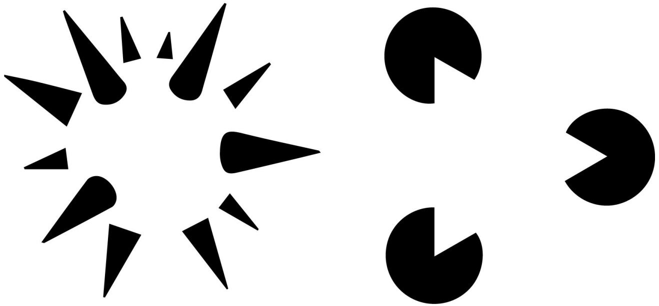
State Reification Networks

Alex Lamb, Jonathan Binas, Anirudh Goyal, Sandeep Subramanian, Denis Kazakov, Ioannis Mitliagkas, Yoshua Bengio, Michael Mozer

Reification in Cognitive Psychology

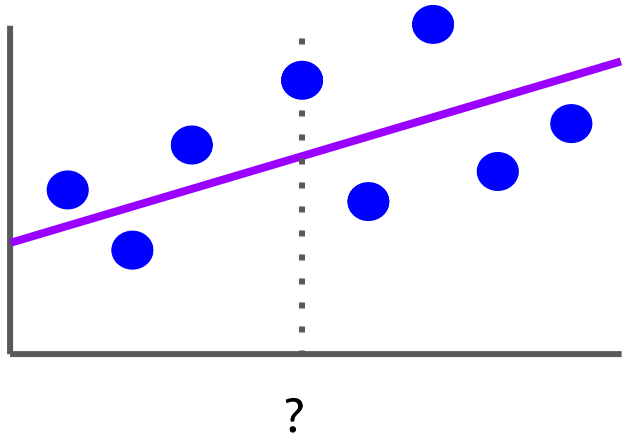
- Human visual perception involves interpreting scenes that can be noisy, missing features, or ambiguous.

- Reification refers to the fact that the output of perception is a coherent whole, not the raw features.



Reification in Machine Learning

- Models are more useful for prediction than are the raw data.
- If that's true for real-world data, might it also be true for data that originate from within the model (i.e., its hidden states)?
- Reification = exchanging inputs with points that are likely under the model.

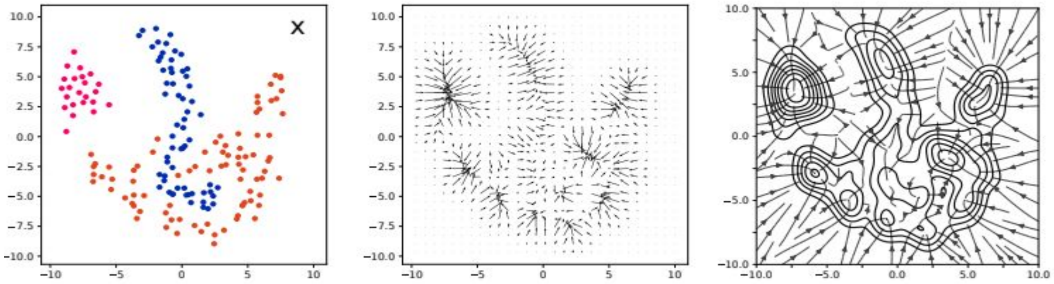


Examples of Reification in Machine Learning

- Batch normalization
  - Performs extremely well, yet only considers 1st and 2nd moments
- Radial Basis Function Networks
  - Projects to “prototypes” around each class ➛ very restrictive
- Generative Classifiers
  - Requires extremely strong generative model, poor practical performance

