

What can neural networks reason about?

Shashank Shekhar

Masters in Applied Science candidate,
Machine Learning Research Group, University of Guelph
Vector Scholar, Vector Institute

Presentation for Deep Learning Reading Club, UWaterloo, Dec 10, 2020

Credits

Eric J Taylor for presentation layout

Parts of slides borrowed from [Anushree Hede](#)

Key takeaways (spoiler alert!)

Published as a conference paper at ICLR 2020

WHAT CAN NEURAL NETWORKS REASON ABOUT?

**Keyulu Xu[†], Jingling Li[‡], Mozhi Zhang[‡], Simon S. Du[§], Ken-ichi Kawarabayashi[¶],
Stefanie Jegelka[†]**

[†]Massachusetts Institute of Technology (MIT)

[‡]University of Maryland

[§]Institute for Advanced Study (IAS)

[¶]National Institute of Informatics (NII)

{keyulu, stefje}@mit.edu

* Theoretical paper but I will be talking at a high level about the key idea proposed and its implications

Key takeaways

- The reasoning process has a certain algorithmic structure
- Whether a neural net architecture is able to learn to perform a reasoning task depends on how much its structure aligns with this algorithmic structure
- Authors present a theoretical framework for measuring this in terms of sample efficiency called **ALGORITHMIC ALIGNMENT**
- Demonstrate empirically the usefulness of this measure:
Reasoning tasks: Reasoning problems like Visual question answering, Shortest path etc can be solved using algorithmic solutions. Use 3 increasingly complex algorithmic solution approaches: summary statistics, relational argmax, dynamic programming
NN architectures: MLP, Deep Sets, GNN

Presentation outline

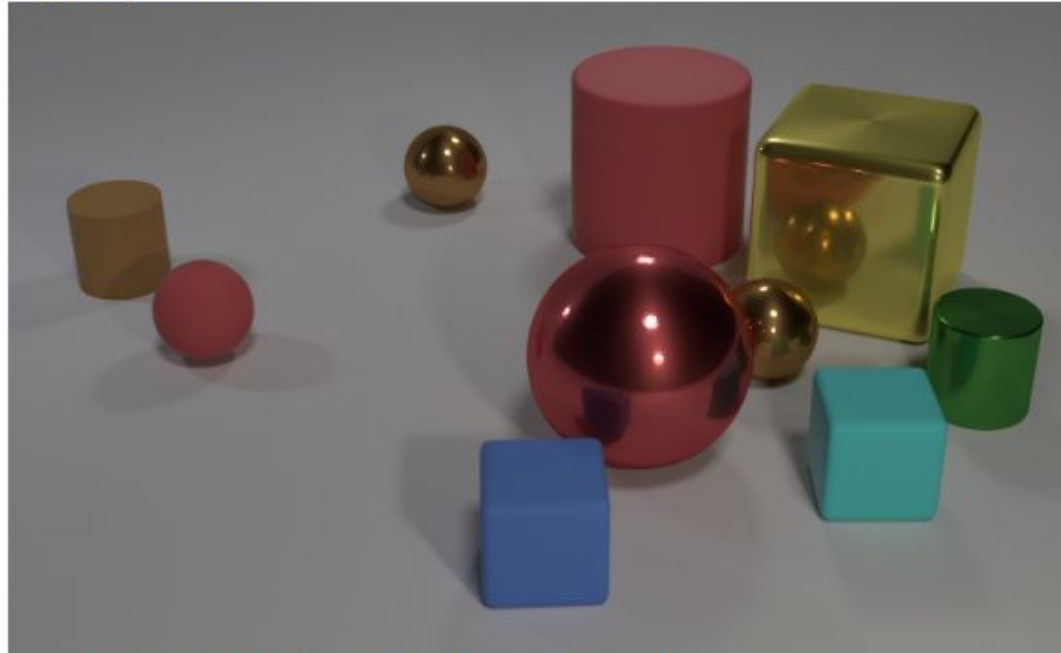
1. Reasoning and algorithms
2. Neural Network structure
3. Algorithmic Alignment
4. Results

Presentation outline

1. Reasoning and algorithms
2. Neural Network structure
3. Algorithmic Alignment
4. Results

Reasoning

Questions in CLEVR test various aspects of visual reasoning including **attribute identification**, **counting**, **comparison**, **spatial relationships**, and **logical operations**.



