Let's Agree to Degree: Comparing Graph Convolutional Networks in the Message-Passing Framework

Floris Geerts (University of Antwerp, Belgium), Guillermo A. Perez (University of Antwerp, Belgium) & Filip Mazowiecki (Max Planck Institute, Germany)

Graph learning: vertex & graph classification, regression,...

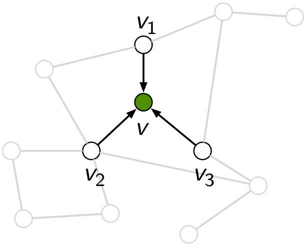
- Graph learning by means of graph embeddings
  - Graph embeddings computed by Message-Passing Neural Networks (MPNNs)
    - Distinguishing power of MPNNs
      - Theoretical analysis

$$\text{discrete world of graphs} \xrightarrow{\textit{graph embedding}} \text{continuous world of vectors in } \mathbb{R}^s$$

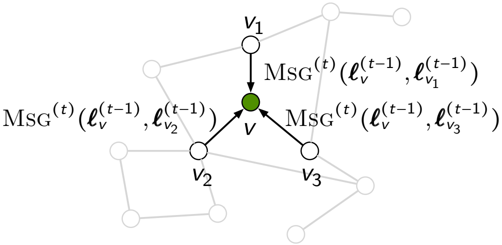
- Parameters underlying embedding methods are learned for specific graph learning tasks.
- Many graph embeddings methods can be seen as a Message-Passing Neural Network.[1]

[1] ☞ Gilmer et al. *Neural message passing for quantum chemistry*. ICML 2017

- Initially: $\boldsymbol{\ell}_v^{(0)} \in \mathbb{R}^{s_0}$ is a hot-one encoding of the label of vertex v
- In layer $t > 0$: Each vertex v receives messages from its neighbors based on the previously computed vertex embeddings, which are then aggregated, and then further updated based on the vertex own previous embedding:

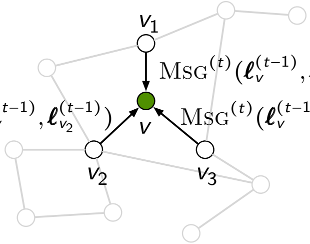


- Initially: $\boldsymbol{\ell}_v^{(0)} \in \mathbb{R}^{s_0}$ is a hot-one encoding of the label of vertex v
- In layer $t > 0$: Each vertex v receives messages from its neighbors based on the previously computed vertex embeddings, which are then aggregated, and then further updated based on the vertex own previous embedding:



Message-Passing Neural Networks (MPNNs)

- Initially: $\boldsymbol{\ell}_v^{(0)} \in \mathbb{R}^{s_0}$ is a hot-one encoding of the label of vertex v
- In layer $t > 0$: Each vertex v receives messages from its neighbors based on the previously computed vertex embeddings, which are then aggregated, and then further updated based on the vertex own previous embedding:

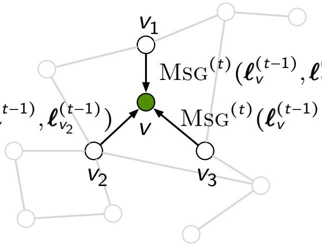


$\textsc{Msg}^{(t)}(\boldsymbol{\ell}_v^{(t-1)}, \boldsymbol{\ell}_{v_1}^{(t-1)})$

$\textsc{Msg}^{(t)}(\boldsymbol{\ell}_v^{(t-1)}, \boldsymbol{\ell}_{v_2}^{(t-1)})$

$\textsc{Msg}^{(t)}(\boldsymbol{\ell}_v^{(t-1)}, \boldsymbol{\ell}_{v_3}^{(t-1)}) \longrightarrow \mathsf{m}_v^{(t)} = \sum_{u \in N_G(v)} \textsc{Msg}^{(t)}(\boldsymbol{\ell}_v^{(t-1)}, \boldsymbol{\ell}_u^{(t-1)}) \in \mathbb{R}^{s_t'}$

