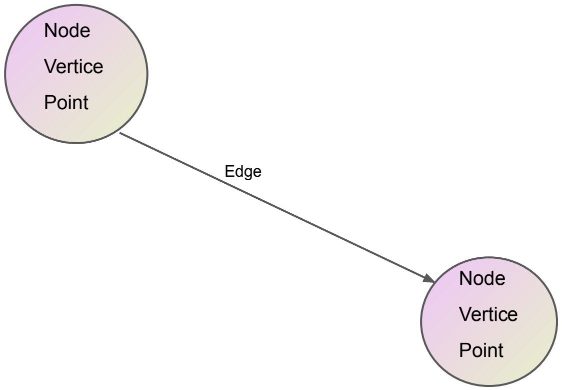
- Fundamentals to Graph Neural Networks
- Mathematics of GNN
- GNN Programming with PyG
- Deep Learning Experimentation with GNN
- Creating Neural Network for a GNN
- Real world GNN Examples

Fundamentals of Graphs

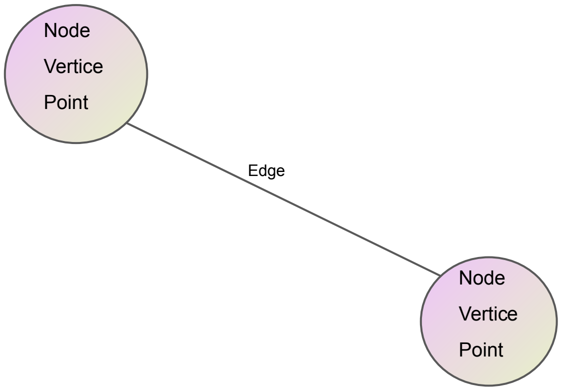


A Graph is a collection of vertices and edges.
Node, Vertice, Point:

You Define what part of data will be used as Node, vertice or Point, It’s your design.



Directed



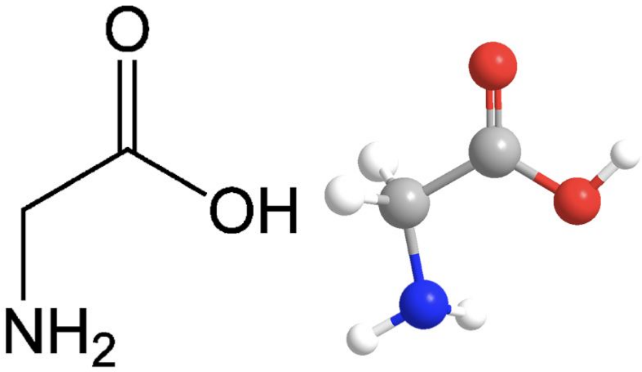
Undirected

Node, Vertice, Point:

You Define what part of data will be used as Node, vertice or Point, It’s your design.

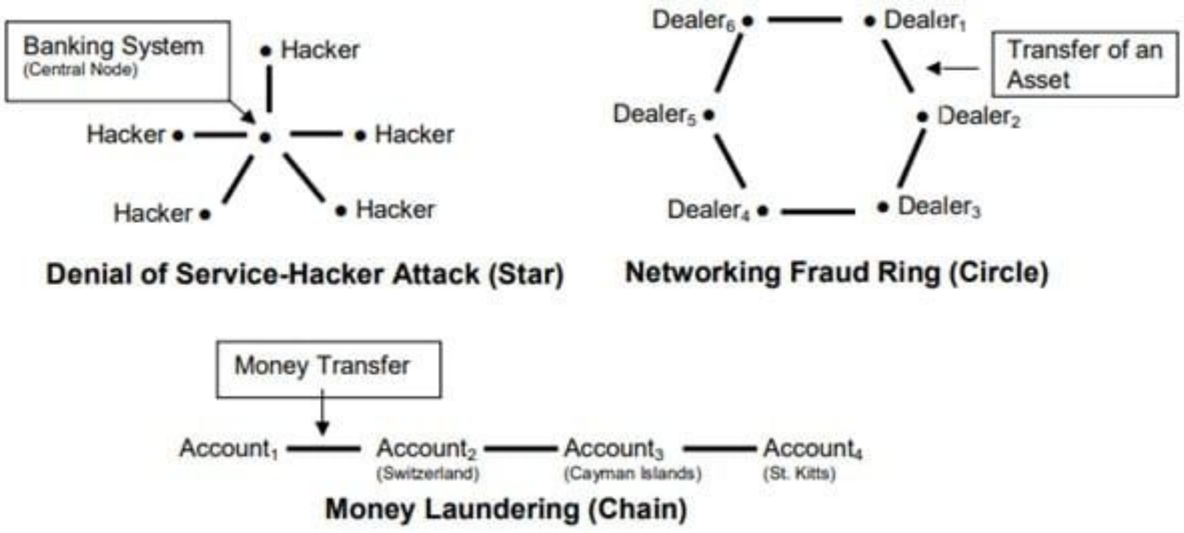
Examples	Nodes	Edges	Example Usages
Google Knowledge Graph	People, Places, Things	Connections	SEO
Chemical Molecular Structure	Atoms	Bonds	Molecule Structure
Document citation Network	Documents	Citation by a person	Cora Dataset
Social Media Networks	Person, properties	Connections	Virality, Influence
Network Design Security	Devices	Connections	Relationships
Financial Transactions	Transections	Connectivity	Fraud, AML

Example	**Nodes**	**Edges**	**Example**
Chemical Molecular Structure	Atoms	Bonds	Molecule Structure (Glycine)



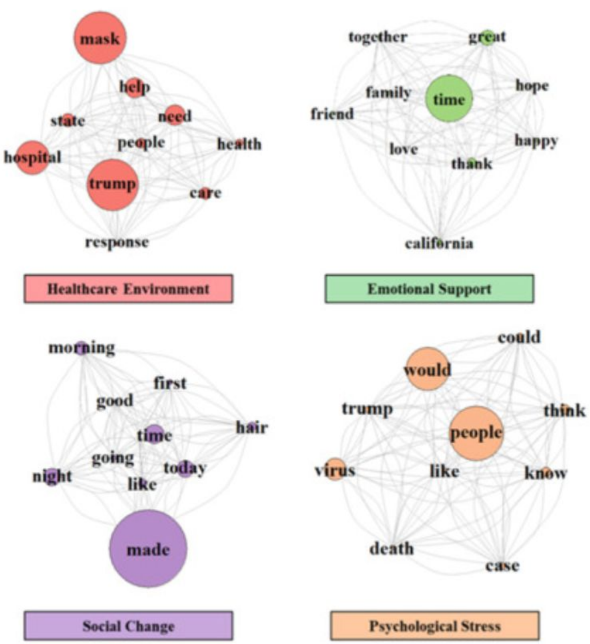
https://www.acs.org/content/acs/en/molecule-of-the-week/archive/g/glycine.html

Example	Nodes	Edges	Example Usages
Financial Transactions	Transections	Connectivity	Fraud, AML



https://www.cgi.com/sites/default/files/white-papers/Implementing-social-network-analysis-for-fraud-prevention.pdf

Example	Nodes	Edges	Example Usages
Social Media Networks	Person, Entities	Connections	Virality, Influence



Social network centrality measures of the top 10 words on major COVID-19 themes.