- Q: Are there an **equal number** of **large things** and **metal spheres**?
- Q: **What size** is the **cylinder that is left of** the **brown metal** thing **that is left of** the **big sphere**?
- Q: There is a **sphere** with the **same size as** the **metal cube**; is it **made of the same material as** the **small red sphere**?
- Q: **How many** objects are **either small cylinders** or **red things**?

Reasoning: Problem Formalization

- Universe S = set of objects to reason about
- Each object $s \in S$ is represented by a feature vector $X = [h_1, h_2, \dots, h_k]$
- Vector X can be state descriptions, features learned from data (images, questions etc.)
- Information about a specific 'question' can be encoded in X
- Given a set of universes $\{S_1, S_2, \dots, S_m\}$ and answer labels $y = \{y_1, y_2, \dots, y_m\}$

learn a function which can answer questions about unseen universe $y = g(S)$

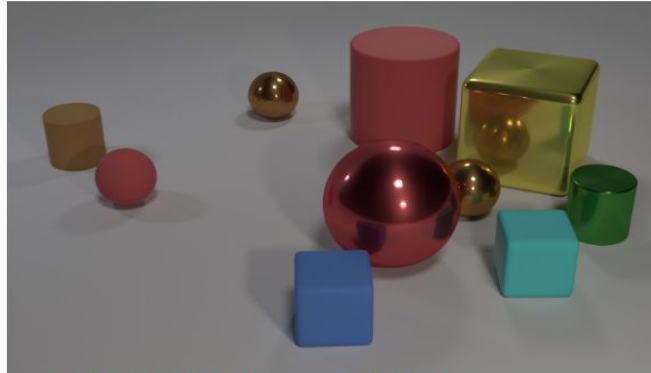
Reasoning: Problem Formalization

- **VQA**
 - **Universe:** Images/Questions
 - **Objects:** Objects in the image/question
 - **Answers:** Answer

- **Shortest Path**
 - **Universe:** Graph
 - **Objects:** Nodes/Edges
 - **Answers:** Shortest path

Reasoning: summary statistics

Questions in CLEVR test various aspects of visual reasoning including **attribute identification**, **counting**, **comparison**, **spatial relationships**, and **logical operations**.



Q: Are there an **equal number** of **large things** and **metal spheres**?

Q: **What size** is the **cylinder that is left of the brown metal thing that is left of the big sphere**?

Q: There is a **sphere** with the **same size as the metal cube**; is it **made of the same material as the small red sphere**?

Q: **How many** objects are **either small cylinders or red things**?

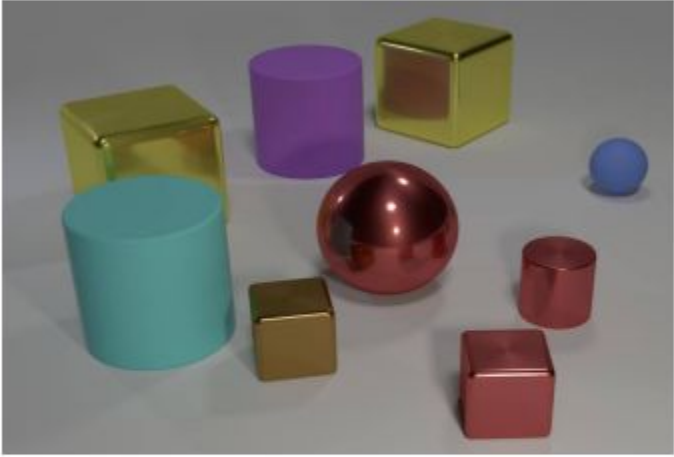


Summary statistics

What is the maximum value difference among treasures?

