

# Implicit Biases in Transformers and SSMs: Distribution Shifts Can Improve Generalization

Nadav Cohen



**Banff International Research Station (BIRS)**  
**Workshop on High-dimensional Learning Dynamics**

3 February 2026

## Outcome-Based RL Provably Leads Transformers to Reason, but Only With the Right Data

---

Ran-Milo\* + Alexander\* + Mendel + C

*Preprint 2026*

## The Implicit Bias of Structured State Space Models Can Be Poisoned with Clean Labels

---

Slutzky\* + Alexander\* + Razin + C

*NeurIPS 2025, Spotlight*

## Learning Low Dimensional State Spaces with Overparameterized Recurrent Neural Nets

---

Cohen-Karlik + Menuhin-Gruman + Giryas + C + Globerson

*ICLR 2023*

## On the Implicit Bias of Gradient Descent for Temporal Extrapolation

---

Cohen-Karlik + Ben David + C + Globerson

*AISTATS 2022*

# Ph.D. Students



Yotam Alexander



Yonatan Slutzky



Yuval Ran-Milo

# Outline

- 1 Implicit Biases in Deep Learning
- 2 Transformers: Simple Examples Are Required for Generalization
- 3 SSMs: Cleanly Labeled Examples Can Disrupt Generalization
- 4 Conclusion

# Deep Learning (DL)

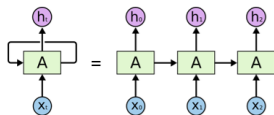
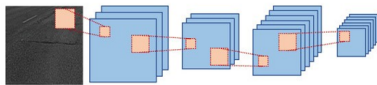
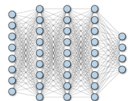
# Deep Learning (DL)

2010s

# Deep Learning (DL)

## 2010s

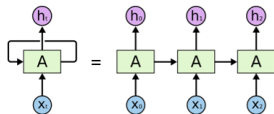
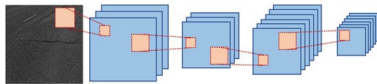
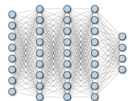
Feedforward/convolutional/recurrent neural networks



# Deep Learning (DL)

## 2010s

Feedforward/convolutional/recurrent neural networks

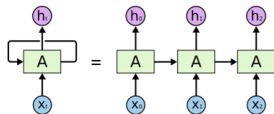
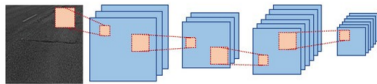
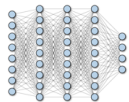


Test distribution  $\approx$  training distribution

# Deep Learning (DL)

## 2010s

Feedforward/convolutional/recurrent neural networks



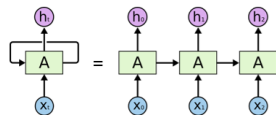
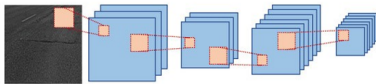
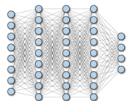
Test distribution  $\approx$  training distribution

## 2020s

# Deep Learning (DL)

## 2010s

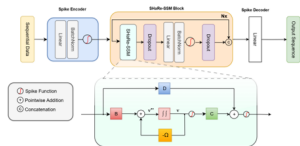
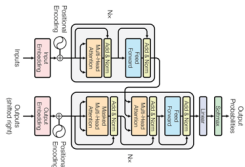
Feedforward/convolutional/recurrent neural networks



Test distribution  $\approx$  training distribution

## 2020s

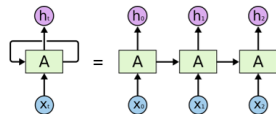
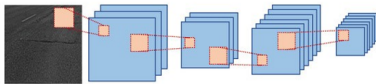
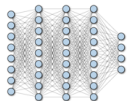
Foundation Models based on **Transformers** and **State Space Models (SSMs)**



# Deep Learning (DL)

## 2010s

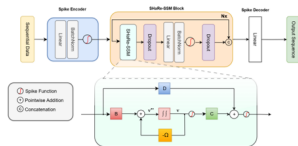
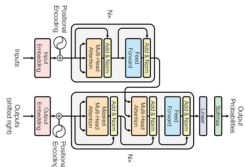
Feedforward/convolutional/recurrent neural networks



Test distribution  $\approx$  training distribution

## 2020s

Foundation Models based on **Transformers** and **State Space Models (SSMs)**

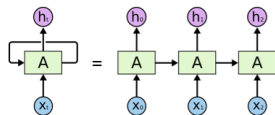
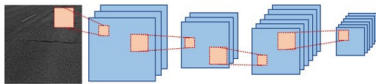
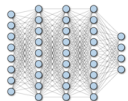


Test distribution  $\neq$  training distribution

# Deep Learning (DL)

## 2010s

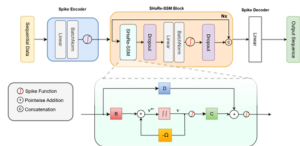
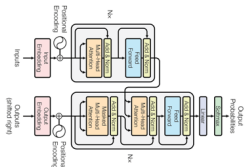
Feedforward/convolutional/recurrent neural networks



Test distribution  $\approx$  training distribution

## 2020s

Foundation Models based on **Transformers** and **State Space Models (SSMs)**



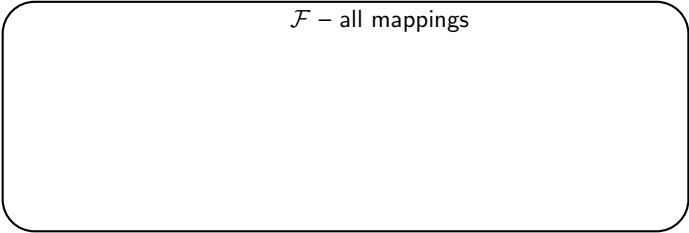
Test distribution  $\neq$  training distribution  $\leftarrow$  **distribution shift**

# Main Theoretical Pillars

Understanding DL calls for addressing three theoretical pillars:

# Main Theoretical Pillars

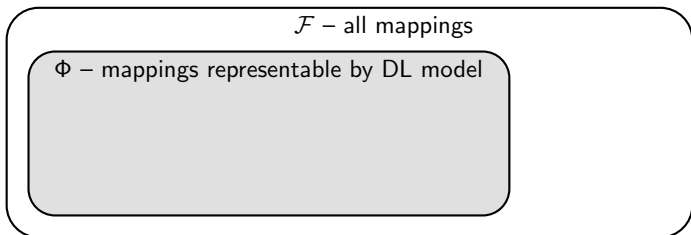
Understanding DL calls for addressing three theoretical pillars:



$\mathcal{F}$  – all mappings

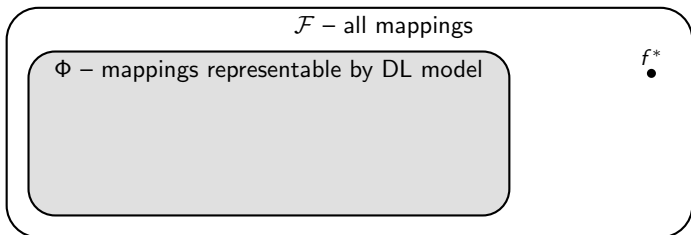
# Main Theoretical Pillars

Understanding DL calls for addressing three theoretical pillars:



# Main Theoretical Pillars

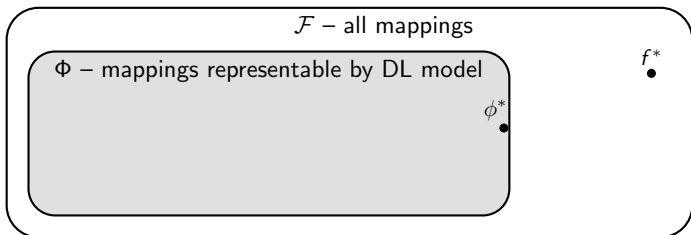
Understanding DL calls for addressing three theoretical pillars:



- $f^*$  – best fit to **test distribution**  $\mathcal{P}$  within  $\mathcal{F}$

# Main Theoretical Pillars

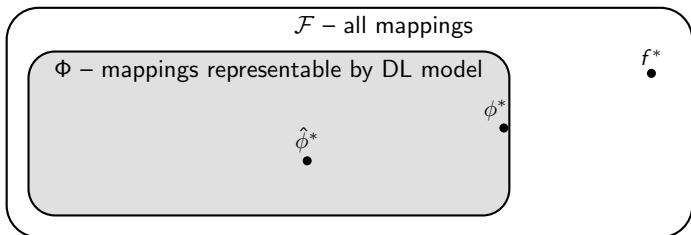
Understanding DL calls for addressing three theoretical pillars:



- $f^*$  – best fit to **test distribution**  $\mathcal{P}$  within  $\mathcal{F}$
- $\phi^*$  – best fit to  $\mathcal{P}$  within  $\Phi$

# Main Theoretical Pillars

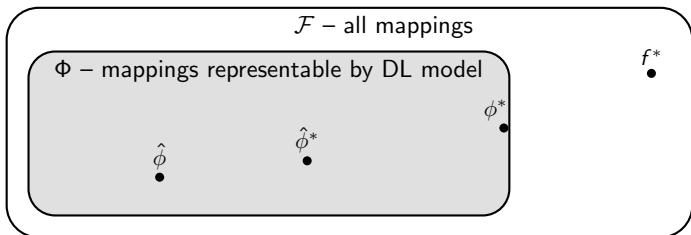
Understanding DL calls for addressing three theoretical pillars:



- $f^*$  – best fit to **test distribution**  $\mathcal{P}$  within  $\mathcal{F}$
- $\phi^*$  – best fit to  $\mathcal{P}$  within  $\Phi$
- $\hat{\phi}^*$  – best fit to **training set**  $\mathcal{T}$  within  $\Phi$

