Massachusetts Institute of Technology

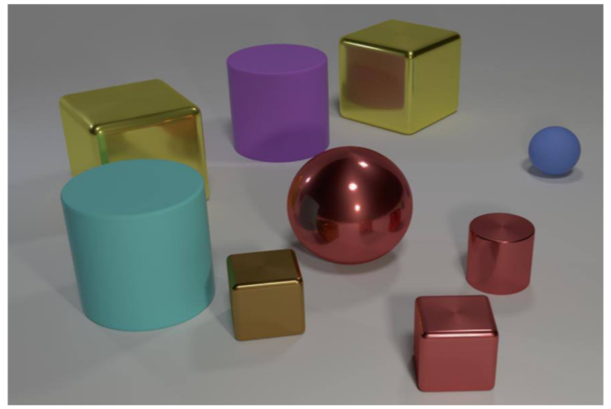
CSAIL

Task Structure and Generalization in Graph Neural Networks

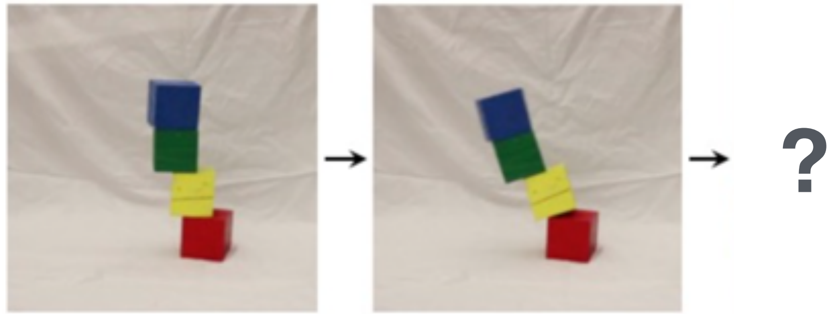
Stefanie Jegelka
MIT

joint work with
Keyulu Xu, Jingling Li, Mozhi Zhang,
Simon S. Du, Ken-ichi Kawarabayashi

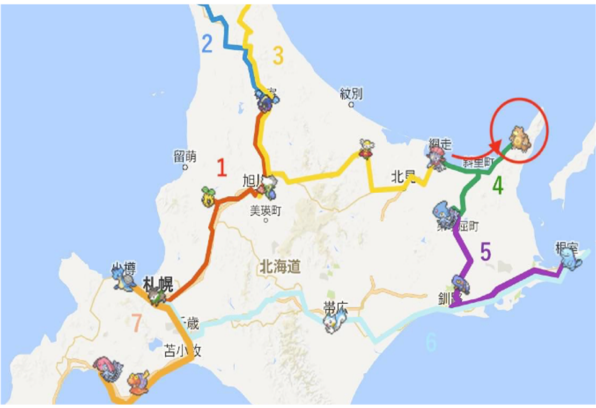
Algorithmic Reasoning Tasks



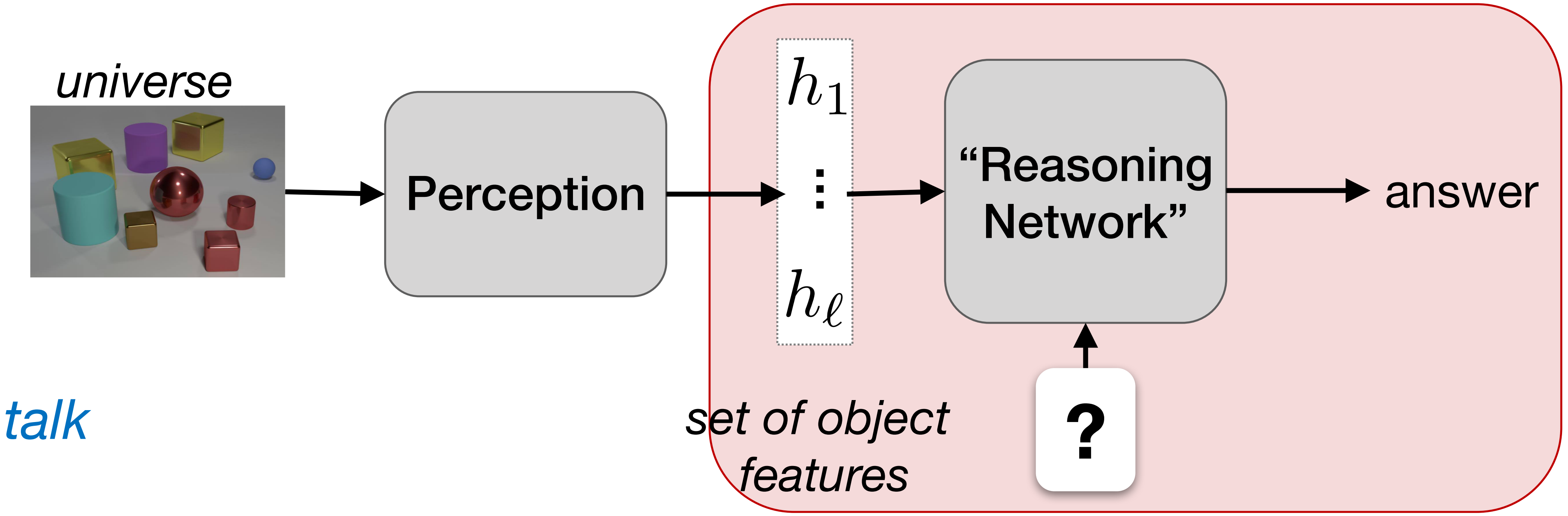
What are the colors of the farthest pair of objects?



What is the next state of the system?



What is the shortest path to the monster?



See also Petar's talk on Tuesday

(Johnson et al., 2017a; Weston et al., 2015; Hu et al., 2017; Fleuret et al., 2011; Antol et al., 2015; Battaglia et al., 2016; Watters et al., 2017; Fragkiadaki et al., 2016; Chang et al., 2017; Saxton et al., 2019; Chang et al., 2019; Santoro et al., 2018; Zhang et al., 2019, …)

How well do (Graph) Neural Networks learn such tasks?

What does this depend on?

