Learning Useful Representations of Recurrent Neural Network Weight Matrices

Vincent Herrmann, Francesco Faccio, Jürgen Schmidhuber

Istituto Dalle Molle di studi sull'intelligenza artificiale
IDSIA

Università della Svizzera italiana

Scuola universitaria professionale della Svizzera italiana
SUPSI

KAUST

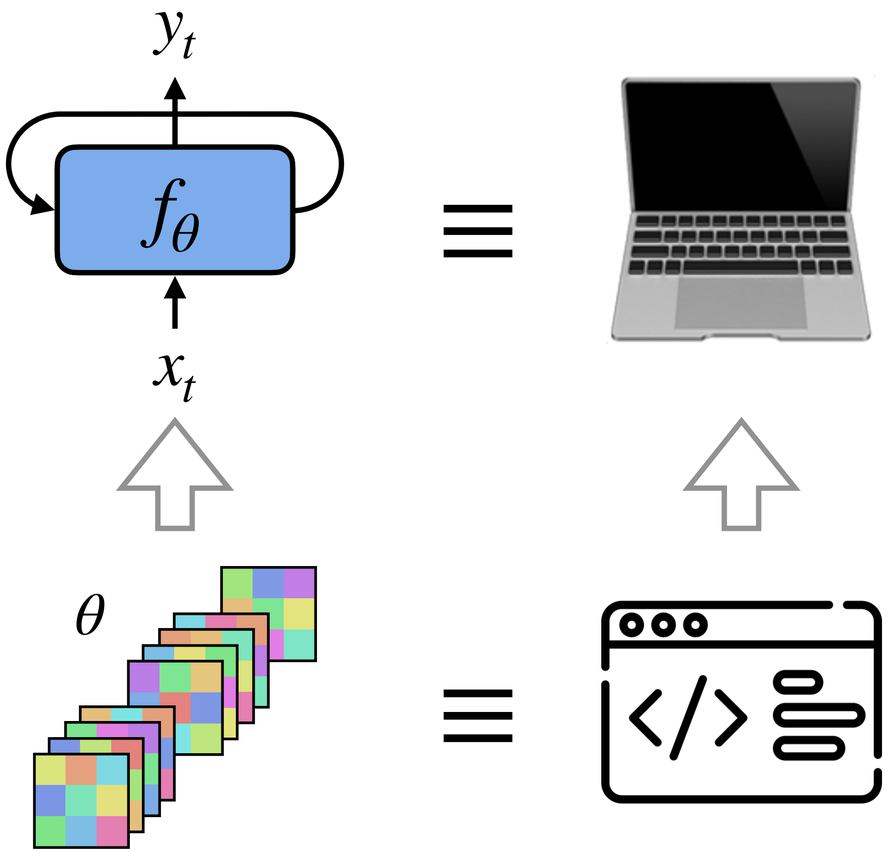
ICML 2024

Learning RNN Weight Representations

- RNNs are universal computers
- Weight matrices are their program

Applications:

- Reinforcement Learning
- Implicit Neural Representations
- Interpretability
- …



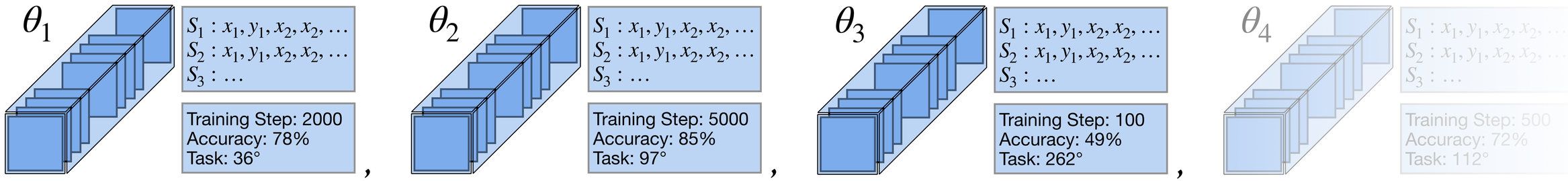
Three Ingredients

- **Datasets**
- **Learning / Pre-Training method**
- **RNN Weight encoder architecture**

Datasets

Two RNN ‘Model Zoos’

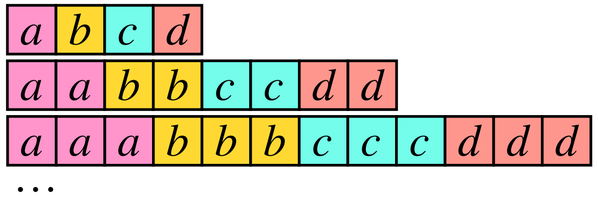
- 1000 LSTM training runs per dataset, 9 snapshots per run
- Each LSTM is trained on a different task from a task family
- Datapoint: LSTM weights, 100 rollouts/trajectories, metadata



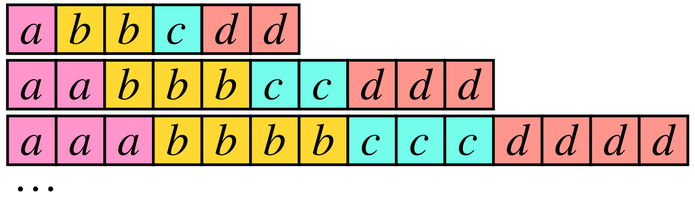
Formal Languages Dataset

- Autoregressive models of 216 different formal languages
- Language family: $L_{m_a,m_b,m_c,m_d} := \{a^{n+m_a}b^{n+m_b}c^{n+m_c}d^{n+m_d} \mid n \in \mathbb{N}\}$
- Examples

