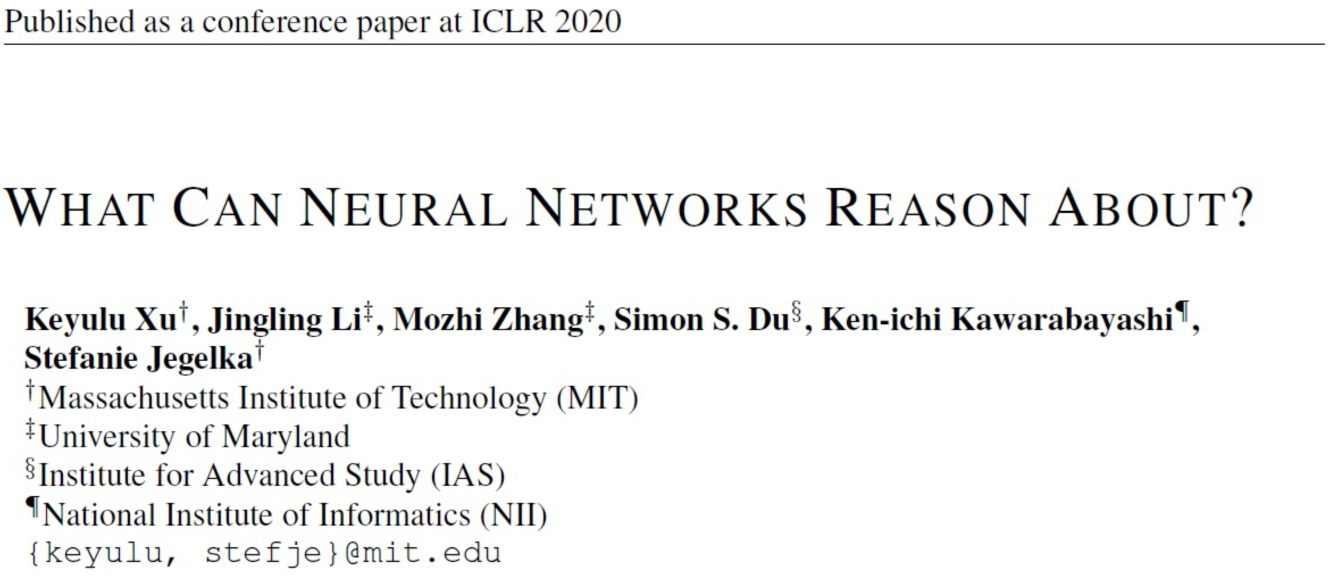


Published as a conference paper at ICLR 2020

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

【深度学习 106】NN Reasoning

张楚珩
清华大学 交叉信息院博士在读

98 人赞同了该文章

最新的 ICLR 2020 的一篇文章，一个理论工作，告诉我们啥样的神经网络结构能做啥样的推理任务。

原文传送门

Xu, Keyulu, et al. "What Can Neural Networks Reason About?." arXiv preprint arXiv:1905.13211 (2019).

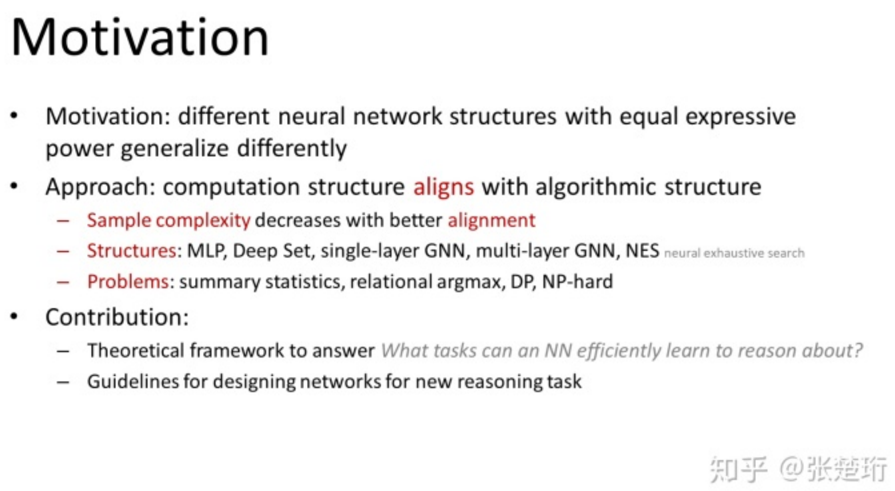
特色

在理论上从简单的 MLP 到结构比较复杂的 GNN 都被证明能拟合任意函数，但是在实际应用中我们发现在某些特定的问题上一些特定的网络结构表现会更好。这篇文章就给出了一个理论框架，来说明何种神经网络结构适合解决何种推理任务。具体地，这篇文章提出了衡量网络结构和任务匹配程度的指标 algorithmic alignment，并证明了好的 algorithmic alignment 将带来更好的 sample complexity。

本文在实验上也做了相应的印证。本文在由易到难的多种任务（包括 summary statistics、relational argmax、DP、NP-hard problem）上做了实验，并且使用了不同的神经网络结构（包括 MLP、Deep Sets、GNN 和本文针对 subset sum 问题设计 neural exhaustive search）。