- Generalization and architectural structure
  K. Xu, J. Li, M. Zhang, S. Du, K. Kawarabayashi, S. Jegelka. ICLR 2020

- Extrapolation, structure and nonlinearities
  K. Xu, J. Li, M. Zhang, S. Du, K. Kawarabayashi, S. Jegelka, ICLR 2021

Generalization Analysis of GNNs

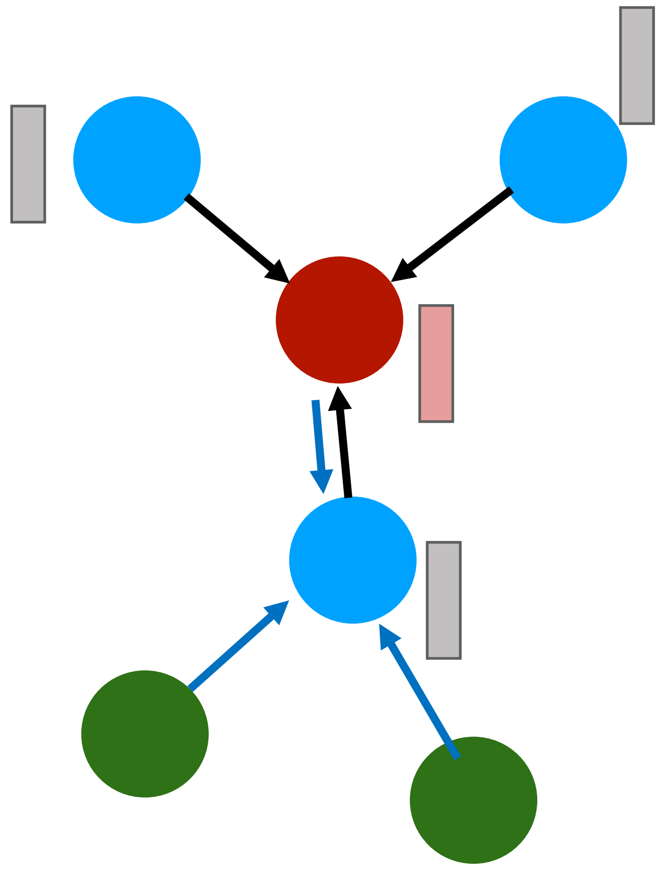
- **Complexity-based**
  (VC-dim/ Rademacher/ PAC-Bayes)
  Scarselli et al 2018, Garg et al 2020, Liao et al 2021

- **Trajectory-based** (NTK)
  Du et al 2019

➡ **Structural Inductive Biases**
(structured functions)
Xu et al 2020, 2021

More Assumptions / More refined

Graph Neural Networks



node embedding

In each round *k*:

Aggregate over neighbors

feature description of node u in round k-1

$$m_{\mathcal{N}(v)}^{(k)} = \mathrm{AGGREGATE}^{(k)}\left(\{\!\{ h_u^{(k-1)} : u \in \mathcal{N}(v) \}\!\}\right)$$

Combine with current node

$$h_v^{(k)} = \mathrm{COMBINE}^{(k)}\left(h_v^{(k-1)}, m_{\mathcal{N}(v)}^{(k)}\right)$$

graph embedding

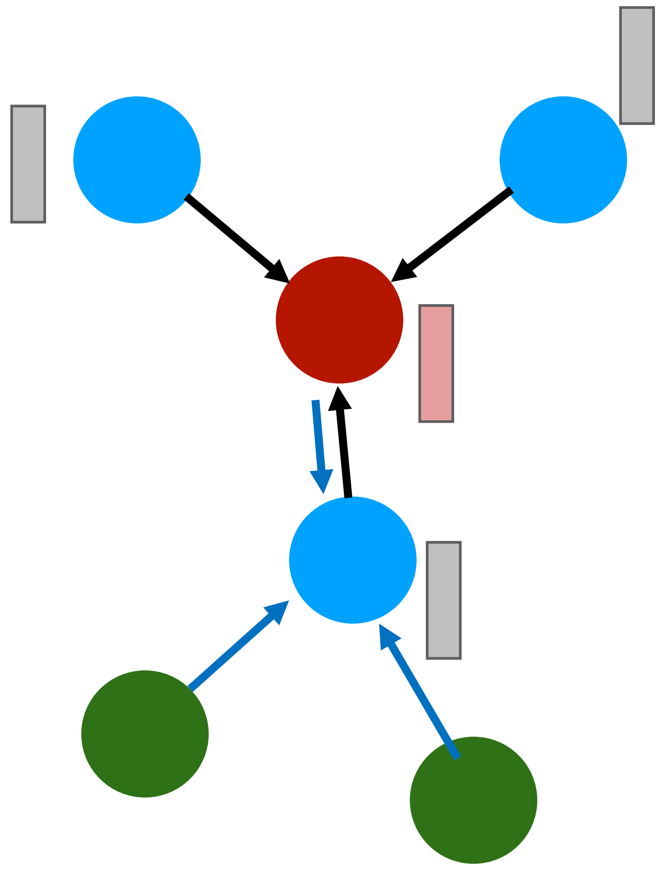
....

Graph-level readout

$$h_{\mathcal{G}} = \mathrm{READOUT}\left(\{\!\{ h_v^{(K)} : v \in \mathcal{G} \}\!\}\right)$$

(Merkwirth & Lengauer 2005; Scarselli et al 2009; Bruna et al 2014; Dai et al 2016; Battaglia et al., 2016; Defferrard et al., 2016; Duvenaud et al., 2015; Hamilton et al., 2017; Kearnes et al., 2016; Kipf & Welling, 2017; Li et al., 2016; Velickovic et al., 2018; Verma & Zhang, 2018; Ying et al., 2018; Zhang et al., 2018; …)

Graph Neural Networks



node embedding

In each round *k*:

Aggregate over neighbors

$$m_{\mathcal{N}(v)}^{(k)} = \sum_{v \in \mathcal{N}(u)} \mathrm{MLP}^{(k)}\left(\boldsymbol{h}_u^{(k-1)}, \boldsymbol{h}_v^{(k-1)}, \boldsymbol{w}_{(v,u)}\right)$$

Combine with current node

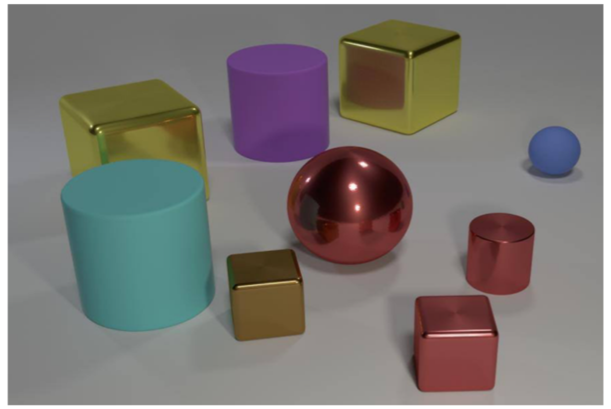
graph embedding

....

