Theoretical Limitations of Graph Neural Networks

Ahmadreza Yousefkhani,

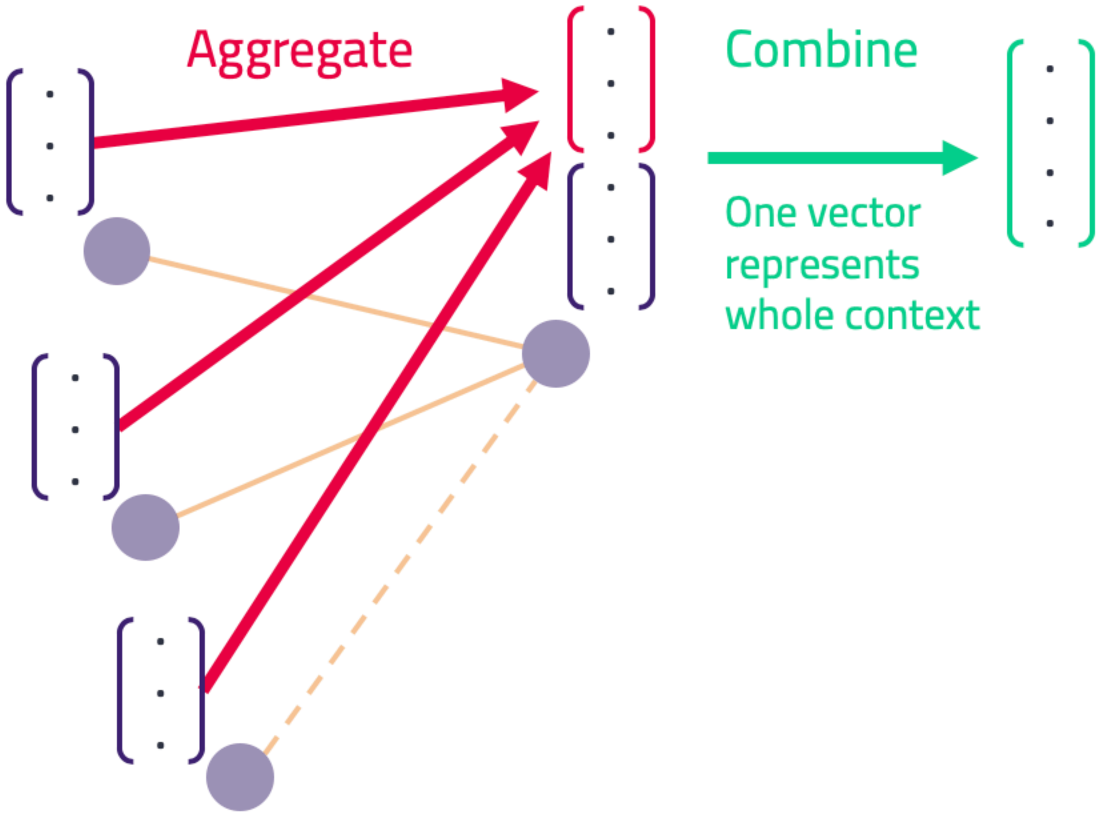
Deep Neural Network seminar, Spring 2021

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

GNN Theoretical Limitations

- Representation learning of graphs
- What can they represent? What kind of problems they solve?
- Better strategies than intuition and heuristics for designing GNNs?

Message Passing Networks (MPNN)

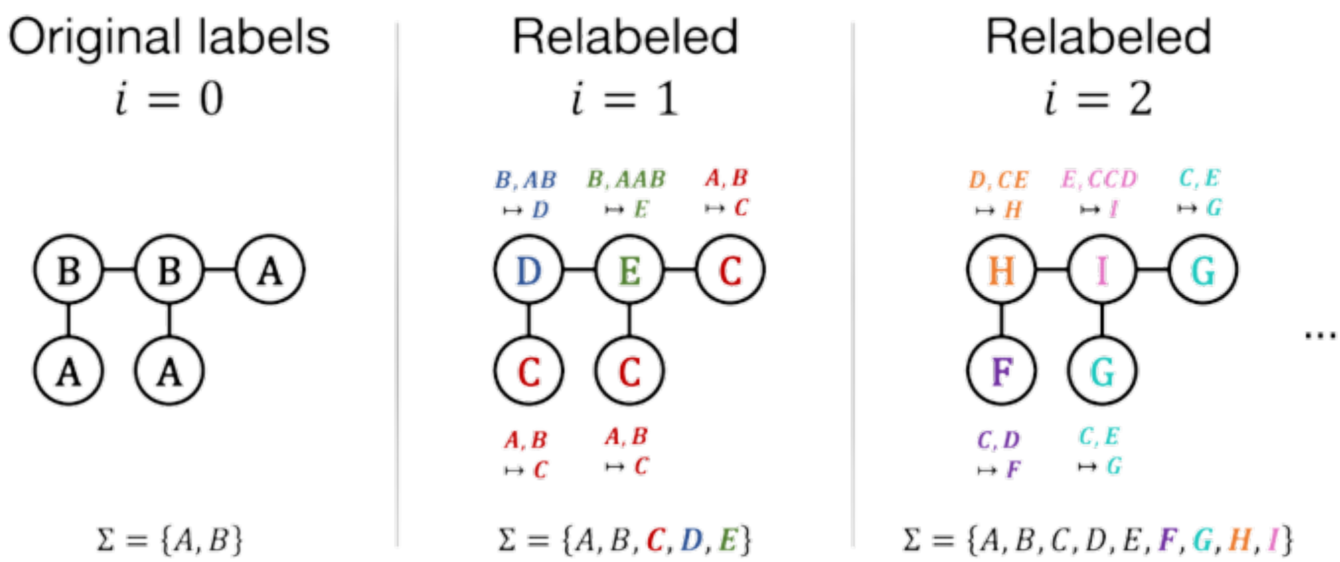


Review on Message Passing Networks

Consider h_v^k to be the feature vector of node v at the k-th iteration/layer:

$$a_v^{(k)} = \mathrm{AGGREGATE}^{(k)}(\{h_u^{(k-1)} : u \in \mathcal{N}(v)\}),$$
$$h_v^{(k)} = \mathrm{COMBINE}^{(k)}(h_v^{(k-1)}, a_v^{(k)})$$
$$h_g = \mathrm{READOUT}(\{h_v^{(K)} | v \in G\})$$

Weisfeiler-Lehman test of isomorphism



Asks whether two graphs are topologically identical

Image from Kriege et al.