# Main Theoretical Pillars

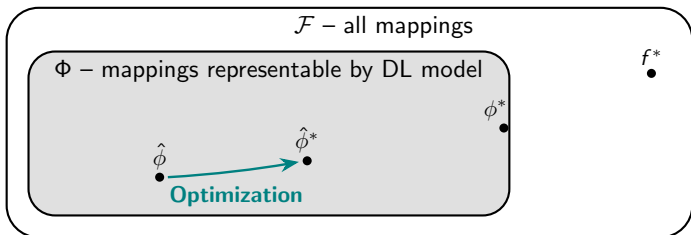
Understanding DL calls for addressing three theoretical pillars:



- $f^*$  – best fit to **test distribution**  $\mathcal{P}$  within  $\mathcal{F}$
- $\phi^*$  – best fit to  $\mathcal{P}$  within  $\Phi$
- $\hat{\phi}^*$  – best fit to **training set**  $\mathcal{T}$  within  $\Phi$
- $\hat{\phi}$  – learned via (variant of) **Gradient Descent (GD)** over  $\mathcal{T}$

# Main Theoretical Pillars

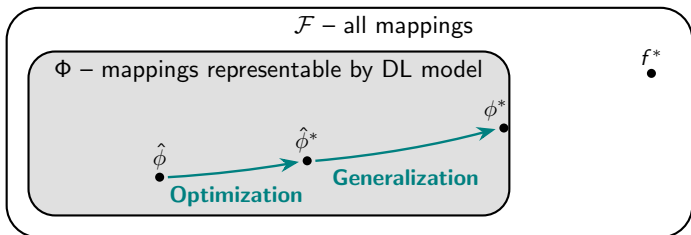
Understanding DL calls for addressing three theoretical pillars:



- $f^*$  – best fit to **test distribution**  $\mathcal{P}$  within  $\mathcal{F}$
- $\phi^*$  – best fit to  $\mathcal{P}$  within  $\Phi$
- $\hat{\phi}^*$  – best fit to **training set**  $\mathcal{T}$  within  $\Phi$
- $\hat{\phi}$  – learned via (variant of) **Gradient Descent (GD)** over  $\mathcal{T}$

# Main Theoretical Pillars

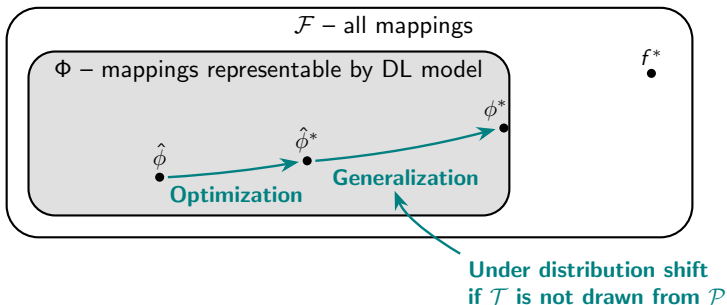
Understanding DL calls for addressing three theoretical pillars:



- $f^*$  – best fit to **test distribution**  $\mathcal{P}$  within  $\mathcal{F}$
- $\phi^*$  – best fit to  $\mathcal{P}$  within  $\Phi$
- $\hat{\phi}^*$  – best fit to **training set**  $\mathcal{T}$  within  $\Phi$
- $\hat{\phi}$  – learned via (variant of) **Gradient Descent (GD)** over  $\mathcal{T}$

# Main Theoretical Pillars

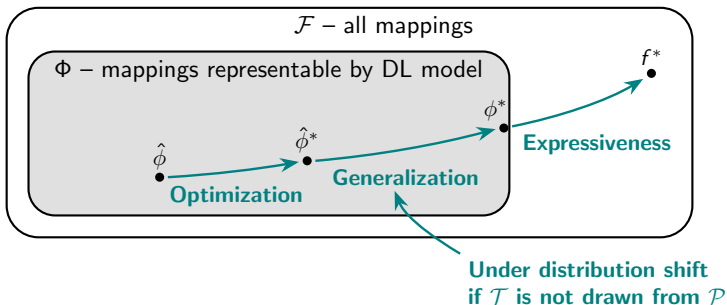
Understanding DL calls for addressing three theoretical pillars:



- $f^*$  – best fit to **test distribution**  $\mathcal{P}$  within  $\mathcal{F}$
- $\phi^*$  – best fit to  $\mathcal{P}$  within  $\Phi$
- $\hat{\phi}^*$  – best fit to **training set**  $\mathcal{T}$  within  $\Phi$
- $\hat{\phi}$  – learned via (variant of) **Gradient Descent (GD)** over  $\mathcal{T}$

# Main Theoretical Pillars

Understanding DL calls for addressing three theoretical pillars:



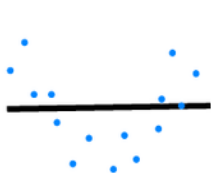
- $f^*$  – best fit to **test distribution**  $\mathcal{P}$  within  $\mathcal{F}$
- $\phi^*$  – best fit to  $\mathcal{P}$  within  $\Phi$
- $\hat{\phi}^*$  – best fit to **training set**  $\mathcal{T}$  within  $\Phi$
- $\hat{\phi}$  – learned via (variant of) **Gradient Descent (GD)** over  $\mathcal{T}$

# Generalization: Classical View

**Generalization** is classically viewed via approximation-estimation tradeoff:

# Generalization: Classical View

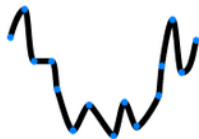
**Generalization** is classically viewed via approximation-estimation tradeoff:



Underfitting



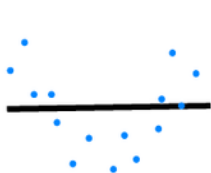
Desired



Overfitting

# Generalization: Classical View

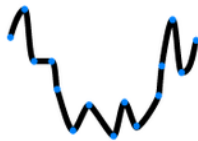
**Generalization** is classically viewed via approximation-estimation tradeoff:



Underfitting



Desired

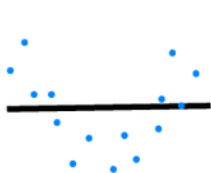


Overfitting

Tradeoff can be controlled through:

# Generalization: Classical View

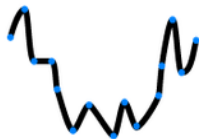
**Generalization** is classically viewed via approximation-estimation tradeoff:



Underfitting



Desired



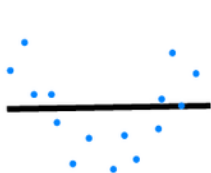
Overfitting

Tradeoff can be controlled through:

- Limiting model size

# Generalization: Classical View

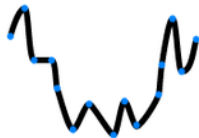
**Generalization** is classically viewed via approximation-estimation tradeoff:



Underfitting



Desired



Overfitting

Tradeoff can be controlled through:

- Limiting model size
- Adding regularization (e.g.  $\ell_2$  penalty)

# Generalization in DL

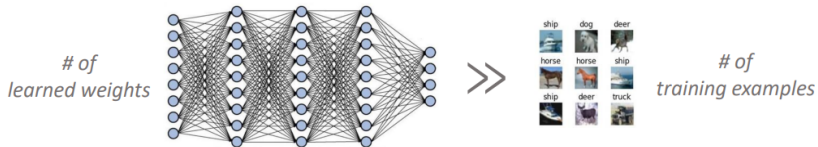
## Phenomenon

A DL model trained by GD often generalizes, even when:

# Generalization in DL

## Phenomenon

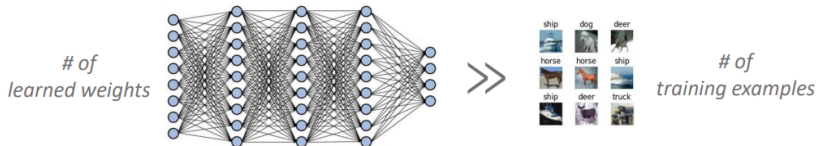
A DL model trained by GD often generalizes, even when:



# Generalization in DL

## Phenomenon

A DL model trained by GD often generalizes, even when:



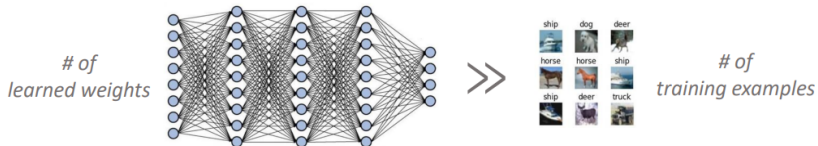
## Contemporary View

There is an **implicit bias** towards generalizing mappings

# Generalization in DL

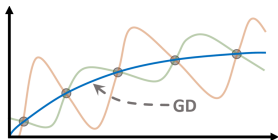
## Phenomenon

A DL model trained by GD often generalizes, even when:



## Contemporary View

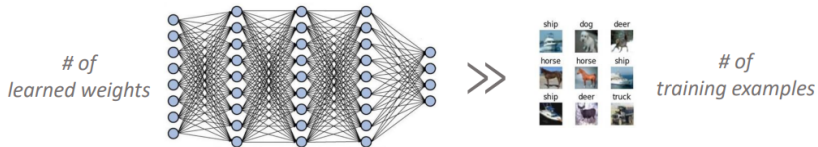
There is an **implicit bias** towards generalizing mappings



# Generalization in DL

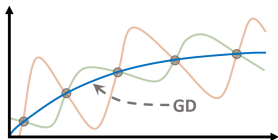
## Phenomenon

A DL model trained by GD often generalizes, even when:



## Contemporary View

There is an **implicit bias** towards generalizing mappings



**Goal:** Theoretically analyze the implicit biases in Transformers and SSMs

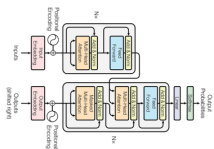
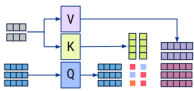
# Outline

- 1 Implicit Biases in Deep Learning
- 2 Transformers: Simple Examples Are Required for Generalization
- 3 SSMs: Cleanly Labeled Examples Can Disrupt Generalization
- 4 Conclusion

# Transformers

# Transformers

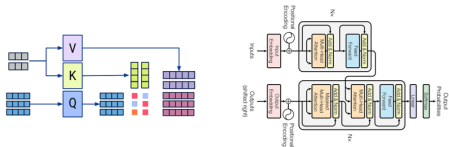
## Learnable attention-based sequence-to-sequence mappings



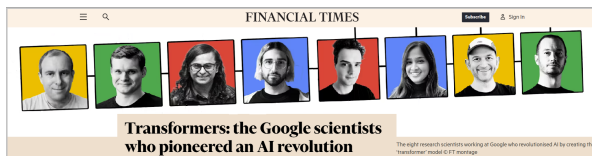


# Transformers

Learnable attention-based sequence-to-sequence mappings



Backbone of modern DL



Pipeline in **Large Language Models (LLMs)**:



# Basic Transformer

# Basic Transformer

## Input

Tokens  $s_1, \dots, s_t \in [d]$  with one-hot representations  $\mathbf{e}_{s_1}, \dots, \mathbf{e}_{s_t} \in \{0, 1\}^d$

$$\mathbf{X} := (\mathbf{e}_{s_1} \mathbf{e}_{s_2} \dots \mathbf{e}_{s_t})^\top \in \mathbb{R}^{t \times d}$$

# Basic Transformer

## Input

Tokens  $s_1, \dots, s_t \in [d]$  with one-hot representations  $\mathbf{e}_{s_1}, \dots, \mathbf{e}_{s_t} \in \{0, 1\}^d$

$$\mathbf{X} := (\mathbf{e}_{s_1} \mathbf{e}_{s_2} \dots \mathbf{e}_{s_t})^\top \in \mathbb{R}^{t \times d}$$

## Parameters

$$\theta = \left( \underbrace{\mathbf{A} \in \mathbb{R}^{d \times d}}_{\text{Attention}}, \underbrace{\mathbf{V} \in \mathbb{R}^{d \times d}}_{\text{Value}} \right)$$

# Basic Transformer

## Input

Tokens  $s_1, \dots, s_t \in [d]$  with one-hot representations  $\mathbf{e}_{s_1}, \dots, \mathbf{e}_{s_t} \in \{0, 1\}^d$

$$\mathbf{X} := (\mathbf{e}_{s_1} \mathbf{e}_{s_2} \dots \mathbf{e}_{s_t})^\top \in \mathbb{R}^{t \times d}$$

## Parameters

$$\theta = \left( \underbrace{\mathbf{A} \in \mathbb{R}^{d \times d}}_{\text{Attention}}, \underbrace{\mathbf{V} \in \mathbb{R}^{d \times d}}_{\text{Value}} \right)$$

## Attention Layer

$$\mathbf{H} = \sigma \left( \text{Mask} \left( \mathbf{XAX}^\top \right) \right) \mathbf{XV},$$

where  $\text{Mask}(\cdot)$  is causal, and  $\sigma = \text{softmax}$  or  $\sigma = \text{id}$  (linear attention)

# Basic Transformer

## Input

Tokens  $s_1, \dots, s_t \in [d]$  with one-hot representations  $\mathbf{e}_{s_1}, \dots, \mathbf{e}_{s_t} \in \{0, 1\}^d$

$$\mathbf{X} := (\mathbf{e}_{s_1} \ \mathbf{e}_{s_2} \ \dots \ \mathbf{e}_{s_t})^\top \in \mathbb{R}^{t \times d}$$

## Parameters

$$\theta = \left( \underbrace{\mathbf{A} \in \mathbb{R}^{d \times d}}_{\text{Attention}}, \quad \underbrace{\mathbf{V} \in \mathbb{R}^{d \times d}}_{\text{Value}} \right)$$

## Attention Layer

$$\mathbf{H} = \sigma \left( \text{Mask} \left( \mathbf{XAX}^\top \right) \right) \mathbf{XV},$$

where  $\text{Mask}(\cdot)$  is causal, and  $\sigma = \text{softmax}$  or  $\sigma = \text{id}$  (linear attention)

## Autoregressive Generation

$$\tau = (s_1, s_2, \dots, s_T) \sim \tau_\theta \quad \text{where} \quad s_{t+1} \sim \text{softmax}(\mathbf{H}_t)$$

# Outcome-Based RL

# Outcome-Based RL

## Reward

Binary signal **based only on final answer** correctness (cf. DeepSeek R1):

$$r(\text{output}) = \begin{cases} 1 & , \text{ if final answer is correct} \\ 0 & , \text{ otherwise} \end{cases}$$

# Outcome-Based RL

## Reward

Binary signal **based only on final answer** correctness (cf. DeepSeek R1):

$$r(\text{output}) = \begin{cases} 1 & , \text{ if final answer is correct} \\ 0 & , \text{ otherwise} \end{cases}$$

## Objective

Maximize expected reward under **training distribution**  $\mathcal{D}$ :

$$\max_{\theta} J(\theta) := \max_{\theta} \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} \left[ \mathbb{E}_{\mathcal{T} \sim \tau_{\theta}(\cdot | \mathbf{x})} [r(\mathcal{T})] \right]$$

# Outcome-Based RL

## Reward

Binary signal **based only on final answer** correctness (cf. DeepSeek R1):

$$r(\text{output}) = \begin{cases} 1 & , \text{ if final answer is correct} \\ 0 & , \text{ otherwise} \end{cases}$$

## Objective

Maximize expected reward under **training distribution**  $\mathcal{D}$ :

$$\max_{\theta} J(\theta) := \max_{\theta} \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [\mathbb{E}_{\tau \sim \tau_{\theta}(\cdot | \mathbf{x})} [r(\tau)]]$$

## Optimization

**Policy gradient:**  $\theta_{t+1} = \theta_t + \eta \nabla_{\theta} J(\theta_t)$  for  $t = 0, 1, 2, \dots$

# Outcome-Based RL

## Reward

Binary signal **based only on final answer** correctness (cf. DeepSeek R1):

$$r(\text{output}) = \begin{cases} 1 & , \text{ if final answer is correct} \\ 0 & , \text{ otherwise} \end{cases}$$

## Objective

Maximize expected reward under **training distribution**  $\mathcal{D}$ :

$$\max_{\theta} J(\theta) := \max_{\theta} \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [\mathbb{E}_{\tau \sim \tau_{\theta}(\cdot | \mathbf{x})} [r(\tau)]]$$

## Optimization

**Policy gradient:**  $\theta_{t+1} = \theta_t + \eta \nabla_{\theta} J(\theta_t)$  for  $t = 0, 1, 2, \dots$

## Theoretical Setting

Gradient flow ( $\eta \rightarrow 0$ ) over exact  $J(\theta)$

# Chain Identification Task

# Chain Identification Task

## Input

Edges forming **two chains** + **start vertex**

# Chain Identification Task

## Input

Edges forming **two chains** + **start vertex**

## Goal

**Output terminal vertex** of chain containing start vertex

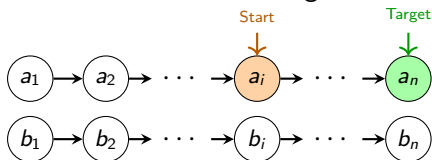
# Chain Identification Task

## Input

Edges forming **two chains** + **start vertex**

## Goal

**Output terminal vertex** of chain containing start vertex



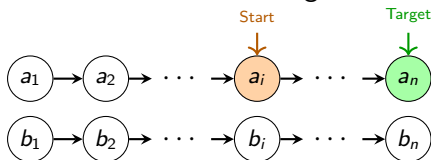
# Chain Identification Task

## Input

Edges forming **two chains** + **start vertex**

## Goal

**Output terminal vertex** of chain containing start vertex



## Example

$(1, 2), (3, 4), (2, 3), (5, 6), (6, 7), (7, 8), 5 \rightarrow 8$

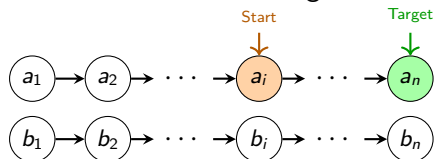
# Chain Identification Task

## Input

Edges forming **two chains** + **start vertex**

## Goal

**Output terminal vertex** of chain containing start vertex



## Example

$(1, 2), (3, 4), (2, 3), (5, 6), (6, 7), (7, 8), 5 \rightarrow 8$

## Task Complexity

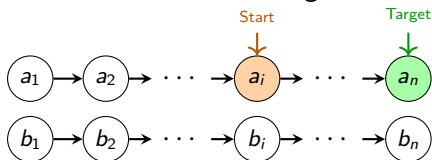
# Chain Identification Task

## Input

Edges forming **two chains** + **start vertex**

## Goal

**Output terminal vertex** of chain containing start vertex



## Example

$(1, 2), (3, 4), (2, 3), (5, 6), (6, 7), (7, 8), 5 \rightarrow 8$

## Task Complexity

Determined by **location of start vertex relative to its terminal vertex**:

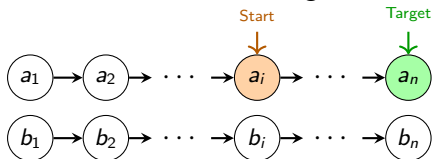
# Chain Identification Task

## Input

Edges forming **two chains** + **start vertex**

## Goal

**Output terminal vertex** of chain containing start vertex



## Example

$(1, 2), (3, 4), (2, 3), (5, 6), (6, 7), (7, 8), 5 \rightarrow 8$

## Task Complexity

Determined by **location of start vertex relative to its terminal vertex**:

- **Near** terminal vertex  $\implies$  **simple** (few reasoning steps required)

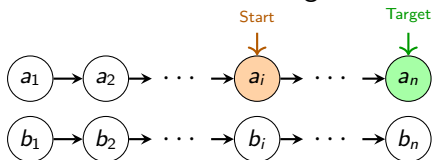
# Chain Identification Task

## Input

Edges forming **two chains** + **start vertex**

## Goal

**Output terminal vertex** of chain containing start vertex



## Example

$(1, 2), (3, 4), (2, 3), (5, 6), (6, 7), (7, 8), 5 \rightarrow 8$

## Task Complexity

Determined by **location of start vertex relative to its terminal vertex**:

- **Near** terminal vertex  $\implies$  **simple** (few reasoning steps required)
- **Far** from terminal vertex  $\implies$  **complex** (many reasoning steps required)

# Reasoning is Necessary and Sufficient for Generalization

# Reasoning is Necessary and Sufficient for Generalization

## Theorem: Reasoning is Necessary

Under standard complexity-theoretic assumptions, there exists a training distribution with which **single-step generation** incurs **suboptimal** reward

# Reasoning is Necessary and Sufficient for Generalization

## Theorem: Reasoning is Necessary

Under standard complexity-theoretic assumptions, there exists a training distribution with which **single-step generation** incurs **suboptimal** reward

## Proposition: Reasoning is Sufficient

For any training distribution, there exist params  $\theta$  s.t. optimal reward is achieved via **multi-step reasoning**

# Efficient vs. Inefficient Reasoning

# Efficient vs. Inefficient Reasoning

Many params achieve optimal reward: some **efficiently**, some **inefficiently**

# Efficient vs. Inefficient Reasoning

Many params achieve optimal reward: some **efficiently**, some **inefficiently**

Proposition: Efficient Solution Exists

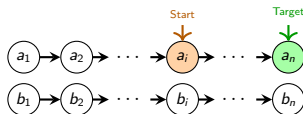
There exist params achieving optimal reward via **chain traversal**

# Efficient vs. Inefficient Reasoning

Many params achieve optimal reward: some **efficiently**, some **inefficiently**

Proposition: Efficient Solution Exists

There exist params achieving optimal reward via **chain traversal**



**Chain traversal:**

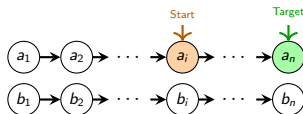
$$a_i \rightarrow a_{i+1} \rightarrow \dots \rightarrow a_n$$

# Efficient vs. Inefficient Reasoning

Many params achieve optimal reward: some **efficiently**, some **inefficiently**

**Proposition: Efficient Solution Exists**

There exist params achieving optimal reward via **chain traversal**



**Chain traversal:**

$$a_i \rightarrow a_{i+1} \rightarrow \dots \rightarrow a_n$$

**Proposition: Inefficient Solutions Exist**

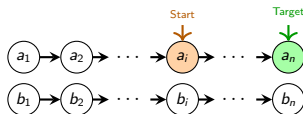
There exist params achieving optimal reward while producing trajectories that are **arbitrarily long**

# Efficient vs. Inefficient Reasoning

Many params achieve optimal reward: some **efficiently**, some **inefficiently**

**Proposition: Efficient Solution Exists**

There exist params achieving optimal reward via **chain traversal**



**Chain traversal:**

$$a_i \rightarrow a_{i+1} \rightarrow \dots \rightarrow a_n$$

**Proposition: Inefficient Solutions Exist**

There exist params achieving optimal reward while producing trajectories that are **arbitrarily long**

Which solution is learned by policy gradient?

# Simple Examples Lead to Efficient Reasoning

# Simple Examples Lead to Efficient Reasoning

Theorem: Simple Examples Suffice for Generalization

# Simple Examples Lead to Efficient Reasoning

## Theorem: Simple Examples Suffice for Generalization

Under technical assumptions, if training distribution includes sufficient mass on **simple examples**, Transformer learns **chain traversal**, thus generalizing under **distribution shifts**

# Simple Examples Lead to Efficient Reasoning

## Theorem: Simple Examples Suffice for Generalization

Under technical assumptions, if training distribution includes sufficient mass on **simple examples**, Transformer learns **chain traversal**, thus generalizing under **distribution shifts**

### Proof Sketch

Assumptions imply rollouts form a Markov chain

It suffices to track probs for three step types:  $p_{\text{fwd}}$ ,  $p_{\text{bwd}}$  and  $p_{\text{other}}$

Careful analysis reveals that with training time  $t$ :

(i)  $p_{\text{bwd}}(t)$ ,  $p_{\text{other}}(t)$  decrease

(ii)  $p_{\text{fwd}}(t)$  increases sufficiently fast

⇒ After polynomial time, each step is forward w.p.  $\geq 1 - \epsilon$

# Without Simple Examples Learning is Intractable

# Without Simple Examples Learning is Intractable

Theorem: Simple Examples are Necessary for Generalization

# Without Simple Examples Learning is Intractable

Theorem: Simple Examples are Necessary for Generalization

Under technical assumptions, if the training distribution contains **only complex examples**, learning is **intractable**

# Without Simple Examples Learning is Intractable

## Theorem: Simple Examples are Necessary for Generalization

Under technical assumptions, if the training distribution contains **only complex examples**, learning is **intractable**

### Proof Sketch

Assumptions imply rollouts form a Markov chain

It suffices to track probs for three step types:  $p_{\text{fwd}}$ ,  $p_{\text{bwd}}$  and  $p_{\text{other}}$

Only complex examples

⇒ Gradients of  $p_{\text{fwd}}$ ,  $p_{\text{bwd}}$ ,  $p_{\text{other}}$  are exponentially small

⇒ Rollout distribution changes exponentially slowly during optimization

# Experiment: Theoretically-Inspired Setting

## Experiment: Theoretically-Inspired Setting

One-layer Transformer trained via REINFORCE on chain identification task

# Experiment: Theoretically-Inspired Setting

One-layer Transformer trained via REINFORCE on chain identification task

Model learns **chain traversal** and **generalizes**:

Complexity (start dist from terminal)		Performance	
<i>Train</i>	<i>Test</i>	<i>Accuracy</i>	<i>Chain Trav.</i>
1 to 3	1 to 3	100%	100%
1 to 7	1 to 7	100%	99%
1 to 11	1 to 11	99%	94%
1 to 3	1 to 11	100%	100%

# Experiment: Theoretically-Inspired Setting

One-layer Transformer trained via REINFORCE on chain identification task

Model learns **chain traversal** and **generalizes**:

Complexity (start dist from terminal)		Performance	
<i>Train</i>	<i>Test</i>	<i>Accuracy</i>	<i>Chain Trav.</i>
1 to 3	1 to 3	100%	100%
1 to 7	1 to 7	100%	99%
1 to 11	1 to 11	99%	94%
1 to 3	1 to 11	100%	100%

**Distribution shift can improve generalization:**

Complexity (start dist from terminal)		Performance	
<i>Train</i>	<i>Test</i>	<i>Accuracy</i>	<i>Chain Trav.</i>
1 to 3	3	100%	100%
3	3	64%	1%

# Experiment: Real-World Setting

# Experiment: Real-World Setting

Qwen2.5 3B trained via GRPO on mathematical reasoning task:

# Experiment: Real-World Setting

Qwen2.5 3B trained via GRPO on mathematical reasoning task:

- **Input:** equations  $x_i = x_j + h$ , one known value  $x_0 = c$ , query variable  $x_{\text{target}}$

# Experiment: Real-World Setting

Qwen2.5 3B trained via GRPO on mathematical reasoning task:

- **Input:** equations  $x_i = x_j + h$ , one known value  $x_0 = c$ , query variable  $x_{\text{target}}$
- **Goal:** evaluate  $x_{\text{target}}$

# Experiment: Real-World Setting

Qwen2.5 3B trained via GRPO on mathematical reasoning task:

- **Input:** equations  $x_i = x_j + h$ , one known value  $x_0 = c$ , query variable  $x_{\text{target}}$
- **Goal:** evaluate  $x_{\text{target}}$

Model learns **efficient reasoning** and **generalizes** to complex examples:

Complexity (number of derivation steps)		Performance
<i>Train</i>	<i>Test</i>	<i>Accuracy</i>
1 to 14	14	96%
1 to 9	14	81%
1 to 4	14	1.2%

# Experiment: Real-World Setting

Qwen2.5 3B trained via GRPO on mathematical reasoning task:

- **Input:** equations  $x_i = x_j + h$ , one known value  $x_0 = c$ , query variable  $x_{\text{target}}$
- **Goal:** evaluate  $x_{\text{target}}$

Model learns **efficient reasoning** and **generalizes** to complex examples:

Complexity (number of derivation steps)		Performance
<i>Train</i>	<i>Test</i>	<i>Accuracy</i>
1 to 14	14	96%
1 to 9	14	81%
1 to 4	14	1.2%

**Distribution shift can improve generalization:**

Complexity (number of derivation steps)		Performance
<i>Train</i>	<i>Test</i>	<i>Accuracy</i>
1 to 14	14	95%
14	14	7%

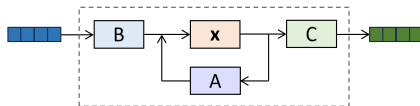
# Outline

- 1 Implicit Biases in Deep Learning
- 2 Transformers: Simple Examples Are Required for Generalization
- 3 SSMs: Cleanly Labeled Examples Can Disrupt Generalization**
- 4 Conclusion