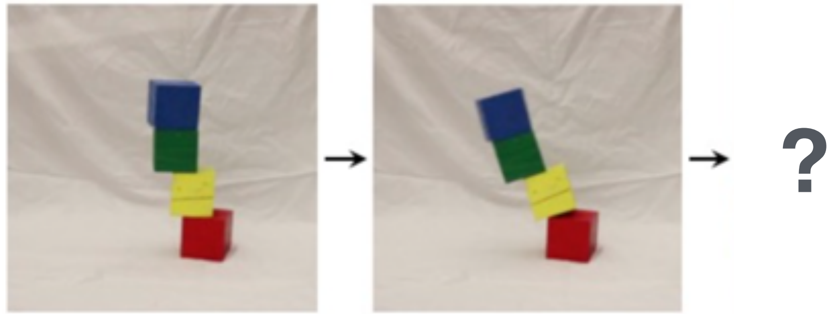
Graph-level readout

(Merkwirth & Lengauer 2005; Scarselli et al 2009; Bruna et al 2014; Dai et al 2016; Battaglia et al., 2016; Defferrard et al., 2016; Duvenaud et al., 2015; Hamilton et al., 2017; Kearnes et al., 2016; Kipf & Welling, 2017; Li et al., 2016; Velickovic et al., 2018; Verma & Zhang, 2018; Ying et al., 2018; Zhang et al., 2018; …)

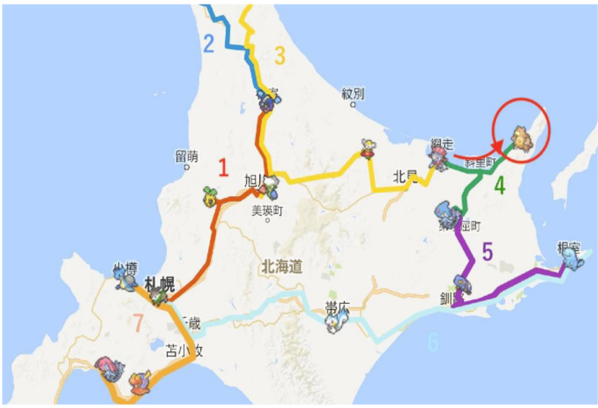
Algorithmic Reasoning Tasks



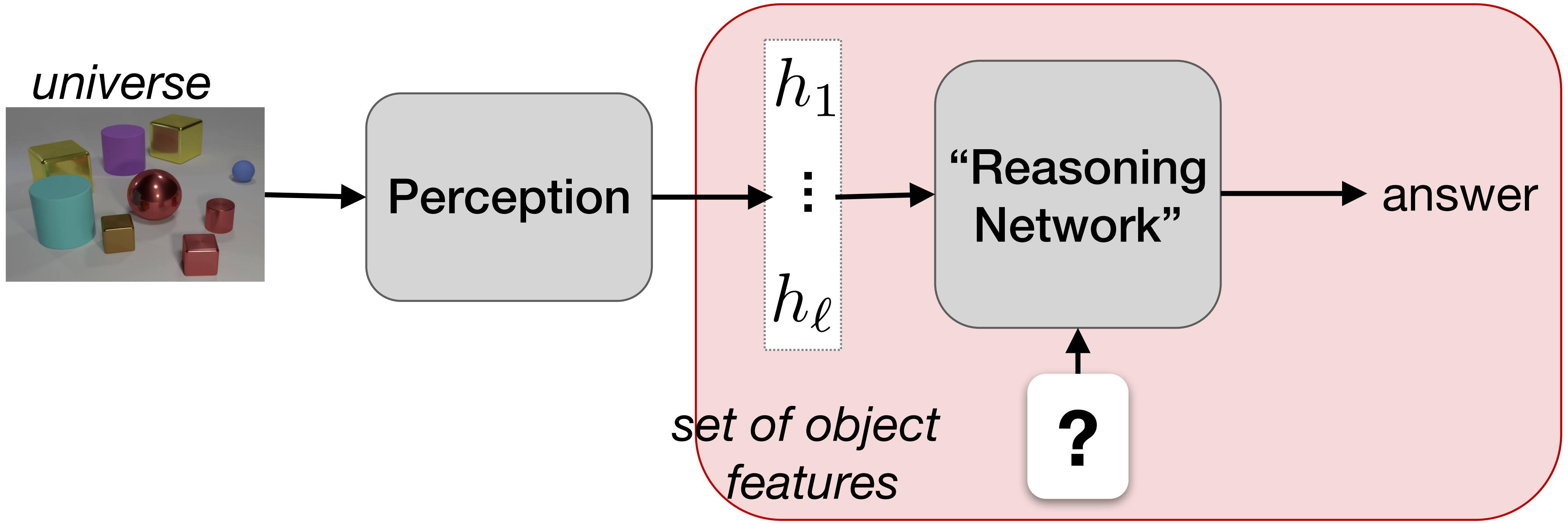
What are the colors of the farthest pair of objects?



What is the next state of the system?



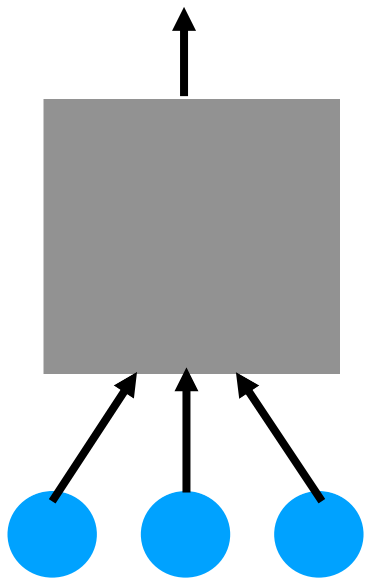
What is the shortest path to the monster?

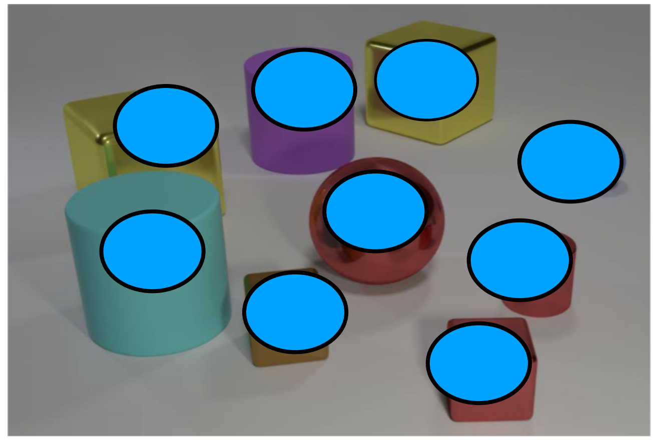


(Johnson et al., 2017a; Weston et al., 2015; Hu et al., 2017; Fleuret et al., 2011; Antol et al., 2015; Battaglia et al., 2016; Watters et al., 2017; Fragkiadaki et al., 2016; Chang et al., 2017; Saxton et al., 2019; Chang et al., 2019; Santoro et al., 2018; Zhang et al., 2019, …)