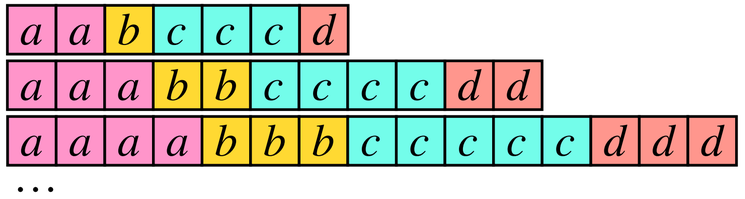
$L_{1,1,1,1}$:



$L_{1,2,1,2}$:



$L_{2,1,3,1}$:

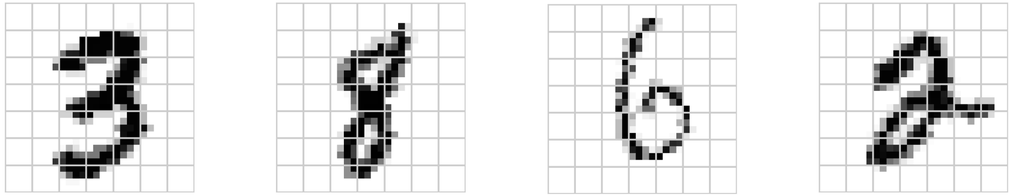


…

Tiled Sequential MNIST Dataset

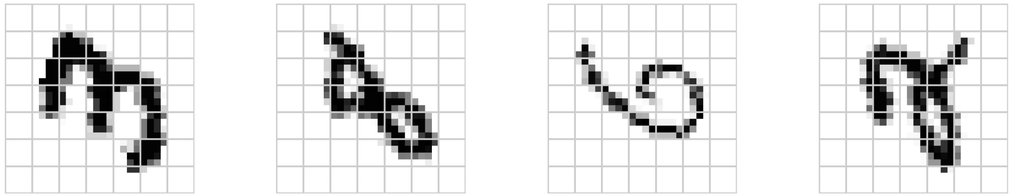
- Classifiers of tiled sequentialised MNIST digits (prediction at every step)
- Every model is trained on a different rotation angle of the digits
- Examples

17° :



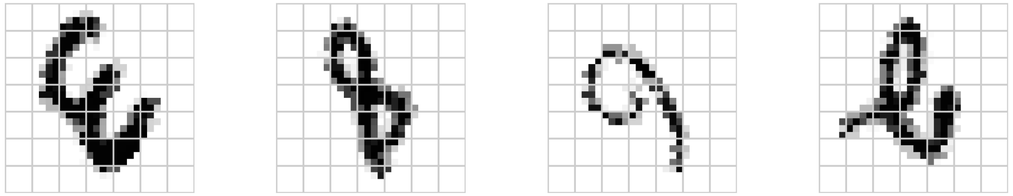
…

74° :



…

231° :



…

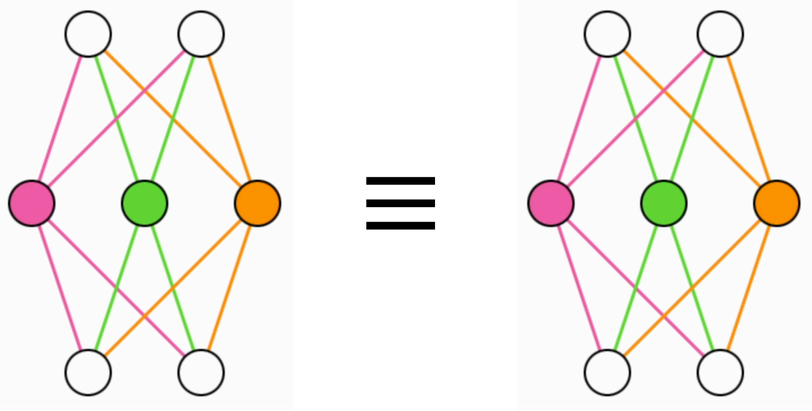
…

Weight Space Symmetries

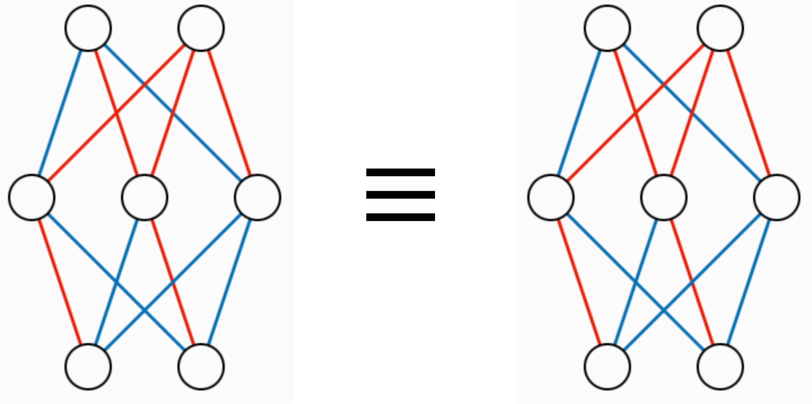
Challenge or Opportunity?