State Reification

Input Space

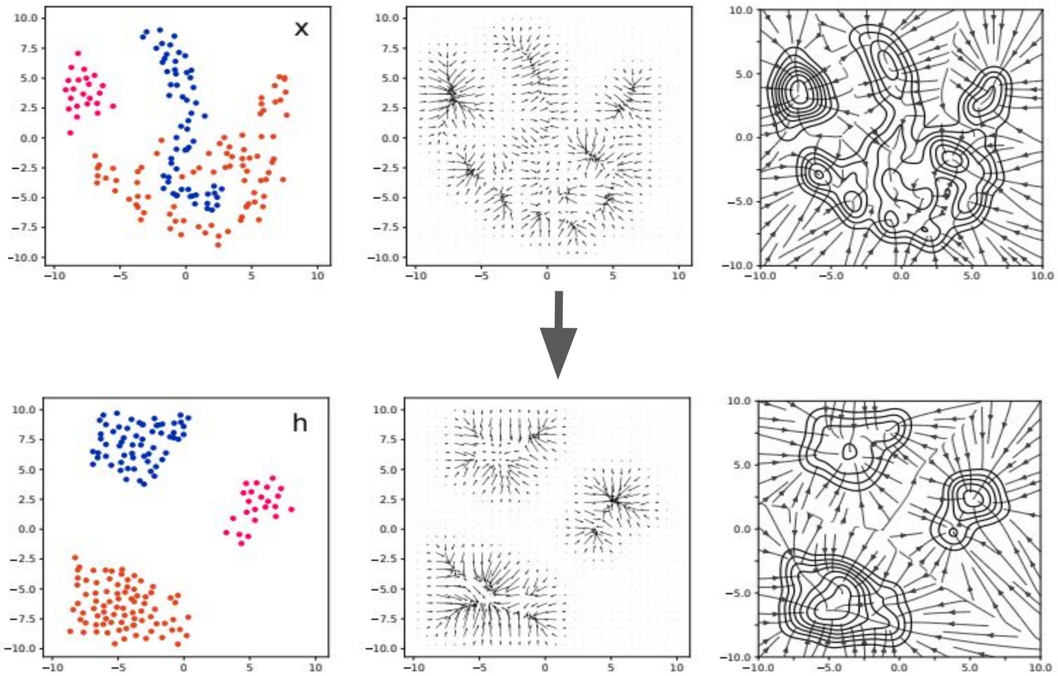


State Reification

- Hidden states can have simpler statistical structure

Input Space

Hidden Space



Explicit Frameworks for State Reification

- Two frameworks for different model types
    - Denoising Autoencoder (CNNs and RNNs)
    - Attractor Networks (RNNs)

$$\mathcal{L} = \mathcal{L}_{\text{task}}(x, y) + \lambda_{\text{rec}} \mathcal{L}_{\text{rec}}(h)$$

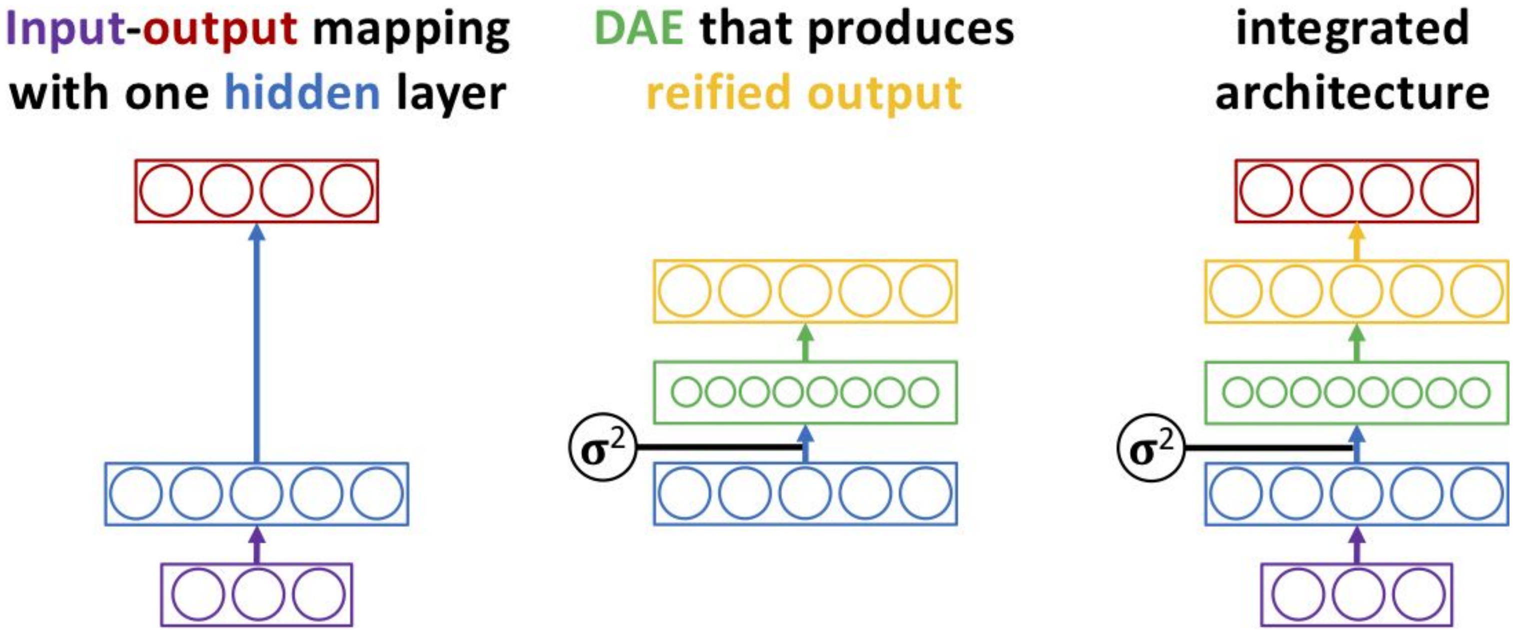
Task Overview

Architecture	**State reification**	**Task**
CNN	Denoising autoencoder	Generalization and adversarial robustness
RNN	Attractor net	Parity Majority Function Reber Grammar Sequence Symmetry
RNN	Denoising autoencoder	Accumulating errors with free running sequence generation