Architectures

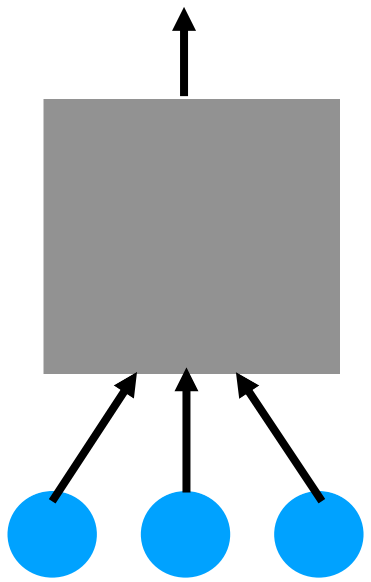
MLP





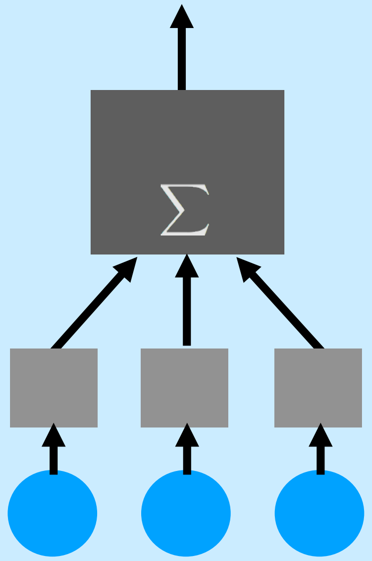
Architectures

MLP

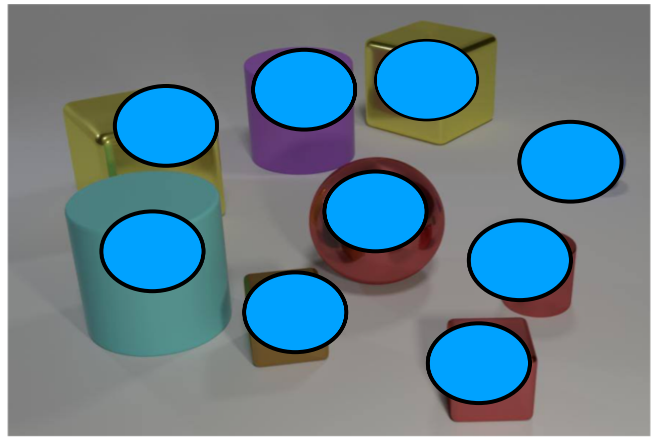


Deep Sets

(Zaheer et al 2017)

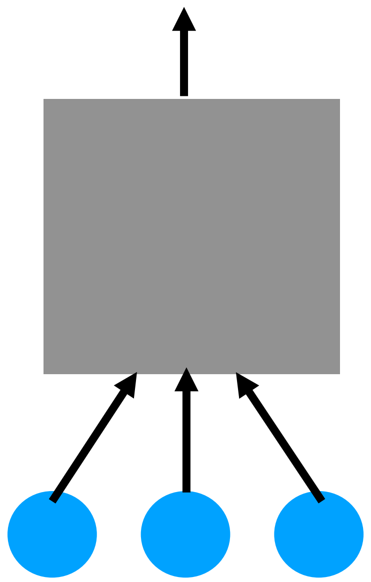


$$y = \mathrm{MLP}_2\left(\sum_{s \in S} \mathrm{MLP}_1(X_s)\right)$$



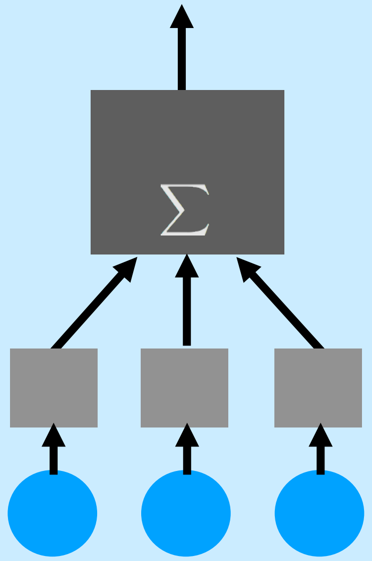
Architectures

MLP



Deep Sets

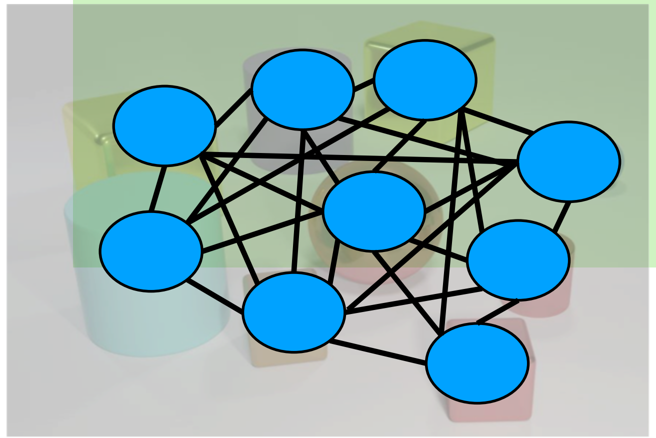
(Zaheer et al 2017)



$$y = \mathrm{MLP}_2\left(\sum_{s \in S} \mathrm{MLP}_1(X_s)\right)$$

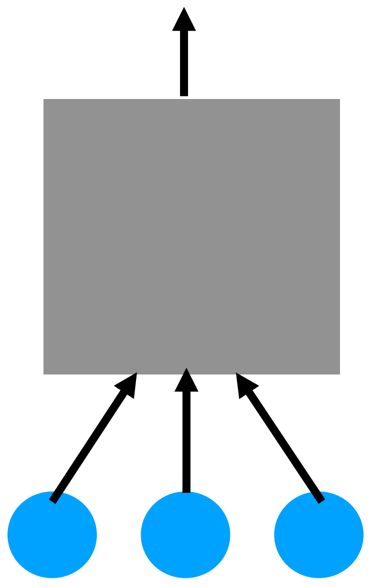
Graph Neural Network

(Battaglia et al, 2018)