Max/Min/Sum etc of (features of) all objects

Reasoning: relational argmax



Compare pairwise relations between objects and answer a question about those pairwise results

Relational argmax

What are the colors of the furthest pair of objects?

Reasoning: dynamic programming



Dynamic programming

What is the cost to defeat monster X
by following the optimal path?

Several relational reasoning tasks can be solved using a dynamic programming algorithm

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

e.g. Shortest path can be solved using Bellman Ford

Reasoning: dynamic programming

VQA: “Starting at object X , if each time we jump to the closest object, which object is K jumps away?”

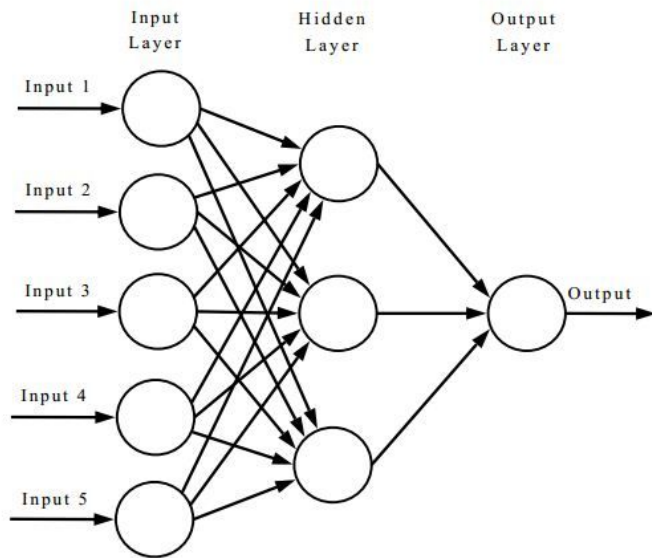
$\text{closest}[1][i] = \arg \min_j d(i, j)$, $\text{closest}[k][i] = \text{closest}[k - 1][\text{closest}[1][i]]$

Intuitive physics: Authors also show how moving objects and force interactions, which are another popular area of AI reasoning research, can be modelled as dynamic programming updates (refer paper)

Presentation outline

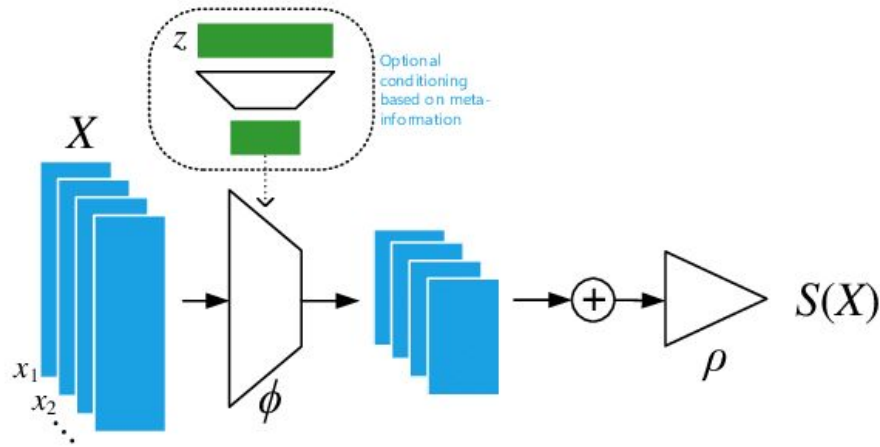
1. Reasoning and algorithms
2. Neural Network structure
3. Algorithmic Alignment
4. Results

