Graph algebra and deep learning for finance

Grégoire Pacreau

CMAP (Doctorant à l'école polytechnique)

November 2022

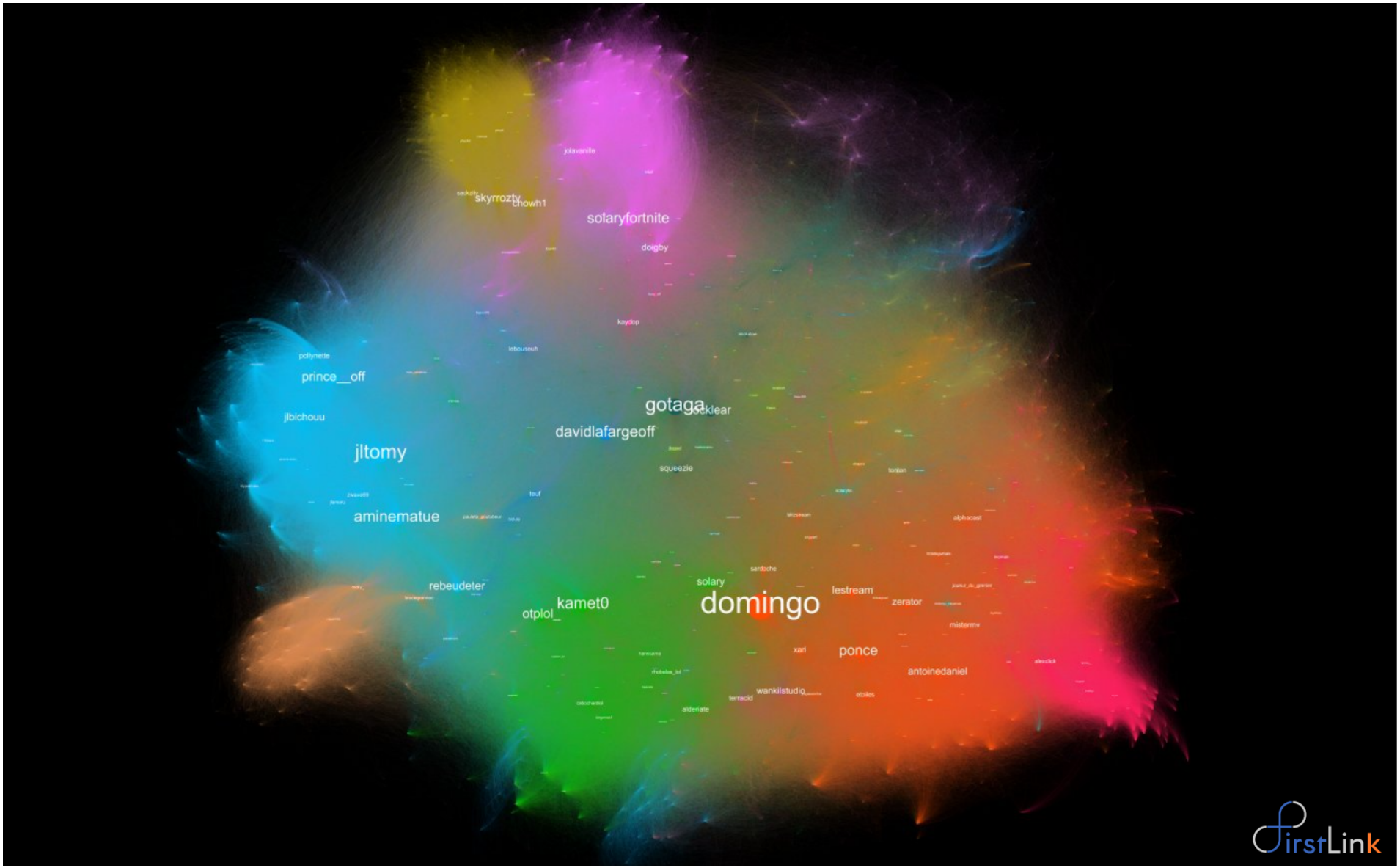
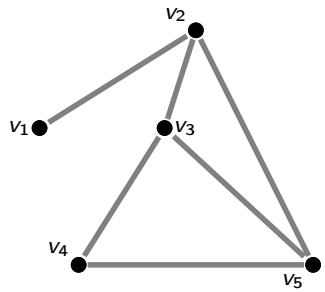


Figure – https://first-link.fr/cartographie-twitch-francais-communautes-audiences/

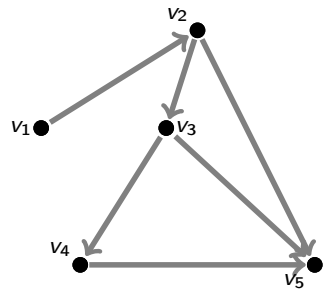
Graph

$G = (V, E)$, $V = [1, N]$, $E \subset V \times V$
Sometimes, we also add a weight function $w : E \to \mathbb{R}$.

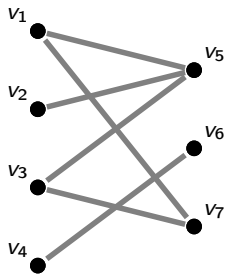
Figure – Examples of graphs



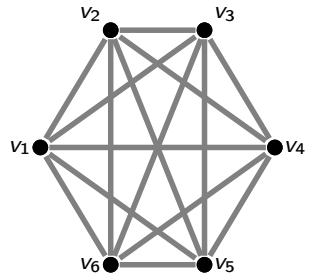
(a) A five nodes graph



(b) Same graph, but directed



(c) A bipartite graph

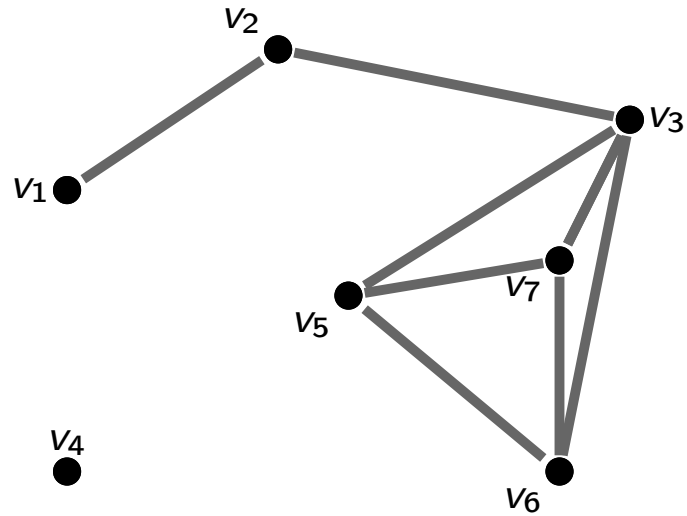


(d) The complete six nodes graph

Adjacency matrix

Let A be the graph G's adjacency matrix, *i.e.* $A_{i,j} > 0 \iff (i,j) \in E$. A is symetric if, and only if, G is undirected.

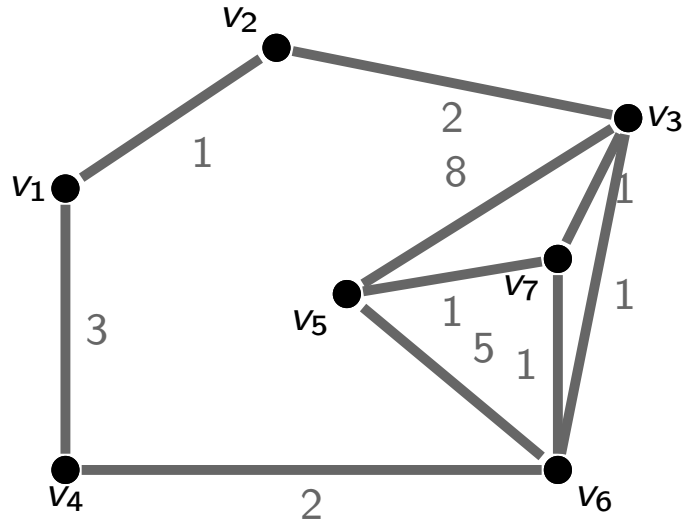
$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$



Weighted graphs

A graph can hold weights on edges. Then, A is no longer binary but holds values in $\mathbb{R}^+$ or $\mathbb{R}$, and $A_{ij} = w(i,j)$.

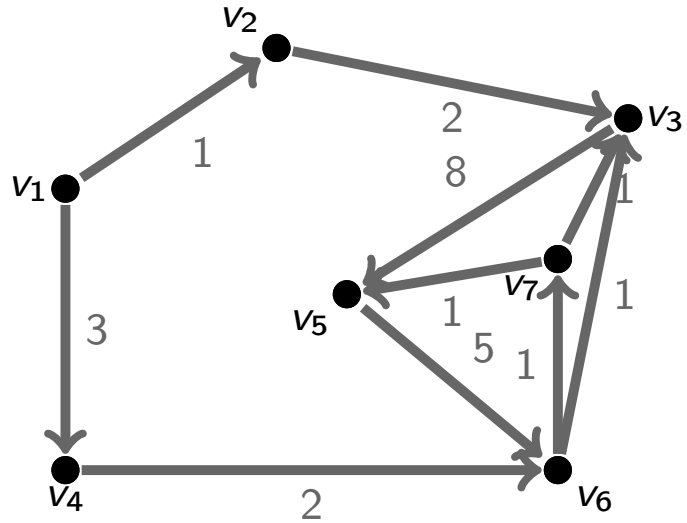
$$A = \begin{pmatrix} 0 & \mathbf{1} & 0 & \mathbf{3} & 0 & 0 & 0 \\ \mathbf{1} & 0 & \mathbf{2} & 0 & 0 & 0 & 0 \\ 0 & \mathbf{2} & 0 & 0 & \mathbf{8} & \mathbf{1} & \mathbf{1} \\ \mathbf{3} & 0 & 0 & 0 & 0 & \mathbf{2} & 0 \\ 0 & 0 & \mathbf{8} & 0 & 0 & \mathbf{5} & \mathbf{1} \\ 0 & 0 & \mathbf{1} & \mathbf{2} & \mathbf{5} & 0 & \mathbf{1} \\ 0 & 0 & \mathbf{1} & 0 & \mathbf{1} & \mathbf{1} & 0 \end{pmatrix}$$



Directed/Undirected graphs

If A is no longer symetric, then the graph is undirected.

$$A = \begin{pmatrix} 0 & \mathbf{1} & 0 & \mathbf{3} & 0 & 0 & 0 \\ 0 & 0 & \mathbf{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{8} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{5} & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} \\ 0 & 0 & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 \end{pmatrix}$$



Any square matrix is a graph

In particular, any similarity metric between time series can be interpreted as a graph. A symetric metric will give an undirected graph (correlation, hellinger, etc), whereas most metrics will provide directed graphs (KL-divergence, Dynamic Time Warping, etc). Prado, 2016 first introduced financial graphs in the markovitz framework and showed they allowed better out of sample performances.