# SSMs

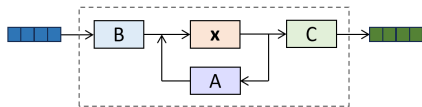
## SSMs

## Learnable sequence-to-sequence mappings



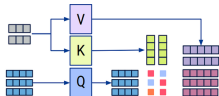
## SSMs

## Learnable sequence-to-sequence mappings

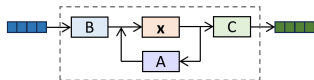


## Viewed as efficient alternative to Transformers

Transformers:  
 $O(H^2)$  complexity

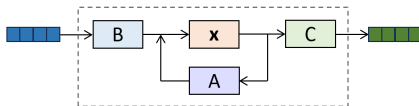


SSMs:  
 $O(H)$  complexity



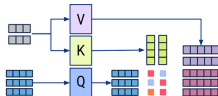
## SSMs

## Learnable sequence-to-sequence mappings

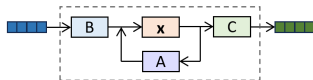


## Viewed as efficient alternative to Transformers

Transformers:  
 $O(H^2)$  complexity



SSMs:  
 $O(H)$  complexity



## Underlie prominent neural networks, e.g., S4, Mamba

## Structured State-Space Sequence Model (S4)

A **Structured State-Space Sequence Model (S4)** is a state space model that leverages **structured state spaces** to efficiently handle long-range dependencies in sequence modeling.

- **Context:**
  - It can typically represent **Continuous Time Series** such as audio and health data, where traditional models like **RNNs** and **Transformers** struggle with long sequences.
  - It can (often) utilize **Linear State Space Systems** techniques to improve both performance and computational efficiency.
  - It can range from being used in **Low-Dimensional Time Series to High-Dimensional Spatio-Temporal Data**.
  - It can typically be implemented in various **Computational Environments**, including those optimized for **Modern Hardware**.

## Mamba (deep learning architecture)

From Wikipedia, the free encyclopedia

Part of a series on **Machine learning and data mining**

**Mamba**<sup>ML</sup> is a deep learning architecture focused on sequence modeling. It was developed by researchers from **Carnegie Mellon University** and **Princeton University** to address some limitations of **transformer models**, especially in processing long



WIKIPEDIA!  
The Free Encyclopedia

# Basic SSM

# Basic SSM

## Parameters

$$\mathbf{A} \in \mathbb{R}^{D \times D} \text{ (Diagonal)}, \quad \mathbf{B} \in \mathbb{R}^{D \times 1}, \quad \mathbf{C} \in \mathbb{R}^{1 \times D}$$

# Basic SSM

## Parameters

$$\mathbf{A} \in \mathbb{R}^{D \times D} \text{ (Diagonal)}, \quad \mathbf{B} \in \mathbb{R}^{D \times 1}, \quad \mathbf{C} \in \mathbb{R}^{1 \times D}$$

## Dynamics

For  $h = 0, 1, 2, \dots$  :

$$\underbrace{\mathbf{x}_{h+1} = \mathbf{A}\mathbf{x}_h + \mathbf{B}u_h}_{\text{state dynamics}}, \quad \underbrace{y_h = \mathbf{C}\mathbf{x}_h}_{\text{output assignment}}$$

# Basic SSM

## Parameters

$$\mathbf{A} \in \mathbb{R}^{D \times D} \text{ (Diagonal)}, \quad \mathbf{B} \in \mathbb{R}^{D \times 1}, \quad \mathbf{C} \in \mathbb{R}^{1 \times D}$$

## Dynamics

$$\text{For } h = 0, 1, 2, \dots : \quad \underbrace{\mathbf{x}_{h+1} = \mathbf{A}\mathbf{x}_h + \mathbf{B}u_h}_{\text{state dynamics}}, \quad \underbrace{y_h = \mathbf{C}\mathbf{x}_h}_{\text{output assignment}}$$

## Input-to-Output Mapping

$$\text{For } h = 0, 1, 2, \dots : \quad y_h = \sum_{h'=0}^{h-1} \mathbf{C}\mathbf{A}^{h'}\mathbf{B}u_{h-1-h'}$$

# Basic SSM

## Parameters

$$\mathbf{A} \in \mathbb{R}^{D \times D} \text{ (Diagonal)}, \quad \mathbf{B} \in \mathbb{R}^{D \times 1}, \quad \mathbf{C} \in \mathbb{R}^{1 \times D}$$

## Dynamics

$$\text{For } h = 0, 1, 2, \dots : \quad \underbrace{\mathbf{x}_{h+1} = \mathbf{A}\mathbf{x}_h + \mathbf{B}u_h}_{\text{state dynamics}}, \quad \underbrace{y_h = \mathbf{C}\mathbf{x}_h}_{\text{output assignment}}$$

## Input-to-Output Mapping

$$\text{For } h = 0, 1, 2, \dots : \quad y_h = \sum_{h'=0}^{h-1} \mathbf{C}\mathbf{A}^{h'}\mathbf{B}u_{h-1-h'}$$

First  $h$  elements of **impulse response**  $(\mathbf{C}\mathbf{A}^h\mathbf{B})_{h=0}^{\infty}$  determine mapping for output at time  $h$

# Teacher-Student Setting

# Teacher-Student Setting

Consider (unknown) **teacher** SSM  $(\mathbf{A}^*, \mathbf{B}^*, \mathbf{C}^*)$  with  $\dim D^*$

# Teacher-Student Setting

Consider (unknown) **teacher** SSM  $(\mathbf{A}^*, \mathbf{B}^*, \mathbf{C}^*)$  with  $\dim D^*$

## Given

Pre-recorded dataset of **horizon**  $H$ :

$$\mathcal{T} = \{(\mathbf{u}^{(n)}, y^{*(n)})\}_{n=1}^N,$$

where  $\mathbf{y}^{*(n)}$  is the teacher output at time  $H$  for  $\mathbf{u}^{(n)} = (u_0^{(n)}, \dots, u_{H-1}^{(n)})$

# Teacher-Student Setting

Consider (unknown) **teacher** SSM  $(\mathbf{A}^*, \mathbf{B}^*, \mathbf{C}^*)$  with  $\dim D^*$

## Given

Pre-recorded dataset of **horizon**  $H$ :

$$\mathcal{T} = \{(\mathbf{u}^{(n)}, y^{*(n)})\}_{n=1}^N,$$

where  $\mathbf{y}^{*(n)}$  is the teacher output at time  $H$  for  $\mathbf{u}^{(n)} = (u_0^{(n)}, \dots, u_{H-1}^{(n)})$

## Goal

Find mapping that fits teacher up to any horizon (guarantees **generalization under distribution shift**)

# Teacher-Student Setting

Consider (unknown) **teacher** SSM  $(\mathbf{A}^*, \mathbf{B}^*, \mathbf{C}^*)$  with  $\dim D^*$

## Given

Pre-recorded dataset of **horizon**  $H$ :

$$\mathcal{T} = \{(\mathbf{u}^{(n)}, y^{*(n)})\}_{n=1}^N,$$

where  $y^{*(n)}$  is the teacher output at time  $H$  for  $\mathbf{u}^{(n)} = (u_0^{(n)}, \dots, u_{H-1}^{(n)})$

## Goal

Find mapping that fits teacher up to any horizon (guarantees **generalization under distribution shift**)

## Method

Train **student** SSM  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  with  $\dim D \gg \max\{D^*, H\}$  via **GD** over:

$$\mathcal{L}_H(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \frac{1}{N} \sum_{n=1}^N (y_H^{(n)} - y^{*(n)})^2,$$

where  $y_H^{(n)}$  is the student output at time  $H$  for  $\mathbf{u}^{(n)}$

# Quantifying Generalization

First  $h$  elements of impulse response determine output mapping at time  $h$

# Quantifying Generalization

First  $h$  elements of impulse response determine output mapping at time  $h$

Impulse response of teacher SSM ( $\mathbf{A}^*$ ,  $\mathbf{B}^*$ ,  $\mathbf{C}^*$ ):

$$(\mathbf{C}^*\mathbf{B}^*, \mathbf{C}^*\mathbf{A}^*\mathbf{B}^*, \mathbf{C}^*(\mathbf{A}^*)^2\mathbf{B}^*, \dots)$$

# Quantifying Generalization

First  $h$  elements of impulse response determine output mapping at time  $h$

Impulse response of teacher SSM ( $\mathbf{A}^*, \mathbf{B}^*, \mathbf{C}^*$ ):

$$(\mathbf{C}^*\mathbf{B}^*, \mathbf{C}^*\mathbf{A}^*\mathbf{B}^*, \mathbf{C}^*(\mathbf{A}^*)^2\mathbf{B}^*, \dots)$$

Impulse response of student SSM ( $\mathbf{A}, \mathbf{B}, \mathbf{C}$ ):

$$(\mathbf{CB}, \mathbf{CAB}, \mathbf{CA}^2\mathbf{B}, \dots)$$

# Quantifying Generalization

First  $h$  elements of impulse response determine output mapping at time  $h$

Impulse response of teacher SSM ( $\mathbf{A}^*, \mathbf{B}^*, \mathbf{C}^*$ ):

$$(\mathbf{C}^*\mathbf{B}^*, \mathbf{C}^*\mathbf{A}^*\mathbf{B}^*, \mathbf{C}^*(\mathbf{A}^*)^2\mathbf{B}^*, \dots)$$