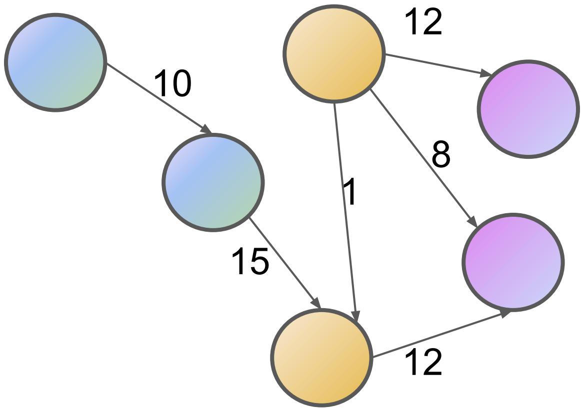
Themes	Degree	Betweenness	Closeness	Eigenvector
Healthcare Environment	18	4.0	0.001885	0.5443
Emotional Support	18	3.6	0.009339	0.5834
Business Economy	18	1.3	0.000421	0.6495
Social Change	18	5.0	0.002602	0.5315
Psychological Stress	18	2.2	0.000656	0.5790

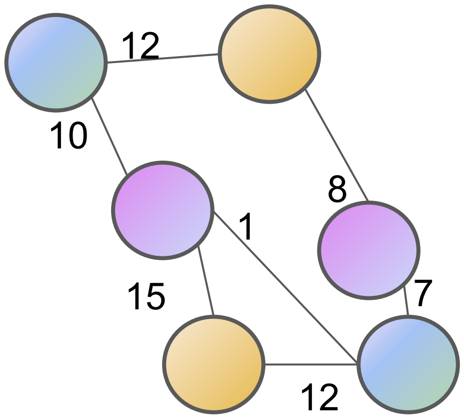
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7438102/

Directed Vs Undirected Graph

https://docs.google.com/presentation/d/11mR1nkIR9fbHegFkcFq8z9oDQ5sjv8E3JJp1LfLGKuk/edit#slide=id.g78327f1586_0_1154

Weighted Graphs





- A Weighted graph is a graph with edges labeled by the numbers
    - I.e. Distance, quantity, price, value etc.
- A weight is a numerical value attached to each individual edge.
- Each branch must have some weight as defined in the weight rule

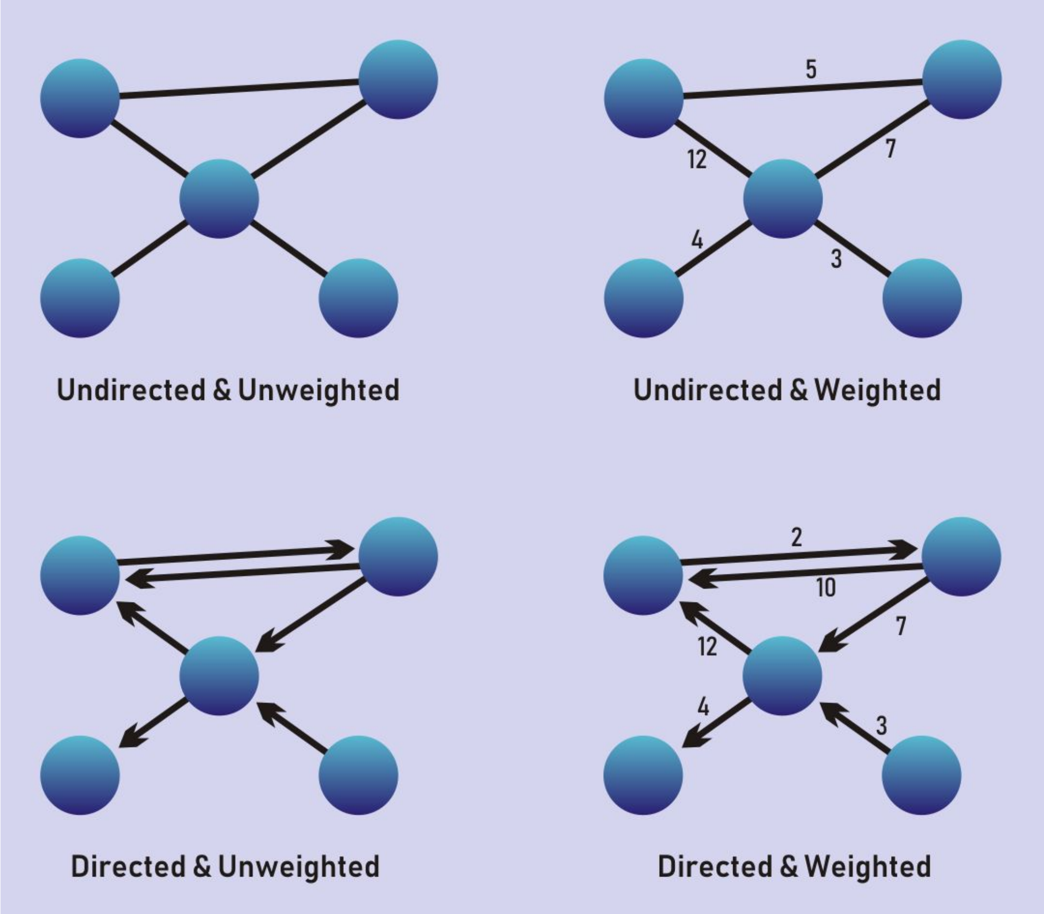


Image Source: https://medium.com/tebs-lab/types-of-graphs-7f3891303ea8

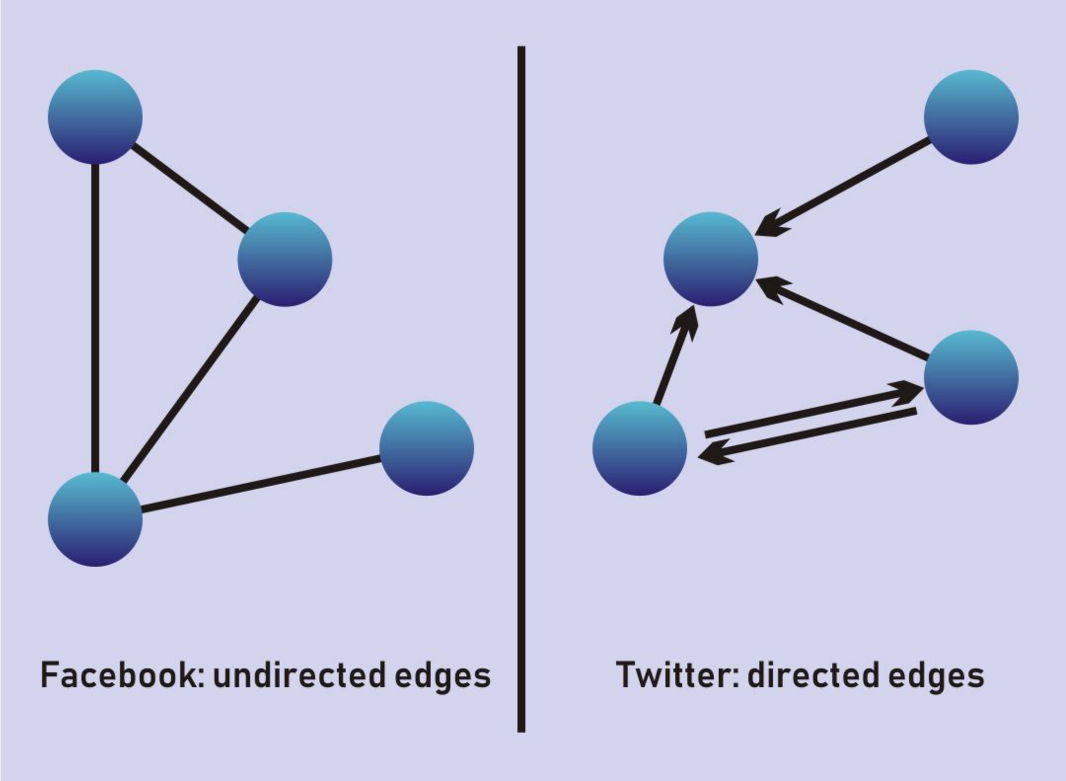
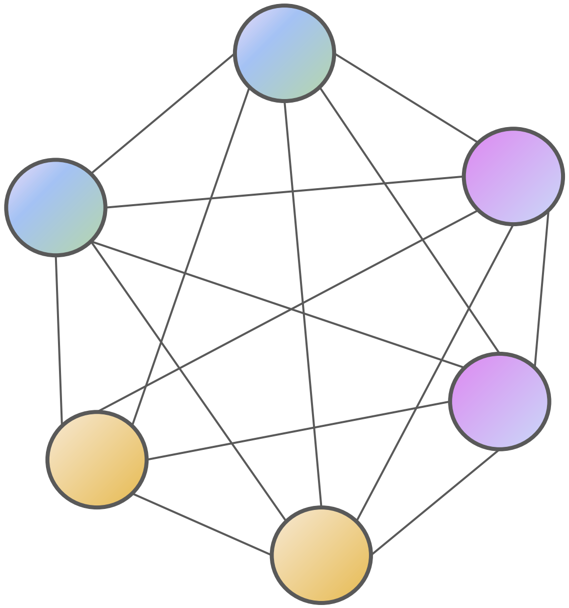