过程

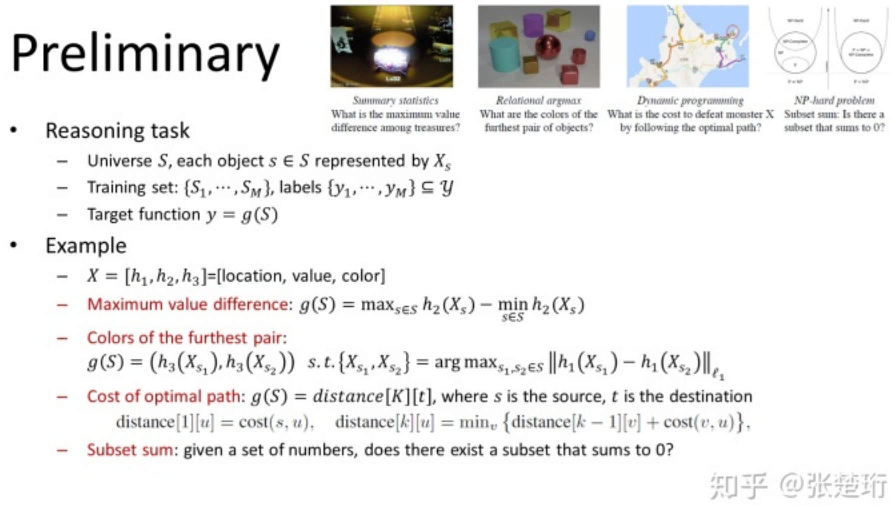
1. 概述



Motivation

- Motivation: different neural network structures with equal expressive power generalize differently
- Approach: computation structure aligns with algorithmic structure
  - Sample complexity decreases with better alignment
  - Structures: MLP, Deep Set, single-layer GNN, multi-layer GNN, NES neural exhaustive search
  - Problems: summary statistics, relational argmax, DP, NP-hard
- Contribution:
  - Theoretical framework to answer *What tasks can an NN efficiently learn to reason about?*
  - Guidelines for designing networks for new reasoning task

文章举例考虑了一些不同的任务，这里列出了这些任务的具体数学表示。



Preliminary

Summary statistics
What is the maximum value difference among treasures?

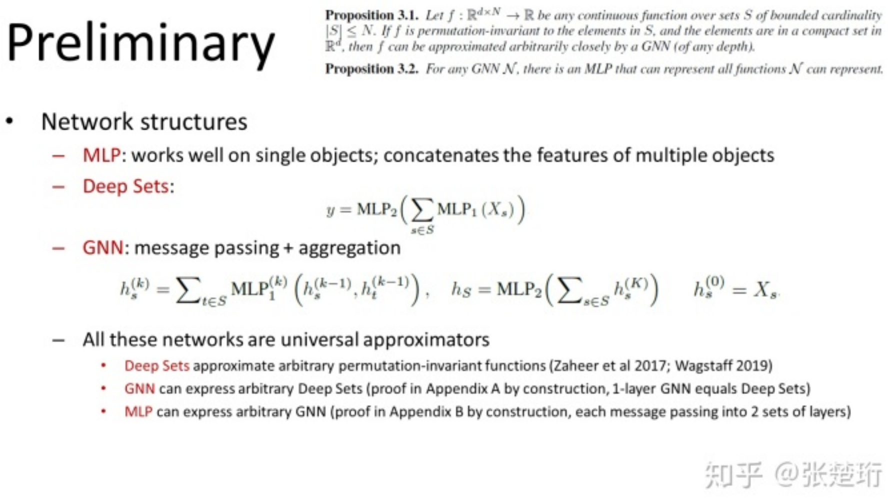
Relational argmax
What are the colors of the furthest pair of objects?

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

NP-hard problem
Subset sum: Is there a subset that sums to 0?

- Reasoning task
  - Universe S, each object $s \in S$ represented by X_s
  - Training set: $\{S_1, \cdots, S_M\}$, labels $\{y_1, \cdots, y_M\} \subseteq \mathcal{Y}$
  - Target function $y = g(S)$
- Example
  - $X = [h_1, h_2, h_3]$=[location, value, color]
  - Maximum value difference: $g(S) = \max_{s \in S} h_2(X_s) - \min_{s \in S} h_2(X_s)$
  - Colors of the furthest pair:
    $g(S) = \left(h_3(X_{s_1}), h_3(X_{s_2})\right) \;\; s.t. \left\{X_{s_1}, X_{s_2}\right\} = \arg\max_{s_1, s_2 \in S} \left\| h_1(X_{s_1}) - h_1(X_{s_2}) \right\|_{\ell_1}$
  - Cost of optimal path: $g(S) = distance[K][t]$, where s is the source, t is the destination
    $\mathrm{distance}[1][u] = \mathrm{cost}(s, u), \quad \mathrm{distance}[k][u] = \min_v \left\{\mathrm{distance}[k-1][v] + \mathrm{cost}(v, u)\right\},$
  - Subset sum: given a set of numbers, does there exist a subset that sums to 0?