Impulse response of student SSM ( $\mathbf{A}, \mathbf{B}, \mathbf{C}$ ):

$$(\mathbf{CB}, \mathbf{CAB}, \mathbf{CA}^2\mathbf{B}, \dots)$$

## Definition

# Quantifying Generalization

First  $h$  elements of impulse response determine output mapping at time  $h$

Impulse response of teacher SSM  $(\mathbf{A}^*, \mathbf{B}^*, \mathbf{C}^*)$ :

$$(\mathbf{C}^* \mathbf{B}^*, \mathbf{C}^* \mathbf{A}^* \mathbf{B}^*, \mathbf{C}^* (\mathbf{A}^*)^2 \mathbf{B}^*, \dots)$$

Impulse response of student SSM  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$ :

$$(\mathbf{C} \mathbf{B}, \mathbf{C} \mathbf{A} \mathbf{B}, \mathbf{C} \mathbf{A}^2 \mathbf{B}, \dots)$$

## Definition

For  $\epsilon > 0$ , we say student SSM  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$   **$\epsilon$ -extrapolates to horizon  $H'$**  if:

$$\forall h \in \{0, \dots, H' - 1\} : \quad |\mathbf{C} \mathbf{A}^h \mathbf{B} - \mathbf{C}^* (\mathbf{A}^*)^h \mathbf{B}^*| \leq \epsilon$$

# Existence of Non-Generalizing Solutions

# Existence of Non-Generalizing Solutions

## Proposition

$\forall \epsilon > 0 \forall H' > H$  there exist assignments for  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  which:

# Existence of Non-Generalizing Solutions

## Proposition

$\forall \epsilon > 0 \forall H' > H$  there exist assignments for  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  which:

- (i) minimize the loss  $\mathcal{L}_H(\cdot)$

# Existence of Non-Generalizing Solutions

## Proposition

$\forall \epsilon > 0 \forall H' > H$  there exist assignments for  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  which:

- (i) minimize the loss  $\mathcal{L}_H(\cdot)$
- (ii) do **not**  $\epsilon$ -extrapolate to horizon  $H'$

# Existence of Non-Generalizing Solutions

## Proposition

$\forall \epsilon > 0 \forall H' > H$  there exist assignments for  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  which:

- (i) minimize the loss  $\mathcal{L}_H(\cdot)$     (ii) do **not**  $\epsilon$ -extrapolate to horizon  $H'$

## Proof Sketch

(i)  $\iff \mathbf{C}\mathbf{A}^h\mathbf{B} = \mathbf{C}^*(\mathbf{A}^*)^h\mathbf{B}^*$  for all  $h \in \{0, \dots, H-1\}$

(ii)  $\iff |\mathbf{C}\mathbf{A}^h\mathbf{B} - \mathbf{C}^*(\mathbf{A}^*)^h\mathbf{B}^*| > \epsilon$  for some  $h \in \{0, \dots, H'-1\}$

Denote  $\mathbf{v} := (\mathbf{C}^*\mathbf{B}^*, \mathbf{C}^*\mathbf{A}^*\mathbf{B}^*, \dots, \mathbf{C}^*(\mathbf{A}^*)^{H-1}\mathbf{B}^*, \mathbf{C}^*(\mathbf{A}^*)^H\mathbf{B}^* + 2\epsilon)^\top$

It suffices to show that there exist  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  with

$$\mathbf{v} = (\mathbf{C}\mathbf{B}, \mathbf{C}\mathbf{A}\mathbf{B}, \dots, \mathbf{C}\mathbf{A}^H\mathbf{B})^\top = \mathbf{V}(\mathbf{A})(\mathbf{C}^\top \odot \mathbf{B}),$$

where  $\mathbf{V}(\mathbf{A})$  is a Vandermonde matrix holding powers of  $\mathbf{A}$ 's diagonal

This follows from Vandermonde matrices having full rank

# Various Examples Lead to Generalization

# Various Examples Lead to Generalization

## Theorem

Consider learning the SSM  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  via GD initialized  $\approx 0$

# Various Examples Lead to Generalization

## Theorem

Consider learning the SSM  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  via GD initialized  $\approx 0$

Assume the input sequences  $\mathbf{u}^{(1)}, \dots, \mathbf{u}^{(N)}$  span  $\mathbb{R}^H$

# Various Examples Lead to Generalization

## Theorem

Consider learning the SSM  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  via GD initialized  $\approx 0$

Assume the input sequences  $\mathbf{u}^{(1)}, \dots, \mathbf{u}^{(N)}$  span  $\mathbb{R}^H$

Then, if  $\mathcal{L}_H(\cdot) = 0$ ,  $\forall \epsilon > 0 \forall H' > H$  there is  $\epsilon$ -extrapolation to horizon  $H'$

# Various Examples Lead to Generalization

## Theorem

Consider learning the SSM  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  via GD initialized  $\approx 0$

Assume the input sequences  $\mathbf{u}^{(1)}, \dots, \mathbf{u}^{(N)}$  span  $\mathbb{R}^H$

Then, if  $\mathcal{L}_H(\cdot) = 0$ ,  $\forall \epsilon > 0 \forall H' > H$  there is  $\epsilon$ -extrapolation to horizon  $H'$

## Proof Sketch

$\exists$  random var  $\mathcal{Z}^*$  taking  $D^*$  values s.t.  $\forall h: \mathbf{C}^*(\mathbf{A}^*)^h \mathbf{B}^* = \mathbb{E}[(\mathcal{Z}^*)^h]$

GD initialized  $\approx 0 \implies \exists$  random var  $\mathcal{Z}$  taking  $D$  values s.t.  $\forall h: \mathbf{C}\mathbf{A}^h \mathbf{B} = \mathbb{E}[\mathcal{Z}^h]$

$\mathcal{L}_H(\cdot) = 0 \implies \forall h = 0, \dots, H-1: \mathbb{E}[\mathcal{Z}^h] = \mathbf{C}\mathbf{A}^h \mathbf{B} = \mathbf{C}^*(\mathbf{A}^*)^h \mathbf{B}^* = \mathbb{E}[(\mathcal{Z}^*)^h]$

**Moment Matching Theorem** (Cohen & Yeredor 2011)  $\implies \mathcal{Z} = \mathcal{Z}^*$

$\implies \forall h: \mathbf{C}\mathbf{A}^h \mathbf{B} = \mathbf{C}^*(\mathbf{A}^*)^h \mathbf{B}^* \implies \epsilon$ -generalization to horizon  $H'$

# Various Examples Lead to Generalization

## Theorem

Consider learning the SSM  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  via GD initialized  $\approx 0$

Assume the input sequences  $\mathbf{u}^{(1)}, \dots, \mathbf{u}^{(N)}$  span  $\mathbb{R}^H$

Then, if  $\mathcal{L}_H(\cdot) = 0$ ,  $\forall \epsilon > 0 \forall H' > H$  there is  $\epsilon$ -extrapolation to horizon  $H'$

## Proof Sketch

$\exists$  random var  $\mathcal{Z}^*$  taking  $D^*$  values s.t.  $\forall h: \mathbf{C}^*(\mathbf{A}^*)^h \mathbf{B}^* = \mathbb{E}[(\mathcal{Z}^*)^h]$

GD initialized  $\approx 0 \implies \exists$  random var  $\mathcal{Z}$  taking  $D$  values s.t.  $\forall h: \mathbf{C}\mathbf{A}^h \mathbf{B} = \mathbb{E}[\mathcal{Z}^h]$

$\mathcal{L}_H(\cdot) = 0 \implies \forall h = 0, \dots, H-1: \mathbb{E}[\mathcal{Z}^h] = \mathbf{C}\mathbf{A}^h \mathbf{B} = \mathbf{C}^*(\mathbf{A}^*)^h \mathbf{B}^* = \mathbb{E}[(\mathcal{Z}^*)^h]$

**Moment Matching Theorem** (Cohen & Yeredor 2011)  $\implies \mathcal{Z} = \mathcal{Z}^*$

$\implies \forall h: \mathbf{C}\mathbf{A}^h \mathbf{B} = \mathbf{C}^*(\mathbf{A}^*)^h \mathbf{B}^* \implies \epsilon$ -generalization to horizon  $H'$

## Experiments

Demonstrate the phenomenon and show it **extends to non-linear SSMs**

# Certain Examples Can Disrupt Generalization

# Certain Examples Can Disrupt Generalization

## Theorem

With  $D^* = 1$ ,  $\forall \epsilon > 0 \forall H' > H + 1$ , there exist:

# Certain Examples Can Disrupt Generalization

## Theorem

With  $D^* = 1$ ,  $\forall \epsilon > 0 \forall H' > H + 1$ , there exist:

- a training set  $\mathcal{T}$

# Certain Examples Can Disrupt Generalization

## Theorem

With  $D^* = 1$ ,  $\forall \epsilon > 0 \forall H' > H + 1$ , there exist:

- a training set  $\mathcal{T}$
- a certain (“special”) input sequence  $\mathbf{u}^\dagger$  with a **clean label**  $y^\dagger$

# Certain Examples Can Disrupt Generalization

## Theorem

With  $D^* = 1$ ,  $\forall \epsilon > 0 \forall H' > H + 1$ , there exist:

- a training set  $\mathcal{T}$
- a certain (“special”) input sequence  $\mathbf{u}^\dagger$  with a **clean label**  $y^\dagger$
- an open set  $\mathcal{I}$  of  $\text{init} \approx 0$

# Certain Examples Can Disrupt Generalization

## Theorem

With  $D^* = 1$ ,  $\forall \epsilon > 0 \forall H' > H + 1$ , there exist:

- a training set  $\mathcal{T}$
- a certain (“special”) input sequence  $\mathbf{u}^\dagger$  with a **clean label**  $y^\dagger$
- an open set  $\mathcal{I}$  of init  $\approx 0$

s.t. in learning  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  via GD init in  $\mathcal{I}$ ,  $\epsilon$ -generalization to horizon  $H'$ :