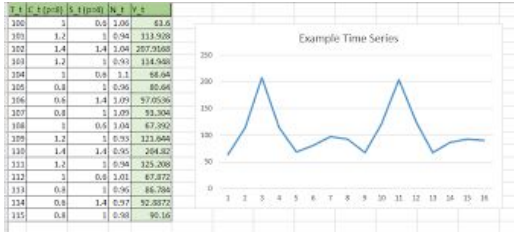
Image Source: https://medium.com/tebs-lab/types-of-graphs-7f3891303ea8

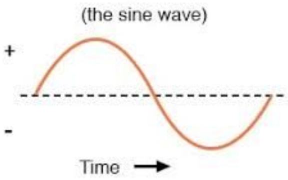
Complete Graph
Fully Connected Graph

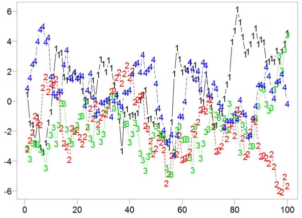


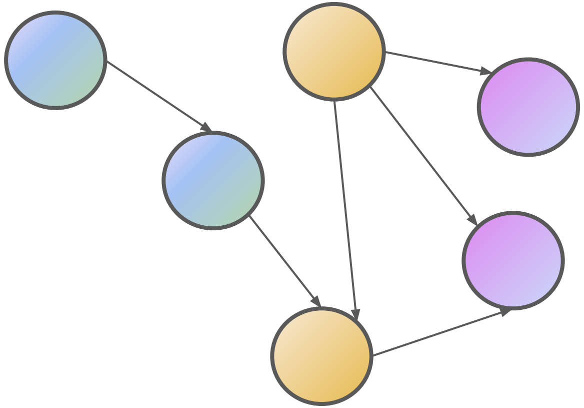
All nodes are connected with each other

Why Graphs are hard to understand?





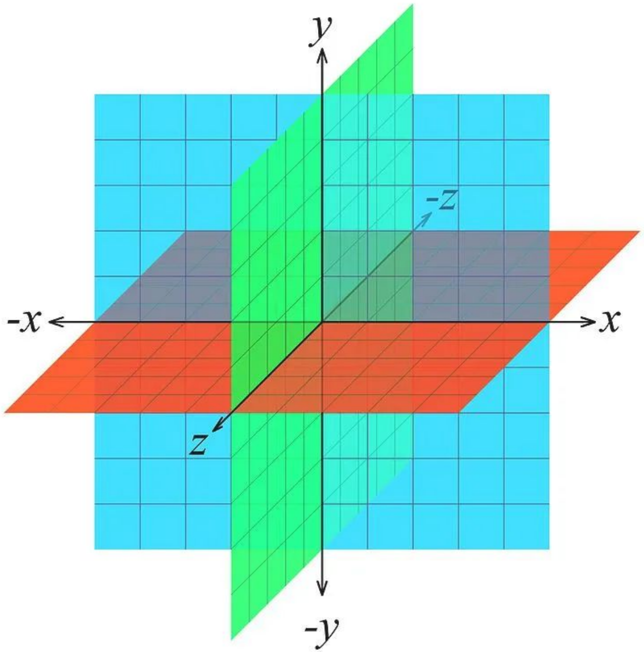


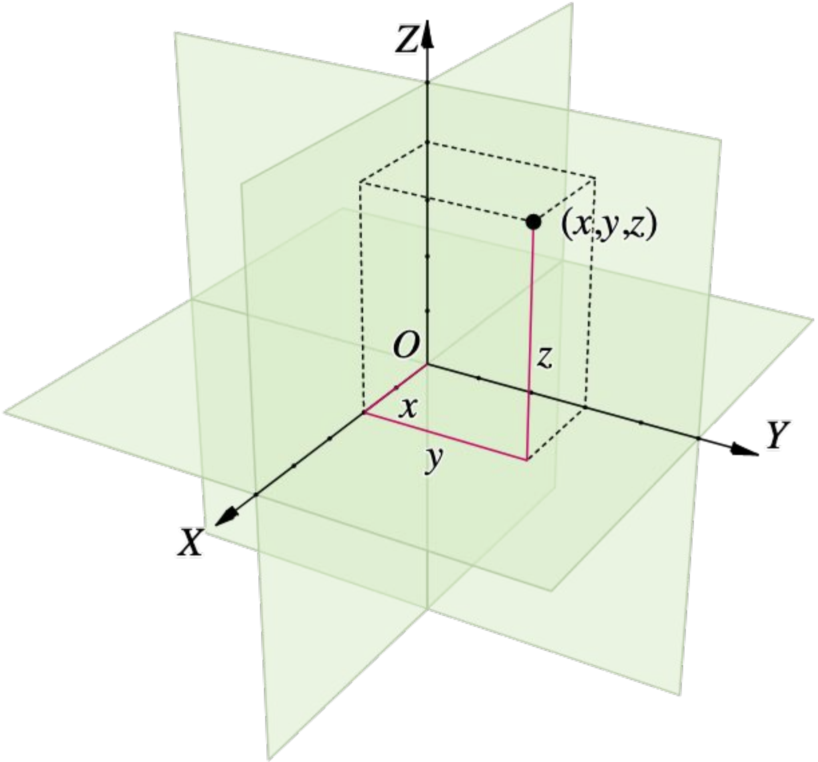


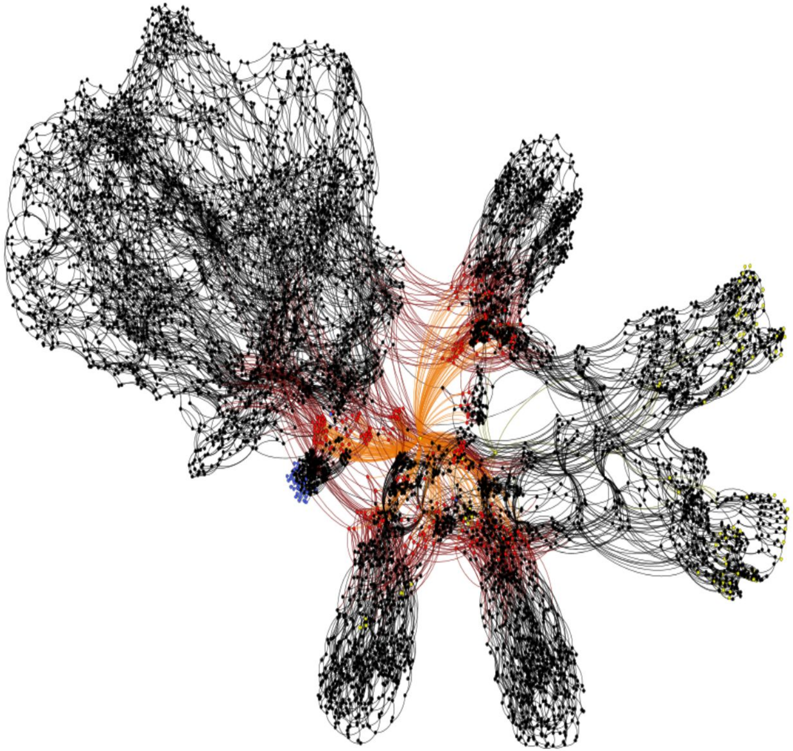
Can be represented into the Euclidean space also have the fixed form or representation.