Network Structure: MLP



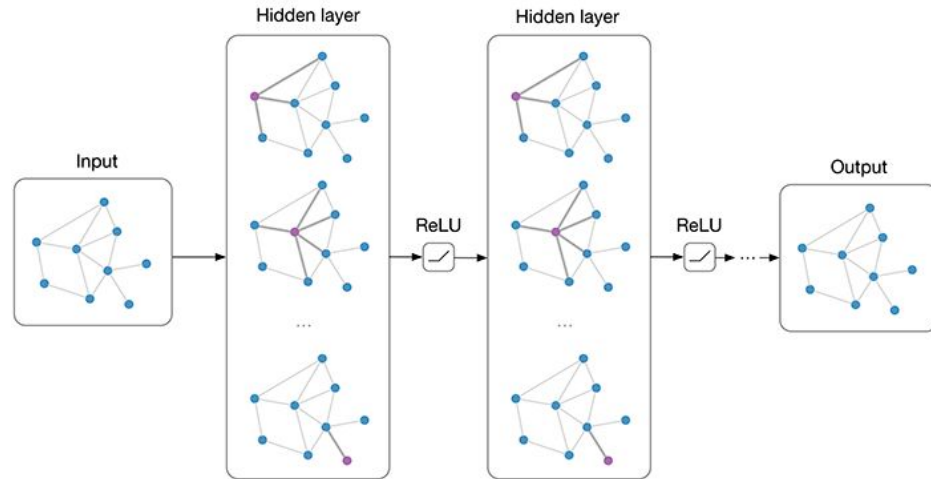
- No inherent relational structure
- Works well for single object universes (image classification)
- Has a hard time generalizing to multi-object universes if trained on concatenated object representations

Network Structure: Deep Sets



- $y = \text{MLP}_2 [\sum_{s \in S} \text{MLP}_1 (X_s)]$
- Can (in principle) learn permutation invariant functions of objects
- Think summary statistics: MLP_1 can learn features of objects and MLP_2 can then learn summary statistics

Network Structure: GNNs



- Message passing scheme where at iteration k the representation $h_s^{(k)}$ is recursively updated by aggregating representations of neighbouring states

- $$h_s^{(k)} = \sum_{t \in S} \text{MLP}^{(k)} [h_s^{(k-1)}, h_t^{(k-1)}]$$
$$h_s = \text{MLP}_2(\sum_{s \in S} h_s^{(k-1)})$$

$$h_s = \text{output}, K = \# \text{layers}, h_s^{(0)} = X_s$$

- Permutation invariant (like deep sets). Unlike deep sets which focus on individual objects, GNNs can also focus on pairwise relations (think relational argmax)

Network structure

- Empirically, GNNs are able to learn 'better' than Deep Sets when relations of objects are involved
- In theory, all three networks should be able to learn any permutation invariant continuous function over sets of object representations with bounded cardinality (for proofs see paper)
- Thus, the difference in accuracy must be arising from different generalization ability

Presentation outline

1. Reasoning and algorithms
2. Neural Network structure
3. Algorithmic Alignment
4. Results

Network structure and algorithms

- Intuitively, a network may generalize better if its able to represent a function more 'easily'
- e.g. CNNs perform great on images because convolution filters are translation invariant for objects in the images
- The inductive bias of *'the neural network's architecture induces a computational structure on the function it computes'*

Network structure and algorithms

Graph Neural Network

for $k = 1 \dots$ GNN iter:

for u in S : *No need to learn for-loops*

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

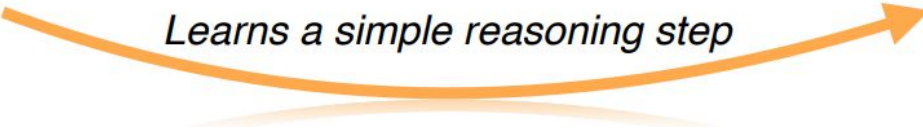
Bellman-Ford algorithm

for $k = 1 \dots |S| - 1$:

for u in S :

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

Learns a simple reasoning step



- e.g. Bellman-Ford algorithm outlines the correct reasoning process to solve a shortest path problem.
- GNN can simulate Bellman Ford if it's able to learn the relaxation step in the last line (sum->min over neighboring nodes v) via its aggregation operation
- However, an MLP or Deep Set would have to learn the structure of the entire for loop.