WL test and expressivity of MPNNs

Represent the set of feature vectors of a given node's neighbors as a *multiset*

A maximally powerful GNN would *never* map two different neighborhoods, *i.e.*, multisets of feature vectors, to the same representation.

- Injective aggregation scheme
- Isomorphic graphs have to be mapped to the same representation

Injective MPNN

When every function in between is injective, the output function is injective as well

- How can we build such model?

Tools for making GIN

Consider a multiset $\mathcal{X}$, and ϵ to be any number

Assume a function $g(c, X)$, where $c, X \in \mathcal{X}$, which maps each pair of inputs to a unique number

There exists an f, such that for some function φ, g can be decomposed as follows:

$$g(c, X) = \varphi((1 + \epsilon) f(c) + \Sigma_{x \in X} f(x))$$

Xu, Keyulu, et al. "How powerful are graph neural networks?." *arXiv preprint arXiv:1810.00826* (2018).

GIN

- Model the $f^{(k+1)} \circ \phi^{(k)}$ with one MLP

$$h_v^{(k)} = \mathrm{MLP}^{(k)}((1+\epsilon^{(k)}).h_v^{(k-1)} + \Sigma_{u \in \mathcal{N}(v)} h_u^{(k-1)})$$

- ϵ can be a learnable parameter or a scalar

- READOUT:
  - More iterations gives better representational power
  - But less generalization
  - So GIN concatenates the embedding (information) from all layers

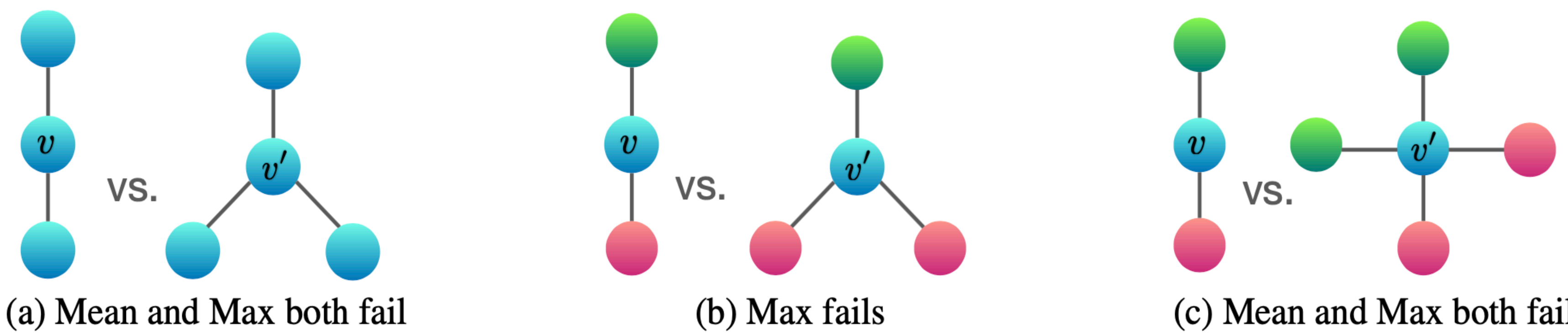
Xu, Keyulu, et al. "How powerful are graph neural networks?." *arXiv preprint arXiv:1810.00826* (2018).

What makes GIN powerful?

- 1-layer perceptron instead of MLP
- Linear mapping can map two multisets to the same representations
- Unlike models using MLPs, the 1-layer perceptron (even with the bias term) is *not a universal approximator* of multiset functions.

Use an MLP with more than 1 layer

What makes GIN powerful?

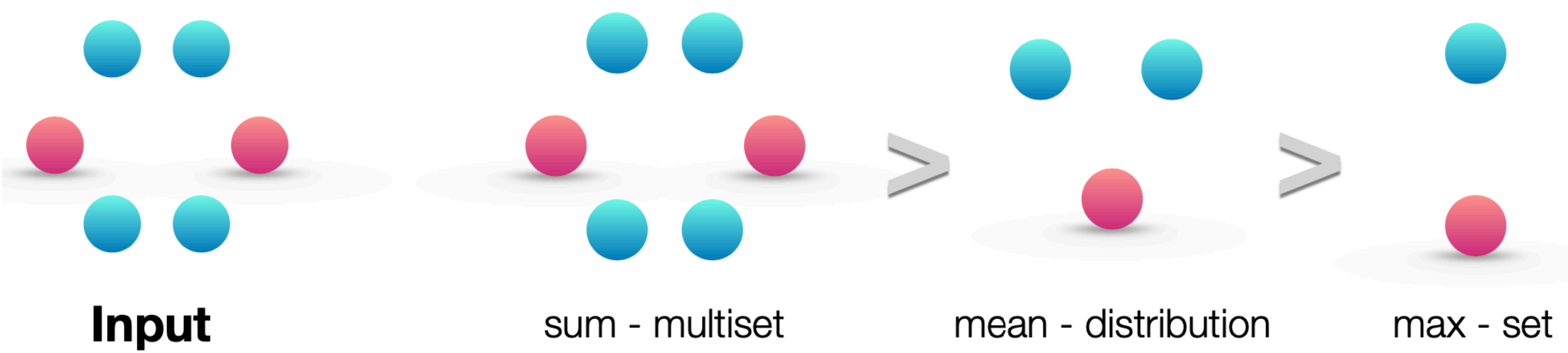


- The mean captures the *distribution* (proportions) of elements in a multiset, but not the *exact* multiset