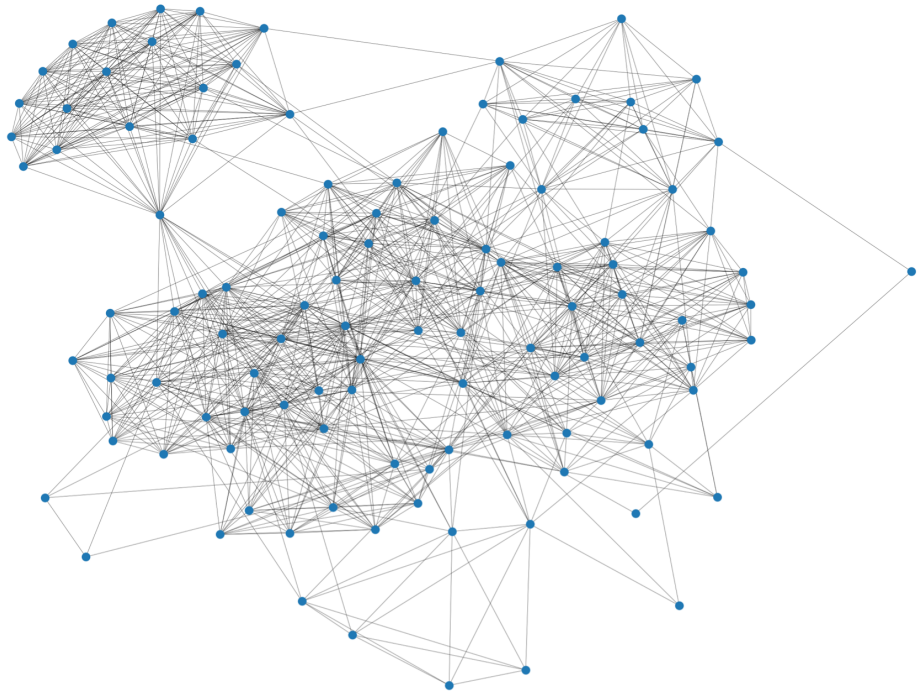


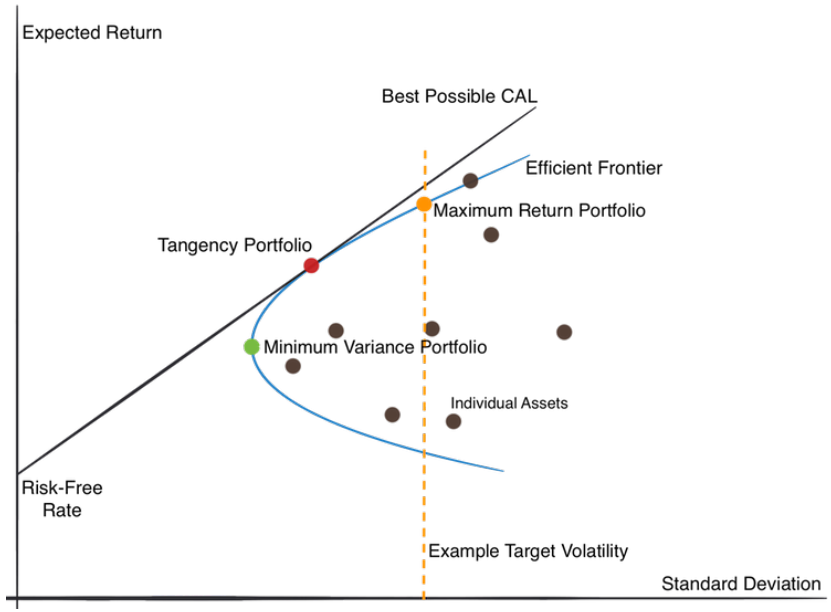
Figure – Graph constructed using a mix of correlation metrics and other information.

Markowitz's portfolio allocation

Given $\boldsymbol{r}$ the returns of assets in a market, and given an expected return μ, the optimal mean-variance portfolio w is the solution to the minimisation problem :

$$\min_w wRw^\top \text{ s.t. } ||w||_2 = 1 \text{ and } w^\top \mathbb{E}(r) = \mu$$

where $R = \text{cov(r)}$ is the covariance matrix of the portfolios.



Prado, 2016's solution

They argue that the markowitz minimisation is akin to finding a path in a complete graph (represented by the correlation matrix). Instead, they propose a allocation method based on clustering and edge pruning, which sparsify the structure of the markets.

The new optimisation

Given $\boldsymbol{r}$ the returns of assets in a market, and given an expected return μ, the optimal mean-variance portfolio w is the solution to the minimisation problem :

$$\min_{w} wAw^{\top} \text{ s.t. } ||w||_2 = 1 \text{ and } w^{\top}\mathbb{E}(r) = \mu$$

where A is the adjacency matrix of a graph constructed through various methods.

Supply Chain information

Grants a directed graph from customer to seller, or vice versa.
Can be weighted by size of the transaction (in percentage of total revenue for the seller) or any other information about the importance of the transaction (how vital it is to the activity of the client).

Ownership and investment

Directed graph from owner to subsidiary, from investment firm to holding, etc.

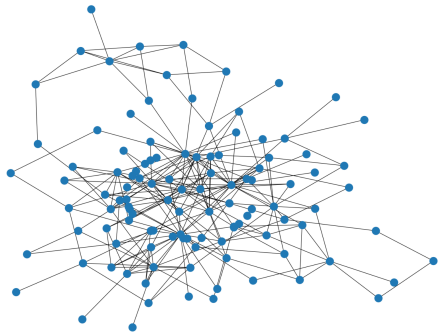


Figure – Undirected graph based on supply chain information on the SP500 index.

Graph of banking ecosystem

Linking accounts to institutions to track transactions between accounts (fraud detection)
Very useful to track blockchain portfolio ownership

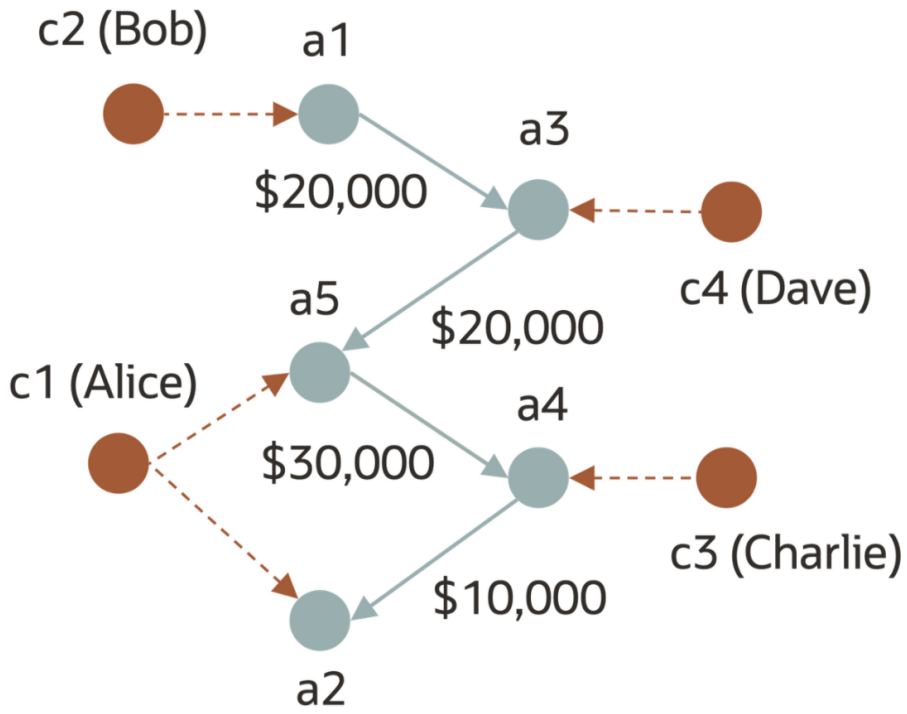


Figure – An example of transaction graph between four people. [1]

1. image from https://medium.com/oracledevs/analyze-bank-transaction-data-using-graph-part-1-3-2088c6024f81

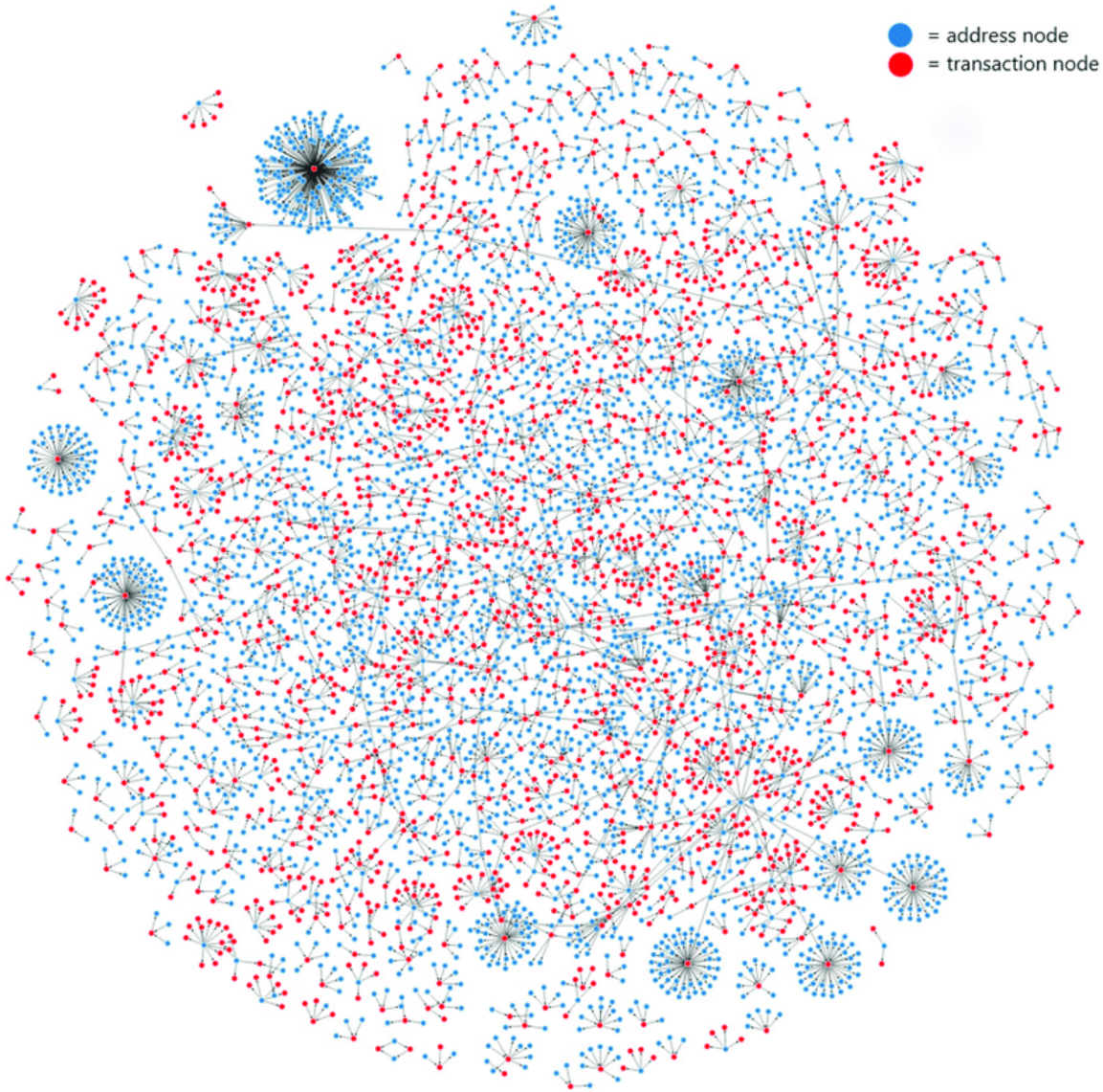


Figure – Address-transaction graph computed with one block of the Bitcoin mainnet. [2]

2. Zola, Francesco & Bruse, Jan & Eguimendia, Maria & Galar, Mikel & Urrutia, Raul. (2019). Bitcoin and Cybersecurity : Temporal Dissection of Blockchain Data to Unveil Changes in Entity Behavioral Patterns.