Graphs does not have a fixed form and can NOTE be represented into the Euclidean space

Euclidean Space - 3 Dimensional (x,y,z) Plane







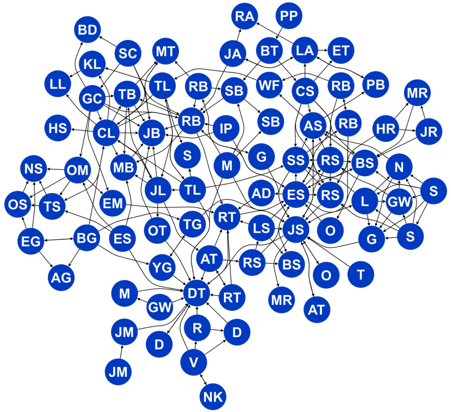
Example of a giant graph: circuit netlist. Figure from J. Baehr et. al. “Machine Learning and Structural Characteristics of Reverse Engineering”

Why we should use Graphs?

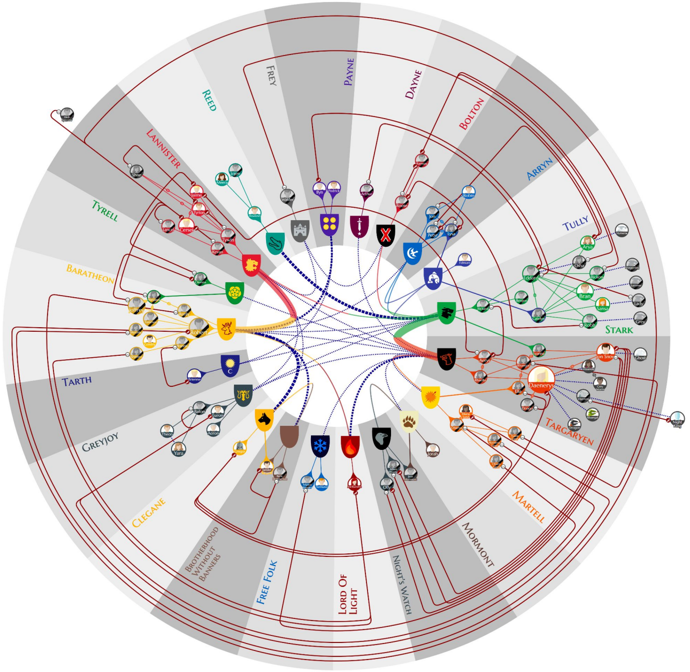
- Give intuitive representation to abstract concepts i.e. relationships and interactions.
- Intuitively visual representation of information.
- Form a Natural basis for analyzing relationships in a Social context.
- Break down complex problems into simpler representations
- Transform the complex problems into representations from different perspectives.