- Max can capture the *skeleton*

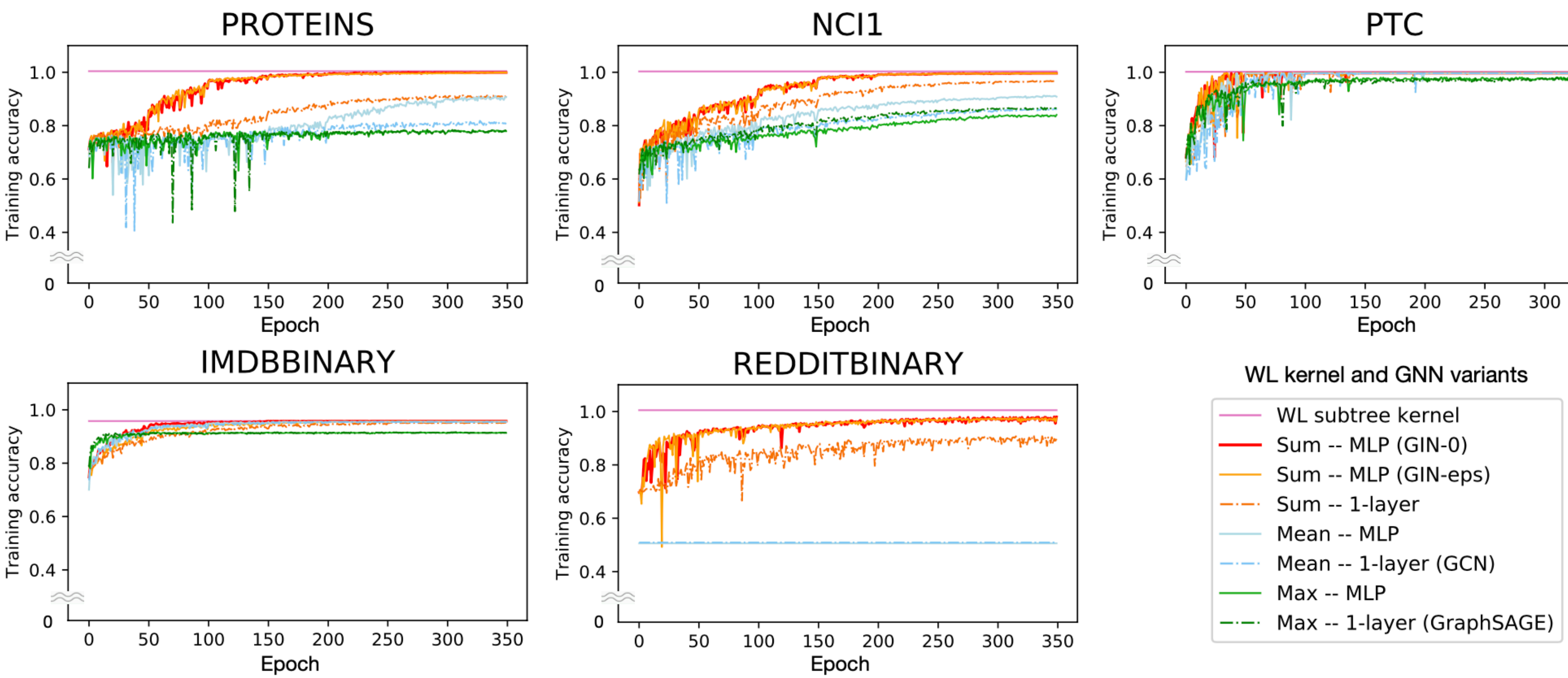
What makes GIN powerful?

- Instead of sum in $h(X) = \Sigma_{x \in X} f(x)$, what if we use mean or max?



ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

GIN-results


PROTEINS
NCI1
PTC
IMDBBINARY
REDDITBINARY
Training accuracy
Epoch
1.0
0.8
0.6
0.4
0
0 50 100 150 200 250 300 350
WL kernel and GNN variants
WL subtree kernel
Sum -- MLP (GIN-0)
Sum -- MLP (GIN-eps)
Sum -- 1-layer
Mean -- MLP
Mean -- 1-layer (GCN)
Max -- MLP
Max -- 1-layer (GraphSAGE)

WL test and expressivity of GNNs

- GNNs are *at most* as powerful as the WL test in distinguishing graph structures
- Conditions on aggregation and the readout function to be as powerful as WL test
- Can we have the universal approximation theorem for GNNs?
- Can we compare different GNNs from their architecture?