- Initially: $\boldsymbol{\ell}_v^{(0)} \in \mathbb{R}^{s_0}$ is a hot-one encoding of the label of vertex v
- In layer $t > 0$: Each vertex v receives messages from its neighbors based on the previously computed vertex embeddings, which are then aggregated, and then further updated based on the vertex own previous embedding:

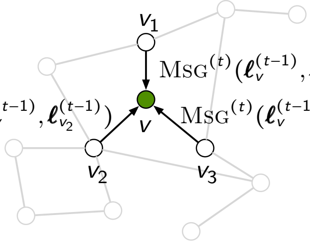


$$\longrightarrow \mathrm{m}_v^{(t)} = \sum_{u \in N_G(v)} \textsc{Msg}^{(t)}(\boldsymbol{\ell}_v^{(t-1)}, \boldsymbol{\ell}_u^{(t-1)}) \in \mathbb{R}^{s'_t}$$

$$\downarrow$$

$$\boldsymbol{\ell}_v^{(t)} := \textsc{Upd}^{(t)}\left(\boldsymbol{\ell}_v^{(t-1)}, \mathrm{m}_v^{(t)}\right) \in \mathbb{R}^{s_t}$$

- Initially: $\boldsymbol{\ell}_v^{(0)} \in \mathbb{R}^{s_0}$ is a hot-one encoding of the label of vertex v
- In layer $t > 0$: Each vertex v receives messages from its neighbors based on the previously computed vertex embeddings, which are then aggregated, and then further updated based on the vertex own previous embedding:



$\textsc{Msg}^{(t)}(\boldsymbol{\ell}_v^{(t-1)}, \boldsymbol{\ell}_{v_1}^{(t-1)})$

$\textsc{Msg}^{(t)}(\boldsymbol{\ell}_v^{(t-1)}, \boldsymbol{\ell}_{v_2}^{(t-1)})$ $\quad$ $\textsc{Msg}^{(t)}(\boldsymbol{\ell}_v^{(t-1)}, \boldsymbol{\ell}_{v_3}^{(t-1)}) \longrightarrow \mathrm{m}_v^{(t)} = \sum_{u \in N_G(v)} \textsc{Msg}^{(t)}(\boldsymbol{\ell}_v^{(t-1)}, \boldsymbol{\ell}_u^{(t-1)}) \in \mathbb{R}^{s'_t}$

$$\downarrow$$

$$\boldsymbol{\ell}_v^{(t)} := \textsc{Upd}^{(t)}\left(\boldsymbol{\ell}_v^{(t-1)}, \mathrm{m}_v^{(t)}\right) \in \mathbb{R}^{s_t}$$

- Message and update functions contain learnable parameters.

Message-Passing Neural Networks (MPNNs)

We emphasize:

- The message functions $\mathrm{MSG}^{(t)}$ in MPNNs only depend on the previously computed vertex embeddings:

$$\boldsymbol{\ell}_v^{(t)} := \mathrm{UPD}^{(t)}\Big(\boldsymbol{\ell}_v^{(t-1)}, \sum_{u \in N_G(v)} \mathrm{MSG}^{(t)}(\boldsymbol{\ell}_v^{(t-1)}, \boldsymbol{\ell}_u^{(t-1)})\Big) \in \mathbb{R}^{s_t},$$

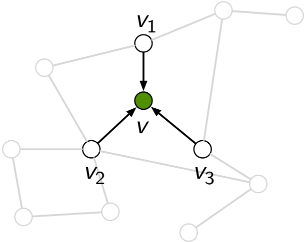
- This will be important later on.

How well can MPNNs distinguish vertices and graphs?