https://prodramp.com

‘Game of Thrones’ Relationship Graph

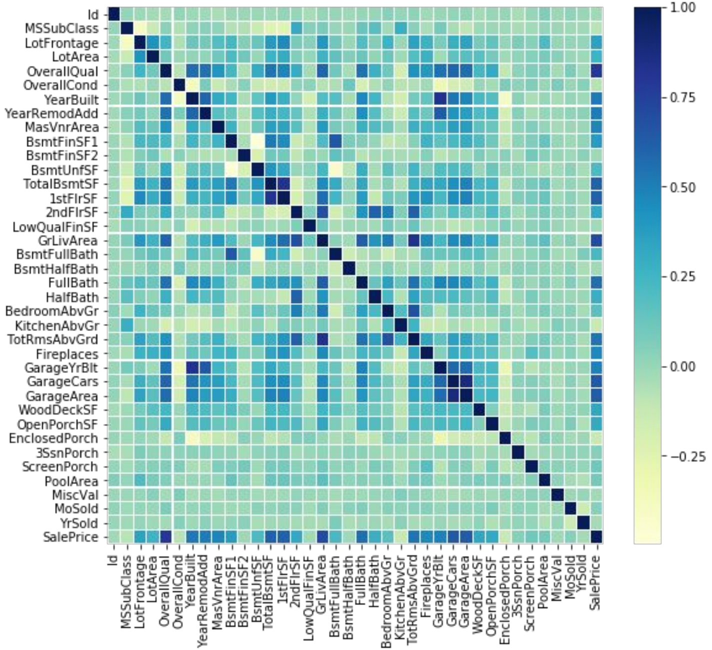


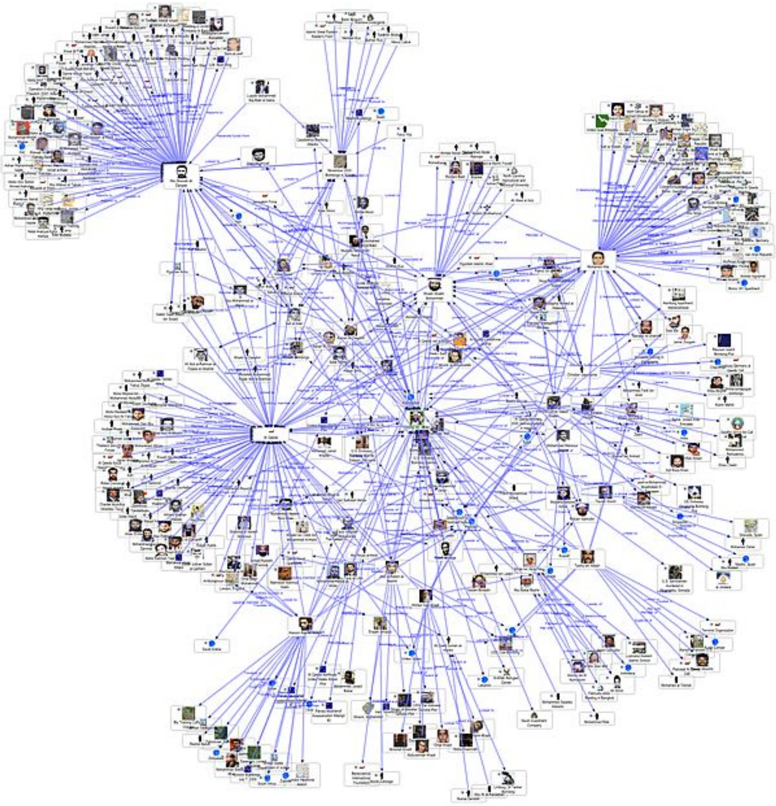
84 Characters

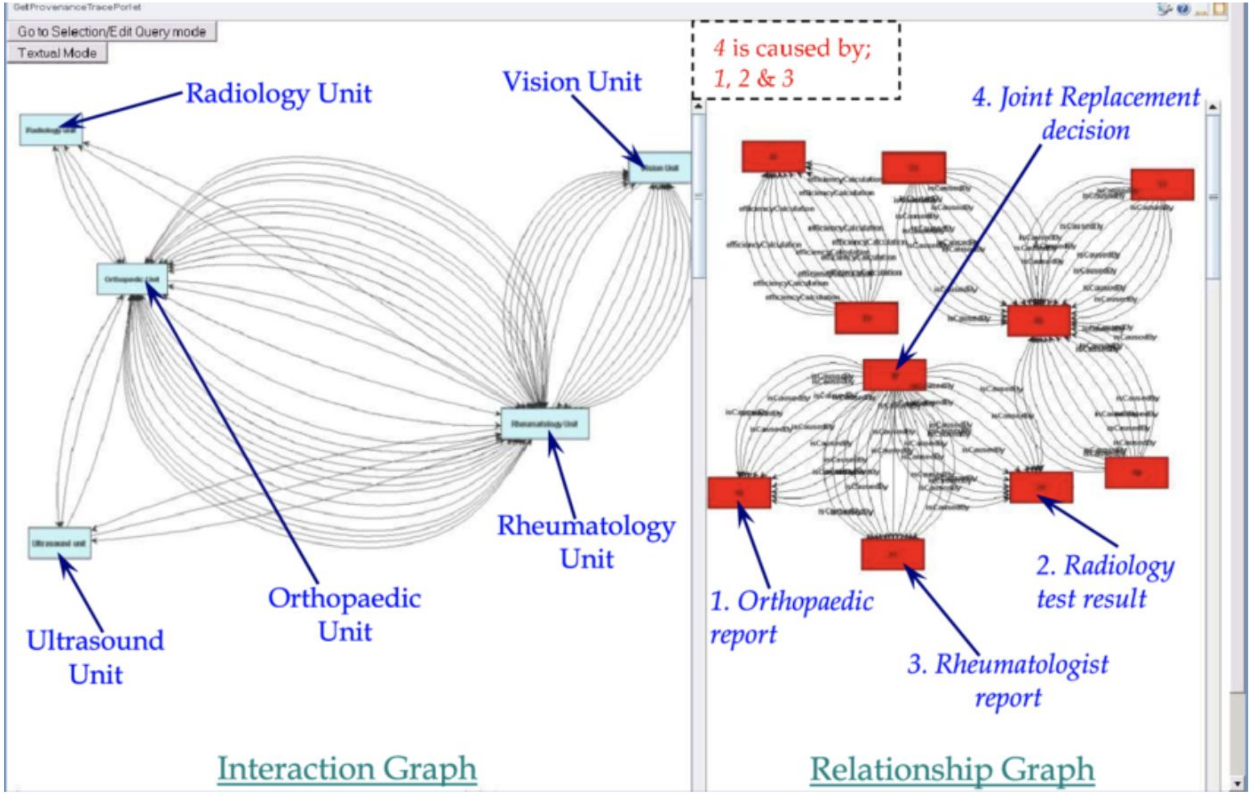


https://www.yworks.com/blog/graph-drawing-contest-2018

Relation & Correlation







https://www.researchgate.net/figure/Interaction-graph-and-relationship-graph_fig1_221158679

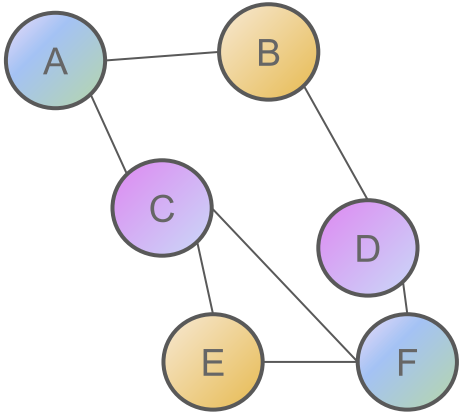
Traditional Graph Analysis Methods

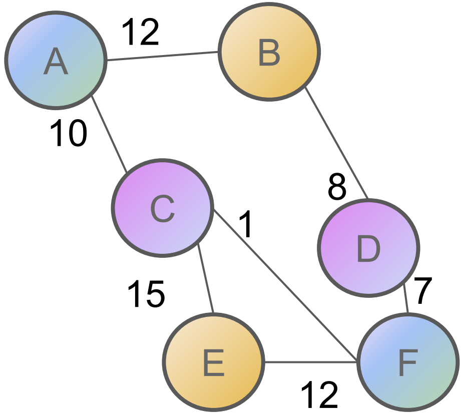
- Searching algorithms, e.g. BFS, DFS
- Shortest path algorithms, e.g. Dijkstra's algorithm, Nearest Neighbour
- Spanning-tree algorithms, e.g. Prim's algorithm
- Clustering methods, e.g. Highly Connected Components, k-mean

Limited based on their use cases

Mathematics of Graph

Mathematical Representation of Graph





Set of Vertices
- V = {A, B, C, D, E, F} ->

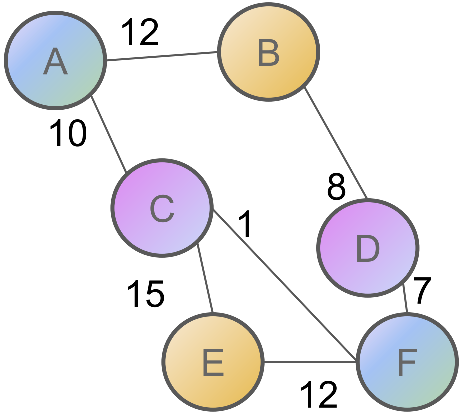
Set of Edges
- E = {AB, AC, BD, CE, CF, DF, EF}
- E = {(A,B), (A,C), (B,D), (C,E), (C,F), (D,F), (E,F)}

Set of Vertices
- V = {A, B, C, D, E, F} ->

Set of Edges
- E = {(A,B,12), (A,C,10), (B,D,8), (C,E,15), (C,F,1), (D,F,7), (E,F,12)}

Graph G = (V, E)

Neighbors



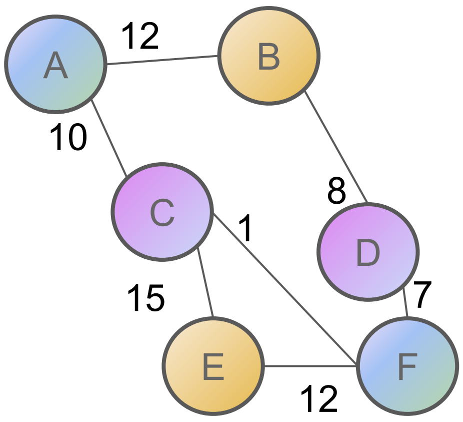
Neighbors:

- Two nodes that are connected with an edge are called neighbors.