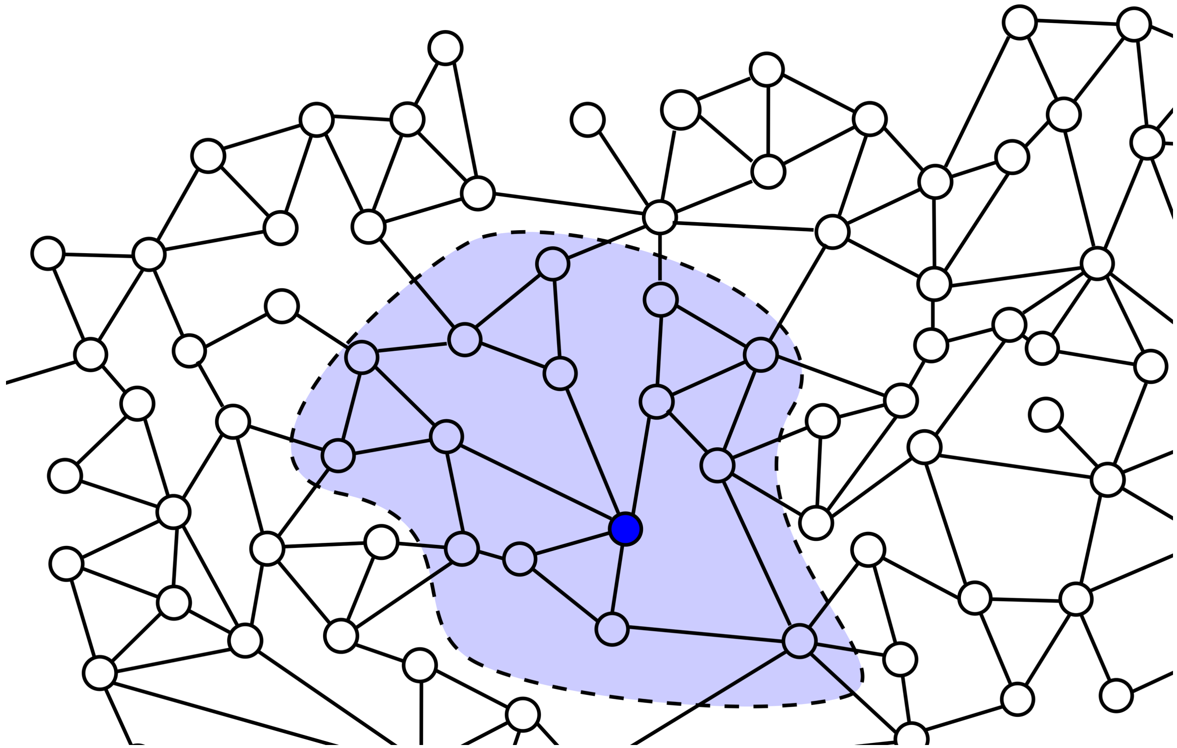
Graph Classification in Theory

- Training a graph classifier = finding the properties shared in one class
  - then deciding whether new graphs abide to said learned properties

- If the problem cannot be learned by a GNN of a certain depth
  - No matter what learning algorithm you use
  - The problem is not solvable!

Find the lower bounds!

Distributed Computing



LOCAL

- computation starts simultaneously and unfolds in synchronous rounds

- 3 possibilities in each round
  - each node receives a string of *unbounded* size from its incoming neighbors
  - each node updates its internal state by performing some local computation
  - Each node sends a string to every one of its outgoing neighbors

Turing Completeness

- δ_g : length of the longest shortest path between any two nodes
- Depth d: number of layers in the network
- Width: largest dimensions of node’s state over all layers and all nodes

Loukas, Andreas. "What graph neural networks cannot learn: depth vs width." *arXiv preprint arXiv:1907.03199* (2019).

Turing Universality

MPNN is Turing universal over connected attributed graphs if:

- each node is uniquely identified
- AGGREGATE and COMBINE are Turing complete for each layer
- the width is unbounded and $d \geq \delta_g$

Loukas, Andreas. "What graph neural networks cannot learn: depth vs width." *arXiv preprint arXiv:1907.03199* (2019).

CONGEST

- Assume we *constraint the number bits in the communication* to be at most b

- If a problem P cannot be solved by CONGEST, it cannot be solved by a MPNN of depth d, and

$$w \leq (b - \log_2 n)/p = \mathcal{O}(b/\log n)$$

Loukas, Andreas. "What graph neural networks cannot learn: depth vs width." *arXiv preprint arXiv:1907.03199* (2019).

K-cycle lower bound

- Finding a K-cycle in a graph
  - undirected graph of k nodes each having exactly two neighbors
- There exists a MPNN of width w and $d = \Omega(\sqrt{n}/w \log n)$ for even $k \geq 4$, and $d = \Omega(n/w \log n)$ for odd $k \geq 5$ which can detect if the input graph contains a K-cycle

Loukas, Andreas. "What graph neural networks cannot learn: depth vs width." *arXiv preprint arXiv:1907.03199* (2019).

Similar to K-Cycle

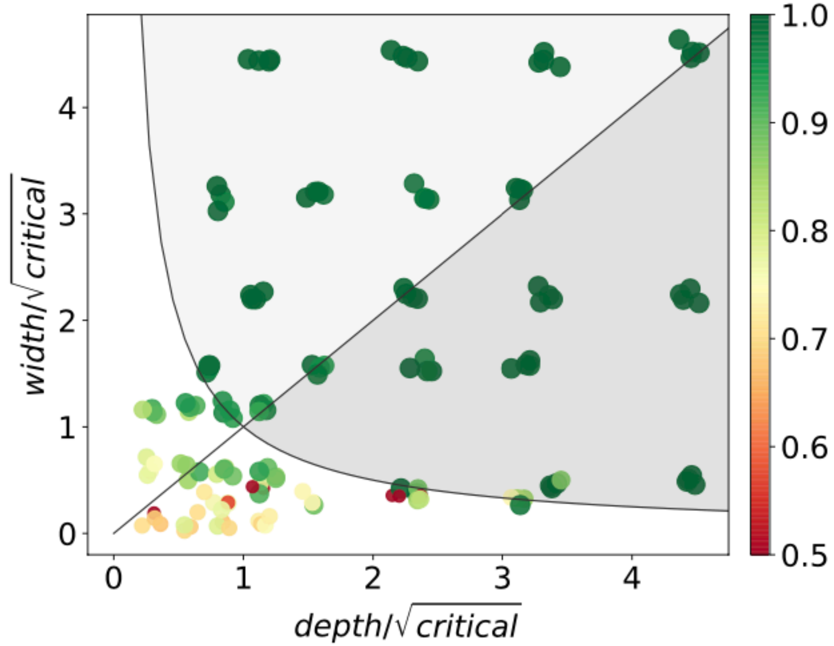
- Similar bounds for:
  - Subgraph detection
  - Subgraph verification
  - Computation problems

- All related to classification!

- Depth and width should exhibit a linear dependence on n, the number of nodes of the input
  - Counter intuitive!
  - Locality

Capacity can be approximated by dw

Capacity in 4-Cycle Classification



dw should pass the critical threshold

Loukas, Andreas. "What graph neural networks cannot learn: depth vs width." *arXiv preprint arXiv:1907.03199* (2019).