Huge number of equivalent networks

- Hidden Neuron Permutation

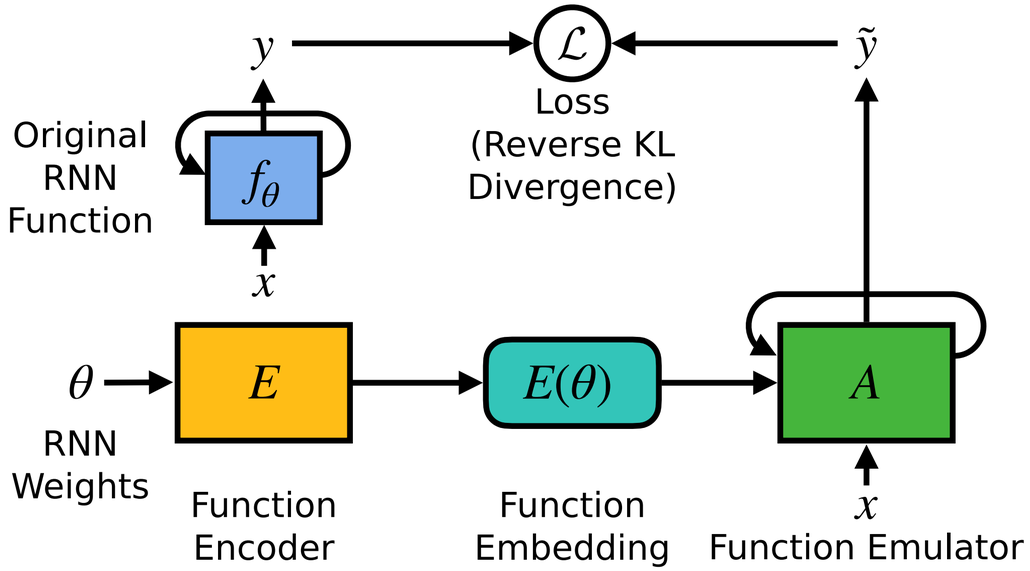


- Sign Flip / Scaling
  (depends on non-linearity)



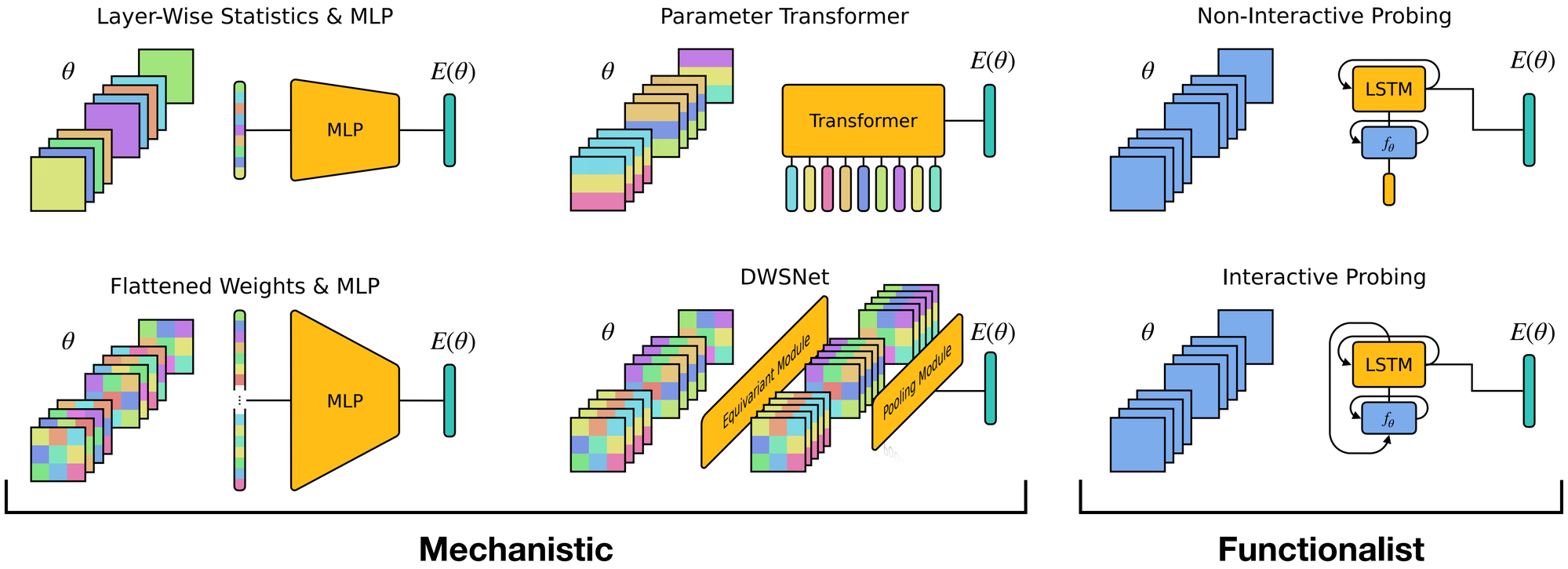
Self-Supervised Training Method

- Reconstruction or naive contrastive methods? 🚨*Weight Space Symmetries*🚨
- *Instead*: Emulate the functionality of the network!
- Encoder E generates representation $E(\theta)$ of RNN f_θ
- Emulator A is conditioned on $E(\theta)$ and imitates f_θ