PAC learning framework

Definition 3.3. (PAC learning and sample complexity). Fix an error parameter $\epsilon > 0$ and failure probability $\delta \in (0, 1)$. Suppose $\{x_i, y_i\}_{i=1}^M$ are i.i.d. samples from some distribution \mathcal{D} , and the data satisfies $y_i = g(x_i)$ for some underlying function g . Let $f = \mathcal{A}(\{x_i, y_i\}_{i=1}^M)$ be the function generated by a learning algorithm \mathcal{A} . Then g is (M, ϵ, δ) -*learnable* with \mathcal{A} if

$$\mathbb{P}_{x \sim \mathcal{D}} [\|f(x) - g(x)\| \leq \epsilon] \geq 1 - \delta. \quad (3.1)$$

The *sample complexity* $\mathcal{C}_{\mathcal{A}}(g, \epsilon, \delta)$ is the minimum M so that g is (M, ϵ, δ) -learnable with \mathcal{A} .

Algorithmic Alignment

Definition 3.4. (Algorithmic alignment). Let g be a reasoning function and \mathcal{N} a neural network with n modules \mathcal{N}_i . The module functions f_1, \dots, f_n generate g for \mathcal{N} if, by replacing \mathcal{N}_i with f_i , the network \mathcal{N} simulates g . Then \mathcal{N} (M, ϵ, δ) -algorithmically aligns with g if (1) f_1, \dots, f_n generate g and (2) there are learning algorithms \mathcal{A}_i for the \mathcal{N}_i 's such that $n \cdot \max_i C_{\mathcal{A}_i}(f_i, \epsilon, \delta) \leq M$.

Formally, a neural network aligns with an algorithm if

1. It can simulate it via a limited number of modules
2. Each module is simple i.e. has low sample complexity

'Good algorithmic alignment, i.e., small M , implies that all algorithm steps f_i to simulate the algorithm g are easy to learn.'

Calculating algorithmic alignment

Paper does not derive any end-to-end learning results. A derivation for sequential training of modules with auxiliary labels for MLPs is provided.

- Functions that are “simple” when expressed as a polynomial (e.g. via a Taylor expansion) can be sample-efficiently learned by an MLP. Algorithm steps which perform computations over many objects (e.g. for loops) lead to higher sample complexity for MLPs

In same simplified setting, paper proves that sample complexity bound increases with increase in algorithmic alignment value M

(*For proofs and derivations please refer to original paper)

Presentation outline

1. Reasoning and algorithms
2. Neural Network structure
3. Algorithmic Alignment
4. Results

Theoretical result

Corollary 3.7. *Suppose universe S has ℓ objects X_1, \dots, X_ℓ , and $g(S) = \sum_{i,j} (X_i - X_j)^2$. In the setting of Theorem 3.6, the sample complexity bound for MLP is $O(\ell^2)$ times larger than for GNN.*

Theoretically, for a summary statistic task (sum of pairwise squares), the sample complexity bound for MLP is $O(\text{objects}^2)$ times larger than for GNN

Empirical Settings

- **Summary Statistics**

- **Maximum value difference** : Each object is a treasure $X = [h_1, h_2, h_3]$ with location h_1 , value h_2 and color h_3 . Models have to predict difference in value between most and least valuable treasure

$$y(S) = \max_{s \in S} h_2(X_s) - \min_{s \in S} h_2(X_s).$$

- **Relational argmax**

- **Furthest pair**: same object setting as before. Train models to find colors of objects with the largest distance (encoded as an integer category representing pair of colors)

$$y(S) = (h_3(X_{s_1}), h_3(X_{s_2})) \quad \text{s.t.} \quad \{X_{s_1}, X_{s_2}\} = \arg \max_{s_1, s_2 \in S} \|h_1(X_{s_1}) - h_1(X_{s_2})\|_{\ell_1}$$