接下来，我们看一下文章考虑的若干种不同的网络结构；同时，我们注意到，这些网络结构都是 universal approximators，即任意的函数它们都可以拟合。因此，它们之间的区别在于它们对于不同的任务具有不同的泛化能力；从 PAC learning theory 的角度，它们在不同任务上的 sample complexity 不一样。



Preliminary

Proposition 3.1. *Let $f : \mathbb{R}^{d\times N} \to \mathbb{R}$ be any continuous function over sets S of bounded cardinality $|S| \le N$. If f is permutation-invariant to the elements in S, and the elements are in a compact set in $\mathbb{R}^d$, then f can be approximated arbitrarily closely by a GNN (of any depth).*

Proposition 3.2. *For any GNN $\mathcal{N}$, there is an MLP that can represent all functions $\mathcal{N}$ can represent.*

- Network structures
  - MLP: works well on single objects; concatenates the features of multiple objects
  - Deep Sets:

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

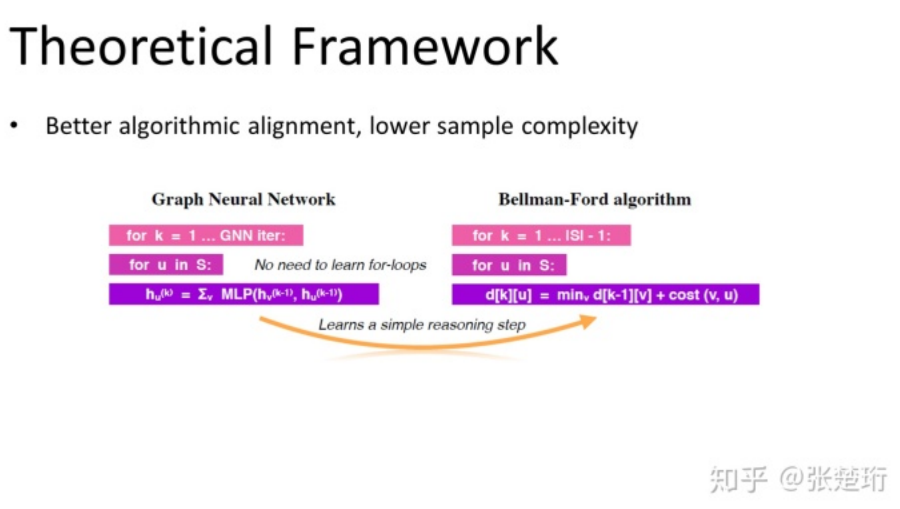
  - GNN: message passing + aggregation

$$h_s^{(k)} = \sum\nolimits_{t\in S}\mathrm{MLP}_1^{(k)}\left(h_s^{(k-1)}, h_t^{(k-1)}\right), \quad h_S = \mathrm{MLP}_2\Big(\sum\nolimits_{s\in S} h_s^{(K)}\Big) \qquad h_s^{(0)} = X_s$$

  - All these networks are universal approximators
    - Deep Sets approximate arbitrary permutation-invariant functions (Zaheer et al 2017; Wagstaff 2019)
    - GNN can express arbitrary Deep Sets (proof in Appendix A by construction, 1-layer GNN equals Deep Sets)
    - MLP can express arbitrary GNN (proof in Appendix B by construction, each message passing into 2 sets of layers)

2. 理论

本文的理论工作上最关键的点在于如下观察：特定的任务一般对应了一些可以*精确求解*这些任务的一些算法，如果神经网络的计算结构和这些算法的计算结构比较相似，那么神经网络就应该更容易在该任务上泛化。比如，文章的一个重要发现就是 GNN 的计算结构和动态规划问题的算法结构比较相似，因此 GNN 更适合解决那些能够被动态规划解决的问题。



Theoretical Framework

- Better algorithmic alignment, lower sample complexity

Graph Neural Network

```
for k = 1 ... GNN iter:
  for u in S:
    $h_u^{(k)} = \Sigma_v \mathrm{MLP}(h_v^{(k-1)}, h_u^{(k-1)})$
```

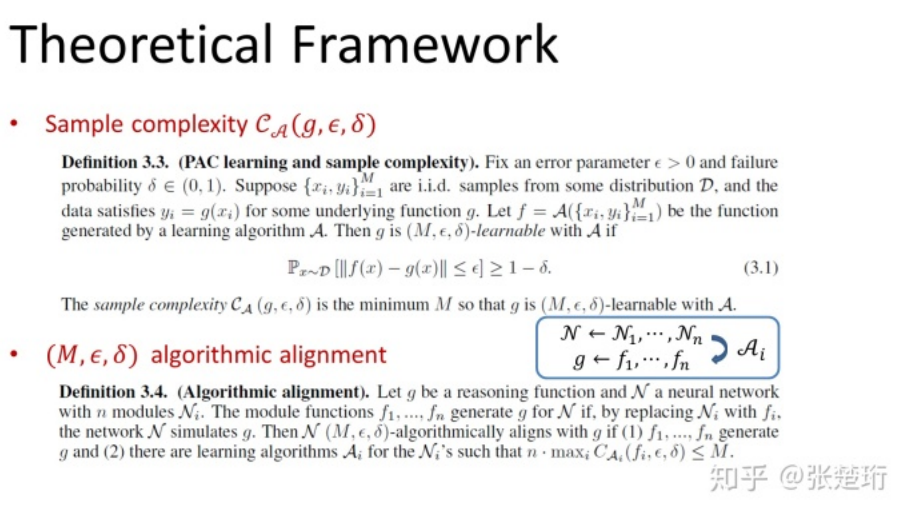
No need to learn for-loops

Bellman-Ford algorithm

```
for k = 1 ... ISI - 1:
  for u in S:
    d[k][u] = $\min_v$ d[k-1][v] + cost (v, u)
```