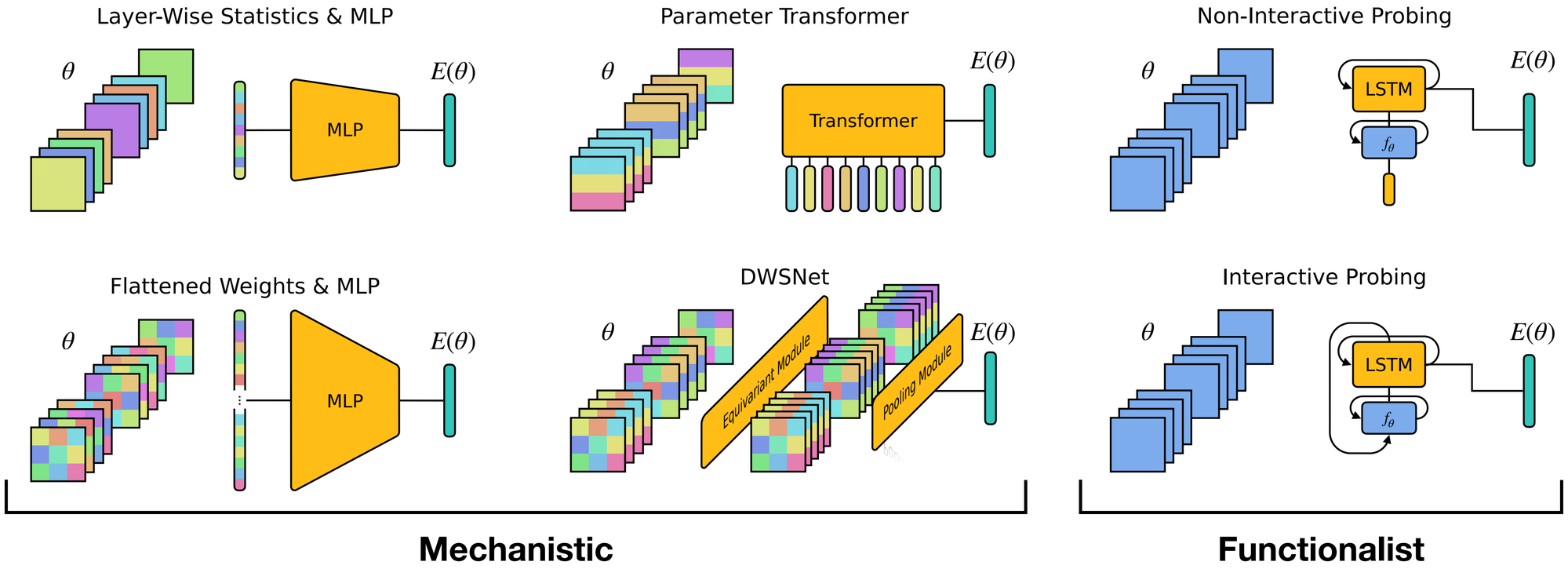
RNN Weight Encoder Architectures

- Distinction: Mechanistic Encoders & Functionalist Encoders
- **Mechanistic** encoders look at the weights θ directly
- **Functionalist** encoders look at the input-output mapping of f_θ



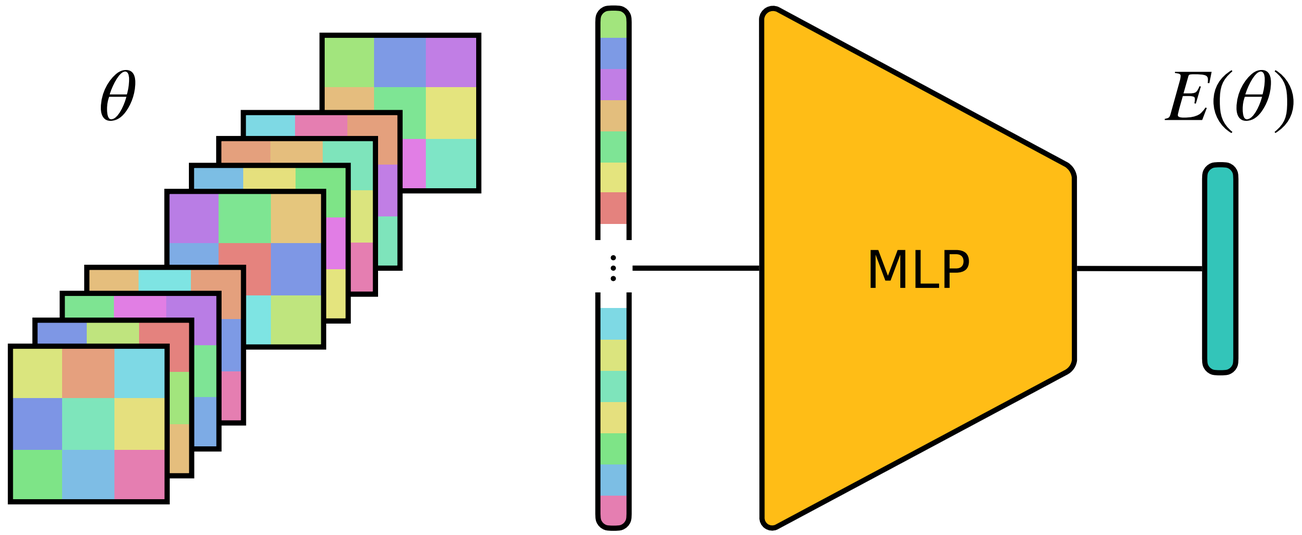
Layer-Wise Statistics & MLP (Mechanistic)

- Concatenate various global statistics of each weight matrix and feed into MLP
- Invariant to hidden neuron permutation 👍
- Not universal 👎