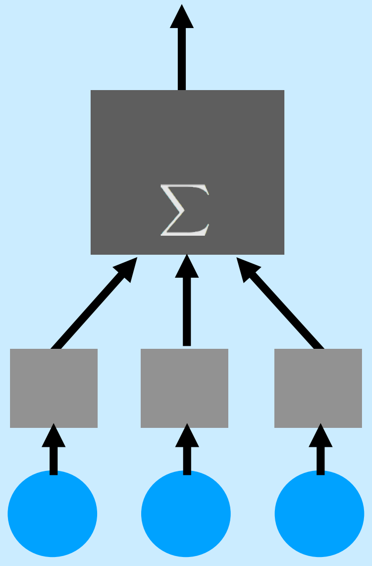
Architectures

MLP



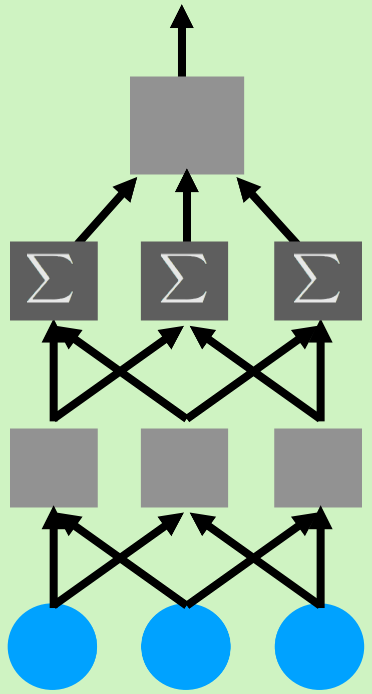
Deep Sets

(Zaheer et al 2017)



Graph Neural Network

(Battaglia et al, 2018)



$$y = \mathrm{MLP}_2\left(\sum_{s \in S} \mathrm{MLP}_1(X_s)\right)$$

$$h_s^{(k+1)} = \sum_{t \in S} \mathrm{MLP}_1^{(k)}\left(h_s^{(k)}, h_t^{(k)}\right) \qquad y = \mathrm{MLP}_2\left(\sum_{s \in S} h_s^{(K)}\right)$$

“equivalent” by representational power,

But big empirical differences in learning!

Idea

- Algorithms are structured arrangements of subroutines

- Neural networks are structured arrangements of learnable “modules”

  formalize ***inductive bias?***

Algorithmic Alignment: Network can mimic algorithm
via *few, easy-to-learn* “modules”

Hypothesis: Alignment facilitates learning

Algorithmic Alignment

Algorithmic Alignment: Network can mimic algorithm via *few, easy-to-learn* "modules"

Bellman-Ford

for k = 1 … |S| - 1:

  for u in S:

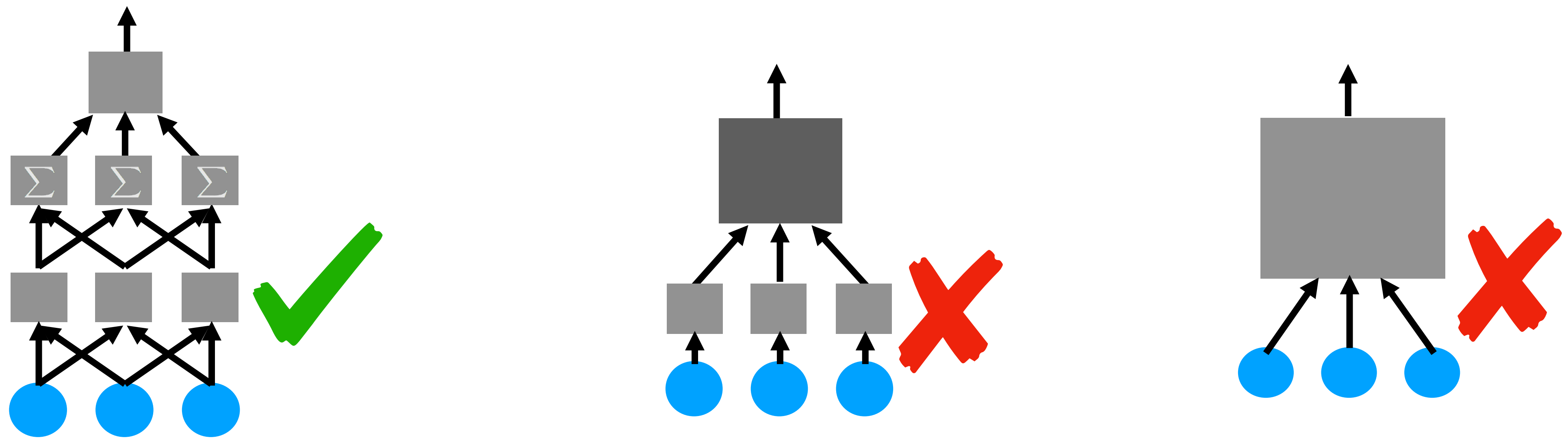
    $d[k][u] = \min_v d[k-1][v] + \text{cost}(v, u)$

GNN

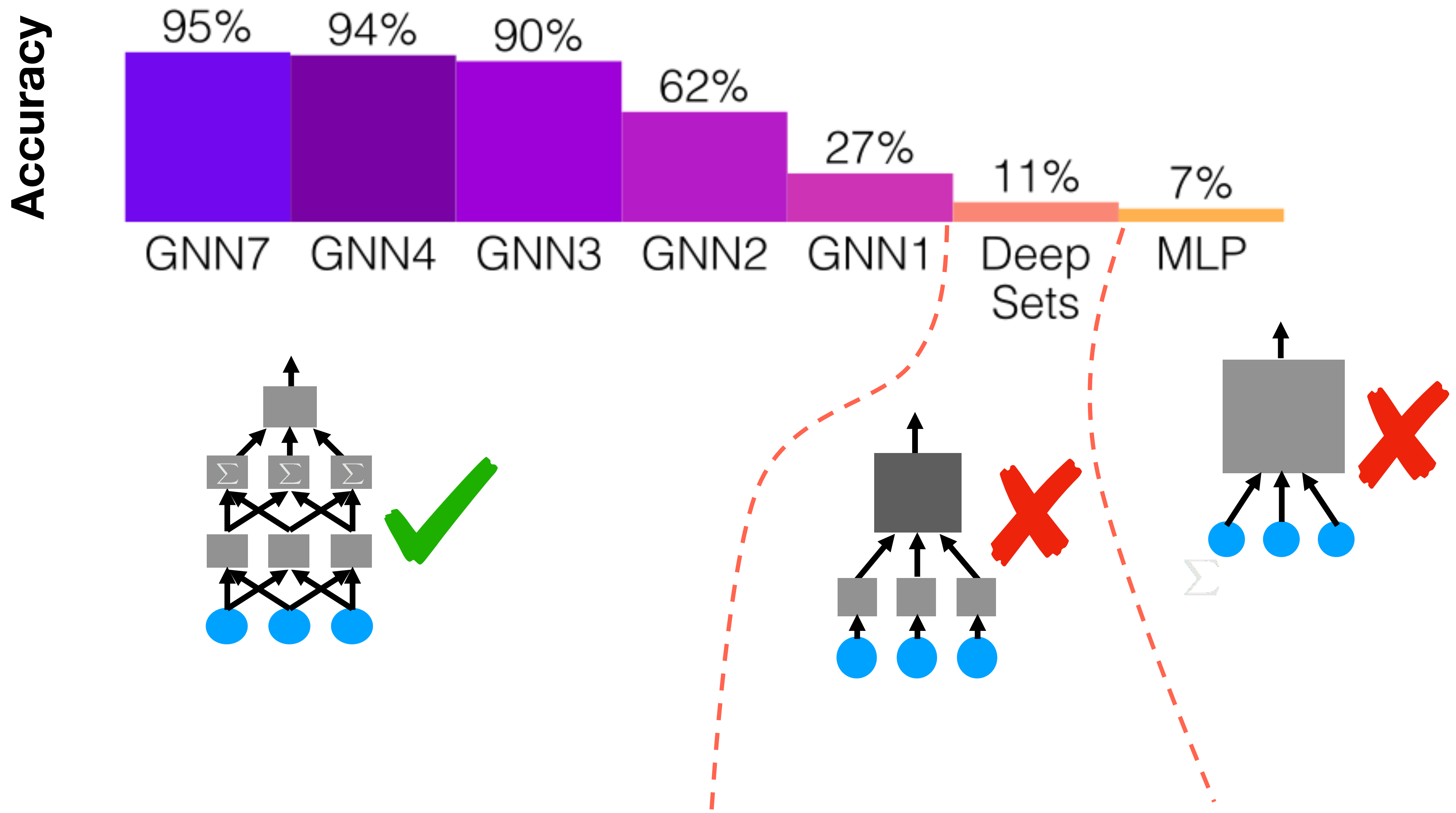
for k = 1 … GNN iter:

  for u in S:

    $h_u^{(k)} = \Sigma_v \text{MLP}(h_v^{(k-1)}, h_u^{(k-1)})$



Empirical Evidence



Alignment leads to a **hierarchy / classification of tasks**.
Predicts which architectures suit which tasks.

Alignment more generally

More generally: **GNNs align with Dynamic Programming**

$$\text{Answer}[k][i] = \text{DP-Update}(\{\text{Answer}[k-1][j],\ j = 1 \ldots n\})$$

$$h_s^{(k)} = \sum\nolimits_{t \in S} \text{MLP}_1^{(k)}\left(h_s^{(k-1)}, h_t^{(k-1)}\right)$$

Many algorithmic / physical reasoning tasks are DPs!

Formalization:

A neural network (M, ϵ, δ)-aligns with an algorithm if it can mimic the algorithm via *n* different (shared) network modules, each of which is PAC-learnable with at most M/n samples.

Implications

Theorem (informal)

Assume the network and some algorithm for the target task (M, ϵ, δ)-align.
Then, under assumptions*, the task is $(M, O(\epsilon), O(\delta))$-learnable by the network.

* *algorithmic stability, Lipschitz continuous modules, layer-wise training.*
Implemented e.g. in Veličković et al 2020.

E.g. via NTK-bounds, can get separation of sample complexity for MLP and GNN.

How well do (Graph) Neural Networks learn such tasks?

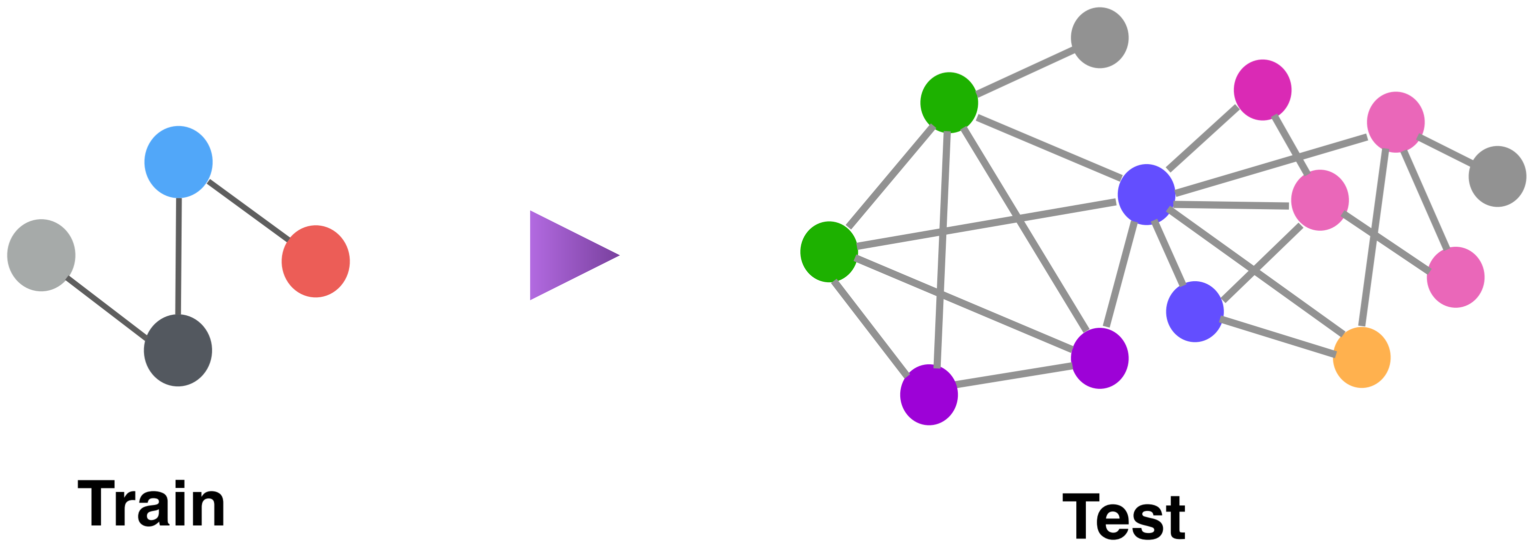
What does this depend on?

- Generalization and architectural structure
  K. Xu, J. Li, M. Zhang, S. Du, K. Kawarabayashi, S. Jegelka. ICLR 2020

- **Extrapolation, structure and nonlinearities**
  K. Xu, J. Li, M. Zhang, S. Du, K. Kawarabayashi, S. Jegelka, ICLR 2021

Extrapolation

What happens outside the support of the training distribution?