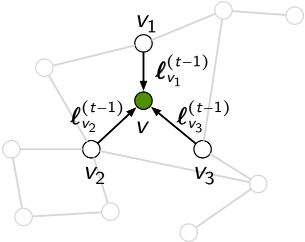
- The distinguishing power reflect the ability to distinguish vertices/graphs by means of their vector embeddings.
- Important to understand, since it measures the loss of information by the embedding method.
- For MPNNs, the distinguish power can be characterized in terms of the Weisfeiler-Lehman graph isomorphism test.[2]

[2] ☞ Weisfeiler and Lehman. *A reduction of a graph to a canonical form and an algebra arising during this reduction*. Nauchno-Technicheskaya Informatsiya, 1968

- Let $\ell_v^{(0)}$ be the initial label of vertex v
- In round $t > 0$, same recipe as for MPNNs:

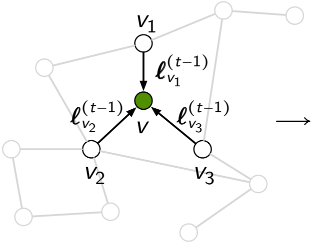


- Let $\ell_v^{(0)}$ be the initial label of vertex v
- In round $t > 0$, same recipe as for MPNNs:



Weisfeiler-Lehman (WL) test

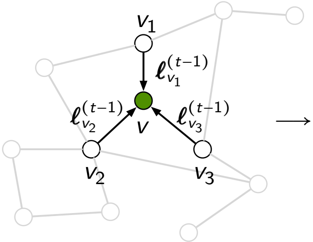
- Let $\ell_v^{(0)}$ be the initial label of vertex v
- In round $t > 0$, same recipe as for MPNNs:



$$\longrightarrow \mathrm{m}_v^{(t)} = \text{multiset } \{\!\!\{ \ell_u^{(t-1)} \mid u \in N_G(v) \}\!\!\}$$

Weisfeiler-Lehman (WL) test

- Let $\boldsymbol{\ell}_v^{(0)}$ be the initial label of vertex v
- In round $t > 0$, same recipe as for MPNNs:



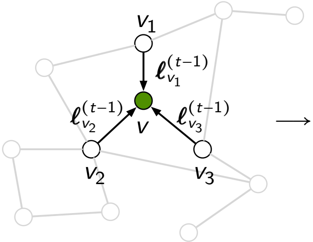
$$\longrightarrow \mathrm{m}_v^{(t)} = \text{multiset } \{\!\{ \boldsymbol{\ell}_u^{(t-1)} \mid u \in N_G(v) \}\!\}$$

$$\downarrow$$

$$\boldsymbol{\ell}_v^{(t)} := \textsc{Hash}\Big(\boldsymbol{\ell}_v^{(t-1)}, \mathrm{m}_v^{(t)}\Big)$$

Weisfeiler-Lehman (WL) test

- Let $\boldsymbol{\ell}_v^{(0)}$ be the initial label of vertex v
- In round $t > 0$, same recipe as for MPNNs:



$$\longrightarrow \mathrm{m}_v^{(t)} = \text{multiset } \{\!\{ \boldsymbol{\ell}_u^{(t-1)} \mid u \in N_G(v) \}\!\}$$

$$\downarrow$$

$$\boldsymbol{\ell}_v^{(t)} := \textsc{Hash}\Big(\boldsymbol{\ell}_v^{(t-1)}, \mathrm{m}_v^{(t)}\Big)$$

- In contrast to MPNNs: No learnable parameters, $\textsc{Hash}$ function is injective=most distinguishing.

- Classical, well-studied algorithm, used in graph isomorphism tests.
- WL is said to distinguish graphs G and H in t rounds when the multisets of labels computed by WL in t rounds on both graphs differ. Formally:

$$\{\!\{\boldsymbol{\ell}_v^{(t)} \mid v \in V_G\}\!\} \neq \{\!\{\boldsymbol{\ell}_w^{(t)} \mid w \in V_H\}\!\}$$

- The distinguishing power of WL is well-understood.[3,4]

[3] ☞ Grohe, M. *Word2vec, node2vec, graph2vec, x2vec: To-wards a theory of vector embeddings of structured data*. PODS, 2020

[4] ☞ Sato, R. *A survey on the expressive power of graph neural networks*. ArXiv, 2020

The following is known:[5,6]