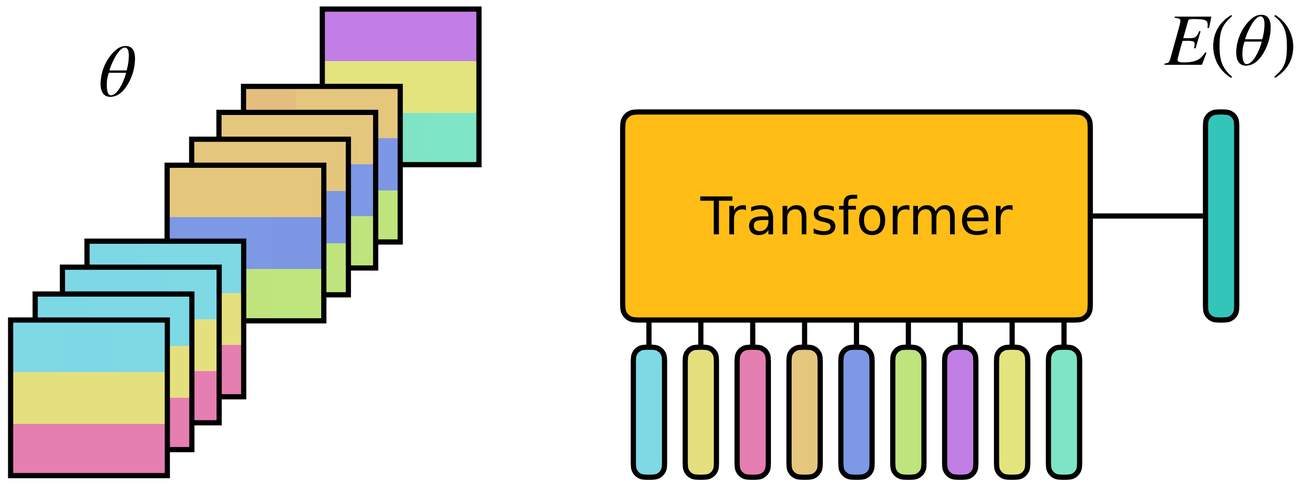
Flattened Weights & MLP (Mechanistic)

- Flatten all weights into one big vector and feed into MLP
- Not invariant to hidden neuron permutation 👎
- Universal 👍



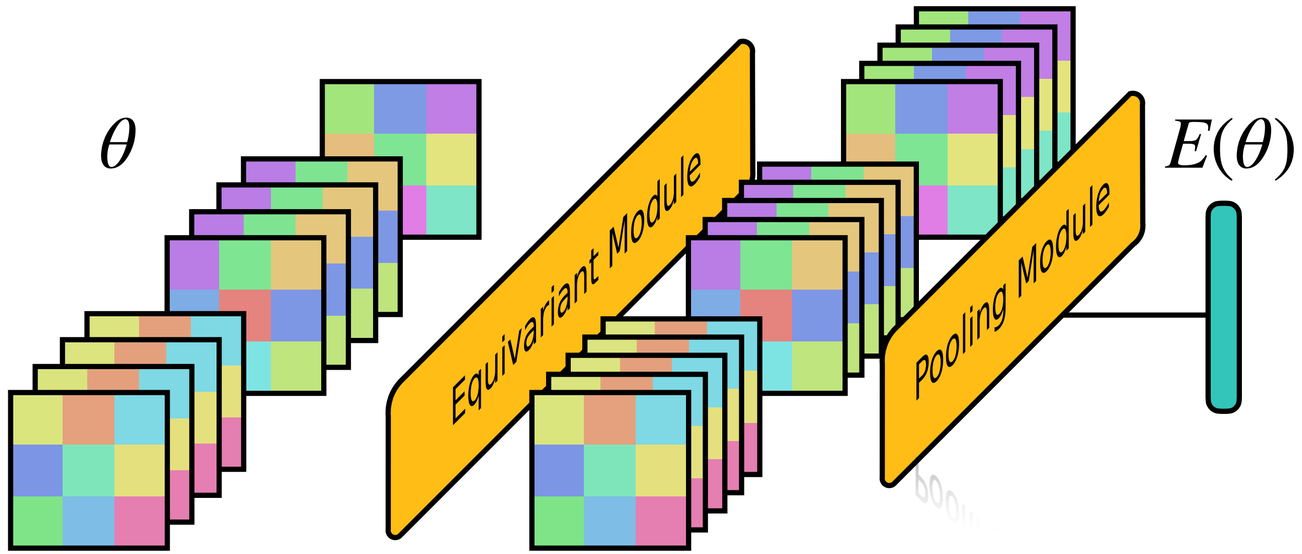
Parameter Transformer (Mechanistic)

- Feed neuron weight vectors as sequence to a transformer model
- Not invariant to hidden neuron permutation 👎
- Universal 👍



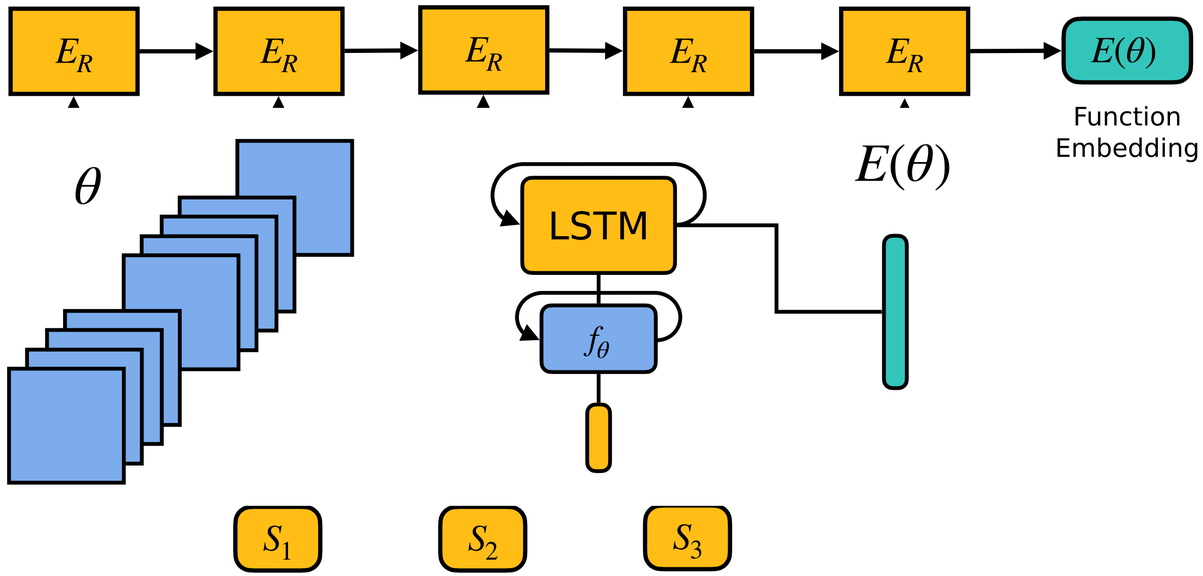
Deep Weight Space Net (Mechanistic)