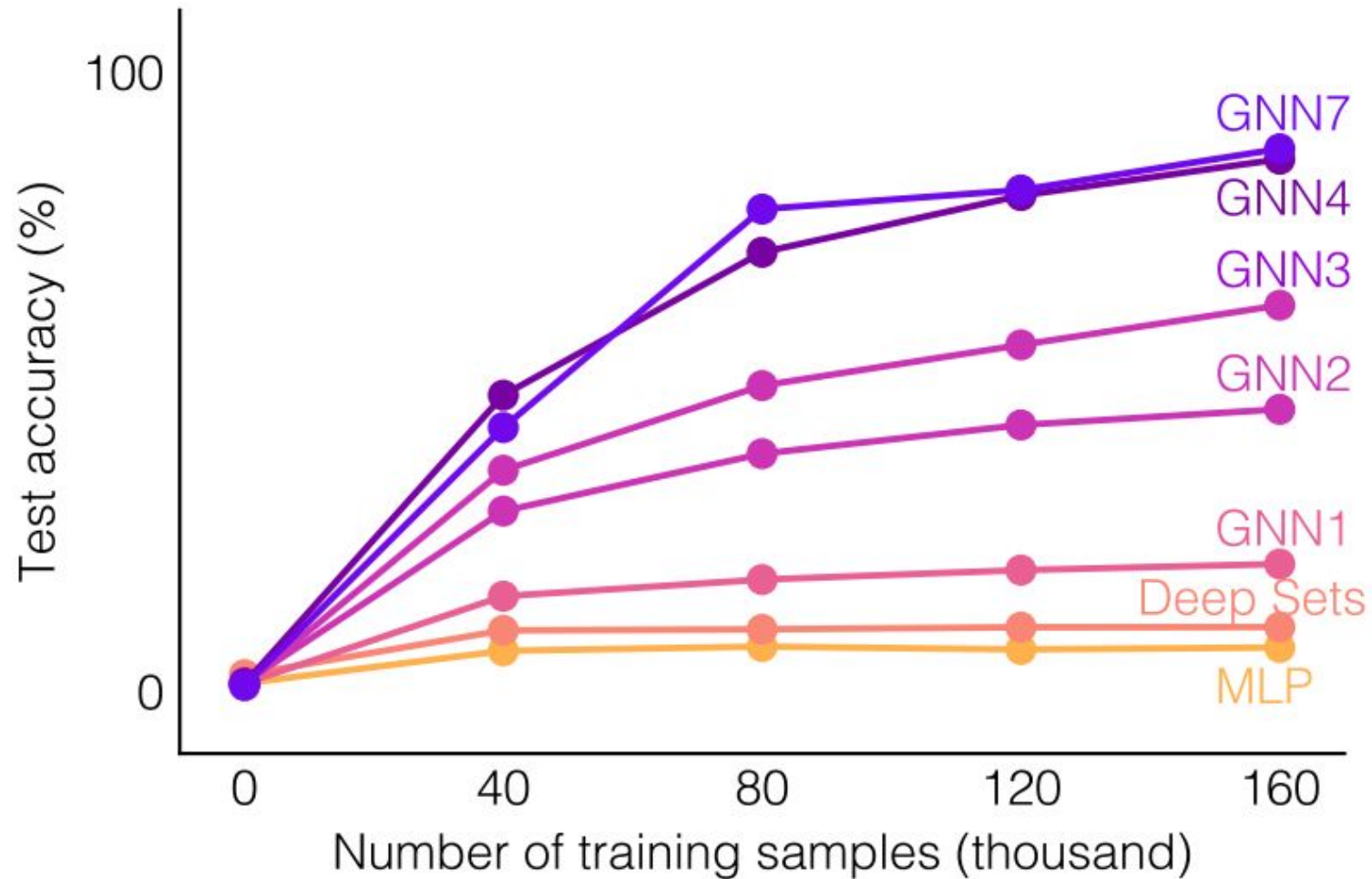
- **Dynamic programming**

- **Shortest Path**: solved using Bellman-Ford previously discussed

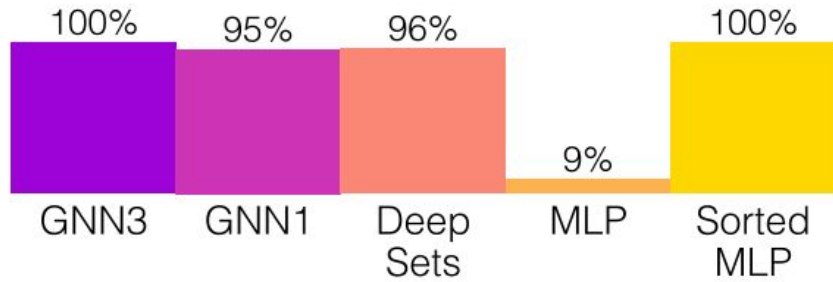
- **NP-Hard problems**

- **Subset sum**: Given a set of numbers, does there exist a subset of numbers which sums to zero?