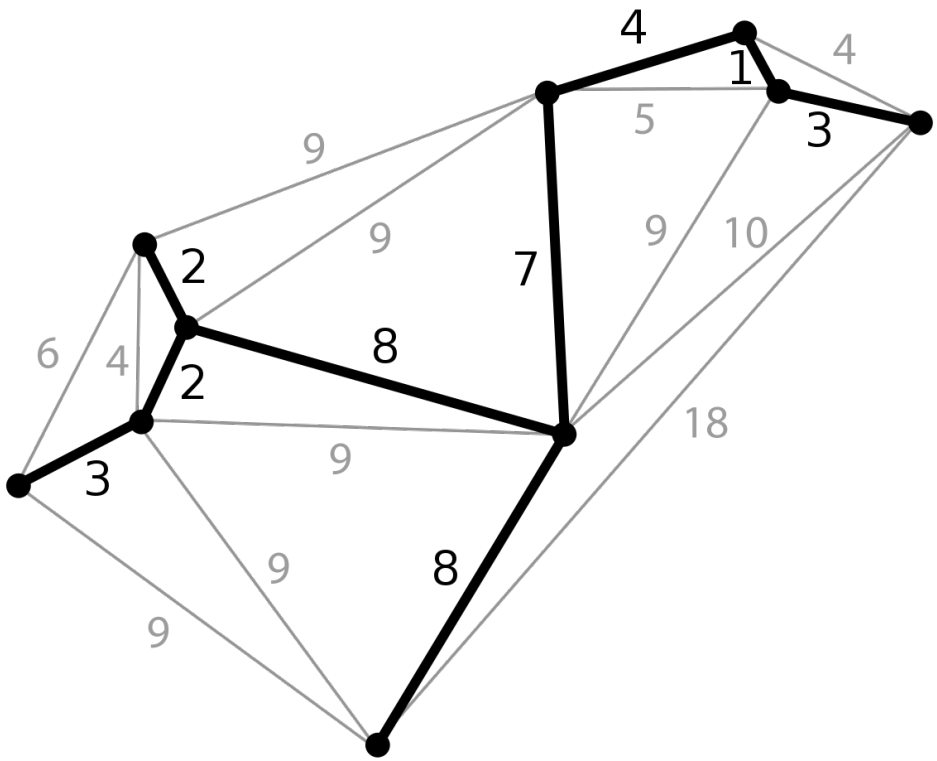
Some easy things to look for in a transaction graph

- loops, ie money that was rerouted to the original node (potential money laundering)
- closeness to known "evil" node (to try to find stollen money funeled through other accounts)
- diversification of neighbours (accounts belonging to major exchanges should communicate with many different accounts)

Pruning the graphs

Covariance and distances in general yield complete graphs. Several methods exist for reducing the number of edges :

- thresholding (for covariance and correlation, only keep those above a level) ;
- Minimum spanning Trees (Kruskal's and Prim's algorithms).



Networkx

Python library for all things graph
https://networkx.org

List of useful features

- Creates graph from numpy array (adjacency matrix) or pandas dataframe (based on common column value).
- Most common algorithms on graph are well optimised.
- Many representation tools linked with matplotlib or seaborn.

Spring layout representations

Main idea :

- unlinked nodes repulses each other like springs ;
- linked nodes attract each other.

Then, simulate the physical system resulting from these laws.

Kamada-Kawai

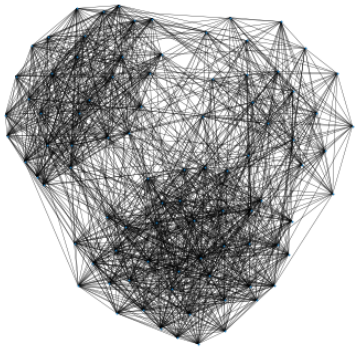
Based on shortest path from each nodes to the rest.

Spectral

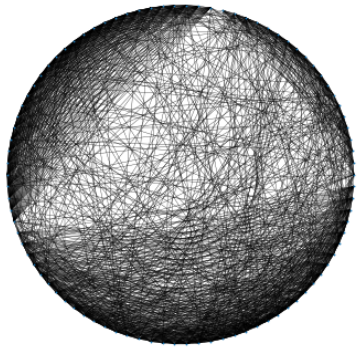
Representation according to the eigenvectors of the adjacency matrix.

Common representations

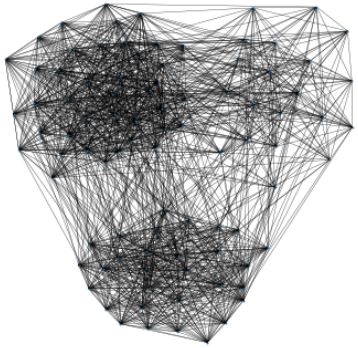
kamada-kawai



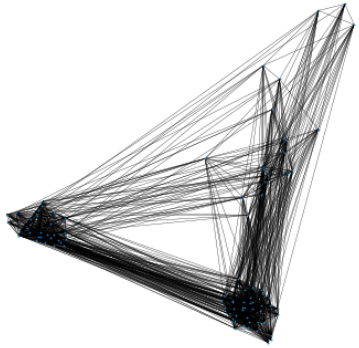
circle



spring-layout



spectral



Some complexity issues

Networks become very large very fast, while graph based algortihms have bottlenecks

Some complexity orders

Let n be number of nodes and m be the number of edges, then :

- shortest path algorithm are in $O(n \log n)$ (Dijkstra's algorithm) for positively weighted graphs or in $O(nm)$ (Bellman-Ford) in general.
- cycle detection is in $O(n + m)$ (depth first search).
- detecting isomorphisms between sub-graphs is **NP-complete**.

Node centrality

A node is central if it connects to many parts of the graph (traffic junction, metro hub, infection source, etc). There are many ways to define centrality.

Degree centrality

Centrality as the number of neighbours :

$$C_d(i) = d(i) = \sum_{j \in V} A_{i,j}$$

Problem : A node connecting two highly populated subgraphs should be central even though it hasn't many neighbours (a bridge is important for traffic).

Random walk centrality

Centrality as the probability that a random walk reaches a node :

$$C_e(i) = \frac{1}{\lambda} \sum_{j \in V} A_{i,j} C_e(j)$$

$$\iff A\mathbf{C}_e = \lambda \mathbf{C}_e$$

Problem : A bias remains towards highly connected nodes, bottlenecks with high importance will be set aside.

PageRank centrality

Normalisation of random walk centrality by the number of neighbours :

$$C_p(i) = \frac{1}{\lambda} \sum_{j \in V} A_{i,j} \frac{C_p(j)}{\max\left(1, \sum_{k \in V} A_{k,j}\right)}$$

Examples

- Centrality on supply chain data correlated to market exposure
- It can also correlate to the impact of companies on market fluctuations

Directed graphs' centrality

Centrality measures can be used in two ways in directed graphs (such as for supply chain modelling) :

- **Hub centrality** counts how many nodes point towards the central node (supplier centrality)
- **Authority centrality** counts to how many nodes are pointed by the central node (customer centrality)

Asset Allocation based on centrality

Výrost et al., 2019 and Olmo, 2021 look at how centrality can be used in asset allocation problems. Network measures based on centrality can reduce portfolio risk.
In particular, Olmo, 2021 provides insight on the relationships between the weights provided by Markowitz optimisation and centrality measures. In some circumstances, highly central nodes tend to be overweighted and their risk underappreciated.

Definition

The length of the longest shortest path in the graph from the node.

$$e(i) = \max_j l(i,j)$$

where $l(i,j)$ is the shortest path from i to j (potentially not symmetric).

Financial indicators

Kaya, 2013 looks for financial indicators based on the eccentricity of multi-asset graphs and minimum spanning tree.
In particular, during crises, they observe that asset networks tend to contract (i.e. eccentricity decreases).

Graph diameter

The maximum eccentricity over all nodes.

$$\max_i e(i)$$

By definition, any path from one part to another of the network will be upper bounded by the diameter.

Graph radius

Smallest eccentricity.

$$\min_i e(i)$$

Gives an idea of how distant the most eccentric node is from the most central one.

Graph differential calculus

Graph calculus is a subset of discrete calculus, we can define a gradient operator on edges and a divergent on nodes.
Let $f : V \to \mathbb{R}$ be a function on vertices and $F : E \to \mathbb{R}$ a function of edges.

Graph gradient

$$(\nabla f)_{i,j} = (f_j - f_i)\mathbb{1}_{(i,j)\in E}$$

Graph divergent

$$(\text{div}F)_i = \sum_{j\in\mathcal{N}(i)} A_{i,j}F_{i,j}$$

Graph Laplacian

We can then define a Laplacian operator on graphs, which behaves like the Laplace-Beltrami operator for differential equations :

$$L = \Delta = -\text{div}\nabla$$

$$\iff L_i = \sum_{j \in \mathcal{N}(i)} A_{i,j}(f_i - f_j)$$

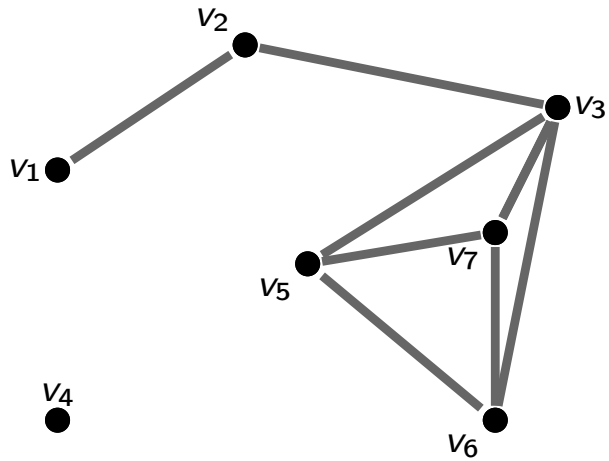
$$\iff L = D - A$$

with D the degree matrix.

Definitions

- degree of a node : $d(i) = |\mathcal{N}(i)| = \sum_{j \in V} A_{i,j}$
- degree matrix : $D_{i,i} = d(i)$
- Laplacian : $L = D - A$.

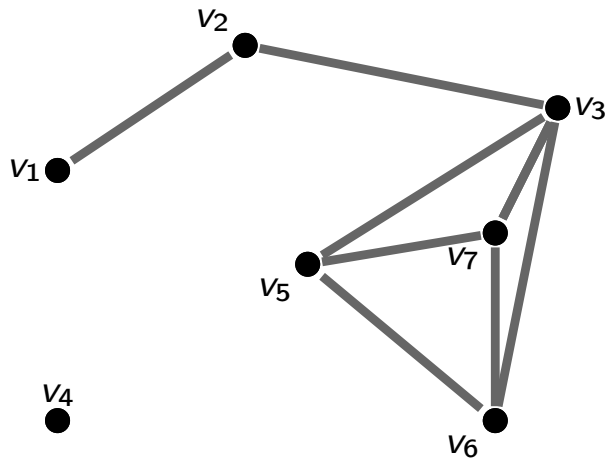
$$L = \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & 0 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 3 & -1 & -1 \\ 0 & 0 & -1 & 0 & -1 & 3 & -1 \\ 0 & 0 & -1 & 0 & -1 & -1 & 3 \end{pmatrix}$$



Normalised Laplacian

$$\tilde{L} = D^{-1/2} L D^{1/2} = I_N - D^{1/2} A D^{1/2}$$

$$\tilde{L} = \begin{pmatrix} 1 & -\frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 \\ -\frac{1}{\sqrt{2}} & 1 & -\frac{1}{2\sqrt{2}} & 0 & 0 & 0 \\ 0 & -\frac{1}{2\sqrt{2}} & 1 & -\frac{1}{2\sqrt{3}} & -\frac{1}{2\sqrt{3}} & -\frac{1}{2\sqrt{3}} \\ 0 & 0 & -\frac{1}{2\sqrt{3}} & 1 & -0.33 & -0.33 \\ 0 & 0 & -\frac{1}{2\sqrt{3}} & -0.33 & 1 & -0.33 \\ 0 & 0 & -\frac{1}{2\sqrt{3}} & -0.33 & -0.33 & 1 \end{pmatrix}$$



Notion of cut

For $G = (V, E)$, a cut is a set $C \subset E$ such that suppressing the edges in C leads to two graphs.
The min-cut problem is a very well studied one in graphs, with algorithms such as Edmonds-Karp (in polynomial time) or Karger (probabilistic solving).