Universal GNN

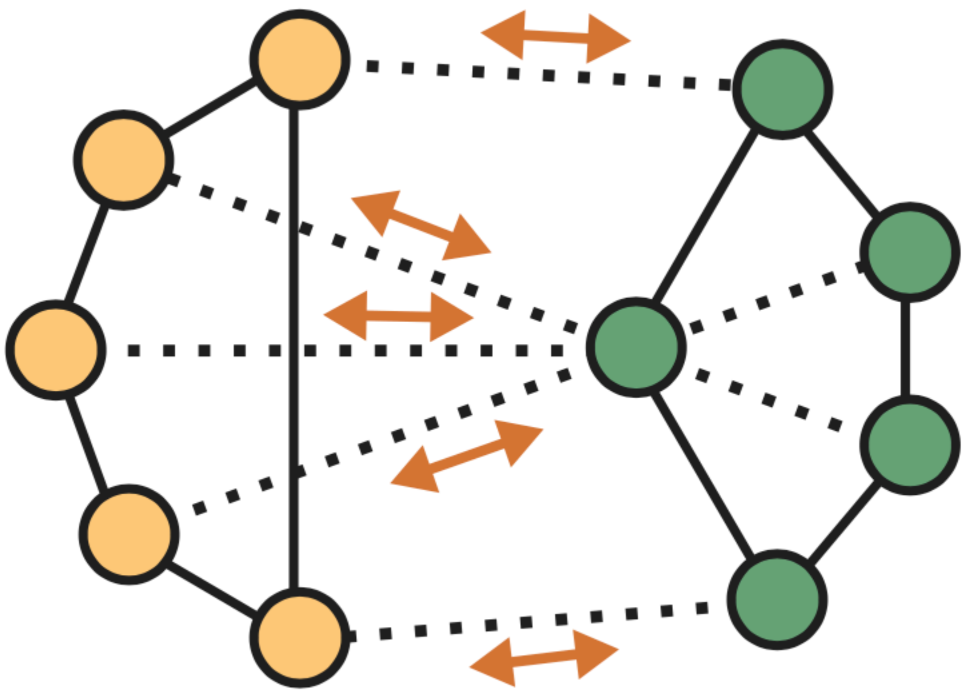
- Enough layers of sufficient expressiveness and width
- Nodes can uniquely distinguish each other
- Turing universality
  - > universal approximation theorem
- Can do the graph isomorphism task (better than 1-WL)

Distinguishing Graphs

- How hard is to distinguish graphs with graph neural networks?

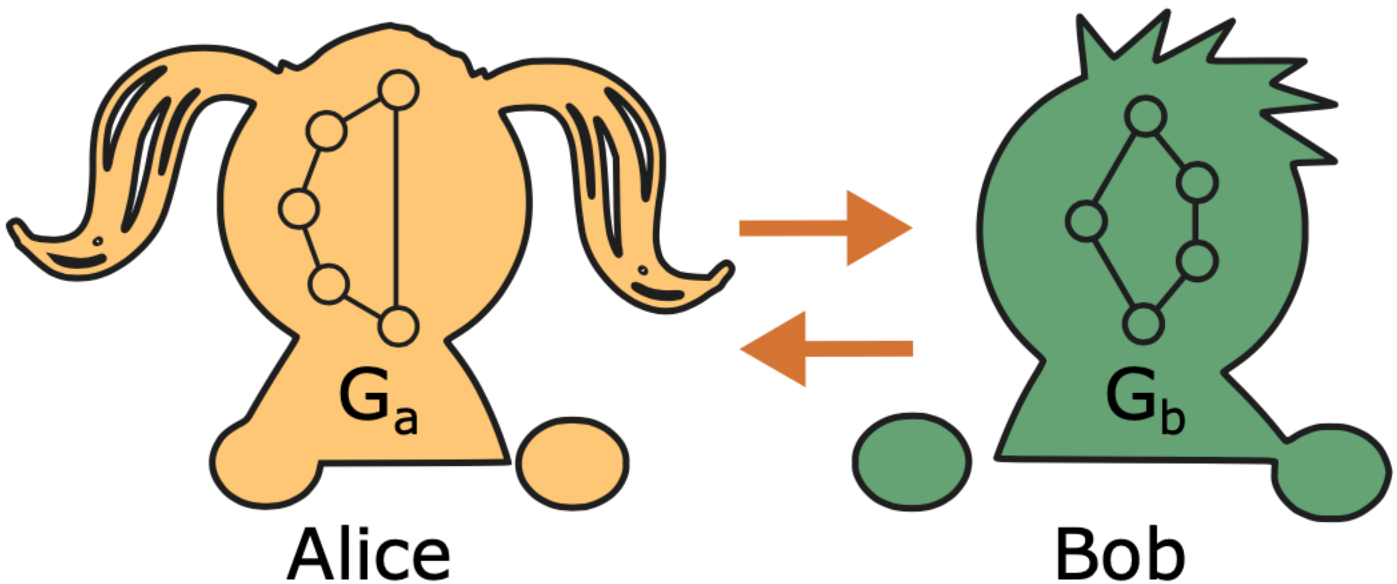
- How much information the nodes of a network can exchange during the forward pass?
  - Communication capacity
  - Generalization of the previous capacity notion