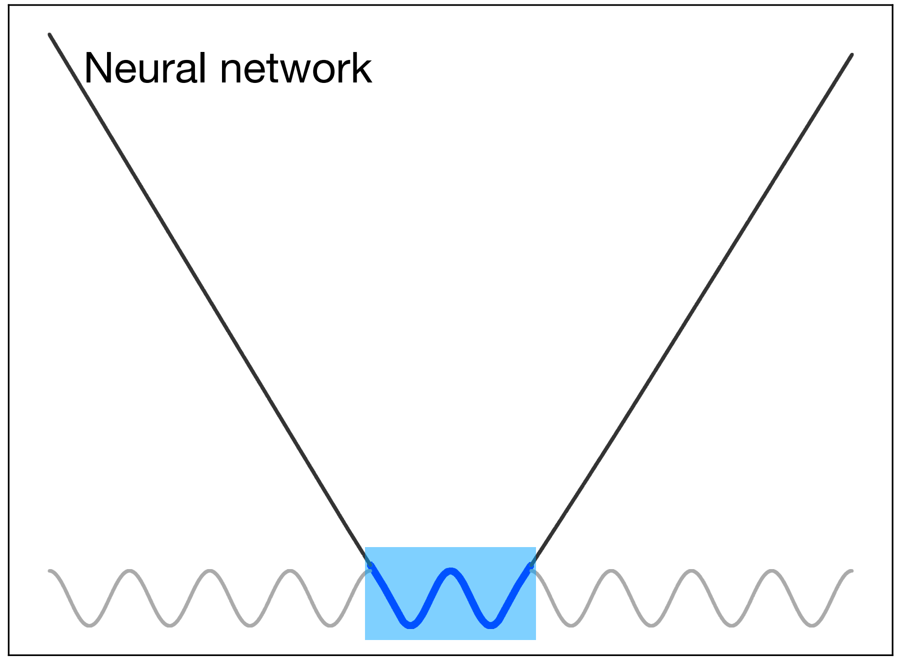
Generalize across graph structures, degrees, node features…?

Prior works:

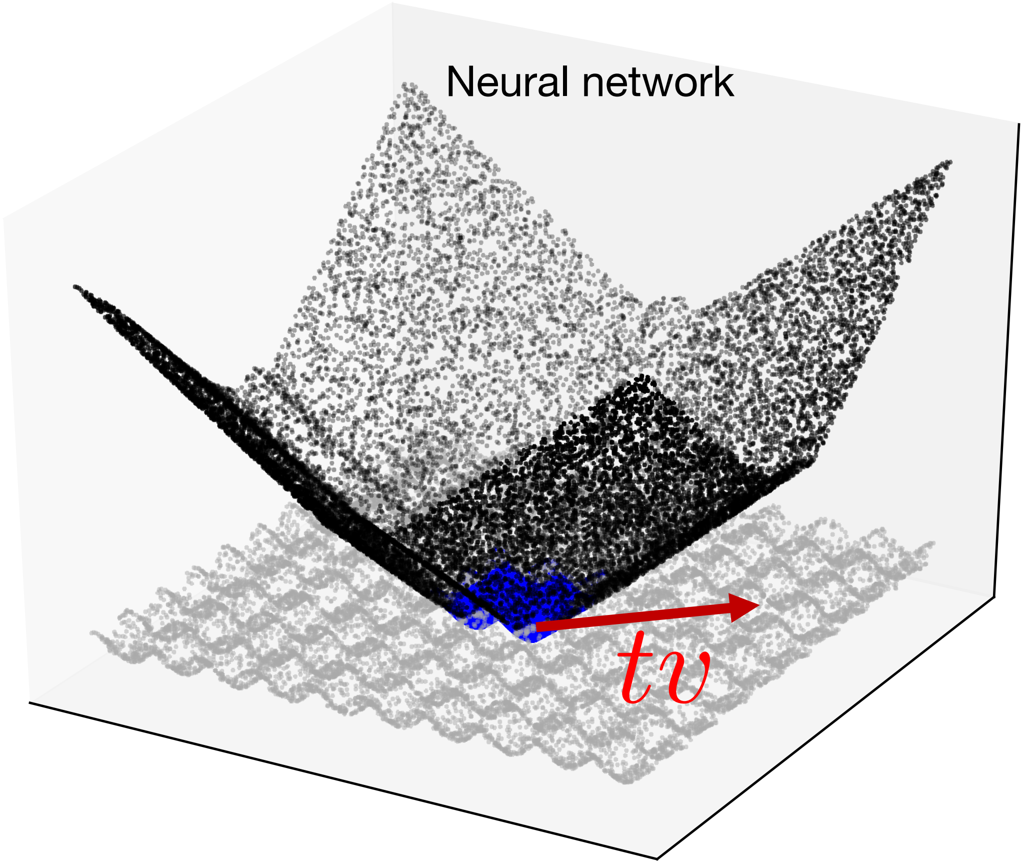
GNNs can sometimes successfully extrapolate to larger graphs.
(Battaglia et al. 2016, 2018; Lample and Charton 2020, Velickovic et al., 2020 ...)

MLPs and ConvNets can “fail” out of distribution.
(Barnard and Wessels,1992; Haley and Soloway, 1992; Santoro et al. 2018; Arjovsky et al. 2019...)

ReLu feedforward networks



Training data



Theorem *(XLZDKJ21)*
Let f be a 2-layer ReLu MLP tained by GD. For any direction $v \in \mathbb{R}^d$ let $x = tv$. As $t \to \infty$: $f(x + hv) - f(x) \to \beta_v h$ with rate $O(1/t)$.

(Linear regions: Montufar et al 2014, Arora et al 2018, Hanin & Rolnick, 2019; Hein et al., 2019, XZDKJ20)

What could this mean for GNNs?

Shortest Path (target): $d[k][u] = \min_{v \in \mathcal{N}(u)} d[k-1][v] + \boldsymbol{w}(v,u)$

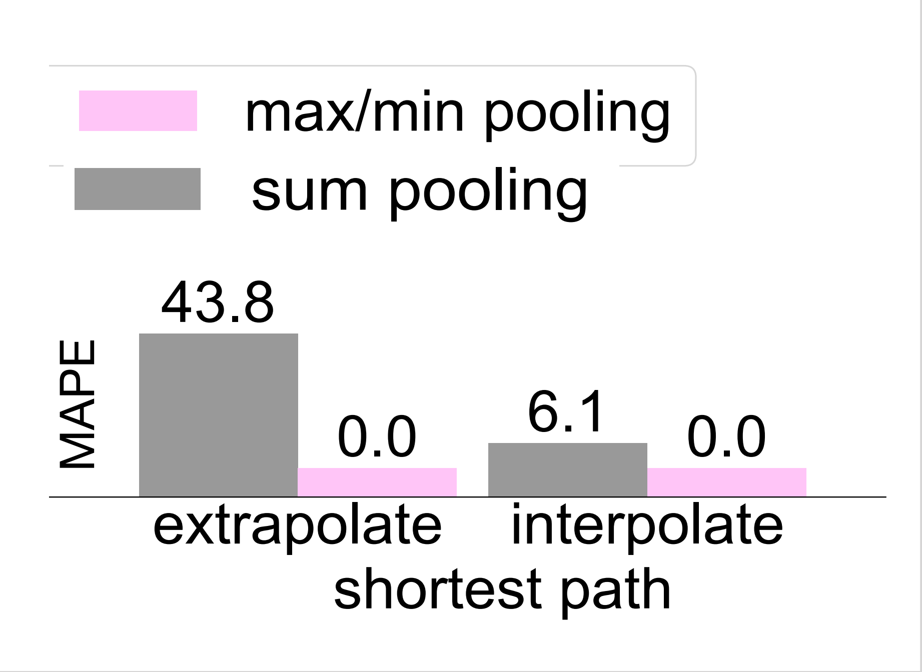
GNN (sum): $\boldsymbol{h}_u^{(k)} = \sum_{v \in \mathcal{N}(u)} \text{MLP}^{(k)}\left(\boldsymbol{h}_u^{(k-1)}, \boldsymbol{h}_v^{(k-1)}, \boldsymbol{w}_{(v,u)}\right)$