Learns a simple reasoning step

下面，我们要做的就是考虑如何在理论上刻画这件事情。首先，神经网络结构在某一个问题上是否能够很好的泛化应该如何刻画？这里采用 PAC learning 的框架用 sample complexity 来刻画泛化的难易程度。下一步，我们如何刻画神经网络结构和某个能够精确解决给定任务的算法之间的匹配关系呢？我们假设对应的算法（ground true function）可以拆成和神经网络结构一一对应的若干个部分，假设我们能够有效地训练神经网络的每一个部分去拟合算法中相应的部分，那么我们就认为该结构和该任务比较匹配（aligned）。



Theoretical Framework

- Sample complexity $\mathcal{C}_{\mathcal{A}}(g, \epsilon, \delta)$

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

$$\mathbb{P}_{x \sim \mathcal{D}} \left[\| f(x) - g(x) \| \le \epsilon \right] \ge 1 - \delta. \tag{3.1}$$

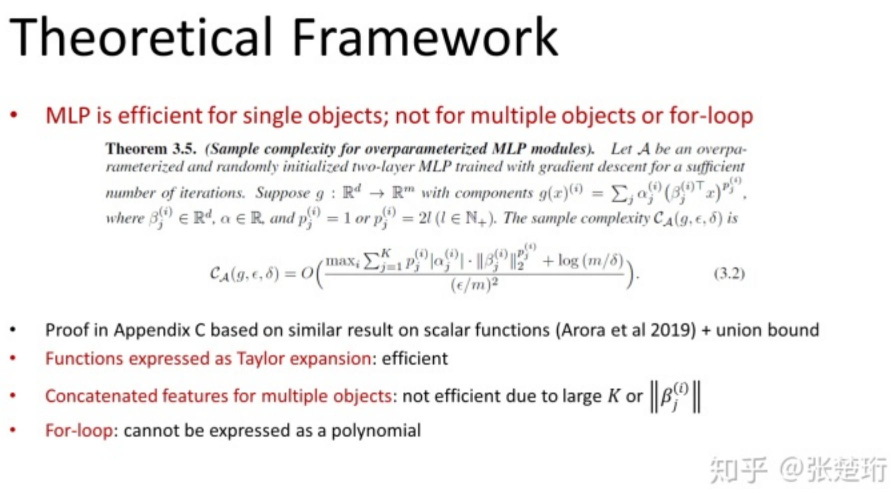
The *sample complexity* $\mathcal{C}_{\mathcal{A}}(g, \epsilon, \delta)$ is the minimum M so that g is (M, ϵ, δ)-learnable with $\mathcal{A}$.

- (M, ϵ, δ) algorithmic alignment

$$\begin{matrix} \mathcal{N} \leftarrow \mathcal{N}_1, \cdots, \mathcal{N}_n \\ g \leftarrow f_1, \cdots, f_n \end{matrix} \; \mathcal{A}_i$$

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 $f_1, ..., f_n$ 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) $f_1, ..., f_n$ generate g and (2) there are learning algorithms $\mathcal{A}_i$ for the $\mathcal{N}_i$'s such that $n \cdot \max_i C_{\mathcal{A}_i}(f_i, \epsilon, \delta) \le M$.

由于文章里面分析的各个神经网络结构都要到了 MLP 作为基本的结构，因此我们先来看一下 MLP 的 sample complexity；特别地，我们来看一下如果一个函数能够被写成 Tayler expansion 的形式，那么 MLP 需要大概这么多样本去拟合该函数。



