Improving algorithmic alignment with autoregressive memory

Nikita Okhotnikov

2024

Introduction

Problem

The role of the metric structure of the input space in model training

Goal

Decresase sample complexity of processor network in NAR pipeline

Proposal

Use less constraint architecture with contrastive objective that allows to utilize prior knowledge of input space structure

Algorithmic learnability

$\mathcal{D}$ – data distribution, $\{x_i, y_i\}_{i=1}^{M}$ – i.i.d samples from $\mathcal{D}$, $\exists g : g(x_i) = y_i$
$\varepsilon > 0$ – error parameter, $\delta \in (0, 1)$ – error probability
$\mathcal{A} : 2^{\mathcal{D}} \to \{f | f : \mathcal{X} \to \mathcal{Y}\}$ – learning algorithm, that generates mapping function by given samples

Definition 1 (Function learnability)

Assume $\varepsilon > 0$, $\delta \in (0, 1)$, $\{x_i, y_i\}_{i=1}^{M}$ are chosen and $y_i = g(x_i)$ for some 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

$$\mathbb{P}_{x \sim \mathcal{D}} \left[\|f(x) - g(x)\| \leq \varepsilon \right] \geq 1 - \delta$$

Definition 2 (Sample complexity)

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

---

https://arxiv.org/pdf/1905.13211

Algorithmic alignment

Definition 3 (Algorithmic alignment)

Assume $\mathcal{N}$ is neural network with n modules $\mathcal{N}_i$.
Let $g : \mathcal{X} \to \mathcal{Y}$ be a reasoning function.
Module functions $f_1, \ldots, f_n$ 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 $f_1, \ldots, f_n$ generate g and there are learning algorithms $\mathcal{A}_i$ for the $\mathcal{N}_i$ such that $n \cdot \max_i C_{\mathcal{A}_i}(f_i, \varepsilon, \delta) \leq M$.

Theorem 1 (Keyulu Xu, et. al. 2020)

$\mathcal{A}$ – an overparameterized and randomly initialized 2-layer MLP trained with GD for a sufficient number of iterations. Suppose $g : \mathbb{R}^d \to \mathbb{R}^m$ with components $g(x)^{(i)} = \sum_j \alpha_j^{(i)} \left(\beta_j^{(i)\top} x\right)^{p_j^{(i)}}$, where $\beta_j^{(i)} \in \mathbb{R}^d$, $\alpha \in \mathbb{R}$ and $p_j^{(i)} = 1$ or $p_j^{(i)} = 2l$, $(l \in \mathbb{N})$. Then the sample complexity $\mathcal{C}(g, \varepsilon, \delta)$ is

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

---

https://arxiv.org/pdf/1905.13211

Algorithmic alignment impoves sample complexity

Theorem 2 (Keyulu Xu, et. al. 2020)

For some ε, δ suppose $\{S_i, y_i\}_{i=1}^{M} \sim \mathcal{D},\ |S_i| < N,\ y_i = g(S_i)$ for some g. Suppose $\mathcal{N}_1 \dots \mathcal{N}_n$ are sequential MLP modules of $\mathcal{N}$. Suppose $\mathcal{N}$ and g (M, ε, δ)-algorithmically align via $f_1 \dots f_n$. Then g is $(M, O(\varepsilon), O(\delta))$-learnable by $\mathcal{N}$.

Corollary 1

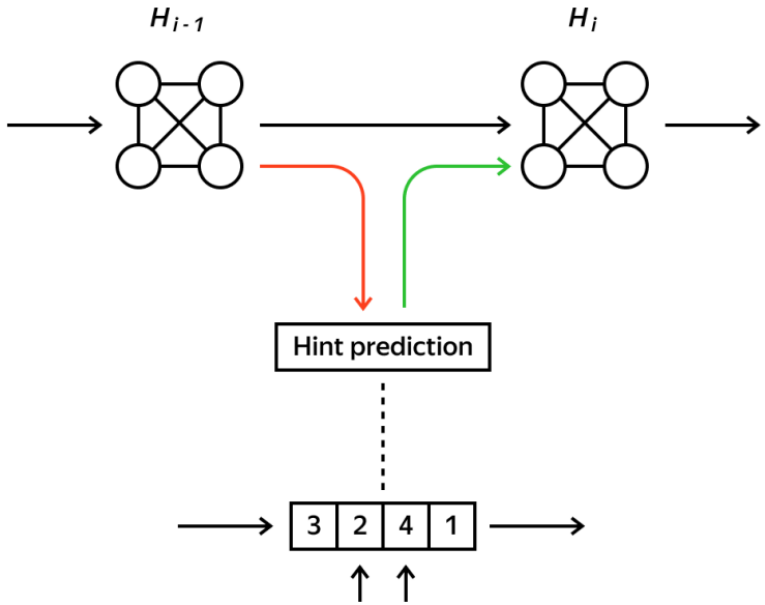
Suppose universe S has n objects $x_1 \dots x_n$ and $g(S) = \sum_{i,j}(x_i - x_j)^2$. Then the sample complexity of MLP is $O(n^2)$ times larger than that of GNN.