Communication Capacity



the maximal amount of information that can be sent across two subgraphs

Communication Complexity



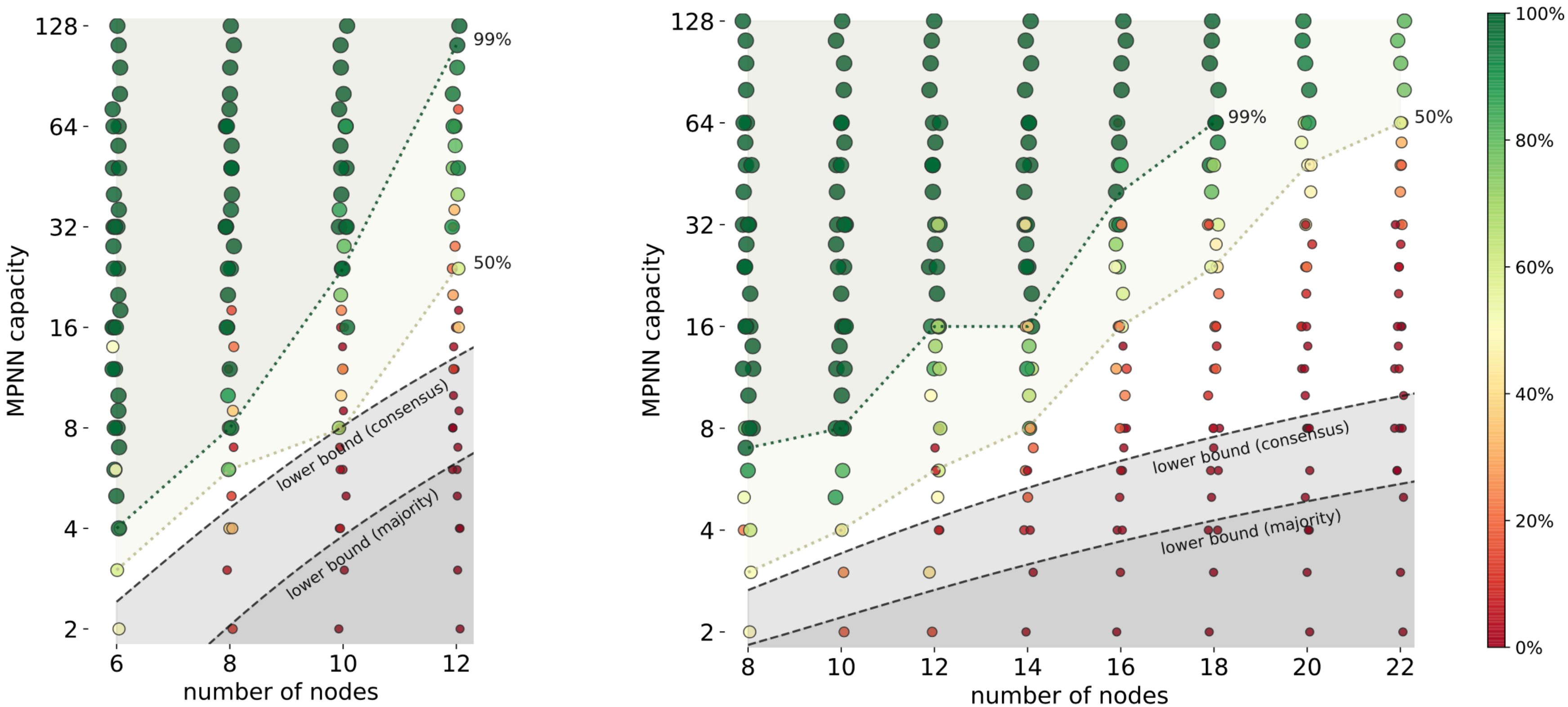
the minimal amount of information needed so that two parties jointly compute a function

Determining Isomorphism Class

- Consider c_g to be the communication capacity of the graph
- For a graph, if $c_g = \Omega(n^2)$, and for a tree if $c_g = \Omega(n)$, isomorphism classes can be learned by the MPNN
- Can be extended to the graphs which are sampled from a distribution by using the expected communication capacity
  - Same bounds

A. Loukas, "How hard is to distinguish graphs with graph neural networks?", NIPS 2020

Determining Isomorphism Class

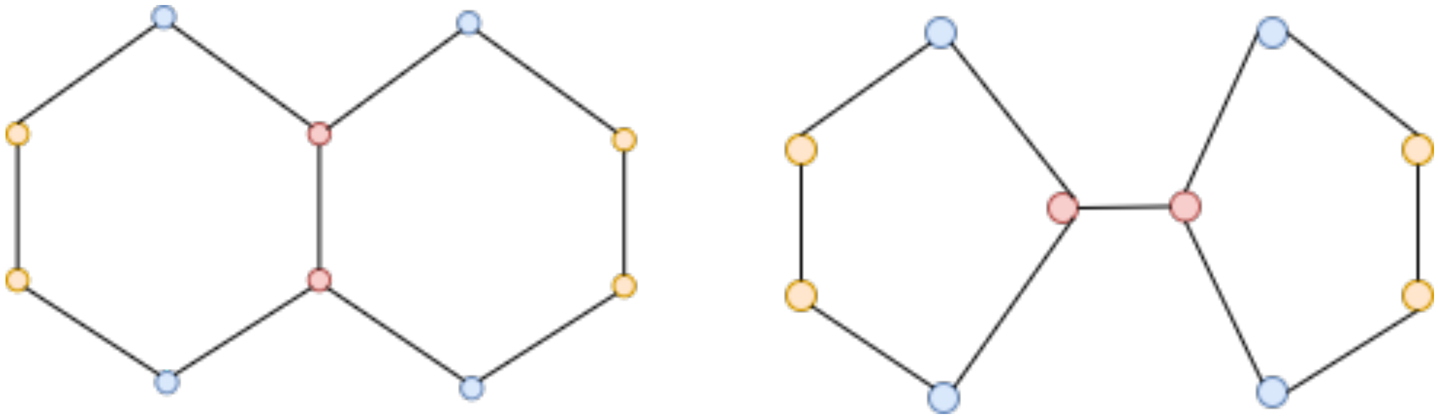


(a) distinguishing graphs

(b) distinguishing trees

A. Loukas, "How hard is to distinguish graphs with graph neural networks?", NIPS 2020

WL is not the most powerful!