Task Overview

Architecture	**State reification**	**Task**
CNN	Denoising autoencoder	Generalization and adversarial robustness
RNN	Attractor net	Parity Majority Function Reber Grammar Sequence Symmetry
RNN	Denoising autoencoder	Accumulating errors with free running sequence generation

Denoising Autoencoder



Denoising Autoencoder

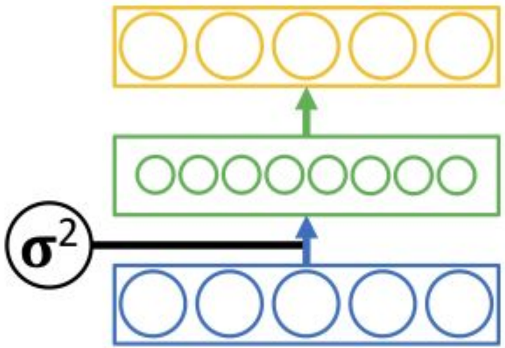
$$\mathcal{L}_{\text{rec}}(x) = \frac{1}{N}\sum_{n=1}^{N}\left(\left\|r_\theta\left(x^{(n)} + a^{(n)}\right) - x^{(n)}\right\|_2^2\right)$$

$a^{(n)} \sim \mathbb{N}(\mathbf{0}, \sigma^2\mathbf{I})$

r_θ Learned denoising function.

$$\frac{r_\sigma(x) - x}{\sigma^2} \to \frac{\partial \log p(x)}{\partial x} \quad \text{as} \quad \sigma \to 0.$$

(Alain and Bengio, 2012)



Adversarial Robustness Setup

- Projected Gradient Descent Attack (PGD):

$$x^{t+1} = \Pi_{x+S}\left(x^t + \alpha\, \mathrm{sgn}(\nabla_x \mathcal{L}_{\text{task}}(x, y))\right)$$

- Train with adversarial examples and DAE reconstruction loss:

$$\mathcal{L} = \mathcal{L}_{\text{task}}(x, y) + \mathcal{L}_{\text{task}}(\widetilde{x}, y) + \lambda_{\text{rec}} \sum_{i \in S} \mathcal{L}^i_{\text{rec}}(h_i)$$

Adversarial Robustness → Improving Generalization

- Improves generalization in adversarial robustness from training set to test set.

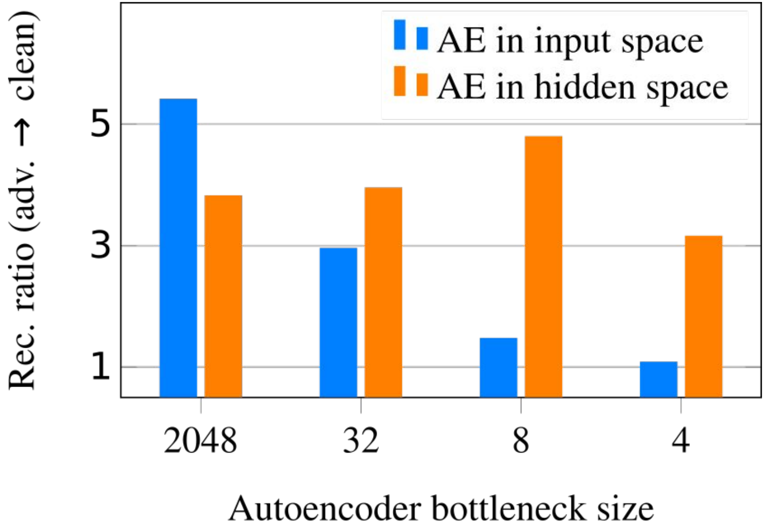
Model	PGD Accuracy (20 steps) baseline	SR
PreActResNet18	37.87	39.20
WideResNet28-10	43.28	44.06

Attack Type	PGD Steps	Attack Epsilon	PGD Accuracy CNN	CNN+	CNN+SR
Normal	7	0.03	33.0	34.2	45.0
Normal	50	0.03	31.6	32.5	42.1
Normal	200	0.03	31.4	32.2	41.5
Normal	100	0.03		35.3	39.2
Normal	100	0.04		24.8	28.0
Normal	100	0.06		14.3	15.6
Normal	100	0.08		12.0	13.0
Normal	100	0.10		11.7	12.9
Normal	100	0.20		10.2	11.3
Normal	100	0.30		8.4	9.6
Normal	100	0.03		33.4	40.1
Noiseless Attack	100	0.03			38.2
BPDA, Skip-DAE	100	0.03			67.1

Adversarial Robustness - some analysis

- Reconstruction error is larger on adversarial examples.