---

https://arxiv.org/pdf/1905.13211

Neural algorithmic reasoning, processor network

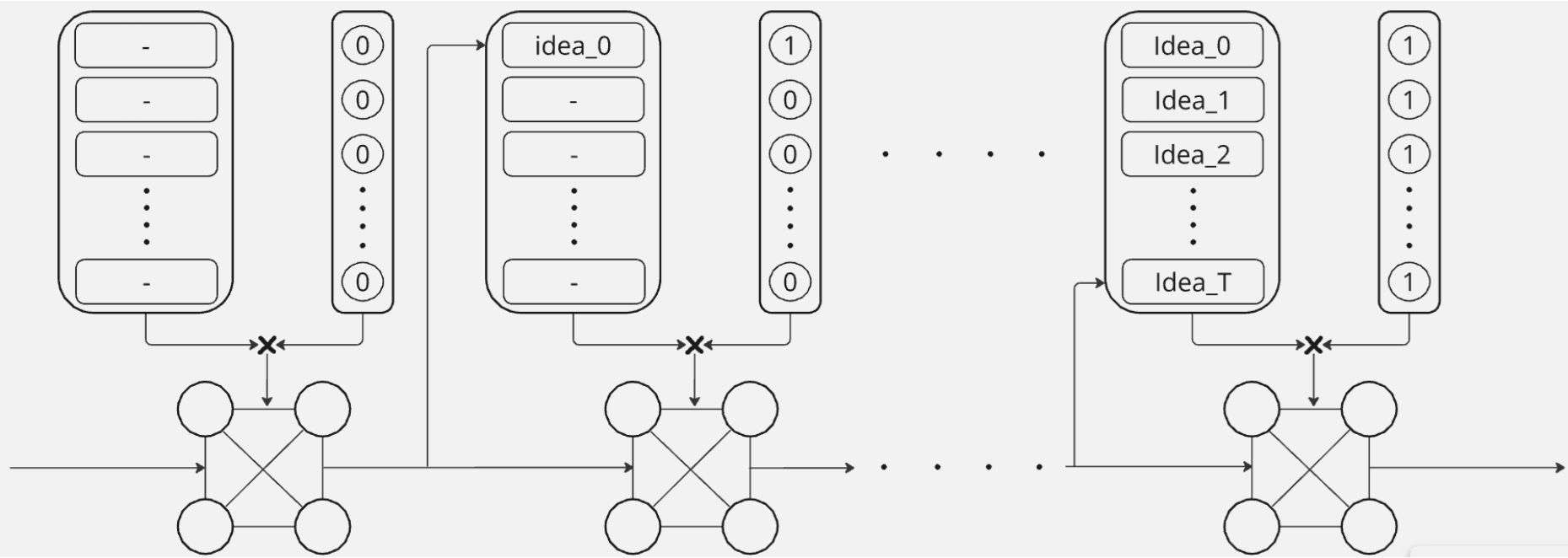
Suppose $g : \mathcal{X} \to \mathcal{Y}$ and $f_1 \dots f_n$ generate g.
$\mathcal{N}$ is a network with modules $\mathcal{N}_i$ that (M, ε, δ)-algorithmically aligns with g.
Then the common way to decrease M is to train the model to explicitly predict $hint_i$ after each module $\mathcal{N}_i$, that are forced to be true $f_i(f_{i-1}(\dots))$



- Trained to follow trajectory of classical algorithm
- Aligns poorly with multiple algorithms at once
- Needs explicit hints on each step to enforce trajectory following

https://arxiv.org/pdf/2306.13411

Proposal: Processor network with autoregressive memory



$$L = -\sum_{X \in \mathcal{X}} \sum_{X_a \in \mathcal{X}_X} \log \frac{\exp\left(\sum_{t=1}^{T} \phi\left(M_t^X, M_t^{X_a}\right)\right)}{\exp\left(\sum_{t=1}^{T} \phi\left(M_t^X, M_t^{X_a}\right)\right) + \sum_{\overline{X_a} \in \overline{\mathcal{X}_X}} \exp\left(\sum_{t=1}^{T} \phi\left(M_t^X, M_t^{\overline{X_a}}\right)\right)}$$

$\mathcal{X}$ – set of inputs, $M_t^X \in \mathbb{R}^k$ – the «idea» generated on step t for input X,
$\mathcal{X}_X$ – set of inputs similar to X, $\overline{\mathcal{X}_X}$ – set of inputs dissimilar to X

Proposal: Processor network with autoregressive memory

- Mimics the step-by-step behaviour of classical algorithms not the exact trajectories
- Much less constraint compared to usual hint-prediction approach
- Does not require explicit hints
- Force processor to extract similar features («ideas») for similar algorithms at each step that might help with multi-algorithm learning

Future plans

- Formalize the proposal
- Mathematically prove the potenitial improvement for a particular task
- Implement and conduct computational experiment on CLRS30 benchmark[1]

[1] https://arxiv.org/abs/2406.04229