Theoretical Framework

- MLP is efficient for single objects; not for multiple objects or for-loop

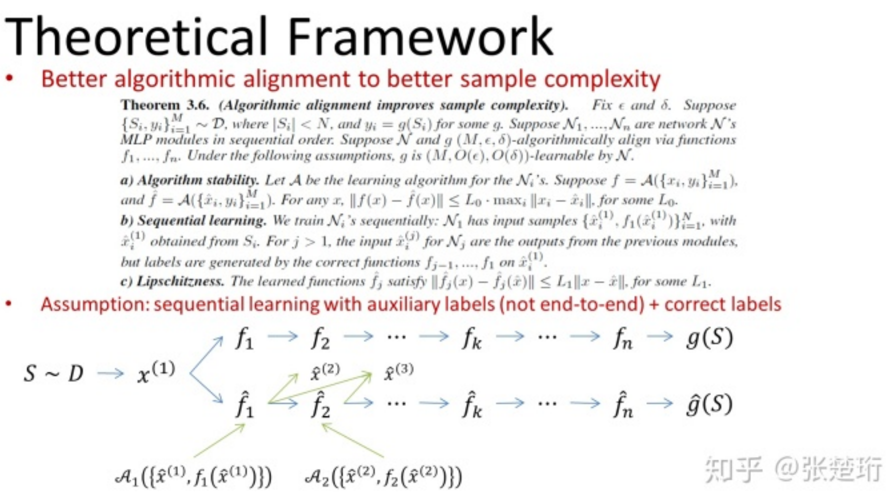
Theorem 3.5. ***(Sample complexity for overparameterized MLP modules).*** *Let $\mathcal{A}$ be an overparameterized and randomly initialized two-layer MLP trained with gradient descent for a sufficient number of iterations. Suppose $g : \mathbb{R}^d \to \mathbb{R}^m$ with components $g(x)^{(i)} = \sum_j \alpha_j^{(i)} (\beta_j^{(i)\top} x)^{p_j^{(i)}}$, where $\beta_j^{(i)} \in \mathbb{R}^d$, $\alpha \in \mathbb{R}$, and $p_j^{(i)} = 1$ or $p_j^{(i)} = 2l$ ($l \in \mathbb{N}_+$). The sample complexity $\mathcal{C}_{\mathcal{A}}(g, \epsilon, \delta)$ is*

$$\mathcal{C}_{\mathcal{A}}(g, \epsilon, \delta) = O\Big(\frac{\max_i \sum_{j=1}^{K} p_j^{(i)} |\alpha_j^{(i)}| \cdot \|\beta_j^{(i)}\|_2^{p_j^{(i)}} + \log(m/\delta)}{(\epsilon/m)^2}\Big). \tag{3.2}$$

- Proof in Appendix C based on similar result on scalar functions (Arora et al 2019) + union bound
- Functions expressed as Taylor expansion: efficient
- Concatenated features for multiple objects: not efficient due to large K or $\left\|\beta_j^{(i)}\right\|$
- For-loop: cannot be expressed as a polynomial

我们看到，对于 MLP 能够较好地拟合单个实体对应的函数；但是实体数量较多的时候，需要的样本数目就会非常多。关于这一点的理解，可以结合后面的例子来理解（Corollary 3.7）。

接下来是文章最重要的一个定理。该定理说明：网络结构和任务之间的 algorithmic alignment 越高（alignment 中的 M 越小），那么网络结构在该任务上的泛化能力将会越好（所需要的 sample complexity M 也越小）。但是个人感觉，这个定理还是比较大的局限：实际的训练中（包括文章后面的实验），都是 end-to-end 的，即不会给定相应算法的中间变量；但是这里需要假设可以拿到“正确”的中间变量来分别训练每个神经网络模块。



Theoretical Framework

- Better algorithmic alignment to better sample complexity

Theorem 3.6. ***(Algorithmic alignment improves sample complexity).*** *Fix ϵ and δ. Suppose $\{S_i, y_i\}_{i=1}^M \sim \mathcal{D}$, where $|S_i| < N$, and $y_i = g(S_i)$ for some g. Suppose $\mathcal{N}_1, ..., \mathcal{N}_n$ are network $\mathcal{N}$'s MLP modules in sequential order. Suppose $\mathcal{N}$ and g (M, ϵ, δ)-algorithmically align via functions $f_1, ..., f_n$. Under the following assumptions, g is $(M, O(\epsilon), O(\delta))$-learnable by $\mathcal{N}$.*

a) Algorithm stability. *Let $\mathcal{A}$ be the learning algorithm for the $\mathcal{N}_i$'s. Suppose $f = \mathcal{A}(\{x_i, y_i\}_{i=1}^M)$, and $\hat{f} = \mathcal{A}(\{\hat{x}_i, y_i\}_{i=1}^M)$. For any x, $\|f(x) - \hat{f}(x)\| \leq L_0 \cdot \max_i \|x_i - \hat{x}_i\|$, for some L_0.*

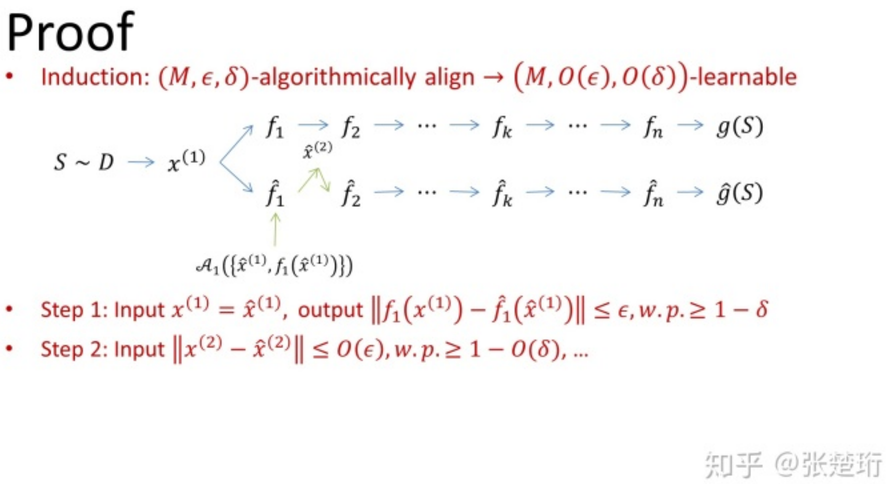
b) Sequential learning. *We train $\mathcal{N}_i$'s sequentially: $\mathcal{N}_1$ has input samples $\{\hat{x}_i^{(1)}, f_1(\hat{x}_i^{(1)})\}_{i=1}^N$, with $\hat{x}_i^{(1)}$ obtained from S_i. For $j > 1$, the input $\hat{x}_i^{(j)}$ for $\mathcal{N}_j$ are the outputs from the previous modules, but labels are generated by the correct functions $f_{j-1}, ..., f_1$ on $\hat{x}_i^{(1)}$.*