- When the autoencoder is in the hidden states, this detection doesn't require a high-capacity autoencoder.

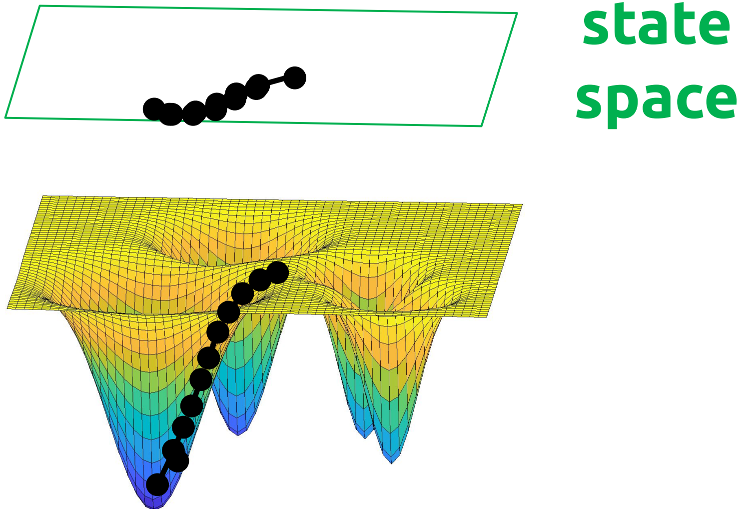


Experiments

Architecture	**State reification**	**Task**
CNN	Denoising autoencoder	Generalization and adversarial robustness
RNN	Attractor net	Parity Majority Function Reber Grammar Sequence Symmetry
RNN	Denoising autoencoder	Accumulating errors with free running sequence generation

Attractor Net

- Network whose dynamics can be characterized as moving downhill in energy, arriving at stable point.



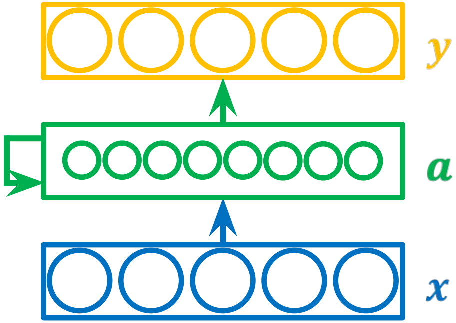
Attractor Net Dynamics

Output $$y = \tanh(v_O + U_O a_\infty)$$

Update $$a_t = W\tanh(a_{t-1}) + x^+$$

Initialization $$a_0 = 0$$

$$x^+ = v_I + U_I \tanh^{-1}(x)$$



To achieve attractor dynamics (Koiran, 1994):