Given Example:

- B and C are neighbors for A
- E and F are neighbors for C
- D is neighbor for B
- F is neighbor for C, D and E

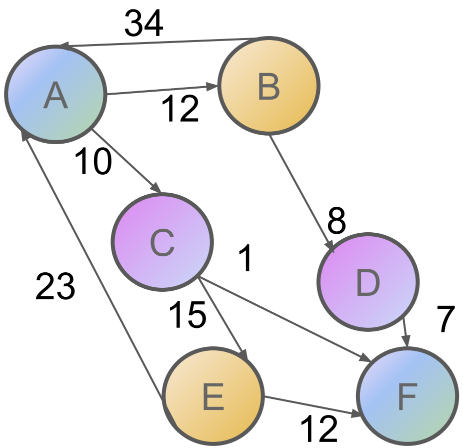
Edge List



A	B	12
A	C	10
B	D	8
C	E	15
C	F	1
D	F	7
E	F	12

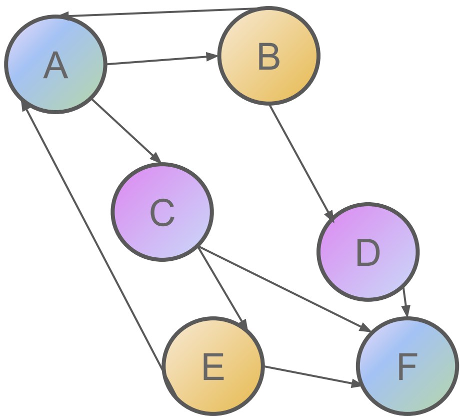
https://prodramp.com

Edge List - Directed



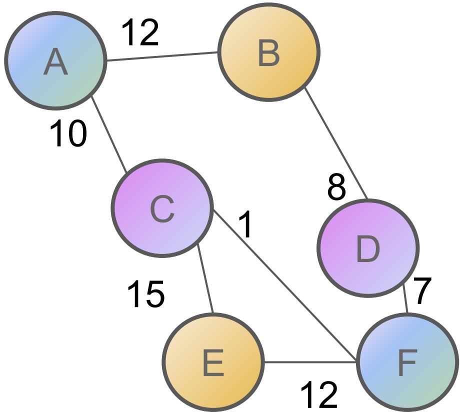
A	B	12
A	C	10
B	D	8
C	E	15
C	F	1
D	F	7
E	F	12
E	A	23
B	A	34

Edge List - Directed & Un-weighted



A	B
A	C
B	D
C	E
C	F
D	F
E	F
E	A
B	A

Adjacency Matrix

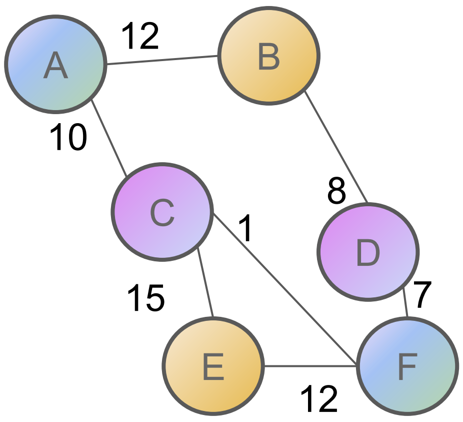


- Adjacency matrix a 2D square matrix
- Each node in the graph has an entry in both dimensions.
- Unweighted graph as T/F or 1/0 values
- Weighted graph as weights, no weights means -1

Representation:

- $A = N\,x\,N$

Adjacency Matrix - Weighted & Undirected

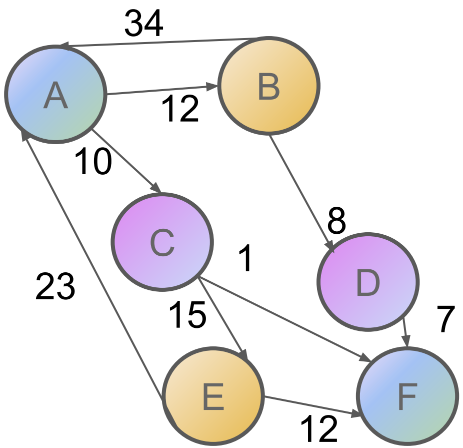


	A	B	C	D	E	F
A	-1	12	10	-1	-1	-1
B	-1	-1	-1	8	-1	-1
C	-1	-1	-1	-1	15	1
D	-1	-1	-1	-1	-1	7
E	-1	-1	-1	-1	-1	12
F	-1	-1	-1	-1	-1	-1

$$A = 6 \, x \, 6$$

https://prodramp.com

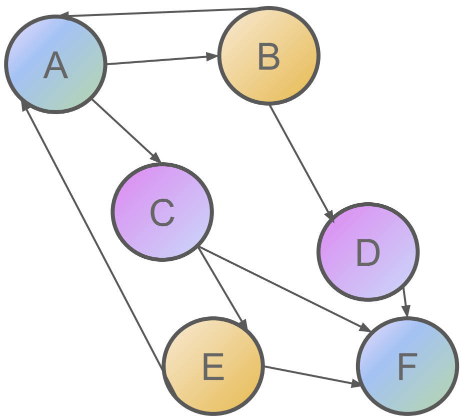
Adjacency Matrix - Weighted & Directed



	A	B	C	D	E	F
A	-1	12	10	-1	-1	-1
B	34	-1	-1	8	-1	-1
C	-1	-1	-1	-1	15	1
D	-1	-1	-1	-1	-1	7
E	23	-1	-1	-1	-1	12
F	-1	-1	-1	-1	-1	-1

$$A = 6 \, x \, 6$$

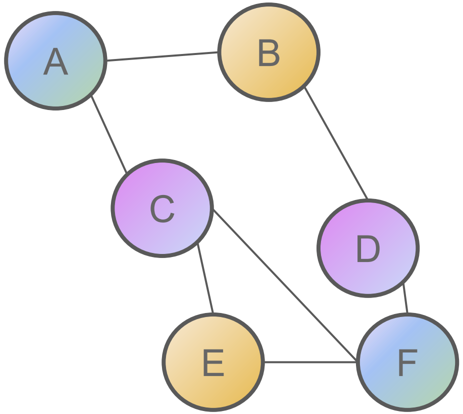
Adjacency Matrix - Un-Weighted & Directed



	A	B	C	D	E	F
A	F	**T**	**T**	F	F	F
B	**T**	F	F	**T**	F	F
C	F	F	F	F	**T**	**T**
D	F	F	F	F	F	**T**
E	**T**	F	F	F	F	**T**
F	F	F	F	F	F	F

$$A = 6\,x\,6$$

Adjacency Matrix - Unweighted & Undirected

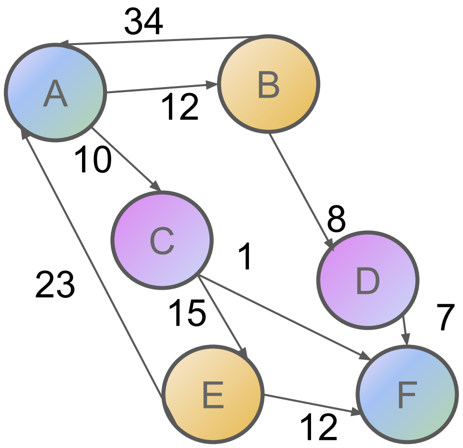


	A	B	C	D	E	F
A	F	T	T	F	F	F
B	F	F	F	T	F	F
C	F	F	F	F	T	T
D	F	F	F	F	F	T
E	F	F	F	F	F	T
F	F	F	F	F	F	F

$A = 6 \, x \, 6$

https://prodramp.com

Adjacency Matrix - Weighted & Directed



	A	B	C	D	E	F
A	-1	12	10	-1	-1	-1
B	34	-1	-1	8	-1	-1
C	-1	-1	-1	-1	15	1
D	-1	-1	-1	-1	-1	7
E	23	-1	-1	-1	-1	12
F	-1	-1	-1	-1	-1	-1

A_{AB}	A-B
A_{AC}	A-C
A_{AF}	A-C-E-F
A_{CD}	C-E-A-B-D
A_{BC}	X

$$A = 6\,x\,6$$

Complete Graph

- All elements of adjacency matrix **A** are 1/T, except along the diagonal
- Path exists for each and every node