Highly Connected Subgraphs

You cut the graph until the subgraphs are connected enough (required connectivity is a parameter).

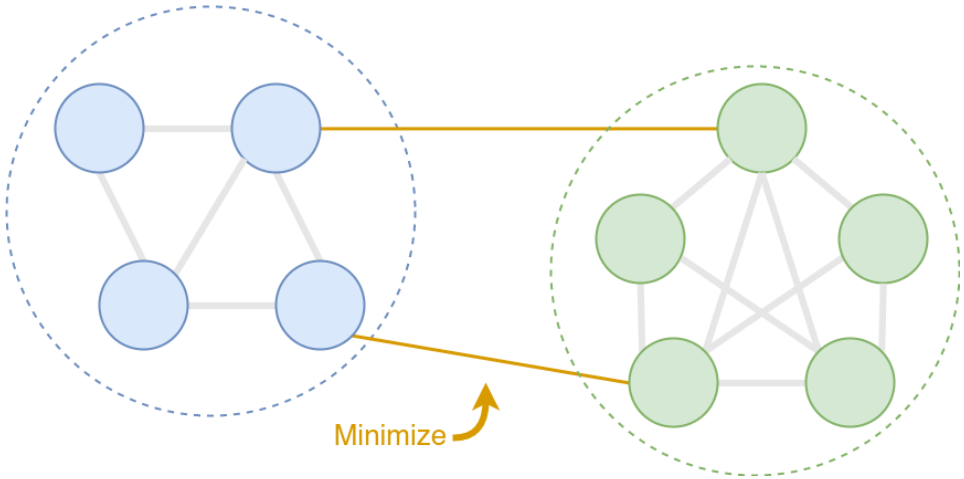


Figure – An illustration of the min-cut problem.

Graph Conductance

Analogy with electrical system. Essentially, the speed at which a random walk in the graph converges to its stationary distribution.
Let C be a cut of the graph, with V_C and V'_C the two subgraphs created by this cut. The conductance of graph G for weight function w is :

$$\phi = \min_{C \text{ a cut}} \frac{\sum_{iinV_C, j \in V'_C} w(i,j)}{\min\left(\sum_{i,j \in V_C} w(i,j), \sum_{i,j \in V'_C} w(i,j)\right)}$$

or, in plain words, how porous the min cut is divided by how connected the least connected side of the cut is.

Cheeger's inequality

Let λ_2 be the second smallest eigenvalue of the Laplacian matrix, then :

$$\frac{\lambda_2}{2} \leq \phi \leq \sqrt{2\lambda_2}$$

In other words, there is a direct link between the spectral world and clustering problems on graphs.

Clustering and portfolio creation

Arroyo et al., 2021 uses graph spectral theory to control diversification.
They use HCS, and the latter the cut arrives, the smaller the weight on the assets in the subgraphs.

Sort of an extension to K-means clustering

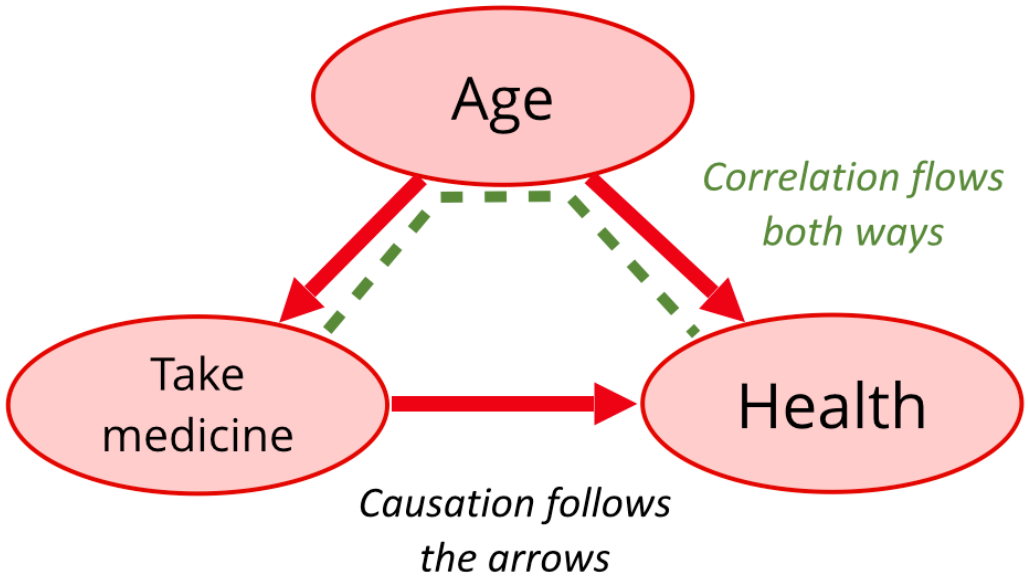
Any clustering in euclidean space can be reframed as a graph clustering problem by taking the complete graph where weights are the euclidean distance between nodes.
However, graph clustering also works on non euclidean relationships.

Principle

Finding causal links between variables by finding confounding variables and finding the true origin of correlation (Pearl, 2009).

DAG

In most of causal frameworks, the variables are set in Directed Acyclic Graphs (DAG). Techniques for causal discovery include Independent Component Analysis (ICA by Shimizu et al., 2006), DyNOTEARS (Pamfil et al., 2020) or normalising flows (Wehenkel et Louppe, 2021), which are all graph discovery algorithms.



Some issues remain

- Generating graphs ? (Large documentation in statistics)
- Large graphs lead to high computational complexity
- Presence of noise in weight functions
- Potentially unknown weight functions
- Need to separate the inner dynamic of nodes from the global dynamic of graphs

Why not use some Deep Learning ?

Perceptron

A perceptron is a linear regression followed by an activation function.

$$y = f(WX + b)$$

- X is the input vector.
- f is a non-linear function (activation function).
- W is a weight matrix (to be learned).
- b is the bias vector (to be learned).

As a generalisation, one can replace the non-linear function by any parametric differentiable function of a vector X.

Multi Layer Perceptron (MLP)

A MLP is a network combining several layers composed of several perceptrons.

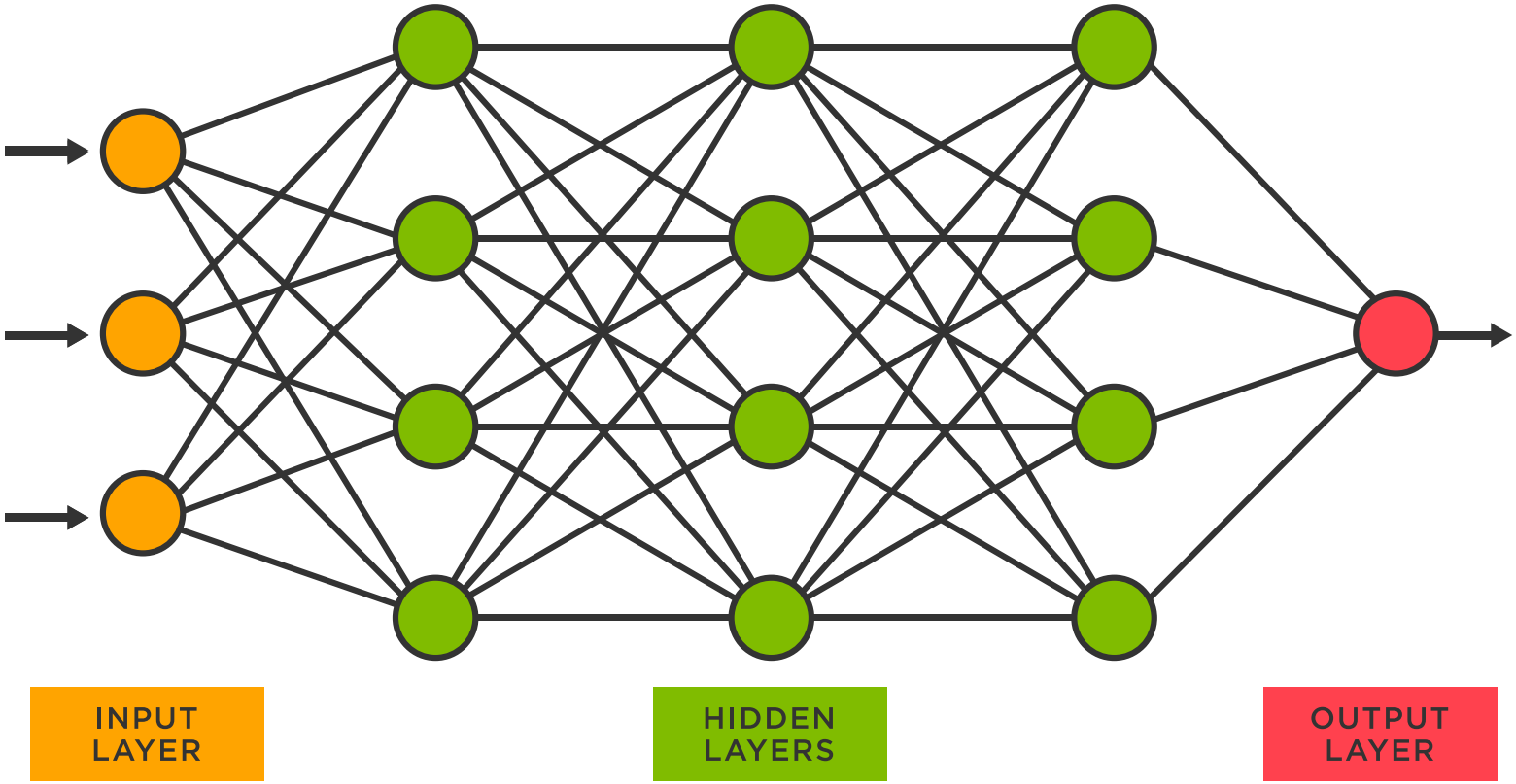
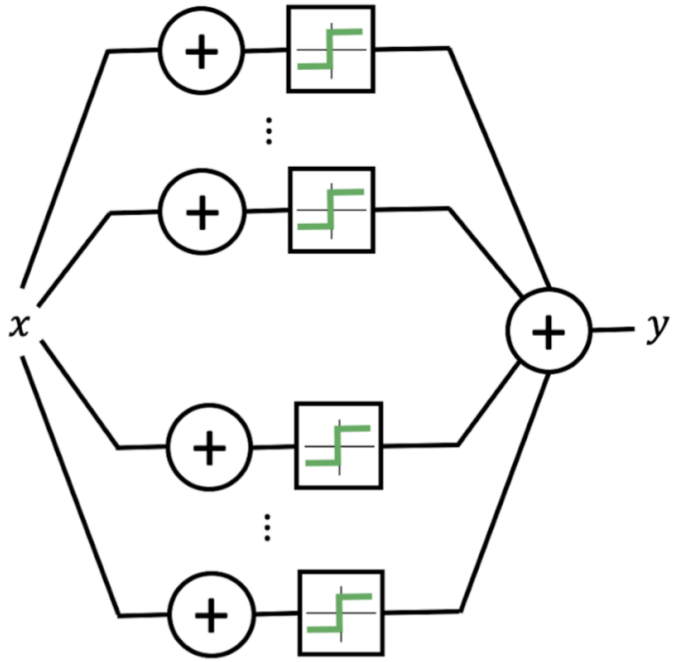
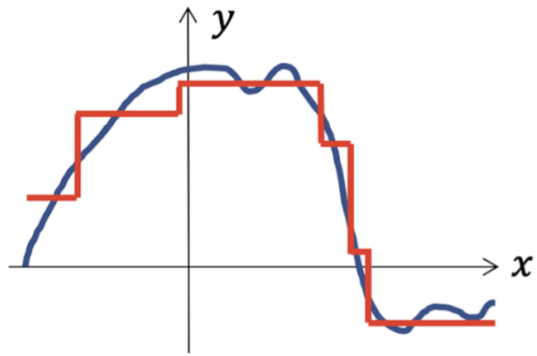