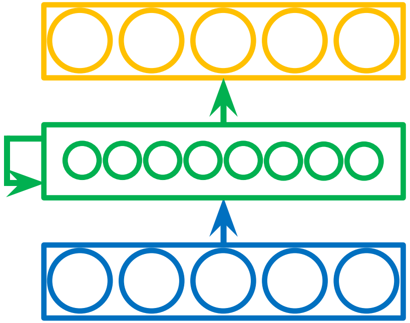
$$w_{ij} = w_{ji}$$

$$w_{ii} \geq 0$$

Attractor Net Training:
Denoising by Convergent Dynamics

Set of target states

$\{\xi_1, \ldots, \xi_n\}$



y $\qquad E = |y - \xi_i|^2$

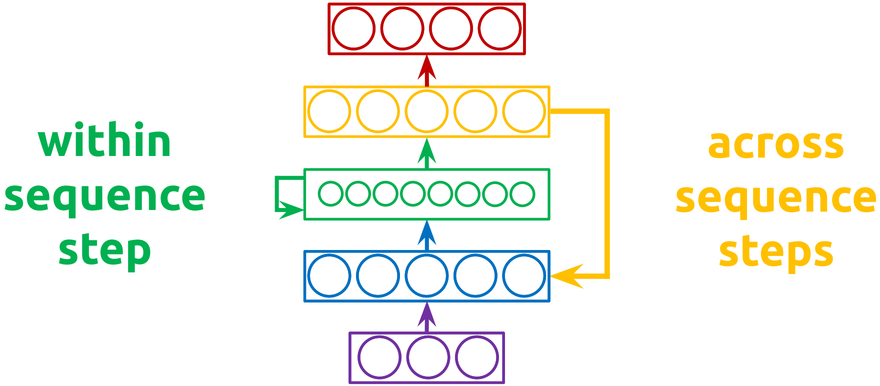
a

$x = \xi_i + \eta \qquad \eta \sim \mathcal{N}(0, \sigma^2)$

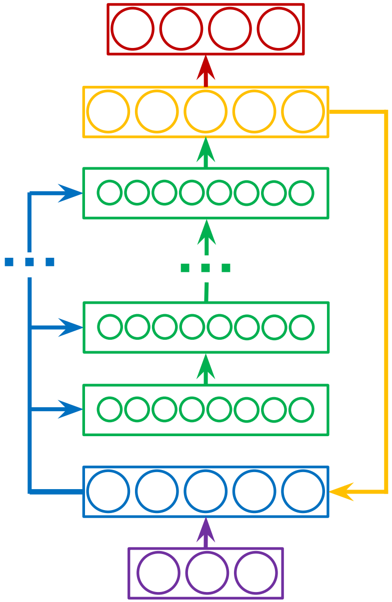
Attractor Nets in RNNs

- ✓ In an imperfectly trained RNN, feedback at each step can inject noise
  - ○ Noise can amplify over time
- ✓ Suppose we could ‘clean up’ the representation at each step to reduce that noise?
  - ○ May lead to better learning and generalization