- Equivariant weight processing modules followed by invariant pooling layer
- Invariant precisely to hidden neuron permutation 👍
- Universal 👍



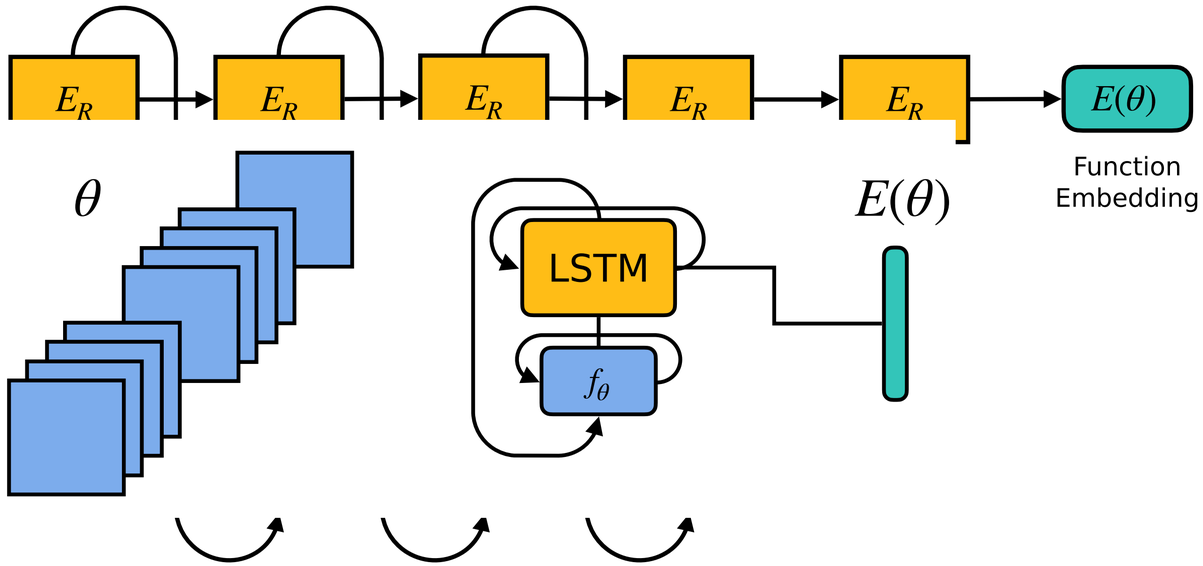
Non-Interactive Probing (Functionalist)

- Fixed but learnable probing sequences are given to f_θ
- Based on the corresponding probing outputs, $E(\theta)$ is computed
- Invariant to everything that leaves f_θ's functionality intact 👍
- Not fully universal 👎



Interactive Probing (Functionalist)

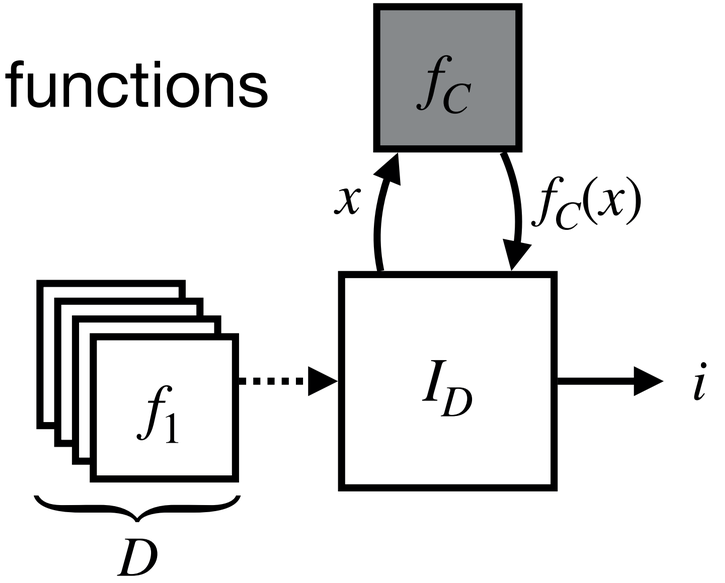
- Probing sequences are dynamically generated by core LSTM
- Next probing inputs depend on f_θ's response to all previous probing inputs
- Invariant to everything that leaves f_θ's functionality intact 👍
- Not fully universal 👎



Functionalist Approach: Theoretical Results

Setup

- Interrogator I_D has to identify f_C from a known set D of functions
- Use as few interactions with f_C as possible



What's the difference between interactive and non-interactive probing?