Assumptions

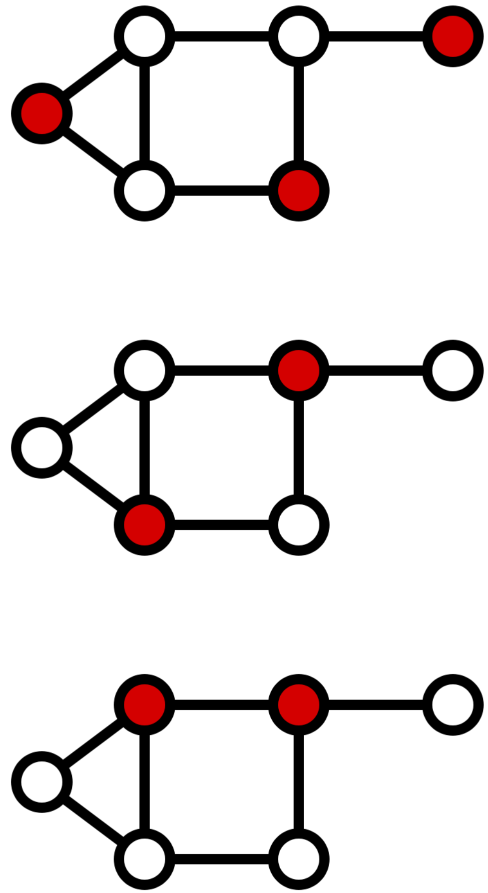
- Bounded degree
  - Which makes sense in most of the cases
- Using an ID for every node
  - One-hot encoding
- No external information other than the graph itself

Minimum Dominating Set

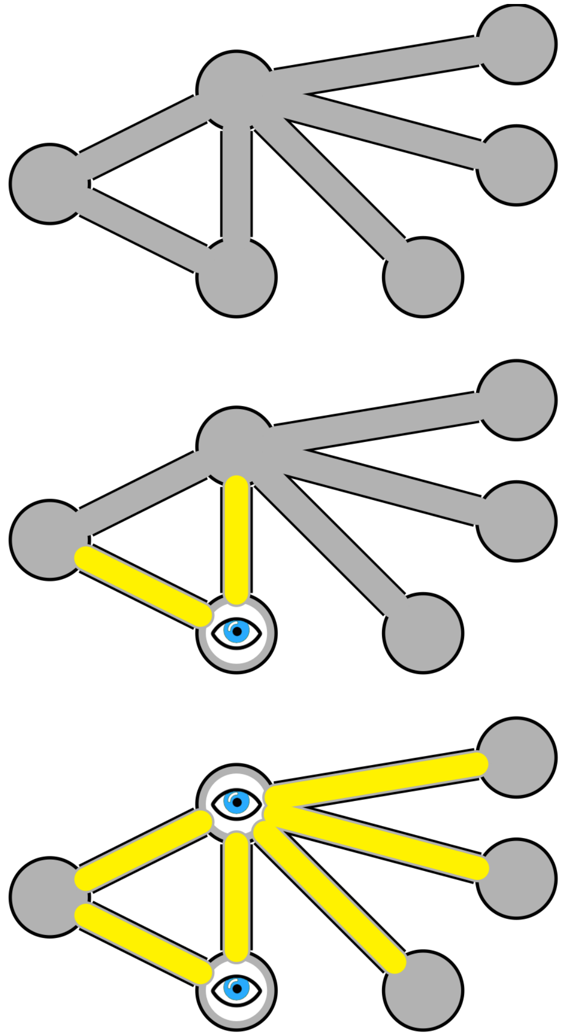
(a)

(b)

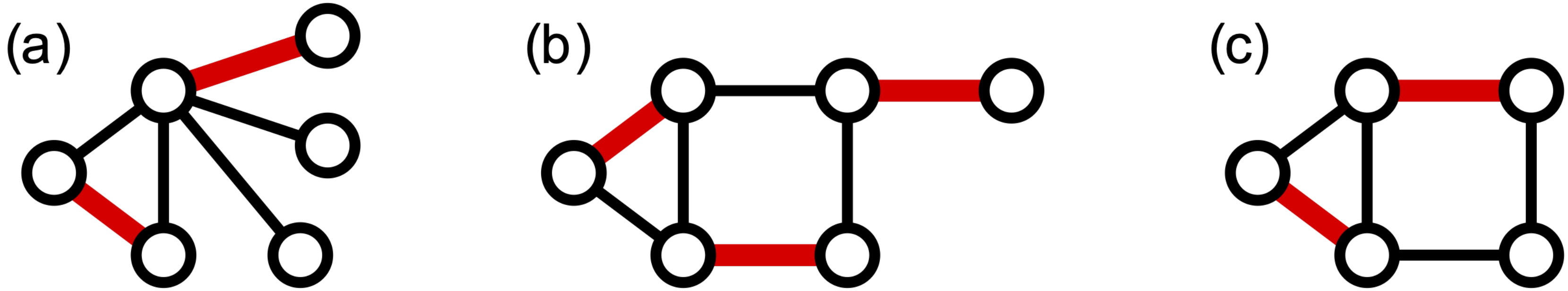
(c)



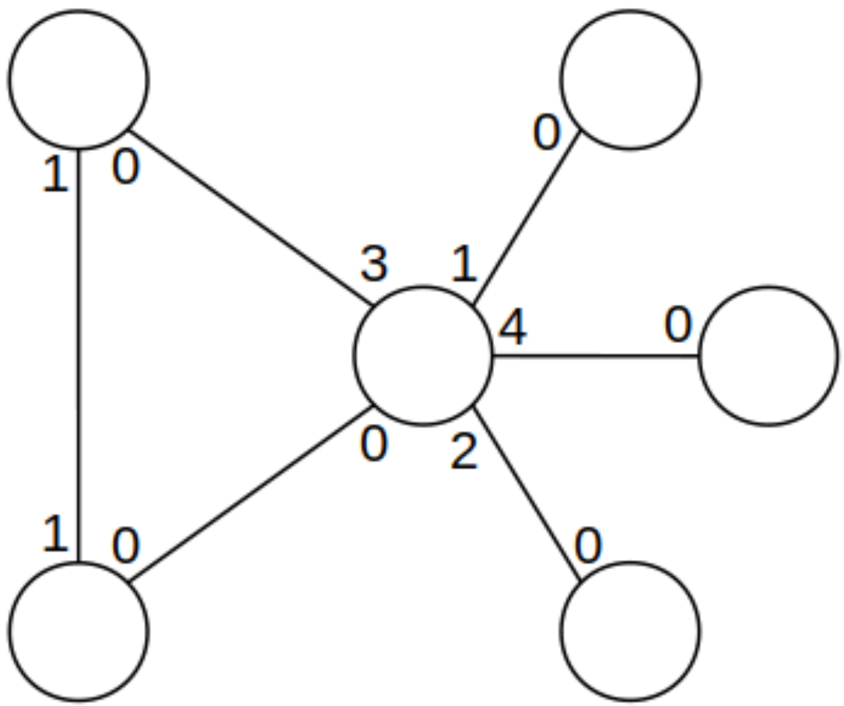
Minimum Vertex Cover



Maximum Matching



Port Numbering



Vector Vector Consistent GNN

- Use port numbering in AGGREGATE
- Linear time (any port numbering)
- Effectively share more information with the neighbors
- CPNGNN

Sato, Ryoma, Makoto Yamada, and Hisashi Kashima. "Approximation ratios of graph neural networks for combinatorial problems." *arXiv preprint arXiv:1905.10261* (2019).