Figure – A multi layer perceptron of hidden depth 3 and layer size 4. [3]

3. https://www.tibco.com/reference-center/what-is-a-neural-network

MLP are universal approximators



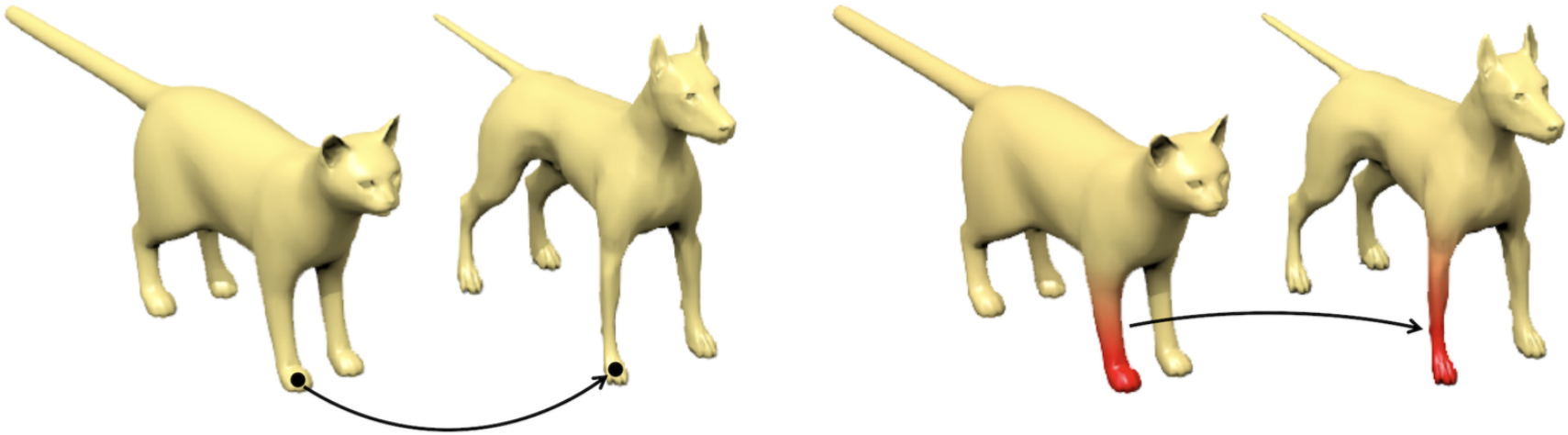


Main idea

We must dissociate structural elements of complex learning problems to focus on transformation invariant properties

Further reading

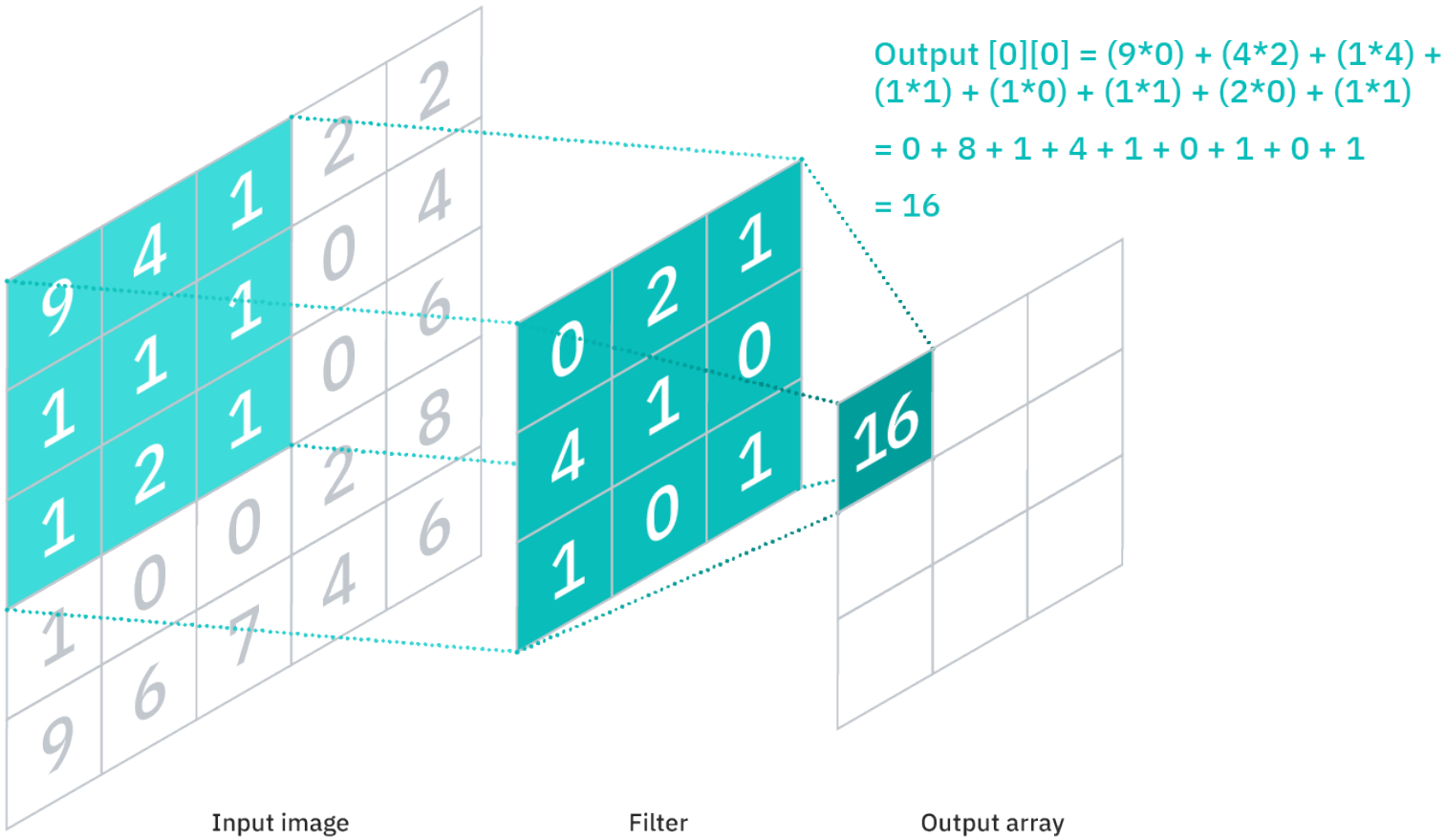
Most of this section is inspired by Bronstein et al., 2021



The Euclidean case : generic convolution

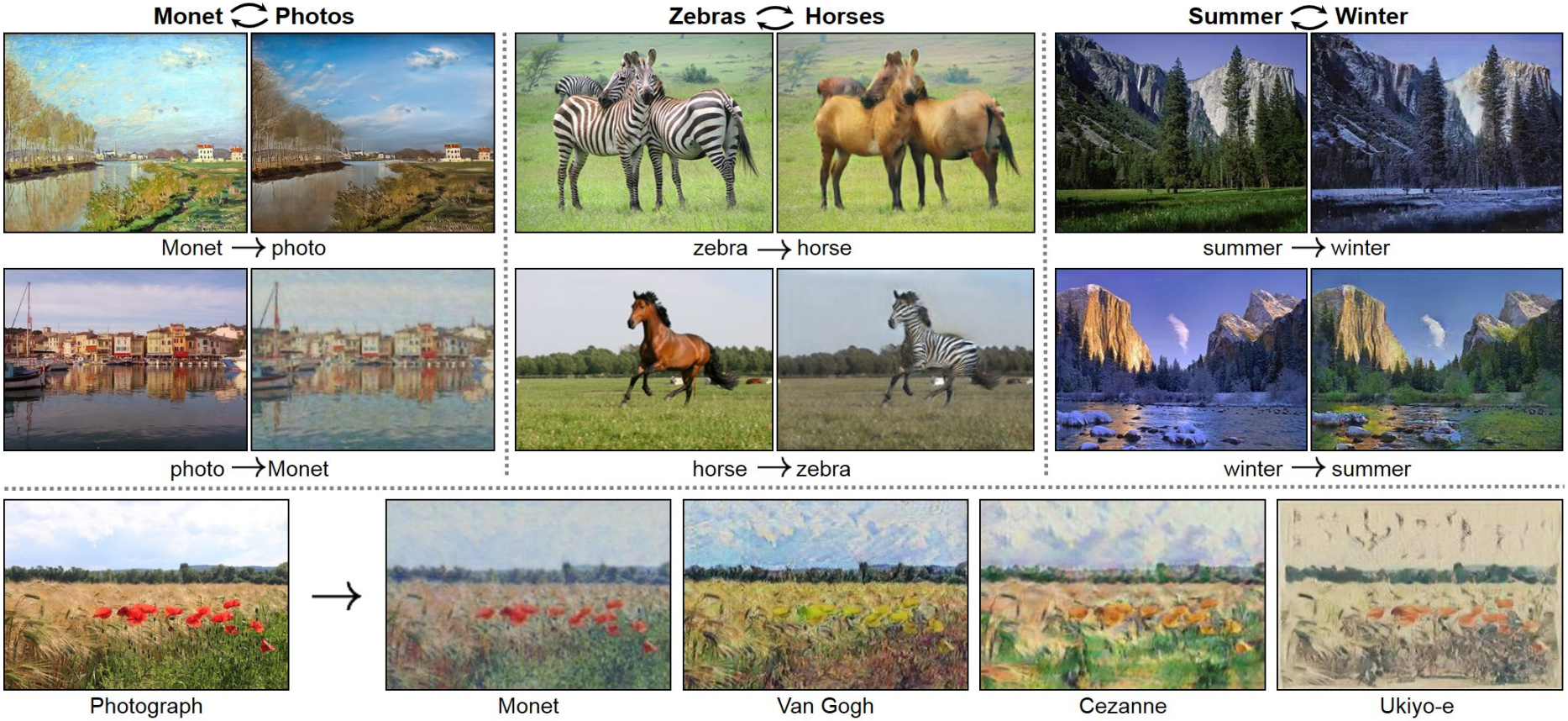
The learned value at one pixel depends on its neighbourhood.
In Torch : **Conv1D**, **Conv2D** and **Conv3D** [a]

a. picture from https://www.ibm.com/cloud/learn/convolutional-neural-networks



Convolution in image, text and video Deep Learning

Yields incredible results on image based learning (see https://junyanz.github.io/CycleGAN/ for instance)