Theorem

- *For any two graphs G and H, if WL cannot distinguish G from H in t rounds, then neither can any t-layer MPNN.*
- *For any two graphs G and H, there exists an MPNN with precisely the same distinguishing power as the WL-test. In fact, this MPNN can be assumed to originate from a "basic" Graph Neural Network (GNN).*

[5] ☞ Morris et al. *Weisfeiler and Leman go neural: Higher-order graph neural networks.* AAAI, 2019

[6] ☞ Xu et al. *How powerful are graph neural networks?* ICLR, 2019

The following is known:[5,6]

Theorem

- *For any two graphs G and H, if WL cannot distinguish G from H in t rounds, then neither can any t-layer MPNN.*
- *For any two graphs G and H, there exists an MPNN with precisely the same distinguishing power as the WL-test. In fact, this MPNN can be assumed to originate from a "basic" Graph Neural Network (GNN).*

[5] ☞ Morris et al. *Weisfeiler and Leman go neural: Higher-order graph neural networks.* AAAI, 2019

[6] ☞ Xu et al. *How powerful are graph neural networks?* ICLR, 2019

The following is known:[5,6]

Theorem

- *For any two graphs G and H, if WL cannot distinguish G from H in t rounds, then neither can any t-layer MPNN.*
- *For any two graphs G and H, there exists an MPNN with precisely the same distinguishing power as the WL-test. In fact, this MPNN can be assumed to originate from a "basic" Graph Neural Network (GNN).*

[5] ☞ Morris et al. *Weisfeiler and Leman go neural: Higher-order graph neural networks*. AAAI, 2019

[6] ☞ Xu et al. *How powerful are graph neural networks?* ICLR, 2019

Our Research Question

- Does this theorem apply to commonly used GNN architectures?
- In other words, can common GNN architectures indeed be cast as MPNNs?
- Are they as powerful as WL?

We next look at:

- Basic Graph Neural Networks (for which the answer to these questions are known)
- Graph Convolutional Networks (for which a new analysis is needed)

- Layers are defined by:[7]
$$L^{(t)} := \sigma\left(L^{(t-1)}W_1^{(t)} + A_G L^{(t-1)}W_2^{(t)} + B^{(t)}\right)$$
- A_G is adjacency matrix of G, $L^{(t)}$ consists of feature vectors,
- $W_1^{(t)}$ and $W_2^{(t)}$ are learnable weight matrices, $B^{(t)}$ is a constant bias matrix.
- Indeed corresponds to an MPNN:
$$\textsc{Msg}^{(t)}(x, y) := yW_2^{(t)} \mapsto \text{neighbors send their weighted features } (y)$$
$$\textsc{Upd}^{(t)}(x, m) := \sigma(xW_1^{(t)} + m + b^{(t)}) \mapsto \text{own weighed feature } (x) \text{ added}$$
$$\text{to aggregations of neighbors features } (m)$$
- So, the Theorem applies: distinguishing power of t layer basic GNNs cannot exceed that of a t round WL test.

[7] ☞ Hamilton et al. *Inductive representation learning on large graphs*. NeurIPS, 2017

- Layers are defined by:[7]

$$\mathsf{L}^{(t)} := \sigma\left(\mathsf{L}^{(t-1)}\mathsf{W}_1^{(t)} + \mathsf{A}_G\mathsf{L}^{(t-1)}\mathsf{W}_2^{(t)} + \mathsf{B}^{(t)}\right)$$

- A_G is adjacency matrix of G, $\mathsf{L}^{(t)}$ consists of feature vectors,
- $\mathsf{W}_1^{(t)}$ and $\mathsf{W}_2^{(t)}$ are learnable weight matrices, $\mathsf{B}^{(t)}$ is a constant bias matrix.

- Indeed corresponds to an MPNN:

$$\textsc{Msg}^{(t)}(\mathsf{x},\mathsf{y}) := \mathsf{y}\mathsf{W}_2^{(t)} \mapsto \text{neighbors send their weighted features } (y)$$

$$\textsc{Upd}^{(t)}(\mathsf{x},\mathsf{m}) := \sigma(\mathsf{x}\mathsf{W}_1^{(t)} + \mathsf{m} + \mathsf{b}^{(t)}) \mapsto \text{own weighed feature } (x) \text{ added to aggregations of neighbors features } (m)$$

- So, the Theorem applies: distinguishing power of t layer basic GNNs cannot exceed that of a t round WL test.