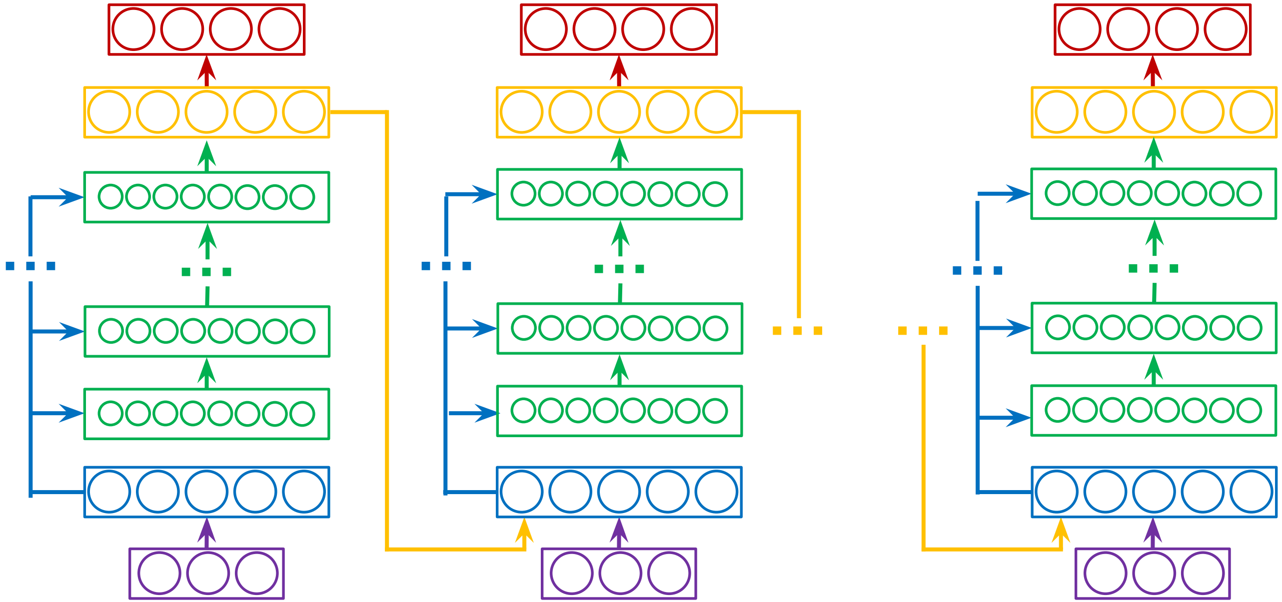
State-Reified RNN



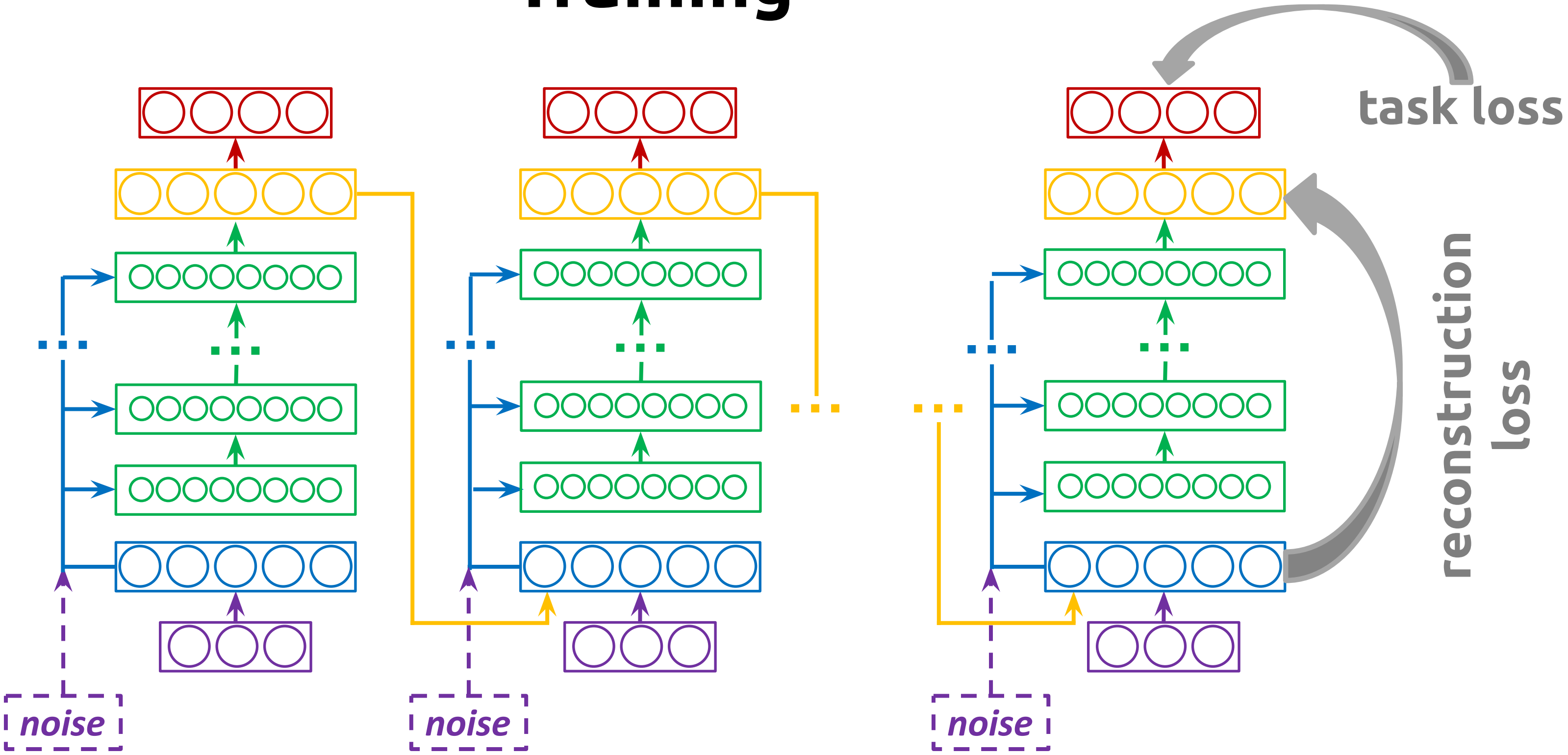
State-Reified RNN



State-Reified RNN



Training



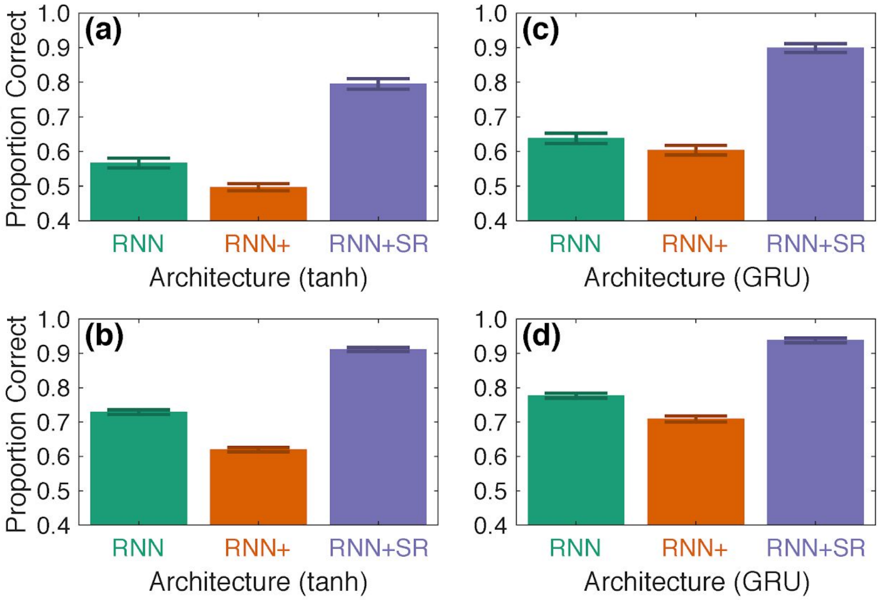
Parity Task

- 10 element sequences
- Training on 256 sequences

1001000101→0
0010101011→1

novel sequences

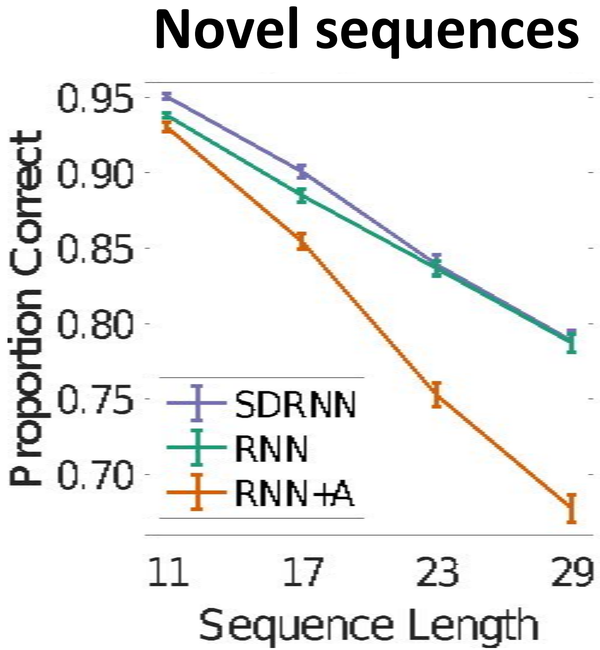
noisy sequences

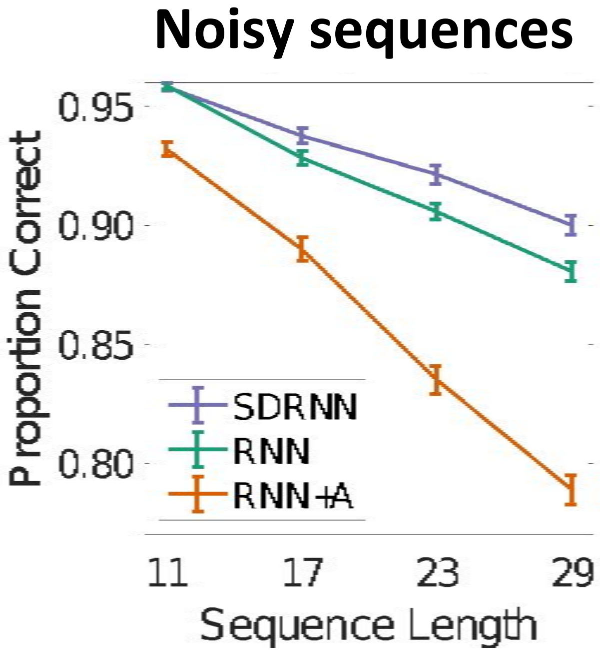


Majority Function

- 100 sequences, length 11-29

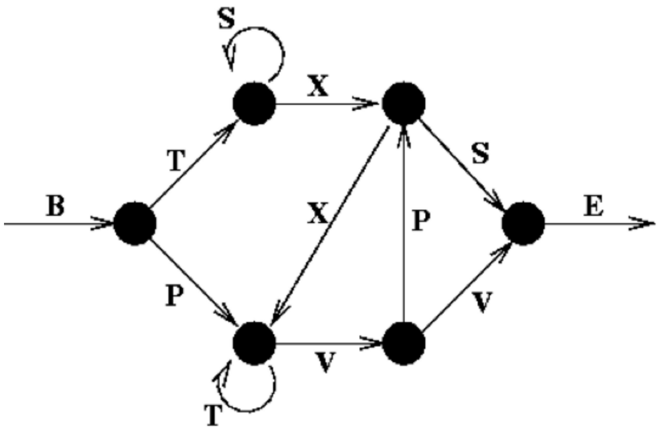
01001000101→0
11010111011→1



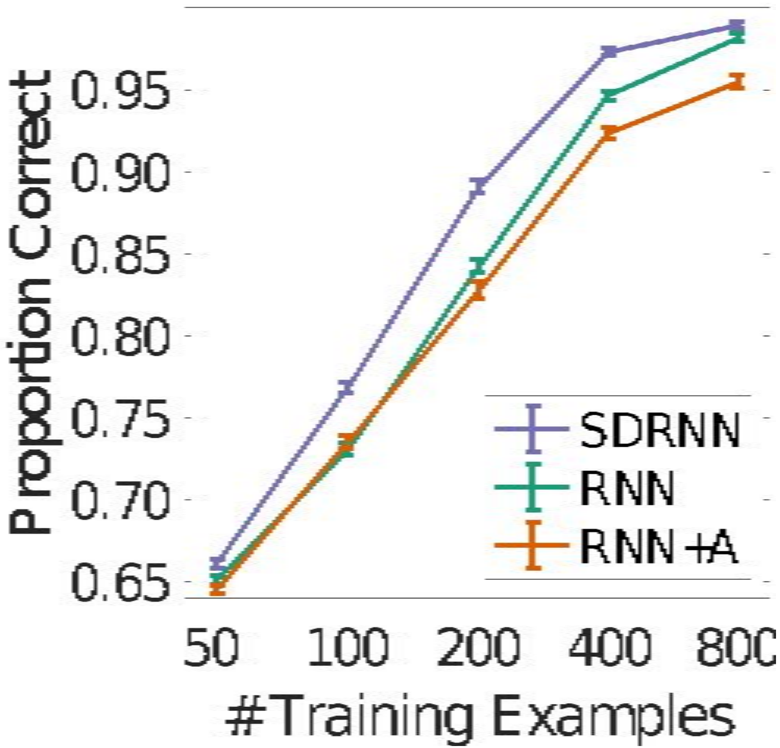


Reber Grammar

- Grammatical or not?
- Vary training set size