c) Lipschitzness. *The learned functions $\hat{f}_j$ satisfy $\|\hat{f}_j(x) - \hat{f}_j(\hat{x})\| \leq L_1\|x - \hat{x}\|$, for some L_1.*

- Assumption: sequential learning with auxiliary labels (not end-to-end) + correct labels

$S \sim D$
$x^{(1)}$
$f_1 \rightarrow f_2 \rightarrow \cdots \rightarrow f_k \rightarrow \cdots \rightarrow f_n \rightarrow g(S)$
$\hat{x}^{(2)}$
$\hat{x}^{(3)}$
$\hat{f}_1 \rightarrow \hat{f}_2 \rightarrow \cdots \rightarrow \hat{f}_k \rightarrow \cdots \rightarrow \hat{f}_n \rightarrow \hat{g}(S)$
$\mathcal{A}_1(\{\hat{x}^{(1)}, f_1(\hat{x}^{(1)})\})$
$\mathcal{A}_2(\{\hat{x}^{(2)}, f_2(\hat{x}^{(2)})\})$

证明的过程用到数学归纳法，剩下的就是把假设和题设条件都一一放进去。

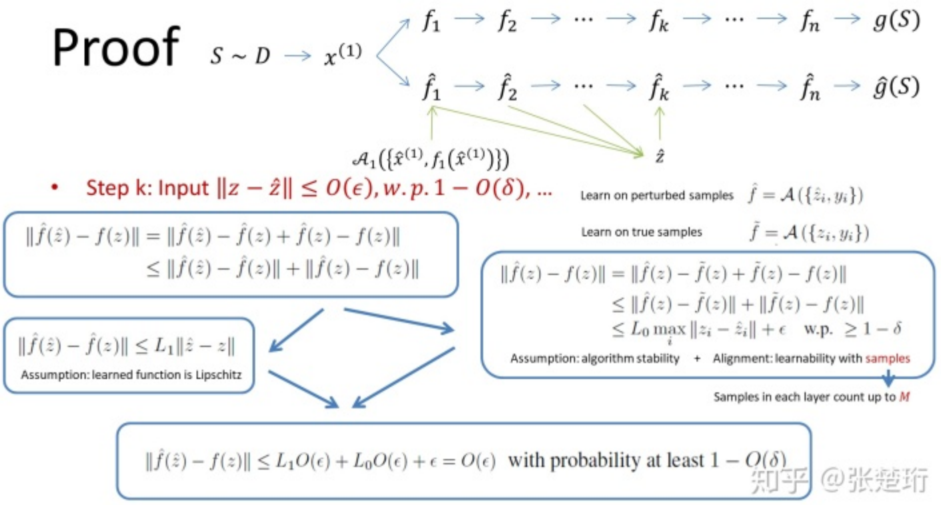


Proof

- Induction: (M, ϵ, δ)-algorithmically align → $(M, O(\epsilon), O(\delta))$-learnable

$S \sim D$
$x^{(1)}$
$f_1 \rightarrow f_2 \rightarrow \cdots \rightarrow f_k \rightarrow \cdots \rightarrow f_n \rightarrow g(S)$
$\hat{x}^{(2)}$
$\hat{f}_1$
$\hat{f}_2 \rightarrow \cdots \rightarrow \hat{f}_k \rightarrow \cdots \rightarrow \hat{f}_n \rightarrow \hat{g}(S)$
$\mathcal{A}_1(\{\hat{x}^{(1)}, f_1(\hat{x}^{(1)})\})$

- Step 1: Input $x^{(1)} = \hat{x}^{(1)}$, output $\|f_1(x^{(1)}) - \hat{f}_1(\hat{x}^{(1)})\| \leq \epsilon, w.p. \geq 1 - \delta$
- Step 2: Input $\|x^{(2)} - \hat{x}^{(2)}\| \leq O(\epsilon), w.p. \geq 1 - O(\delta)$, ...