[7] ☞ Hamilton et al. *Inductive representation learning on large graphs*. NeurIPS, 2017

- Very popular architecture in which layers are defined by:[8]

$$\mathsf{L}^{(t)} := \sigma\left(\mathsf{D}^{-1/2}(\mathsf{I} + \mathsf{A}_G)\mathsf{D}^{-1/2}\mathsf{L}^{(t-1)}\mathsf{W}_2^{(t)} + \mathsf{B}^{(t)}\right)$$

- D is diagonal matrix consisting of degrees d_v for $v \in V$.

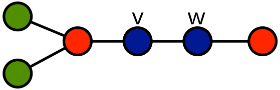
- A GCN can distinguish v from w with one layer, but WL cannot in one round!
- Previous theorem does not apply! A GCN is not a "standard" MPNN.

[8] ☞ Kipf and Welling. *Semi-supervised classification with graph convolutional networks*. ICLR, 2017

- Very popular architecture in which layers are defined by:[8]

$$\mathsf{L}^{(t)} := \sigma\left(\mathsf{D}^{-1/2}(\mathsf{I} + \mathsf{A}_G)\mathsf{D}^{-1/2}\mathsf{L}^{(t-1)}\mathsf{W}_2^{(t)} + \mathsf{B}^{(t)}\right)$$

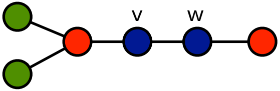
- D is diagonal matrix consisting of degrees d_v for $v \in V$.



- A GCN can distinguish v from w with one layer, but WL cannot in one round!
- Previous theorem does not apply! A GCN is not a “standard” MPNN.

[8] ☞ Kipf and Welling. *Semi-supervised classification with graph convolutional networks*. ICLR, 2017

- What happened?



- A GCN detects immediately that:

  v is adjacent to a red vertex of degree three

  w is adjacent to a red vertex of degree one.

- By contrast, WL only observes the colors.

- To see GCNs as MPNNs, we extend the message functions with degree information:
- Proposal: Degree-aware MPNNs:

- As before, let $\boldsymbol{\ell}_v^{(0)} \in \mathbb{R}^{s_0}$ be a hot-one encoding of the label of vertex v.
- Then, in layer t a degree-aware MPNN computes a new vertex-labelling for each vertex:

$$\boldsymbol{\ell}_v^{(t)} := \textsc{Upd}^{(t)}\Big(\boldsymbol{\ell}_v^{(t-1)}, \sum_{u \in N_G(v)} \textsc{Msg}^{(t)}(\boldsymbol{\ell}_v^{(t-1)}, \boldsymbol{\ell}_u^{(t-1)}, d_v, d_u)\Big) \in \mathbb{R}^{s_t},$$

where now $\textsc{Upd}^{(t)}$ has extra arguments.

Degree-aware MPNNs vs standard MPNNs

Proposition

Any degree-aware MPNN consisting of t layers, can be simulated by an MPNN consisting of $t+1$ layers.

Idea: use the first layer of the MPNN to compute degrees, add these to the labels in subsequent layers.

- Thus, for any two graphs G and H, if WL cannot distinguish G from H in $t+1$ rounds, then neither can any degree-aware t layer MPNN.
- Degree-aware MPNNs (such as GCNs) may have an advantage over standard MPNNs in terms of number of layers. So, let's agree to degree!

Degree-aware MPNNs vs standard MPNNs

Proposition

Any degree-aware MPNN consisting of t layers, can be simulated by an MPNN consisting of $t + 1$ layers.

Idea: use the first layer of the MPNN to compute degrees, add these to the labels in subsequent layers.