MLP learns non-linear function

Battaglia et al 2018, Velickovic et al 2020: extrapolation with

$$\boldsymbol{h}_u^{(k)} = \min_{v \in \mathcal{N}(u)} \text{MLP}^{(k)}\left(\boldsymbol{h}_u^{(k-1)}, \boldsymbol{h}_v^{(k-1)}, \boldsymbol{w}_{(v,u)}\right)$$

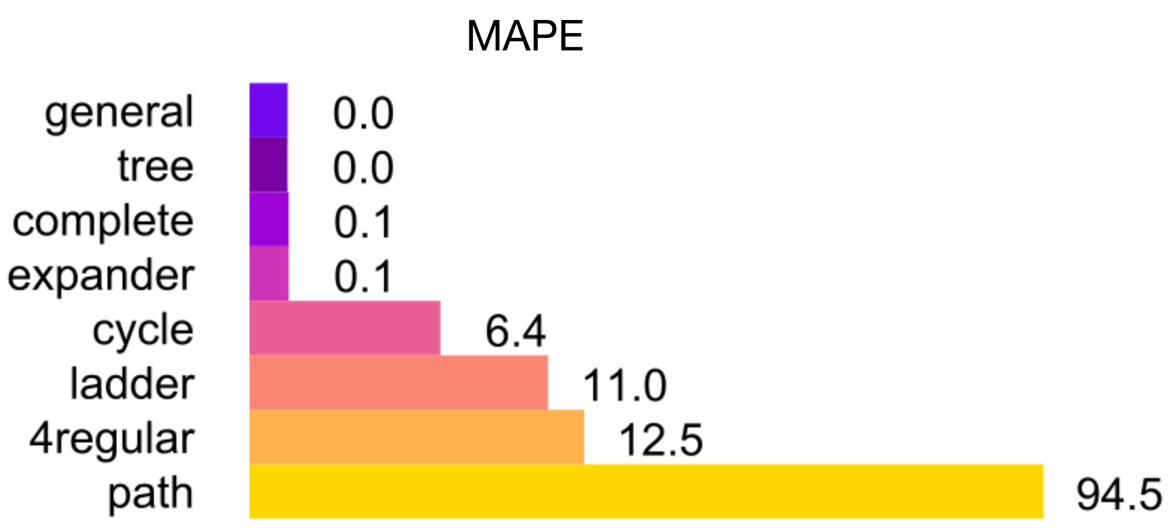
MLP learns linear function



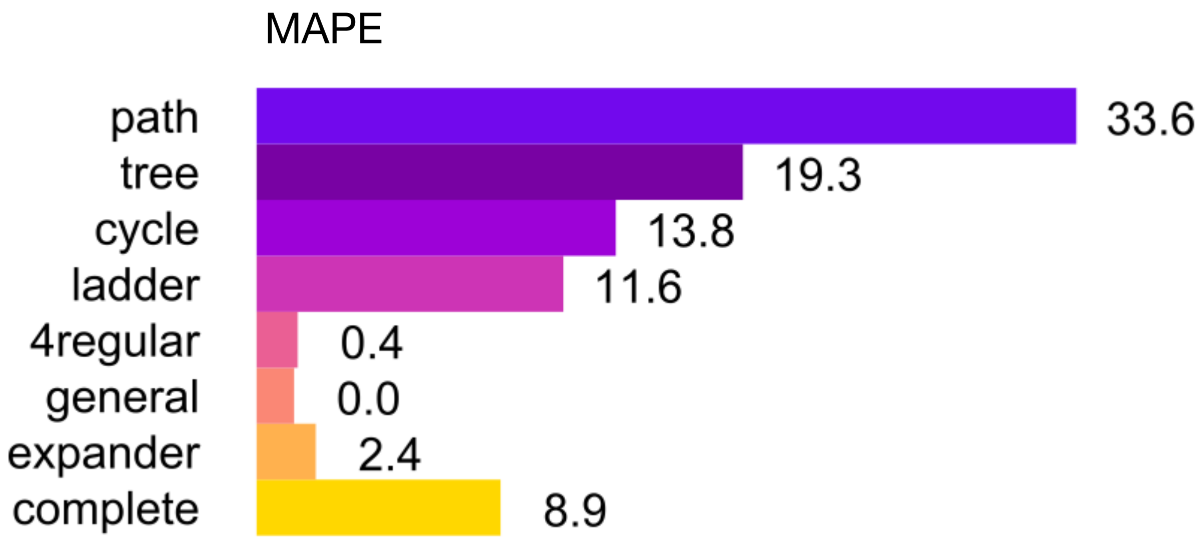
Hypothesis: ***Linear*** algorithmic alignment helps *extrapolation*
(formal proof for special cases)

Encode nonlinearities
in architecture or features.

Importance of training graphs



Max degree



Shortest Path

Lemma *(XZDKJ21)*: A max-aggregation GNN in the NTK regime learns Max-degree in the NTK regime under conditions on the training data: $\{\deg_{\max}(G_i), \deg_{\min}(G_i), N_i^{\max} \deg_{\max}(G_i), N_i^{\min} \deg_{\min}(G_i)\}_{i=1}^{n}$ spans $\mathbb{R}^4$

Summary: Task Structure and generalization

- **Generalization within distribution**:
  algorithmic alignment formalizes inductive bias

- **Extrapolation**:
  nonlinearities matter: *linear algorithmic alignment*
  ➔ encode nonlinearities in architecture (aggregation) or features

K. Xu, J. Li, M. Zhang, S. Du, K. Kawarabayashi, S. Jegelka. What Can Neural Networks Reason About? *ICLR*, 2020.

K. Xu, J. Li, M. Zhang, S. Du, K. Kawarabayashi, S. Jegelka. How Neural Networks Extrapolate: From Feedforward to Graph Neural Networks. *ICLR*, 2021.