Sparse Graph

- Not all elements of adjacency matrix **A** are 1/T, lots of values are -1 or F/0
- Not every node is connected to each other

Extended Reading:

- https://medium.com/@TebbaVonMathenstien/implementations-of-graphs-92eb7f121793

Node Attribute Matrix / Feature Matrix (X)

- The features or attributes of each node
- A Graph with N nodes with the size of node attributes as F,
    - Matrix Shape = $N \, x \, F$

Example:

Document 1: I travelled to Himalaya.
Document 2: I travelled to Las Vegas Nevada

Corpus: {I, travelled, to, Himalaya, Las, Vegas, Nevada}
Size of Corpus (F) = 7

	Document 1	**Document 2**
I	1	1
Travelled	1	1
to	1	1
Himalaya	1	0
Las	0	1
Vegas	0	1
Nevada	0	1

The shape of node attributes matrix X = NxF = 2x7 = 14

Bag-of-words Illustration as node features

Trick Question

All Diagonals are values are 1

	A	B	C	D	E	F
A	T	T	T	F	F	F
B	F	T	F	T	F	F
C	F	F	T	F	T	T
D	F	F	F	T	F	T
E	F	F	F	F	T	T
F	F	F	F	F	F	T

Trick Question - **Self Loop**

All Diagonals are values are 1

	A	B	C	D	E	F
A	T	T	T	F	F	F
B	F	T	F	T	F	F
C	F	F	T	F	T	T
D	F	F	F	T	F	T
E	F	F	F	F	T	T
F	F	F	F	F	F	T

Graph Programming in Python with NetworkX

Hands-on-exercise

- Use networkX Python Library
- Apply Python Code into Jupyter Notebook

Graph Neural Networks (GNN)

GNN

- A neural network that can directly be applied to graphs.
- GNN provides a Classification & Prediction tasks at:
    - node level
    - edge level
    - graph level
- Mainly 3 types:
    - Recurrent Graph Neural Network
    - Spatial Convolutional Network
    - Spectral Convolutional Network
- GNN can help us to perform:
    - Node Classification
    - Link/Edge Prediction
    - Graph Classification

Batch vs Single Mode - GNN Data Models

Single Mode:

- A single graph consists of large collection of nodes
- Example:
    - In document classification a single BIG graph consisting of all the documents as the nodes.

Batch Mode

- A graph collection of various graphs, each graph may have one or multiple nodes-
- Example:
    - In chemical molecules classification, each molecule as 1 different graph
    - The number of the graphs will be as many as the number of the molecules

prodramp
https://prodramp.com

Node Embeddings in GNN

Principle:

- Nodes have neighbors and connections.
- Removing the neighbors and connections around a node, node will lose all its information.
- Neighbors of a node and connections to neighbors define the concept of the node.

Node Embeddings:

- Every node represent its concept as a state *(x)* .
- The node state *(x)* produces the decision about its concept as an output *(o)*
- The final state *(x_n)* of the node is normally called “node embedding”.

The task of all GNN is to determine the “node embedding” of each node, by looking at the information on its neighboring nodes.

Relationship between GNN and CNN

(CNN + GNN = GCN)

GCN

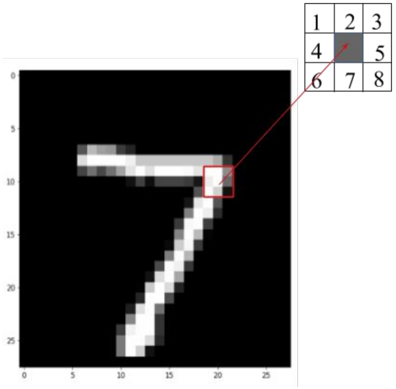
The simplest GCN has only three different operators:

- Graph convolution
- Linear layer
- Nonlinear activation

https://neptune.ai/blog/graph-neural-network-and-some-of-gnn-applications

Image as Graph Data

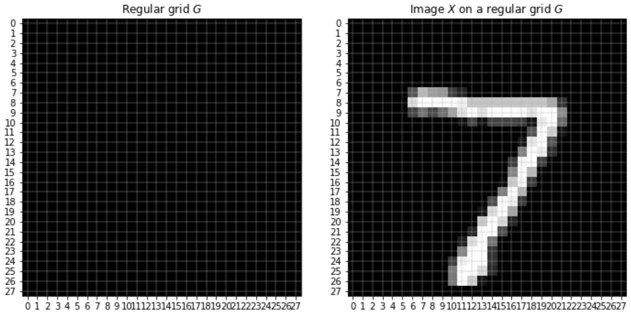
- **Each node** represents **each pixel.**
- **Node feature** represents **the pixel value**.
- **Edge feature** represents **the Euclidean distance between each pixel**.
- The closer 2 pixels are to each other, the larger the edge values.



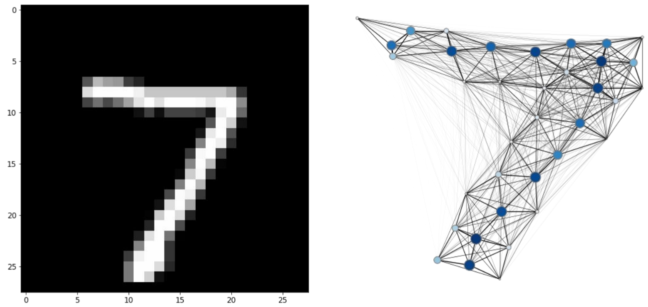
CNN view an image as a graph.

- Adjacent pixels number 2,4,5,7 share the same Euclidean distance with the middle pixel.
- Similarly, diagonal pixels 1,3,6,8 share slightly larger Euclidean distance

https://medium.com/analytics-vidhya/getting-the-intuition-of-graph-neural-networks-a30a2c34280d



MNIST Images - 28x28 Pixels Grid
N = 28x28 = 784 Pixels



Graph G is going to have N=784 nodes and edges will have large values (thicker edges in the Figure below) for closely located pixels and small values (thinner edges) for remote pixels.

An image from the MNIST dataset on the left and an example of its graph representation on the right. Darker and larger nodes on the right correspond to higher pixel intensities. The figure on the right is inspired by Figure 5 in (Fey et al., CVPR, 2018)