- Thus, for any two graphs G and H, if WL cannot distinguish G from H in $t+1$ rounds, then neither can any degree-aware t layer MPNN.
- Degree-aware MPNNs (such as GCNs) may have an advantage over standard MPNNs in terms of number of layers. So, let's agree to degree!

Not all degree-aware MPNNs are one step ahead:

	GNN architectures using degrees[9,10,11]	bounded by
GNN1.	$\sigma\left((\mathsf{D}+\mathsf{I})^{-1/2}(\mathsf{A}+\mathsf{I})(\mathsf{D}+\mathsf{I})^{-1/2}\mathsf{L}^{(t-1)}\mathsf{W}^{(t)}\right)$	WL+1 step
GNN2.	$\sigma\left(\mathsf{D}^{-1/2}\mathsf{A}\mathsf{D}^{-1/2}\mathsf{L}^{(t-1)}\mathsf{W}^{(t)}\right)$	WL+1 step
GNN3.	$\sigma\left((r\mathsf{I}+(1-r)\mathsf{D})^{-1/2}(\mathsf{A}+p\mathsf{I})(r\mathsf{I}+(1-r)\mathsf{D})^{-1/2}\mathsf{L}^{(t-1)}\mathsf{W}^{(t)}\right)$	WL+1 step
GNN4.	$\sigma\left((\mathsf{D}^{-1/2}\mathsf{A}\mathsf{D}^{-1/2}+\mathsf{I})\mathsf{L}^{(t-1)}\mathsf{W}^{(t)}\right)$	WL+1 step
GNN5.	$\sigma\left(\mathsf{D}^{-1}\mathsf{A}\mathsf{L}^{(t-1)}\mathsf{W}^{(t)}\right)$	WL
GNN6.	$\sigma\left((\mathsf{D}+\mathsf{I})^{-1}(\mathsf{A}+\mathsf{I})\mathsf{L}^{(t-1)}\mathsf{W}^{(t)}\right)$	WL

If GNNs only use degrees after aggregation, then they can be cast as standard MPNNs.

[9] ☞ Kipf and Welling. *Semi-supervised classification with graph convolutional networks*. ICLR, 2017
[10] ☞ Wu et al. *Simplifying graph convolutional networks*. ICML, 2019
[11] ☞ Meltzer et al. *Pinet: A permutation invariant graph neural network for graph classification*. arXiv, 2019

Recall:[12]

Theorem

For any two graphs G and H, there exists a basic GNN that can distinguish these graphs when WL can distinguish them too.

- So, the class of basic GNNs is as powerful as WL.
- Still true for GCNs? No!

- WL can distinguish these two vertices, GCNs cannot!

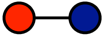
[12] ☞ Morris et al. *Weisfeiler and Leman go neural: Higher-order graph neural networks.* AAAI, 2019

Recall:[12]

Theorem

For any two graphs G and H, there exists a basic GNN that can distinguish these graphs when WL can distinguish them too.

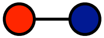
- So, the class of basic GNNs is as powerful as WL.
- Still true for GCNs? No!



- WL can distinguish these two vertices, GCNs cannot!

[12] ☞ Morris et al. *Weisfeiler and Leman go neural: Higher-order graph neural networks*. AAAI, 2019

- Reason that GCNs cannot distinguish vertices in

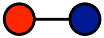


  is not because of degree information but simply because of use of $I + A$ as aggregation matrix.
- For the example graph,

$$I + A = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

  So all features are propagated in the same way for both vertices.
- Solution: consider $pI + A$ for parameter $0 < p < 1$ instead!
- The use of parameter p was empirically motivated by Kipf and Welling.

- Reason that GCNs cannot distinguish vertices in



  is not because of degree information but simply because of use of $I + A$ as aggregation matrix.

- For the example graph,

$$I + A = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

  So all features are propagated in the same way for both vertices.

- Solution: consider $pI + A$ for parameter $0 < p < 1$ instead!
- The use of parameter p was empirically motivated by Kipf and Welling.