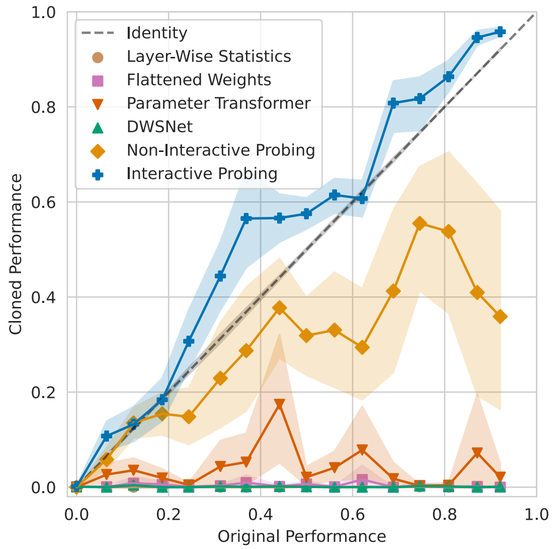
Results

- General upper bound for required interactions is the same
- For certain function sets, interactive probing is exponentially more efficient

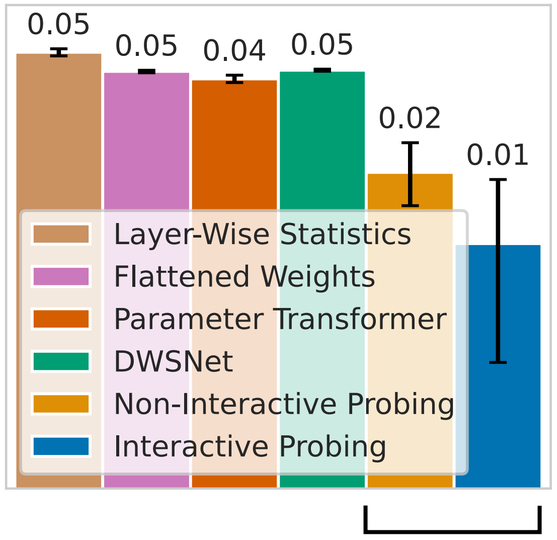
Pre-Training Results

Formal Languages

Original vs. Emulated



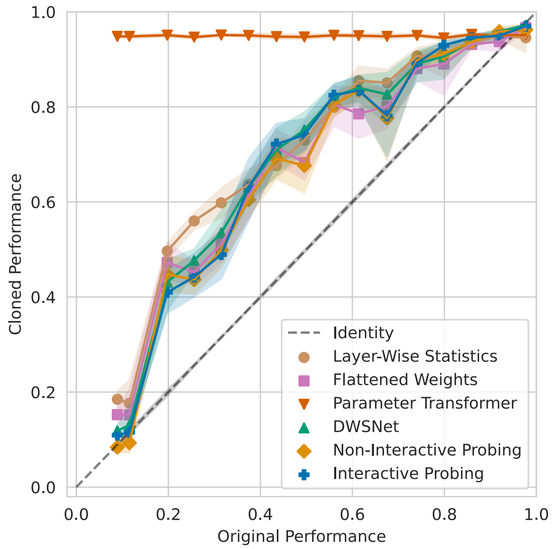
Validation Loss



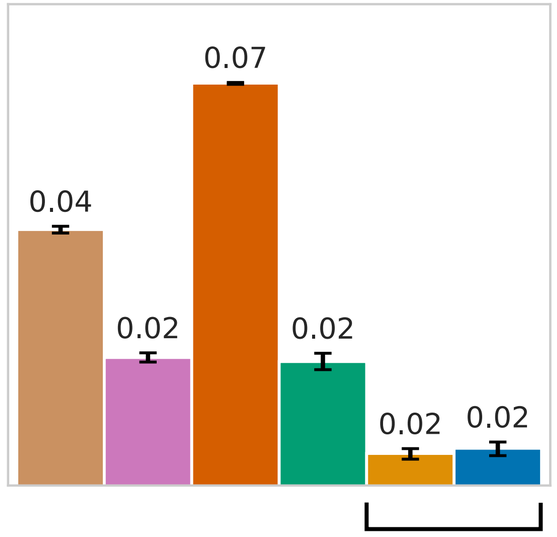
Functionalist