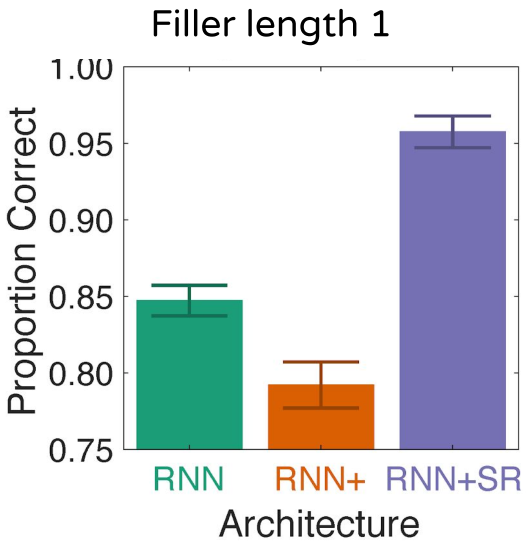
BTTXPVE →0
BPTTVPSE→1

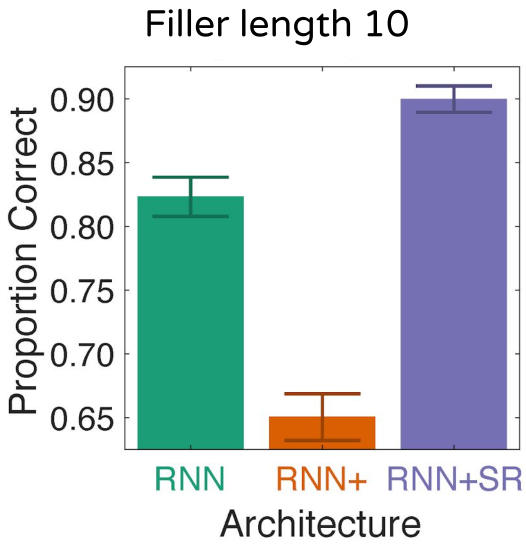


Symmetry

- Is sequence symmetric?
- 5 symbols, filler, 5 symbols

ACAFBXBFACA → 1
ACAFBXBFABA → 0





Experiments

Architecture	**State reification**	**Task**
CNN	Denoising autoencoder	Generalization and adversarial robustness
RNN	Attractor net	Parity Majority Function Reber Grammar Sequence Symmetry
RNN	Denoising autoencoder	Accumulating errors with free running sequence generation

Identifying Failures in Teacher Forcing

- Train LSTM on character-level Text8 dataset for language modeling.
- Train a denoising autoencoder on the hidden states while doing teacher forcing

Sampling Steps	Reconstruction Error Ratio
0	1.00
50	1.03
180	1.12
300	1.34

Open Problems

- How well does state reification scale to harder tasks and larger datasets?

- Denoising autoencoders with quadratic loss may not be ideal for reification.
    - Maybe GANs or better generative models could help?

- Thinking about how the states are changed to make reification easier (are these changes ideal or not)?
    - For example, reification might be made easier by having more compressed representations.

Questions?

- Can also email questions to any of the authors!