CPNGNN Algorithm

Algorithm 2 CPNGNN: The most powerful $\mathrm{VV_C}$-GNN

Require: Graph $G = (V, E, \boldsymbol{X})$; Maximum degree $\Delta \in \mathbb{Z}^+$; Weight matrix $\boldsymbol{W}^{(l)} \in \mathbb{R}^{d_{l+1} \times (d_l + \Delta(d_l+1))} (l = 1, \ldots, L)$.

Ensure: Output for the graph problem $\boldsymbol{y} \in Y^n$

calculate a consistent port numbering p

$\boldsymbol{z}_v^{(1)} \leftarrow \boldsymbol{x}_v \ (\forall v \in V)$

for $l = 1, \ldots, L$ **do**

    for $v \in V$ **do**

        $\boldsymbol{z}_v^{(l+1)} \leftarrow \boldsymbol{W}^{(l)} \textsc{Concat}(\boldsymbol{z}_v^{(l)}, \boldsymbol{z}_{p_{\mathrm{tail}}(v,1)}^{(l)}, p_{\mathrm{n}}(v,1), \boldsymbol{z}_{p_{\mathrm{tail}}(v,2)}^{(l)}, p_{\mathrm{n}}(v,2), \ldots, \boldsymbol{z}_{p_{\mathrm{tail}}(v,\Delta)}^{(l)}, p_{\mathrm{n}}(v,\Delta))$

        $\boldsymbol{z}_v^{(l+1)} \leftarrow \textsc{ReLU}(\boldsymbol{z}_v^{(l+1)})$

    end for

end for

for $v \in V$ **do**

    $\boldsymbol{z}_v \leftarrow \textsc{MultiLayerPerceptron}(\boldsymbol{z}_v^{(L+1)})$ # calculate the final embedding of a node v.

    $\boldsymbol{y}_v \leftarrow \mathrm{argmax}_{i \in [d_{L+1}]} \boldsymbol{z}_{vi}$ # output the index of the maximum element.

end for

return $\boldsymbol{y}$

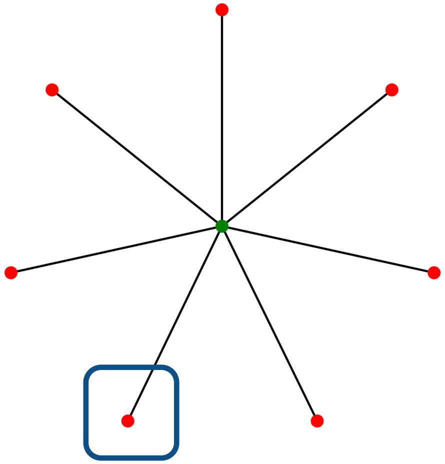
Sato, Ryoma, Makoto Yamada, and Hisashi Kashima. "Approximation ratios of graph neural networks for combinatorial problems." *arXiv preprint arXiv:1905.10261* (2019).

Hierarchy of GNNs

- In terms of the class of problems that they can solve:

$$\mathcal{P}_{\mathit{SB\text{-}GNNs}} \subsetneq \mathcal{P}_{\mathit{MB\text{-}GNNs}} \subsetneq \mathcal{P}_{\mathit{VV_C\text{-}GNNs}}$$

- Example of problems that MB-GNNs cannot solve. Finding a single leaf:



Sato, Ryoma, Makoto Yamada, and Hisashi Kashima. "Approximation ratios of graph neural networks for combinatorial problems." *arXiv preprint arXiv:1905.10261* (2019).

Approximation Ratio

	Without coloring	Weak 2-coloring and degree of nodes	2-coloring (only bipartite)
Minimum Dominating Set	$\delta_g + 1$	$\frac{\delta_g + 1}{2}$	$\frac{\delta_g + 1}{2}$
Minimum Vertex Cover	2	2	2
Maximum Matching	Not possible!	$\frac{\delta_g + 1}{2}$	$any\ \alpha > 1$

Sato, Ryoma, Makoto Yamada, and Hisashi Kashima. "Approximation ratios of graph neural networks for combinatorial problems." *arXiv preprint arXiv:1905.10261* (2019).

Summary

- GNNs are universal when nodes are given unique features (random coloring, one-hot encoding) and the depth and width satisfy some conditions
- The equivalence of anonymous MPNN to the 1st-order Weisfeiler-Lehman (1-WL) graph isomorphism test
- GIN vs VVc-GNN
- Combinatorial (approximation) algorithms

References

- Xu, Keyulu, et al. "How powerful are graph neural networks?." *arXiv preprint arXiv:1810.00826* (2018).
- Loukas, Andreas. "What graph neural networks cannot learn: depth vs width." *arXiv preprint arXiv:1907.03199* (2019).
- A. Loukas, “How hard is to distinguish graphs with graph neural networks?”, NIPS 2020
- Sato, Ryoma, Makoto Yamada, and Hisashi Kashima. "Approximation ratios of graph neural networks for combinatorial problems." *arXiv preprint arXiv:1905.10261* (2019).
- Images:
  - Kriege, Nils & Johansson, Fredrik & Morris, Christopher. (2020). A survey on graph kernels. Applied Network Science. 5. 10.1007/s41109-019-0195-3.
  - https://www.slideshare.net/ssuser0c8361/20200212-227754437, visited on 28.04.2021
  - https://grakn.ai/src/pages/landingpages/MachineLearning/