Proof $S \sim D \to x^{(1)}$

$f_1 \to f_2 \to \cdots \to f_k \to \cdots \to f_n \to g(S)$

$\hat{f}_1 \to \hat{f}_2 \to \cdots \to \hat{f}_k \to \cdots \to \hat{f}_n \to \hat{g}(S)$

$\mathcal{A}_1(\{\hat{x}^{(1)}, f_1(\hat{x}^{(1)})\})$ $\hat{z}$

- Step k: Input $\|z - \hat{z}\| \leq O(\epsilon), w.p. 1 - O(\delta)$, …

Learn on perturbed samples $\hat{f} = \mathcal{A}(\{\hat{z}_i, y_i\})$

Learn on true samples $\tilde{f} = \mathcal{A}(\{z_i, y_i\})$

$$\begin{aligned}\|\hat{f}(\hat{z}) - f(z)\| &= \|\hat{f}(\hat{z}) - \hat{f}(z) + \hat{f}(z) - f(z)\| \\ &\leq \|\hat{f}(\hat{z}) - \hat{f}(z)\| + \|\hat{f}(z) - f(z)\|\end{aligned}$$

$$\|\hat{f}(\hat{z}) - \hat{f}(z)\| \leq L_1\|\hat{z} - z\|$$

Assumption: learned function is Lipschitz

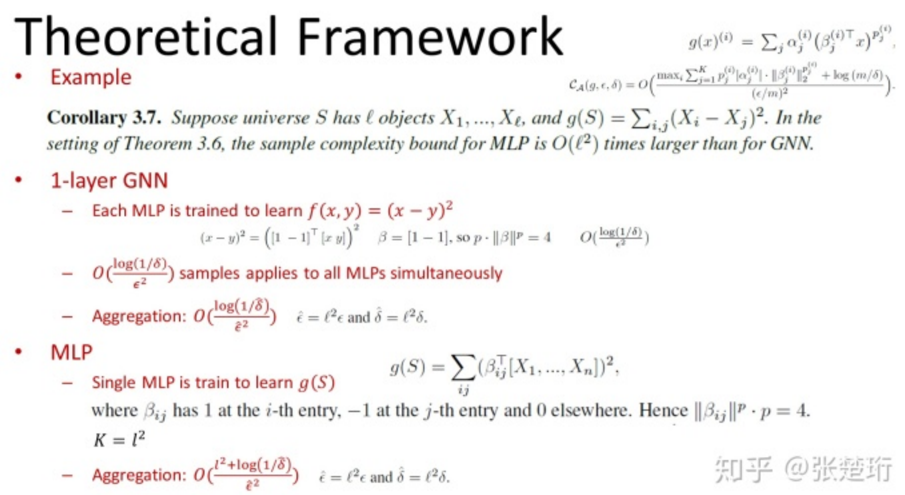
$$\begin{aligned}\|\hat{f}(z) - f(z)\| &= \|\hat{f}(z) - \tilde{f}(z) + \tilde{f}(z) - f(z)\| \\ &\leq \|\hat{f}(z) - \tilde{f}(z)\| + \|\tilde{f}(z) - f(z)\| \\ &\leq L_0 \max_i \|z_i - \hat{z}_i\| + \epsilon \quad \text{w.p.} \ \geq 1 - \delta\end{aligned}$$

Assumption: algorithm stability + Alignment: learnability with samples

Samples in each layer count up to M

$$\|\hat{f}(\hat{z}) - f(z)\| \leq L_1 O(\epsilon) + L_0 O(\epsilon) + \epsilon = O(\epsilon) \ \text{with probability at least } 1 - O(\delta)$$

接下来文章举了一个例子，以此来说明对于某一个特定的任务，GNN 和 MLP 在 sample complexity 上的差距。



Theoretical Framework

$g(x)^{(i)} = \sum_j \alpha_j^{(i)} (\beta_j^{(i)\top} x)^{p_j^{(i)}},$

$\mathcal{C}_{\mathcal{A}}(g, \epsilon, \delta) = O\Big(\frac{\max_i \sum_{j=1}^{K} p_j^{(i)} |\alpha_j^{(i)}| \cdot \|\beta_j^{(i)}\|_2^{p_j^{(i)}} + \log(m/\delta)}{(\epsilon/m)^2}\Big).$

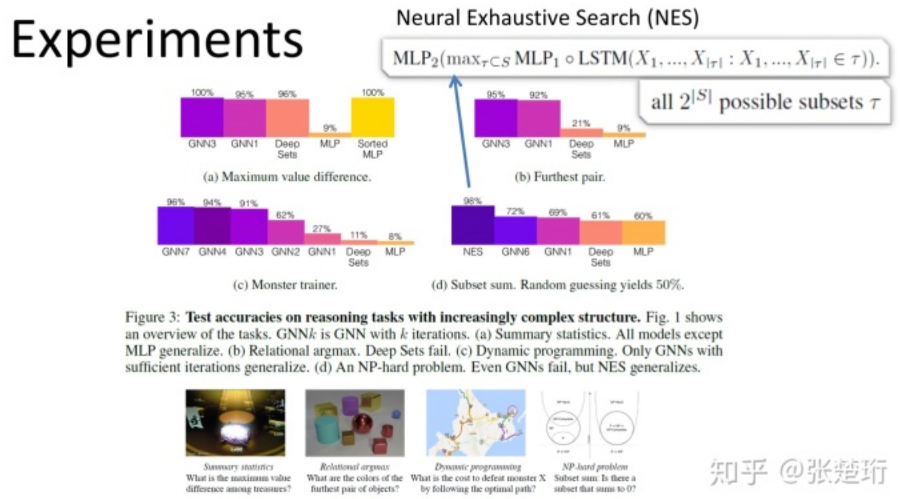
- Example

Corollary 3.7. *Suppose universe S has ℓ objects $X_1, ..., X_\ell$, 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.*