Consider general degree-based MPNNs based on generalized GCNs with layers:

$$\mathsf{L}^{(t)} := \sigma\big(\mathrm{diag}(\mathsf{g})(\mathsf{A} + p\mathsf{I})\mathrm{diag}(\mathsf{h})\mathsf{L}^{(t-1)}\mathsf{W}^{(t)} + \mathsf{B}^{(t)}\big),$$

where g and h are degree-determined vectors (hold identical values for vertices having same degrees)

Theorem (Main result)

For any two graphs G and H, there exists a generalized GCN that can distinguish these graphs when WL can distinguish them too. In addition, the parameter p can be chosen uniformly across layers.

- Applies to basic GCNs by Kipf and Welling: $\sigma\left((\mathsf{D}+\mathsf{I})^{-1/2}(\mathsf{A}+p\mathsf{I})(\mathsf{D}+\mathsf{I})^{-1/2}\mathsf{L}^{(t-1)}\mathsf{W}^{(t)}\right)$
- Theoretical justification of the parameter p!

Consider general degree-based MPNNs based on generalized GCNs with layers:

$$\mathrm{L}^{(t)} := \sigma\big(\mathrm{diag}(\mathrm{g})(\mathrm{A} + p\mathrm{I})\mathrm{diag}(\mathrm{h})\mathrm{L}^{(t-1)}\mathrm{W}^{(t)} + \mathrm{B}^{(t)}\big),$$

where g and h are degree-determined vectors (hold identical values for vertices having same degrees)

Theorem (Main result)

For any two graphs G and H, there exists a generalized GCN that can distinguish these graphs when WL can distinguish them too. In addition, the parameter p can be chosen uniformly across layers.

- Applies to basic GCNs by Kipf and Welling: $\sigma\left((\mathrm{D} + \mathrm{I})^{-1/2}(\mathrm{A} + p\mathrm{I})(\mathrm{D} + \mathrm{I})^{-1/2}\mathrm{L}^{(t-1)}\mathrm{W}^{(t)}\right)$
- Theoretical justification of the parameter p!

WL-powerful GCNs

	GNN architectures using degrees	as strong as WL?
GNN7.	$\sigma\left((A+pI)L^{(t-1)}W^{(t)}\right)$	yes
GNN8.	$\sigma\left((D+I)^{-1/2}(A+pI)(D+I)^{-1/2}L^{(t-1)}W^{(t)}\right)$	yes
GNN3.	$\sigma\left((rI+(1-r)D)^{-1/2}(A+pI)(rI+(1-r)D)^{-1/2}L^{(t-1)}W^{(t)}\right)$	yes
GNN1.	$\sigma\left((D+I)^{-1/2}(A+I)(D+I)^{-1/2}L^{(t-1)}W^{(t)}\right)$	no
GNN2.	$\sigma\left(D^{-1/2}AD^{-1/2}L^{(t-1)}W^{(t)}\right)$	no
GNN4.	$\sigma\left((D^{-1/2}AD^{-1/2}+I)L^{(t-1)}W^{(t)}\right)$	no
GNN5.	$\sigma\left(D^{-1}AL^{(t-1)}W^{(t)}\right)$	no
GNN6.	$\sigma\left((D+I)^{-1}(A+I)L^{(t-1)}W^{(t)}\right)$	no

Conclusions

- When casting GNNs as MPNNs: carefully analyze what information message functions use!
- In case of degree information: distinguishing power still bounded by WL, but one step ahead.
- This is important since in practice GNN consist of a small number of layers.
- WL-powerful degree-aware GNNs: introduce learnable parameter p and use $pI + A$ as aggregation matrix.
- Research direction: Analyze distinguishing power of more general MPNN extensions in which message functions may depend on graph information beyond degrees.

Conclusions

- When casting GNNs as MPNNs: carefully analyze what information message functions use!
- In case of degree information: distinguishing power still bounded by WL, but one step ahead.
- This is important since in practice GNN consist of a small number of layers.
- WL-powerful degree-aware GNNs: introduce learnable parameter p and use $pI + A$ as aggregation matrix.

- Research direction: Analyze distinguishing power of more general MPNN extensions in which message functions may depend on graph information beyond degrees.