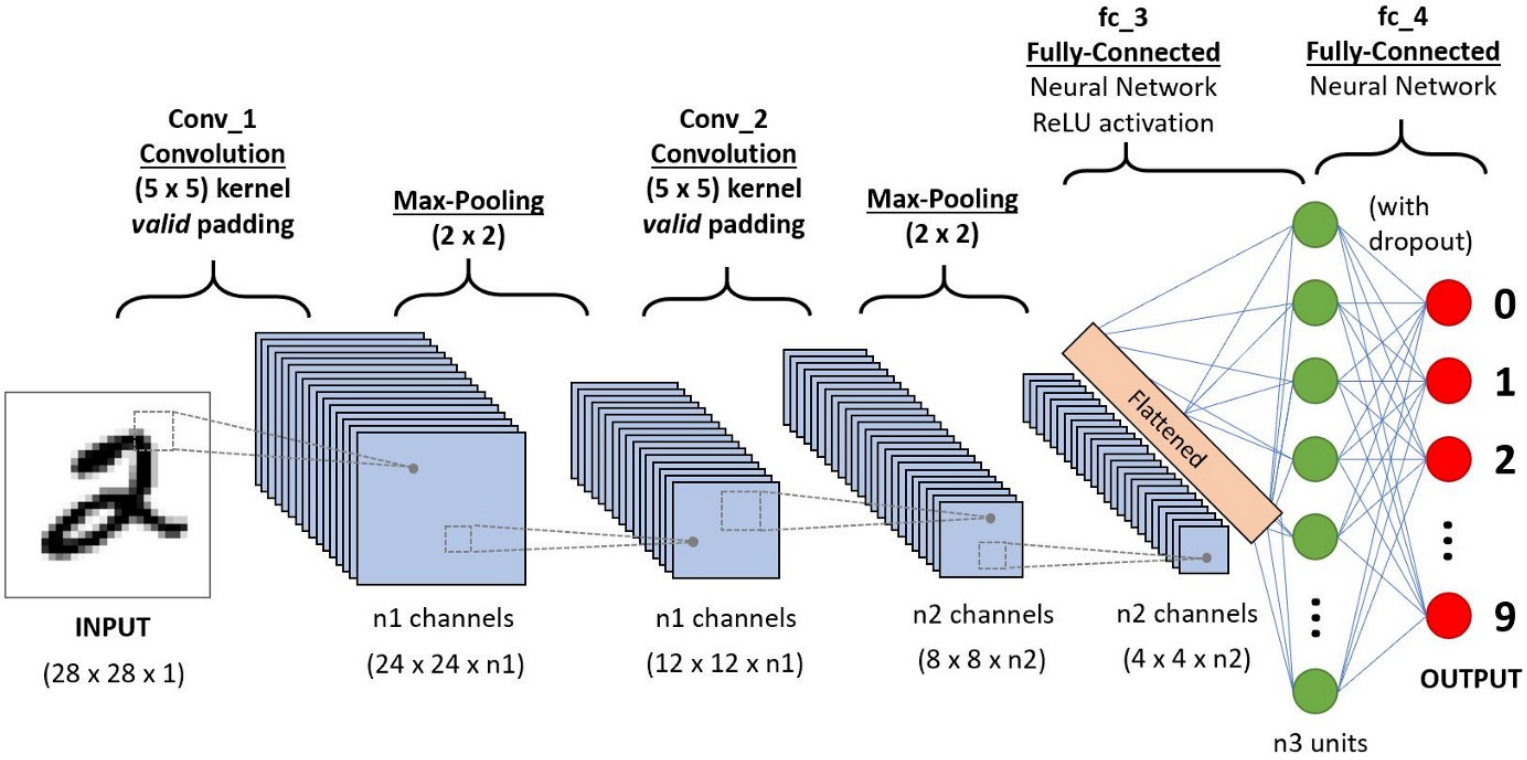


Figure – A classic deep architecture with convolutional layers for image classification (image from https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53)

Graphs and Image Recognition

Convolution is just a special case of graph diffusion !

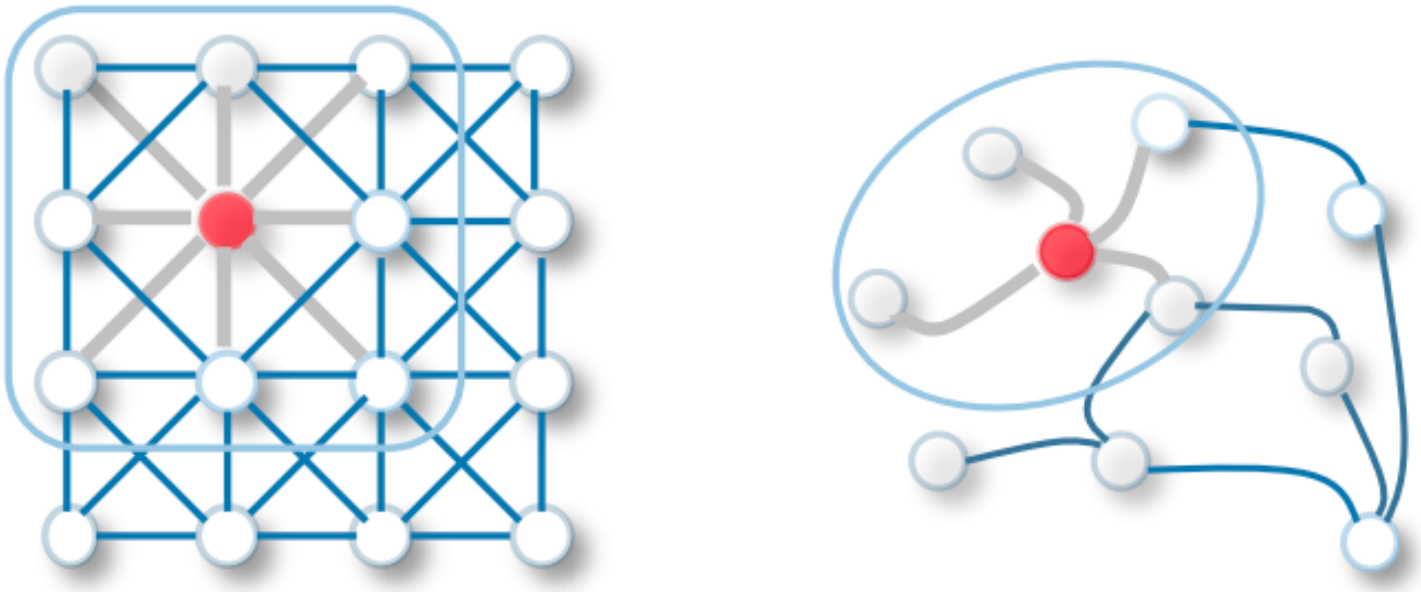


Figure – grid convolution vs graph convolution.

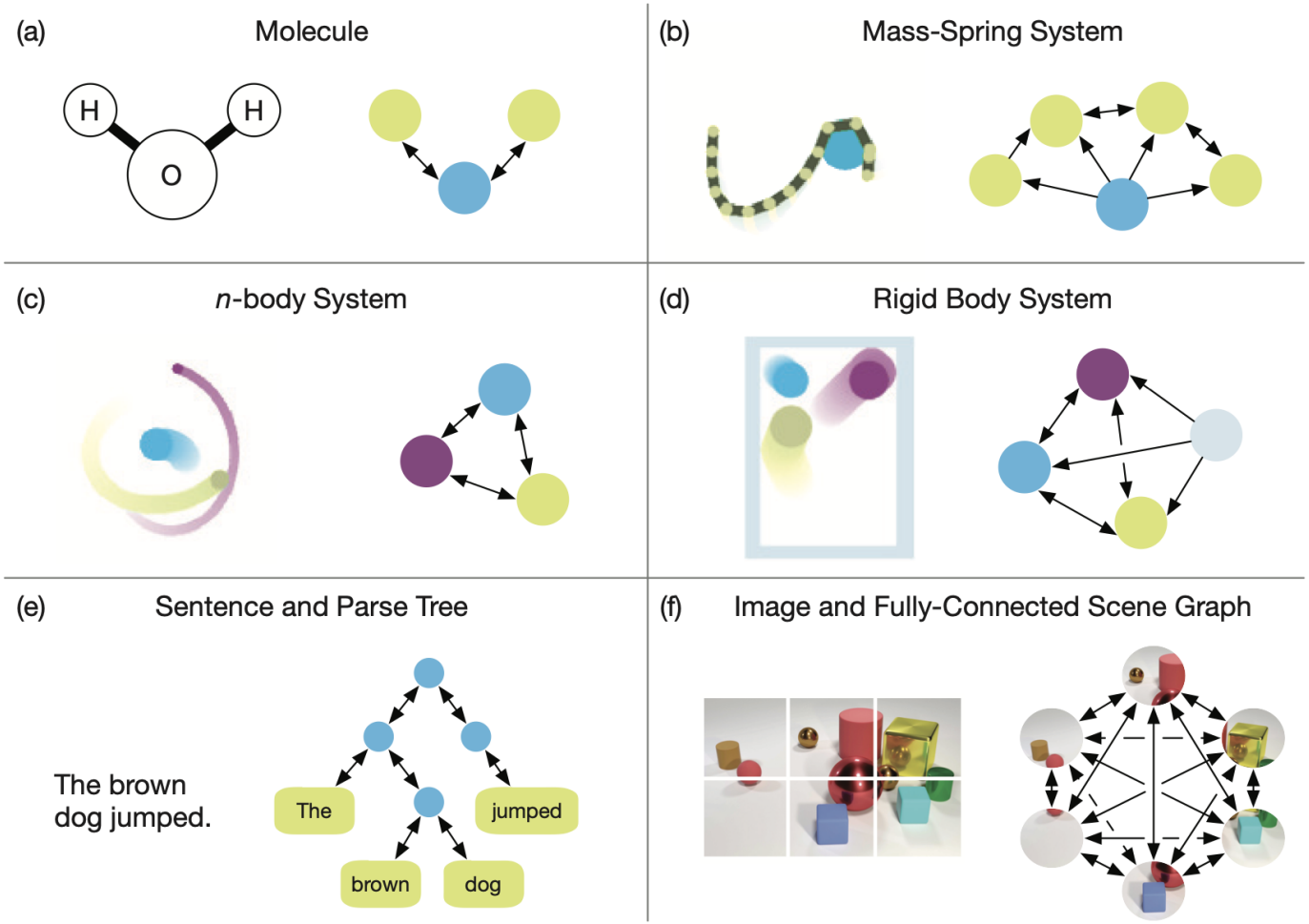


Figure – Different graph representations. (a) A molecule from Duvenaud et al., 2015. (b) A mass-spring system, (c) A n-body system and (d) a rigid body system fromBattaglia et al., 2016 ; Chang et al., 2017. (e) A sentence from Socher et al., 2013. A fully connected graph could be used as in Vaswani et al., 2017. (f) An image Santoro et al., 2017 ; Wang et al., 2018.