- 1-layer GNN
  - Each MLP is trained to learn $f(x, y) = (x - y)^2$
    $(x-y)^2 = \big([1\ -1]^\top [x\ y]\big)^2$ $\quad \beta = [1 - 1]$, so $p \cdot \|\beta\|^p = 4 \quad O(\frac{\log(1/\delta)}{\epsilon^2})$
  - $O(\frac{\log(1/\delta)}{\epsilon^2})$ samples applies to all MLPs simultaneously
  - Aggregation: $O(\frac{\log(1/\hat{\delta})}{\hat{\epsilon}^2})$ $\quad \hat{\epsilon} = \ell^2 \epsilon$ and $\hat{\delta} = \ell^2 \delta$.
- MLP
  - Single MLP is train to learn $g(S)$ $\quad g(S) = \sum_{ij} (\beta_{ij}^\top [X_1, ..., X_n])^2,$
    where β_{ij} has 1 at the i-th entry, -1 at the j-th entry and 0 elsewhere. Hence $\|\beta_{ij}\|^p \cdot p = 4.$
    $K = l^2$
  - Aggregation: $O(\frac{l^2 + \log(1/\hat{\delta})}{\hat{\epsilon}^2})$ $\quad \hat{\epsilon} = \ell^2 \epsilon$ and $\hat{\delta} = \ell^2 \delta$.

3. 实验

文章接下来把前面提到的任务和各种功能神经网络的结构的组合做了实验，实验结果如下。另外注意到，第四组实验 subset sum 是一个 NP-hard 问题，其他神经网络的效果都比较差。而这类问题的解法目前只有暴力搜索，因此这里就涉及了一个基于暴力搜索的神经网络结构 NES。实验表明，该神经网络结构能够有效解决该问题。



Experiments

Neural Exhaustive Search (NES)

$\text{MLP}_2(\max_{\tau \subset S} \text{MLP}_1 \circ \text{LSTM}(X_1, ..., X_{|\tau|} : X_1, ..., X_{|\tau|} \in \tau)).$

all $2^{|S|}$ possible subsets τ

(a) Maximum value difference.

(b) Furthest pair.

(c) Monster trainer.

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

Figure 3: **Test accuracies on reasoning tasks with increasingly complex structure.** Fig. 1 shows an overview of the tasks. GNNk is GNN with k iterations. (a) Summary statistics. All models except MLP generalize. (b) Relational argmax. Deep Sets fail. (c) Dynamic programming. Only GNNs with sufficient iterations generalize. (d) An NP-hard problem. Even GNNs fail, but NES generalizes.

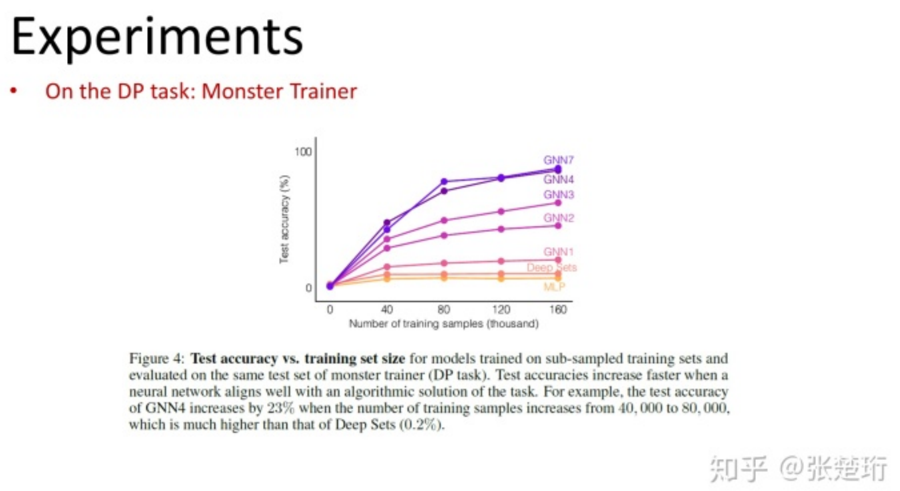
Summary statistics
What is the maximum value difference among treasures?

Relational argmax
What are the colors of the furthest pair of objects?

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

NP-hard problem
Subset sum: Is there a subset that sums to 0?

同时，在 DP 这个任务上，文章还关于 sample complexity 做了实验上的对比。



Experiments

- On the DP task: Monster Trainer

Figure 4: **Test accuracy vs. training set size** for models trained on sub-sampled training sets and evaluated on the same test set of monster trainer (DP task). Test accuracies increase faster when a neural network aligns well with an algorithmic solution of the task. For example, the test accuracy of GNN4 increases by 23% when the number of training samples increases from 40, 000 to 80, 000, which is much higher than that of Deep Sets (0.2%).

发布于 2020-03-02

深度学习（Deep Learning） 机器学习 神经网络

▲ 赞同 98 ▼ 5 条评论 分享 喜欢 收藏 ···

文章被以下专栏收录