# Certain Examples Can Disrupt Generalization

## Theorem

With  $D^* = 1$ ,  $\forall \epsilon > 0 \forall H' > H + 1$ , there exist:

- a training set  $\mathcal{T}$
- a certain (“special”) input sequence  $\mathbf{u}^\dagger$  with a **clean label**  $y^\dagger$
- an open set  $\mathcal{I}$  of  $\text{init} \approx 0$

s.t. in learning  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  via GD  $\text{init}$  in  $\mathcal{I}$ ,  $\epsilon$ -generalization to horizon  $H'$ :

- **Takes place** if  $\mathcal{T}$  is used on its own

# Certain Examples Can Disrupt Generalization

## Theorem

With  $D^* = 1$ ,  $\forall \epsilon > 0 \forall H' > H + 1$ , there exist:

- a training set  $\mathcal{T}$
- a certain (“special”) input sequence  $\mathbf{u}^\dagger$  with a **clean label**  $y^\dagger$
- an open set  $\mathcal{I}$  of  $\text{init} \approx 0$

s.t. in learning  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  via GD  $\text{init}$  in  $\mathcal{I}$ ,  $\epsilon$ -generalization to horizon  $H'$ :

- **Takes place** if  $\mathcal{T}$  is used on its own
- **Does not take place** if  $(\mathbf{u}^\dagger, y^\dagger)$  is appended to  $\mathcal{T}$

# Certain Examples Can Disrupt Generalization

## Theorem

With  $D^* = 1$ ,  $\forall \epsilon > 0 \forall H' > H + 1$ , there exist:

- a training set  $\mathcal{T}$
- a certain (“special”) input sequence  $\mathbf{u}^\dagger$  with a **clean label**  $y^\dagger$
- an open set  $\mathcal{I}$  of init  $\approx 0$

s.t. in learning  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  via GD init in  $\mathcal{I}$ ,  $\epsilon$ -generalization to horizon  $H'$ :

- **Takes place** if  $\mathcal{T}$  is used on its own
- **Does not take place** if  $(\mathbf{u}^\dagger, y^\dagger)$  is appended to  $\mathcal{T}$

## Experiment

Generalization errors with vs. without special sequences

Setting	Without special sequences	With special sequences
SSM per Theorem	$1.34 \times 10^{-3}$	$4.1 \times 10^{-2}$
SSM beyond Theorem	$1.94 \times 10^{-1}$	16.61
SSM in non-linear neural network	$1.61 \times 10^{-3}$	$5.39 \times 10^{-2}$

# Certain Examples Can Disrupt Generalization

## Theorem

With  $D^* = 1$ ,  $\forall \epsilon > 0 \forall H' > H + 1$ , there exist:

- a training set  $\mathcal{T}$
- a certain (“special”) input sequence  $\mathbf{u}^\dagger$  with a **clean label**  $y^\dagger$
- an open set  $\mathcal{I}$  of init  $\approx 0$

s.t. in learning  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  via GD init in  $\mathcal{I}$ ,  $\epsilon$ -generalization to horizon  $H'$ :

- **Takes place** if  $\mathcal{T}$  is used on its own
- **Does not take place** if  $(\mathbf{u}^\dagger, y^\dagger)$  is appended to  $\mathcal{T}$

## Experiment

Generalization errors with vs. without special sequences

Setting	Without special sequences	With special sequences
SSM per Theorem	$1.34 \times 10^{-3}$	$4.1 \times 10^{-2}$
SSM beyond Theorem	$1.94 \times 10^{-1}$	16.61
SSM in non-linear neural network	$1.61 \times 10^{-3}$	$5.39 \times 10^{-2}$

Phenomenon **extends to S4/Mamba2/LRU on real-world data**

# Certain Examples Can Disrupt Generalization

## Theorem

With  $D^* = 1$ ,  $\forall \epsilon > 0 \forall H' > H + 1$ , there exist:

- a training set  $\mathcal{T}$
- a certain (“special”) input sequence  $\mathbf{u}^\dagger$  with a **clean label**  $y^\dagger$
- an open set  $\mathcal{I}$  of  $\text{init} \approx 0$

s.t. in learning  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  via GD init in  $\mathcal{I}$ ,  $\epsilon$ -generalization to horizon  $H'$ :

- **Takes place** if  $\mathcal{T}$  is used on its own
- **Does not take place** if  $(\mathbf{u}^\dagger, y^\dagger)$  is appended to  $\mathcal{T}$

# Certain Examples Can Disrupt Generalization

## Theorem

With  $D^* = 1$ ,  $\forall \epsilon > 0 \forall H' > H + 1$ , there exist:

- a training set  $\mathcal{T}$
- a certain (“special”) input sequence  $\mathbf{u}^\dagger$  with a **clean label**  $y^\dagger$
- an open set  $\mathcal{I}$  of  $\text{init} \approx 0$

s.t. in learning  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  via GD  $\text{init}$  in  $\mathcal{I}$ ,  $\epsilon$ -generalization to horizon  $H'$ :

- **Takes place** if  $\mathcal{T}$  is used on its own
- **Does not take place** if  $(\mathbf{u}^\dagger, y^\dagger)$  is appended to  $\mathcal{T}$

$\implies$  **Clean-label poisoning**

# Certain Examples Can Disrupt Generalization

## Theorem

With  $D^* = 1$ ,  $\forall \epsilon > 0 \forall H' > H + 1$ , there exist:

- a training set  $\mathcal{T}$
- a certain (“special”) input sequence  $\mathbf{u}^\dagger$  with a **clean label**  $y^\dagger$
- an open set  $\mathcal{I}$  of init  $\approx 0$

s.t. in learning  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  via GD init in  $\mathcal{I}$ ,  $\epsilon$ -generalization to horizon  $H'$ :

- **Takes place** if  $\mathcal{T}$  is used on its own
- **Does not take place** if  $(\mathbf{u}^\dagger, y^\dagger)$  is appended to  $\mathcal{T}$

## ⇒ Clean-label poisoning

About 1,300 results (0.09 sec)

Poison frogs! targeted **clean-label poisoning** attacks on neural networks

A.Shafiq, H.Li, H. Wang, M. Nallathambi – Advances in neural information processing systems, 2018 - proceedings.neurips.cc

☆ Save ☰ Cite Cited by 1231 Related articles All 13 versions 39

Transferable **clean-label poisoning** attacks on deep neural nets

C.Zhu, W.S. Hoang, H.Li, G.Jayakumar – International conference on machine learning, 2019 - proceedings.mlr.press

☆ Save ☰ Cite Cited by 342 Related articles All 5 versions 19

Bullseye polytope: A scalable **clean-label poisoning** attack with improved transferability

H.Ashrafian, D.Moag, Y.K. Shih – 2021 IEEE European conference on artificial intelligence, 2021 - ieeeopen.ieee.org

☆ Save ☰ Cite Cited by 113 Related articles All 8 versions 19

CLPA: **Clean-label poisoning** availability attacks using generative adversarial nets

B.Zhao, Y.Luo – Proceedings of the AAAI Conference on Artificial Intelligence, 2022 - ojs.aaai.org

☆ Save ☰ Cite Cited by 24 Related articles All 5 versions 19

# Certain Examples Can Disrupt Generalization

## Theorem

With  $D^* = 1$ ,  $\forall \epsilon > 0 \forall H' > H + 1$ , there exist:

- a training set  $\mathcal{T}$
- a certain (“special”) input sequence  $\mathbf{u}^\dagger$  with a **clean label**  $y^\dagger$
- an open set  $\mathcal{I}$  of init  $\approx 0$

s.t. in learning  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  via GD init in  $\mathcal{I}$ ,  $\epsilon$ -generalization to horizon  $H'$ :

- **Takes place** if  $\mathcal{T}$  is used on its own
- **Does not take place** if  $(\mathbf{u}^\dagger, y^\dagger)$  is appended to  $\mathcal{T}$

⇒ **Clean-label poisoning**

SSMs susceptible to clean-label poisoning!

About 1,300 results (0.09 sec)

Poison frogs! targeted **clean-label poisoning** attacks on neural networks

A.Shafiq, H.Li, H. Wang, M. Nallathambi. – Advances in neural information processing systems, 2018. proceedings.neurips.cc

☆ Save ☰ Cite Cited by 1231 Related articles All 13 versions 39

Transferable **clean-label poisoning** attacks on deep neural nets

C.Zhu, W.S. Han, H.Li, G. Juyuan. – International conference on machine learning, 2019. proceedings.mlr.press

☆ Save ☰ Cite Cited by 342 Related articles All 5 versions 39

Bullseye polytope: A scalable **clean-label poisoning** attack with improved transferability

H.Ashkhanlou, D. Moos, Y.K. Shih. – 2021 IEEE European conference on artificial intelligence, 2021. ieee.ieee.org

☆ Save ☰ Cite Cited by 113 Related articles All 8 versions 39

CLPA: **Clean-label poisoning** availability attacks using generative adversarial nets

B.Zhao, Y.Luo. – Proceedings of the AAAI Conference on Artificial Intelligence, 2022. aaii.org

☆ Save ☰ Cite Cited by 24 Related articles All 5 versions 39

# Certain Examples Can Disrupt Generalization

## Theorem

With  $D^* = 1$ ,  $\forall \epsilon > 0 \forall H' > H + 1$ , there exist:

- a training set  $\mathcal{T}$
- a certain (“special”) input sequence  $\mathbf{u}^\dagger$  with a **clean label**  $y^\dagger$
- an open set  $\mathcal{I}$  of init  $\approx 0$

s.t. in learning  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  via GD init in  $\mathcal{I}$ ,  $\epsilon$ -generalization to horizon  $H'$ :