Graphs invariance

Graph layers have the following advantage :

- they do not limit the number of nodes, only the number of features per nodes ;
- they limit the convolution to pertinent relationships (in the case of the n-body problem) ;
- they are permutation invariant in the ordering of neighbours ;
- isomorphic graphs will behave the same and give the same tensor as a result, up to permutations in one dimension.
-

Graphs are not images

Since graphs are a generic structure, context matters and influences the architecture of the network, moreso than images.

General Framework

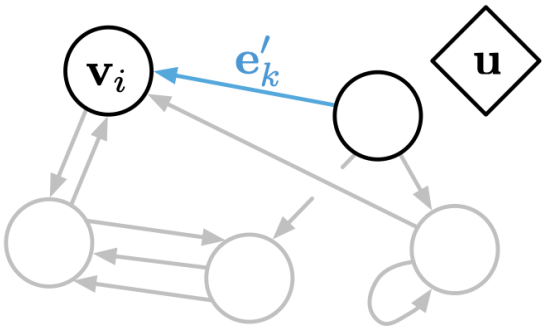
Battaglia et al., 2016 presents a very general framework for graph learning.

Several states

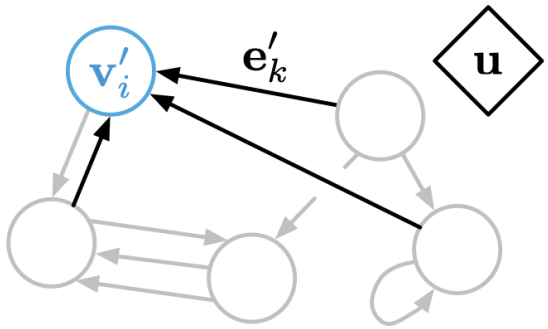
A graph has three main state functions :

- a node state function ;
- an edge state function ;
- a global graph state function.

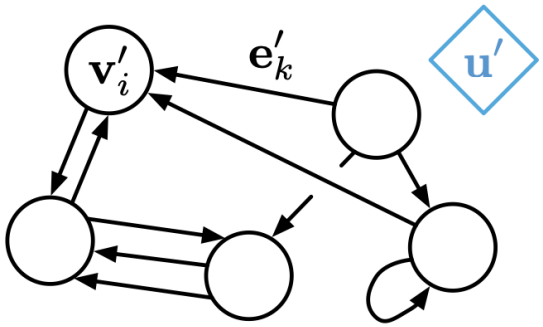
Each convolution needs to update these functions at every epoch.



(a) Edge update



(b) Node update



(c) Global update

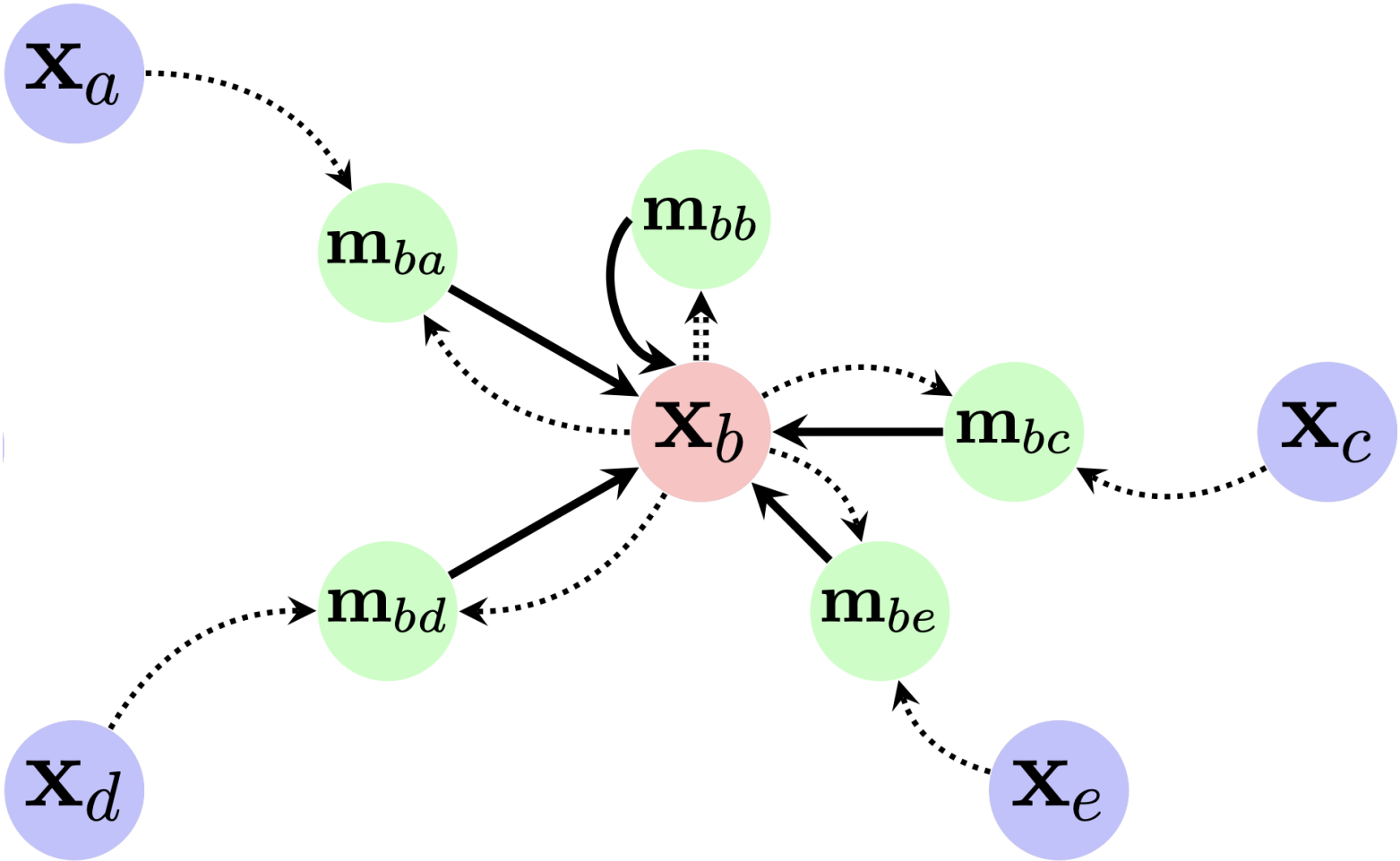
Message Passing

A special case of the previous framework where transmission is only node to node.
In practice, it is the main framework for graph layers.

Mechanism

Three steps :

- Message building : transforms the feature into a message ;
- Transmission : Propagates the message towards the neighbours ;
- Aggregation : each node agregrates its received messages to update their features.

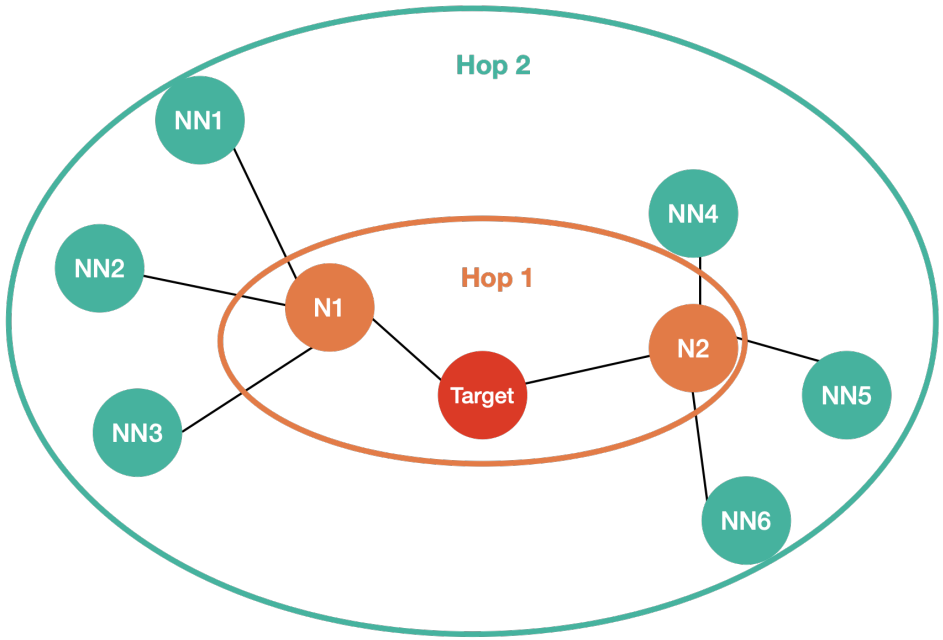


Message-passing

Graph Sage

Averaging of neighbouring node/edge values, inspired by traditional convolutional layers (Hamilton et al., 2018).

$$y_i = W_1 x_i + W_2 \sum_{j \in \mathcal{N}(i)} x_j$$



GCN

Layer inspired by Laplacian based diffusion. Approximation of the spectral convolution of a linear filter and the features (Kipf et Welling, 2017).

Definition

Let $\tilde{A} = A + I_N$ and $\tilde{D} = D + I_N$,

$$Y = \sigma\left(\mathbf{\Theta} * X\right) \sim \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}X\mathbf{\Theta}\right)$$

GCN construction

Spectral convolution of features X with filter g_θ

By definition,

$$(g_\theta * X) = U \Theta U^\top X$$

where U is the eigenvector matrix of the Laplacian. To avoid eigenvalue computation, we approach the filter g_θ with a Chebychev expansion over the Laplacian. We normalise the Laplacian to guaranty stability :

$$(g_\theta * X) \approx \sum_{k=\mathbf{0}}^{K} \theta_k T_k \left(\frac{2}{\lambda_{\mathbf{max}}} \tilde{L} - I_N \right) x$$

Which gives, taken at the first order and knowing that $\lambda_{\mathbf{max}} \approx 2$:

$$(g_\theta * X) \approx \theta_{\mathbf{0}} X + \theta_{\mathbf{1}} \left(\tilde{L} - I_N \right) X$$

Then we suppose $\theta_{\mathbf{0}} = \theta_{\mathbf{1}} = \text{diag}^{-\mathbf{1}} \Theta$ and we renormalise the expression by adding self loops in the graph, naming $\tilde{A}$ the new adjacency matrix and $\tilde{D}$ its degree matrix. This gives :

$$(g_\theta * X) \approx \tilde{D}^{-\mathbf{1/2}} \tilde{A} \tilde{D}^{-\mathbf{1/2}} X \Theta$$

Graph Attention

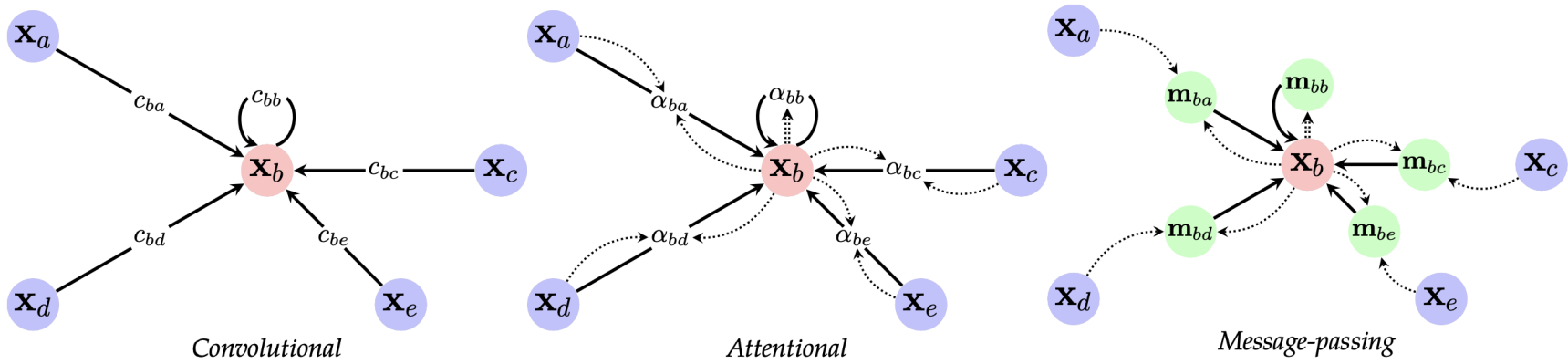
Each node will look at its neighbour with a a different attention, represented by a coefficient α (Veličković et al., 2018).