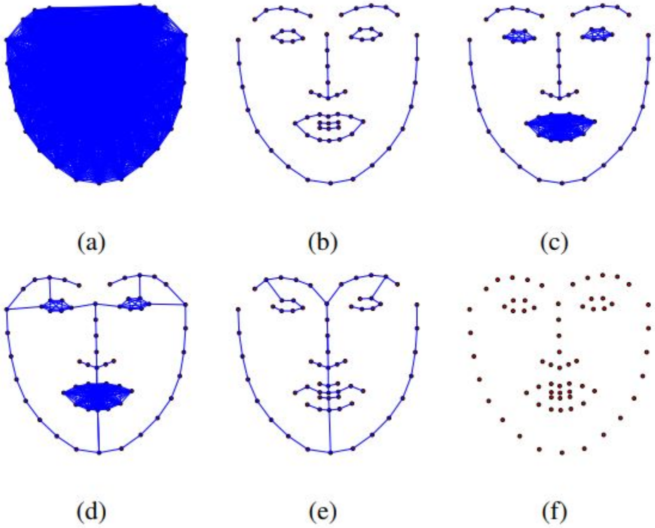
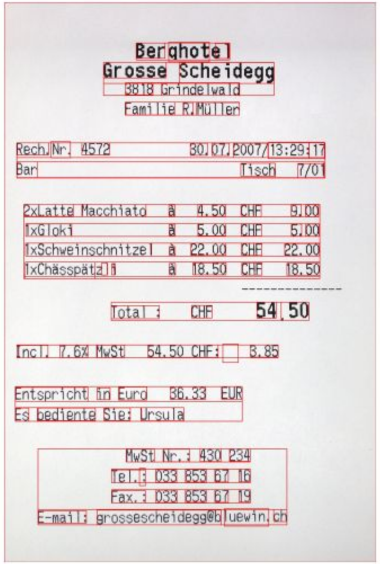
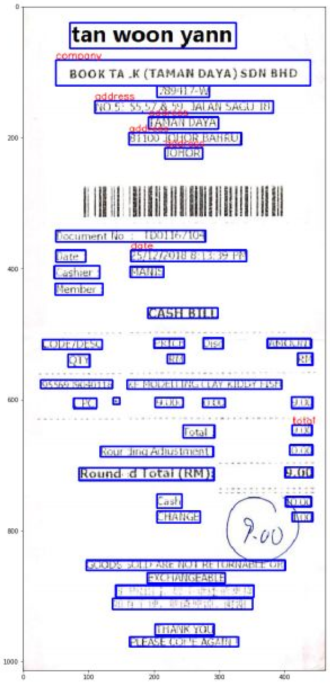
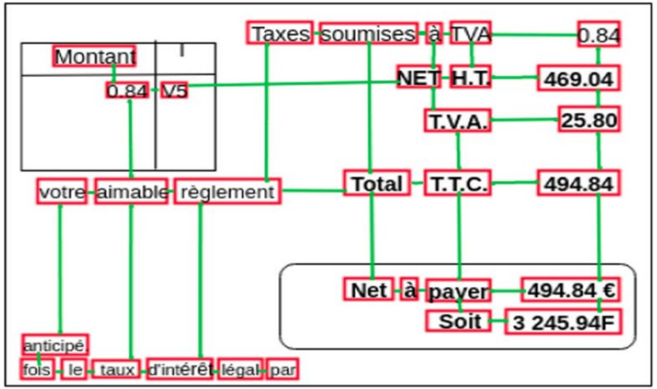


Figure 2: Employed graph structures. *(a)* Complete graph. *(b)* Chain per area. *(c)* Chain and complete per area. *(d)* Chain and complete per area with connections between them. *(e)* Minimum spanning tree. *(f)* Empty graph.

A figure from (Antonakos et al., CVPR, 2015) showing representation of a face as a graph of landmarks. This is an interesting approach, but it is not a sufficient facial representation in many cases, since a lot can be told from the face texture captured well by convolutional networks. In contrast, reasoning over 3D meshes of a face looks like a more sensible approach compared to 2D landmarks (Ranjan et al., ECCV, 2018).



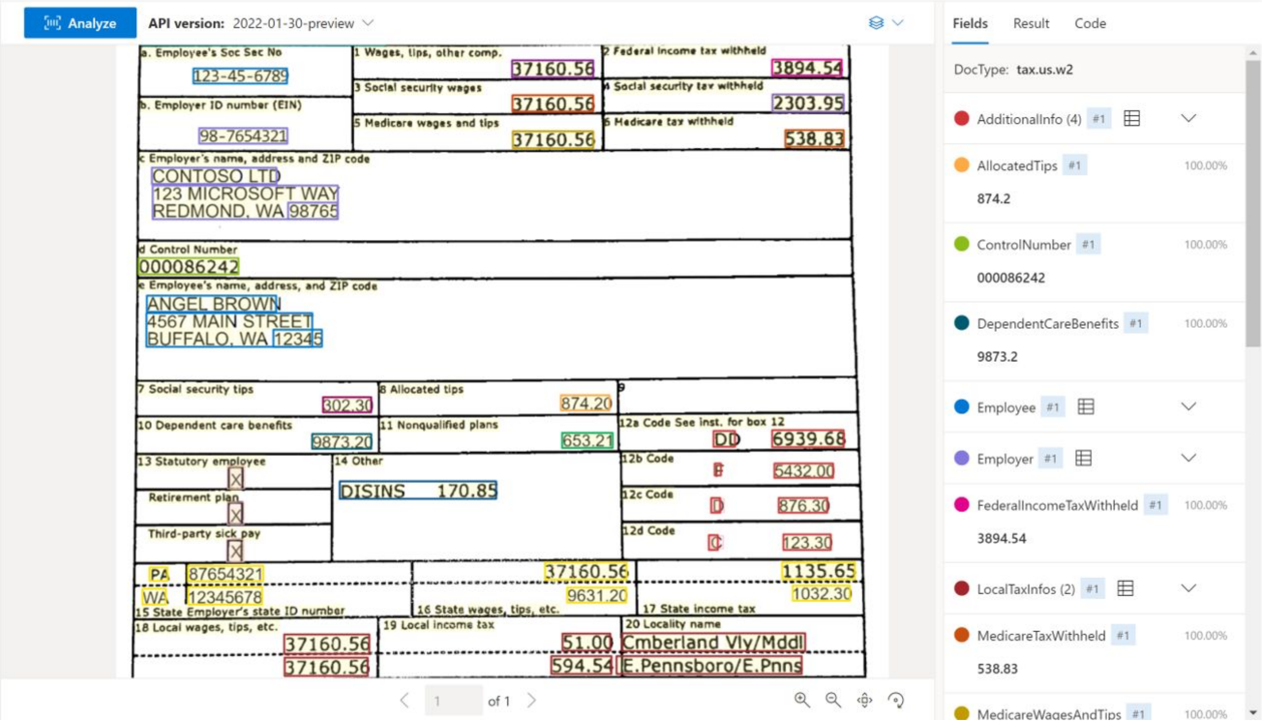




- **Each node** represents **individual text segment (textbox).**
- **Edge feature** represents **the linkage between 2 text segments**, such as horizontal/vertical distances and width/height ratio between text segments.

https://nanonets.com/blog/information-extraction-graph-convolutional-networks/

Azure Form Recognizer



https://fott.azurewebsites.net/projects/create

https://docs.microsoft.com/en-us/azure/applied-ai-services/form-recognizer/concept-w2

Introduction to PyG

(pytorch_geometric)

Resources

Libraries

- https://networkit.github.io/
- https://github.com/danielegrattarola/spektral
- https://networkx.org/
- https://pytorch-geometric.readthedocs.io/en/latest/

Resources: Starters:

- https://www.kdnuggets.com/2018/05/wtf-tensor.html
- https://medium.com/tebs-lab/types-of-graphs-7f3891303ea8
- https://hadrienj.github.io/posts/Deep-Learning-Book-Series-2.1-Scalars-Vectors-Matrices-and-Tensors/
- https://towardsdatascience.com/an-introduction-to-graph-neural-network-gnn-for-analysing-structured-data-afce79f4cfdc
- https://towardsdatascience.com/understanding-graph-convolutional-networks-for-node-classification-a2bfdb7aba7b
- https://neptune.ai/blog/graph-neural-network-and-some-of-gnn-applications
- https://pub.towardsai.net/understanding-social-networks-409dffc785ea

Advance:

- https://towardsdatascience.com/hands-on-graph-neural-networks-with-pytorch-pytorch-geometric-359487e221a8
- https://medium.com/analytics-vidhya/getting-the-intuition-of-graph-neural-networks-a30a2c34280d
- https://medium.com/@BorisAKnyazev/tutorial-on-graph-neural-networks-for-computer-vision-and-beyond-part-1-3d9fada3b80d

Examples

- https://graphsandnetworks.com/the-cora-dataset/

Documentations

- https://networkx.org/documentation/networkx-1.10/tutorial/tutorial.html
- https://pytorch-geometric.readthedocs.io/en/latest/notes/introduction.html