- **Takes place** if  $\mathcal{T}$  is used on its own
- **Does not take place** if  $(\mathbf{u}^\dagger, y^\dagger)$  is appended to  $\mathcal{T}$

## ⇒ Clean-label poisoning

About 1,300 results (0.09 sec)

Poison frogs! targeted **clean-label poisoning** attacks on neural networks

☆ [Save](#) [Cite](#) [Cited by 1231](#) [Related articles](#) [All 13 versions](#) [PDF](#)

Transferable **clean-label poisoning** attacks on deep neural nets

☆ [Save](#) [Cite](#) [Cited by 342](#) [Related articles](#) [All 5 versions](#) [PDF](#)

Bullseye polytope: A scalable **clean-label poisoning** attack with improved

transferability

☆ [Save](#) [Cite](#) [Cited by 113](#) [Related articles](#) [All 8 versions](#) [PDF](#)

CLPA: **clean-label poisoning** availability attacks using generative adversarial nets

☆ [Save](#) [Cite](#) [Cited by 24](#) [Related articles](#) [All 5 versions](#) [PDF](#)

☆ [Save](#) [Cite](#) [Cited by 24](#) [Related articles](#) [All 5 versions](#) [PDF](#)

☆ [Save](#) [Cite](#) [Cited by 24](#) [Related articles](#) [All 5 versions](#) [PDF](#)

☆ [Save](#) [Cite](#) [Cited by 24](#) [Related articles](#) [All 5 versions](#) [PDF](#)

☆ [Save](#) [Cite](#) [Cited by 24](#) [Related articles](#) [All 5 versions](#) [PDF](#)

☆ [Save](#) [Cite](#) [Cited by 24](#) [Related articles](#) [All 5 versions](#) [PDF](#)

☆ [Save](#) [Cite](#) [Cited by 24](#) [Related articles](#) [All 5 versions](#) [PDF](#)

SSMs susceptible to clean-label poisoning!

SSM generalization isn't explained by:

- implicit complexity minimization
- typicality in loss landscape

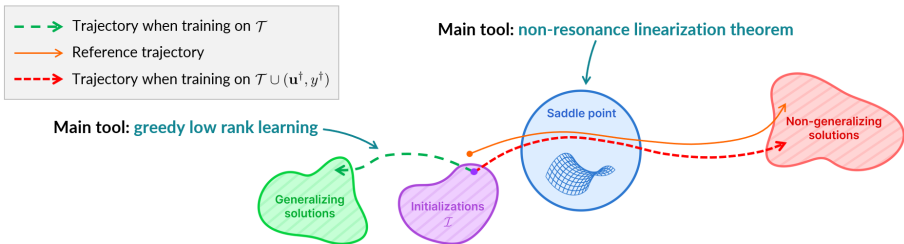
# Certain Examples Can Disrupt Generalization

## Theorem

... in learning  $(\mathbf{A}, \mathbf{B}, \mathbf{C})$  via GD init in  $\mathcal{I}$ ,  $\epsilon$ -generalization to horizon  $H'$ :

- **Takes place** if  $\mathcal{T}$  is used on its own
- **Does not take place** if  $(\mathbf{u}^\dagger, \mathbf{y}^\dagger)$  is appended to  $\mathcal{T}$

## Proof Idea



# Outline

- 1 Implicit Biases in Deep Learning
- 2 Transformers: Simple Examples Are Required for Generalization
- 3 SSMs: Cleanly Labeled Examples Can Disrupt Generalization
- 4 Conclusion

# Recap

# Recap

Modern DL is powered by **implicit biases** in **Transformers** and **SSMs**, which lead them to **generalize under distribution shifts**

# Recap

Modern DL is powered by **implicit biases** in **Transformers** and **SSMs**, which lead them to **generalize under distribution shifts**

We **theoretically analyzed** and **empirically evaluated** these implicit biases

# Recap

Modern DL is powered by **implicit biases** in **Transformers** and **SSMs**, which lead them to **generalize under distribution shifts**

We **theoretically analyzed** and **empirically evaluated** these implicit biases

## Transformers

# Recap

Modern DL is powered by **implicit biases** in **Transformers** and **SSMs**, which lead them to **generalize under distribution shifts**

We **theoretically analyzed** and **empirically evaluated** these implicit biases

## Transformers

In outcome-based RL, on a graph traversal task, provably:

# Recap

Modern DL is powered by **implicit biases** in **Transformers** and **SSMs**, which lead them to **generalize under distribution shifts**

We **theoretically analyzed** and **empirically evaluated** these implicit biases

## Transformers

In outcome-based RL, on a graph traversal task, provably:

- **With simple examples** implicit bias admits **generalization via reasoning**

# Recap

Modern DL is powered by **implicit biases** in **Transformers** and **SSMs**, which lead them to **generalize under distribution shifts**

We **theoretically analyzed** and **empirically evaluated** these implicit biases

## Transformers

In outcome-based RL, on a graph traversal task, provably:

- **With simple examples** implicit bias admits **generalization via reasoning**
- **Without simple examples** learning is **intractable**

# Recap

Modern DL is powered by **implicit biases** in **Transformers** and **SSMs**, which lead them to **generalize under distribution shifts**

We **theoretically analyzed** and **empirically evaluated** these implicit biases

## Transformers

In outcome-based RL, on a graph traversal task, provably:

- **With simple examples** implicit bias admits **generalization via reasoning**
- **Without simple examples** learning is **intractable**

Validated empirically on graph traversal and real-world mathematical task

# Recap

Modern DL is powered by **implicit biases** in **Transformers** and **SSMs**, which lead them to **generalize under distribution shifts**

We **theoretically analyzed** and **empirically evaluated** these implicit biases

## Transformers

In outcome-based RL, on a graph traversal task, provably:

- **With simple examples** implicit bias admits **generalization via reasoning**
- **Without simple examples** learning is **intractable**

Validated empirically on graph traversal and real-world mathematical task

## SSMs

# Recap

Modern DL is powered by **implicit biases** in **Transformers** and **SSMs**, which lead them to **generalize under distribution shifts**

We **theoretically analyzed** and **empirically evaluated** these implicit biases

## Transformers

In outcome-based RL, on a graph traversal task, provably:

- **With simple examples** implicit bias admits **generalization via reasoning**
- **Without simple examples** learning is **intractable**

Validated empirically on graph traversal and real-world mathematical task

## SSMs

In teacher-student setting, provably:

# Recap

Modern DL is powered by **implicit biases** in **Transformers** and **SSMs**, which lead them to **generalize under distribution shifts**

We **theoretically analyzed** and **empirically evaluated** these implicit biases

## Transformers

In outcome-based RL, on a graph traversal task, provably:

- **With simple examples** implicit bias admits **generalization via reasoning**
- **Without simple examples** learning is **intractable**

Validated empirically on graph traversal and real-world mathematical task

## SSMs

In teacher-student setting, provably:

- With **various examples** implicit bias admits **generalization**

# Recap

Modern DL is powered by **implicit biases** in **Transformers** and **SSMs**, which lead them to **generalize under distribution shifts**

We **theoretically analyzed** and **empirically evaluated** these implicit biases

## Transformers

In outcome-based RL, on a graph traversal task, provably:

- **With simple examples** implicit bias admits **generalization via reasoning**
- **Without simple examples** learning is **intractable**

Validated empirically on graph traversal and real-world mathematical task

## SSMs

In teacher-student setting, provably:

- With **various examples** implicit bias admits **generalization**
- **Certain examples** (cleanly labeled) can **disrupt** generalization

# Recap

Modern DL is powered by **implicit biases** in **Transformers** and **SSMs**, which lead them to **generalize under distribution shifts**

We **theoretically analyzed** and **empirically evaluated** these implicit biases

## Transformers

In outcome-based RL, on a graph traversal task, provably:

- **With simple examples** implicit bias admits **generalization via reasoning**
- **Without simple examples** learning is **intractable**

Validated empirically on graph traversal and real-world mathematical task

## SSMs

In teacher-student setting, provably:

- With **various examples** implicit bias admits **generalization**
- **Certain examples** (cleanly labeled) can **disrupt** generalization

Validated empirically on synthetic and real-world tasks

# Outlook: Leveraging Distribution Shifts

# Outlook: Leveraging Distribution Shifts

## Conventional Wisdom

Distribution shifts are an **impediment** to generalization

# Outlook: Leveraging Distribution Shifts

## Conventional Wisdom

Distribution shifts are an **impediment** to generalization

## Our Results

Carefully crafted **distribution shifts can improve generalization**:

# Outlook: Leveraging Distribution Shifts

## Conventional Wisdom

Distribution shifts are an **impediment** to generalization

## Our Results

Carefully crafted **distribution shifts can improve generalization**:

- **Transformers**: add to training set simple examples

# Outlook: Leveraging Distribution Shifts

## Conventional Wisdom

Distribution shifts are an **impediment** to generalization

## Our Results

Carefully crafted **distribution shifts can improve generalization**:

- **Transformers**: add to training set simple examples
- **SSMs**: remove from training set disruptive examples

# Outlook: Leveraging Distribution Shifts

## Conventional Wisdom

Distribution shifts are an **impediment** to generalization

## Our Results

Carefully crafted **distribution shifts can improve generalization**:

- **Transformers**: add to training set simple examples
- **SSMs**: remove from training set disruptive examples

Research on benefits of distribution shifts for Transformers and SSMs may lead to improved curricula for training Foundation Models



# Thank You!

## Work supported by:

ERC Starting Grant (101164614); Apple; Google; Meta; Yandex; ISF Grant (1780/21); Blavatnik Foundation; Adelis Foundation; Tel Aviv University Center for AI and Data Science; and Amnon and Anat Shashua