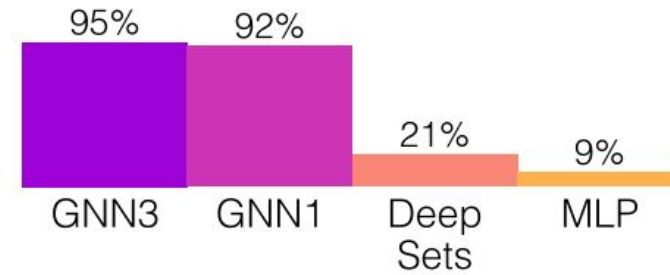
Empirical Results: Sample efficiency (DP task)



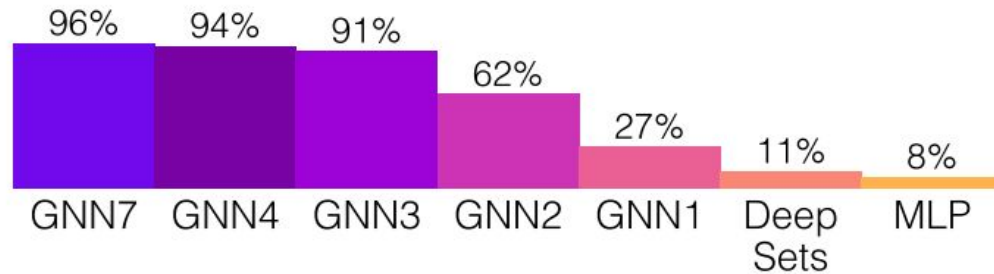
Empirical Results: Test Accuracy



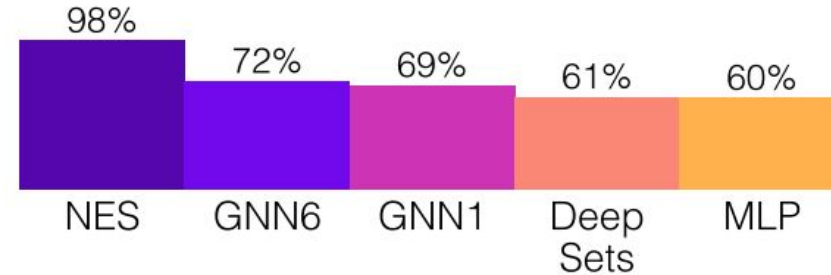
(a) Maximum value difference.



(b) Furthest pair.



(c) Monster trainer.



(d) Subset sum. Random guessing yields 50%.

Thank You