Definition

$$\alpha_{i,j} = \text{softmax}\left(\text{LeakyReLU}\left(\boldsymbol{a}'\left[\boldsymbol{W}X_i||\boldsymbol{W}X_j\right]\right)\right)$$

$$Y = \sigma\left(\sum_{j\in\mathcal{N}_i\cup i}\alpha_{i,j}\boldsymbol{W}X_j\right)$$

The three flavours of graph neural networks



Other layers

- ChebNet : extension of GCN, Chebyshev expansion at order $K > 1$.
- CayleyNet : other Laplacian expansion using Cayley polynomials.
- Node2Vec : older layer using a vector representation of nodes.

The Smoothing Conundrum

Because of the links with heat diffusion, too many iterations of a GCN like layer will dilute signals and lead to a averaging over the entire graph.

Solutions

- Change the aggregation function (concatenation instead of averaging).
- Step away from message passing (GC-LSTM architecture).
- Use only sparse graphs.

Pooling nodes together

Just like images, it is useful to reduce the graph by :

- max-pooling ;
- min pooling ;
- average pooling.

Pooling layers use predefined sets of nodes (such as sector information for assets), or cluster the graph themselves.

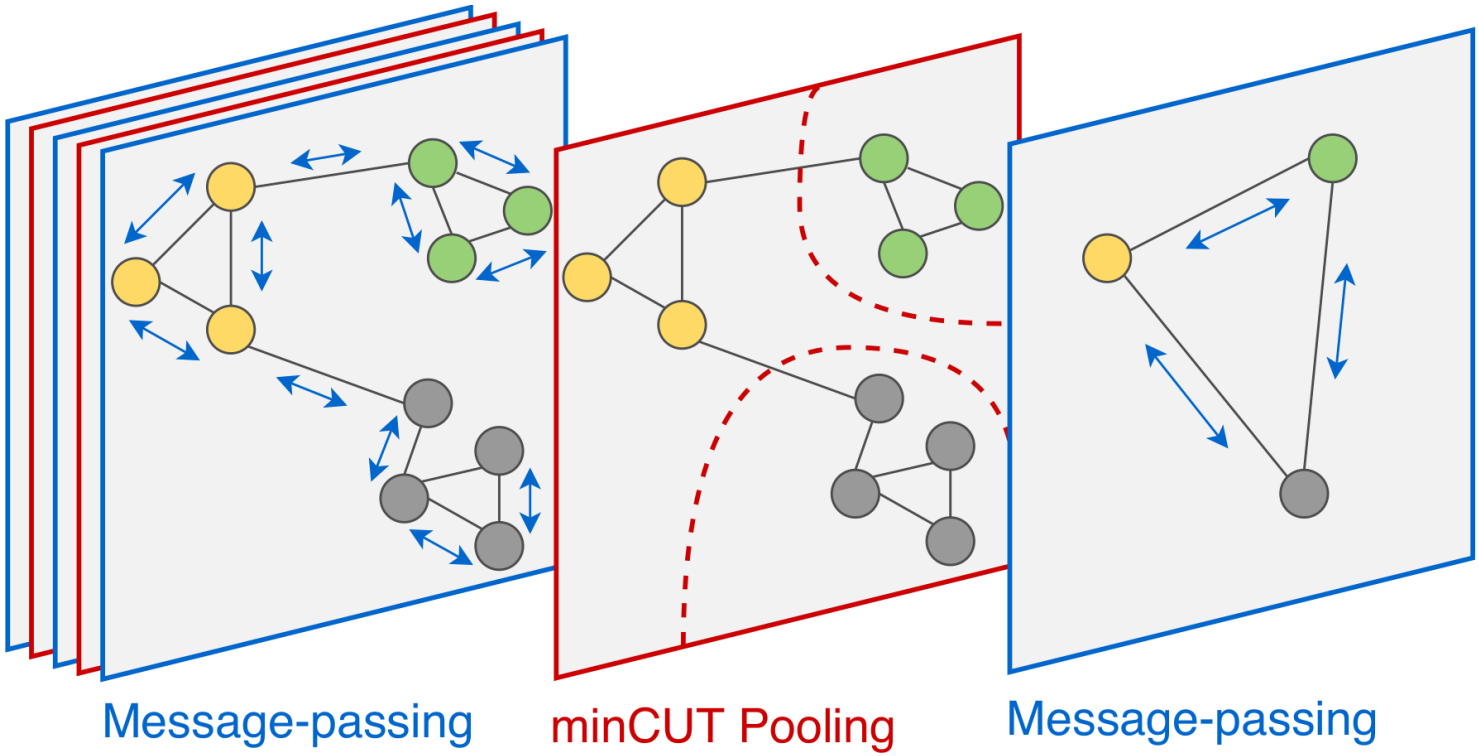


Figure – The min-cut pooling layer from Bianchi et al., 2020 (image from https://danielegrattarola.github.io/posts/2019-07-25/mincut-pooling.html)

Normalisation Layers

- Batch norm normalises according to predefined batches of nodes ;
- Pairnorm combats over smoothing in GCN ;
- Graph size norm controls the values of each nodes compared to the size of the graph ;
- etc

Aggregation Mechanisms

Some papers propose pure message passing with custom aggregation mechanisms which include :

- averaging all messages ;
- taking the max ;
- taking the min ;
- summing ;
- multiplying ;
- concatenating ;
- etc

LSTM and GNN

Packages like Pytorch Geometric Temporal (https://pytorch-geometric-temporal.readthedocs.io/en/latest/index.html) try to combine graph learning with lstms.

Arroyo, A., Scalzo, B., Stankovic, L., & Mandic, D. P. (2021). Dynamic Portfolio Cuts: A Spectral Approach to Graph-Theoretic Diversification [arXiv:2106.03417 [eess, q-fin]].
Comment: 5 pages, 3 Figures, 2 Tables.

Battaglia, P. W., Pascanu, R., Lai, M., Rezende, D., & Kavukcuoglu, K. (2016). *Interaction Networks for Learning about Objects, Relations and Physics* (rapp. tech. arXiv:1612.00222) [arXiv:1612.00222 [cs] type: article]. arXiv
Comment: Published in NIPS 2016.

Bianchi, F. M., Grattarola, D., & Alippi, C. (2020). *Spectral Clustering with Graph Neural Networks for Graph Pooling* (rapp. tech. arXiv:1907.00481) [arXiv:1907.00481 [cs, stat] type: article]. arXiv.

Bronstein, M. M., Bruna, J., Cohen, T., & Veličković, P. (2021). Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges [Number: arXiv:2104.13478 arXiv:2104.13478 [cs, stat]].
Comment: 156 pages. Work in progress – comments welcome!

Chang, M. B., Ullman, T., Torralba, A., & Tenenbaum, J. B. (2017). *A Compositional Object-Based Approach to Learning Physical Dynamics* (rapp. tech. arXiv:1612.00341) [arXiv:1612.00341 [cs] type: article]. arXiv
Comment: Published as a conference paper for ICLR 2017. 15 pages, 6 figures.

Duvenaud, D., Maclaurin, D., Aguilera-Iparraguirre, J., Gómez-Bombarelli, R., Hirzel, T., Aspuru-Guzik, A., & Adams, R. P. (2015). *Convolutional Networks on Graphs for Learning Molecular Fingerprints* (rapp. tech. arXiv:1509.09292) [arXiv:1509.09292 [cs, stat] type: article]. arXiv
Comment: 9 pages, 5 figures. To appear in Neural Information Processing Systems (NIPS).

Hamilton, W. L., Ying, R., & Leskovec, J. (2018). Inductive Representation Learning on Large Graphs [Number: arXiv:1706.02216 arXiv:1706.02216 [cs, stat]].
Comment: Published in NIPS 2017; version with full appendix and minor corrections.

Kaya, H. (2013). Eccentricity in Asset Management.

Kipf, T. N., & Welling, M. (2017). Semi-Supervised Classification with Graph Convolutional Networks [arXiv: 1609.02907]. *arXiv:1609.02907 [cs, stat]*. Récupérée 10 mars 2022, à partir de http://arxiv.org/abs/1609.02907
Comment: Published as a conference paper at ICLR 2017

Olmo, J. (2021). Optimal portfolio allocation and asset centrality revisited [Publisher: Routledge _eprint: https://doi.org/10.1080/14697688.2021.1937298]. *Quantitative Finance*, *21*(9), 1475-1490.

Pamfil, R., Sriwattanaworachai, N., Desai, S., Pilgerstorfer, P., Beaumont, P., Georgatzis, K., & Aragam, B. (2020). *DYNOTEARS: Structure Learning from Time-Series Data* (rapp. tech. arXiv:2002.00498) [arXiv:2002.00498 [cs, stat] type: article]. arXiv
Comment: 23 pages, 13 figures, accepted to AISTATS 2020, corrected version.

Pearl, J. (2009). *Causality* (2e éd.). Cambridge University Press.

Prado, M. L. d. (2016). Building Diversified Portfolios that Outperform Out of Sample [Publisher: Institutional Investor Journals Umbrella]. *The Journal of Portfolio Management*, *42*(4), 59-69.

Santoro, A., Raposo, D., Barrett, D. G. T., Malinowski, M., Pascanu, R., Battaglia, P., & Lillicrap, T. (2017). *A simple neural network module for relational reasoning* (rapp. tech. arXiv:1706.01427) [arXiv:1706.01427 [cs] type: article]. arXiv.

Shimizu, S., Hyvärinen, A., Hoyer, P. O., & Kano, Y. (2006). Finding a causal ordering via independent component analysis. *Computational Statistics & Data Analysis*, *50*(11), 3278-3293.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., & Potts, C. (2013). Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 1631-1642. Récupérée 22 novembre 2022, à partir de https://aclanthology.org/D13-1170

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). *Attention Is All You Need* (rapp. tech. arXiv:1706.03762) [arXiv:1706.03762 [cs] type: article]. arXiv
Comment: 15 pages, 5 figures.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). Graph Attention Networks [Number: arXiv:1710.10903 arXiv:1710.10903 [cs, stat]].
Comment: To appear at ICLR 2018. 12 pages, 2 figures.

Výrost, T., Lyócsa, Š., & Baumöhl, E. (2019). Network-based asset allocation strategies. *The North American Journal of Economics and Finance*, *47*, 516-536.

Wang, X., Girshick, R., Gupta, A., & He, K. (2018). *Non-local Neural Networks* (rapp. tech. arXiv:1711.07971) [arXiv:1711.07971 [cs] type: article]. arXiv
Comment: CVPR 2018, code is available at: https://github.com/facebookresearch/video-nonlocal-net.

Wehenkel, A., & Louppe, G. (2021). *Graphical Normalizing Flows* (rapp. tech. arXiv:2006.02548) [arXiv:2006.02548 [cs, stat] type: article]. arXiv.