Sequential MNIST

Original vs. Emulated



Validation Loss

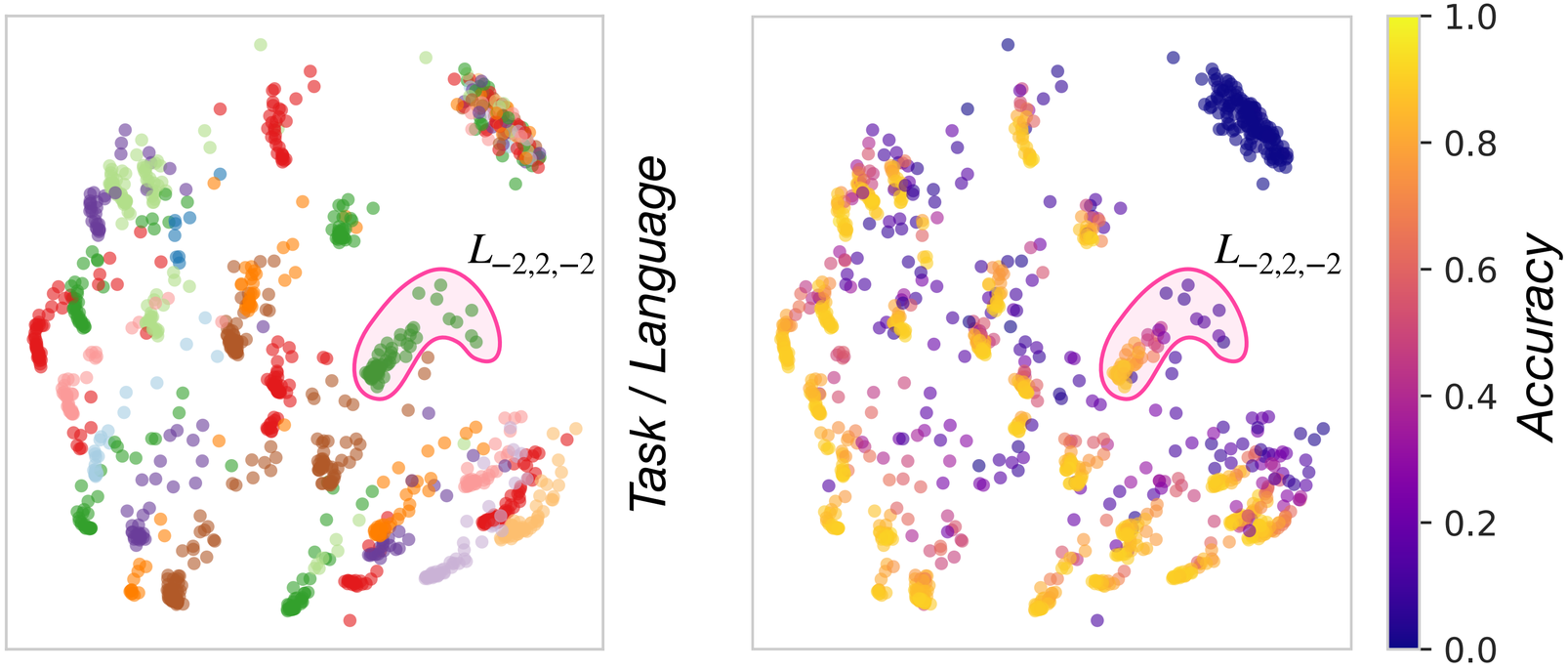


Functionalist

Learned Embedding Spaces

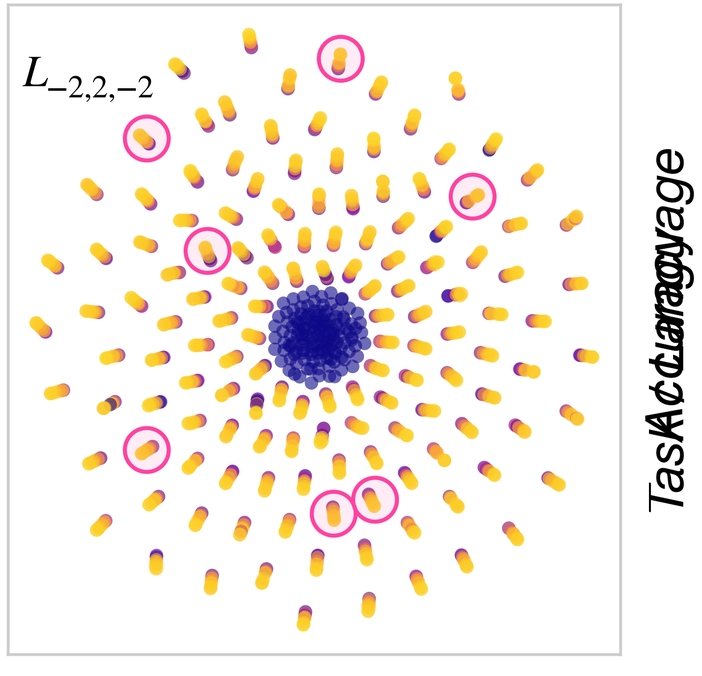
Formal Languages

Interactive Fingerprinting Embeddings (PCA)



Comparison:

t-SNE of weights

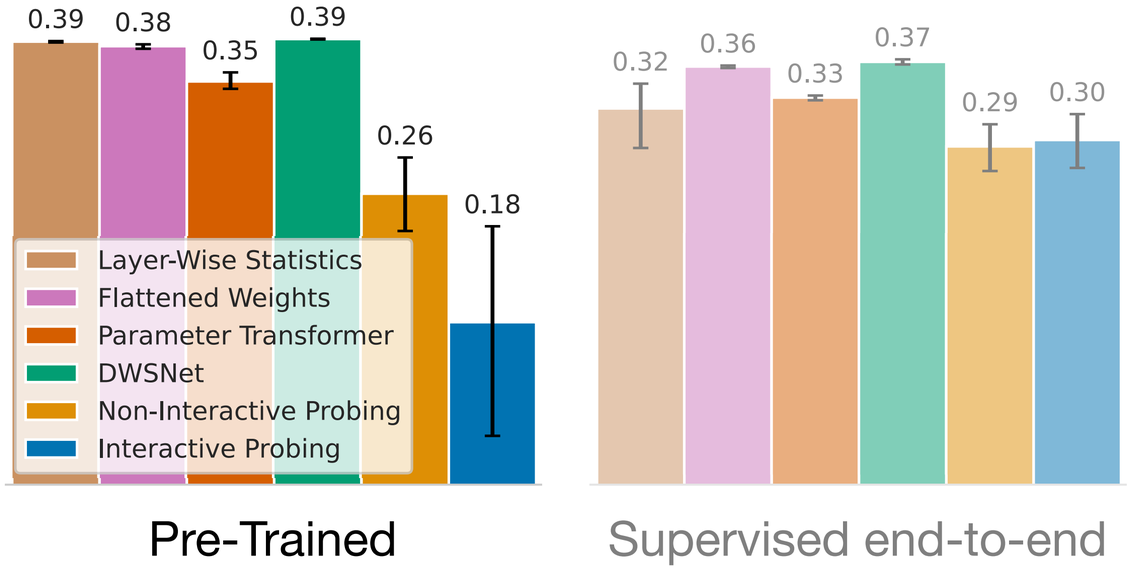


Downstream Results

Predictor MLP on top of learned representations

Formal Languages

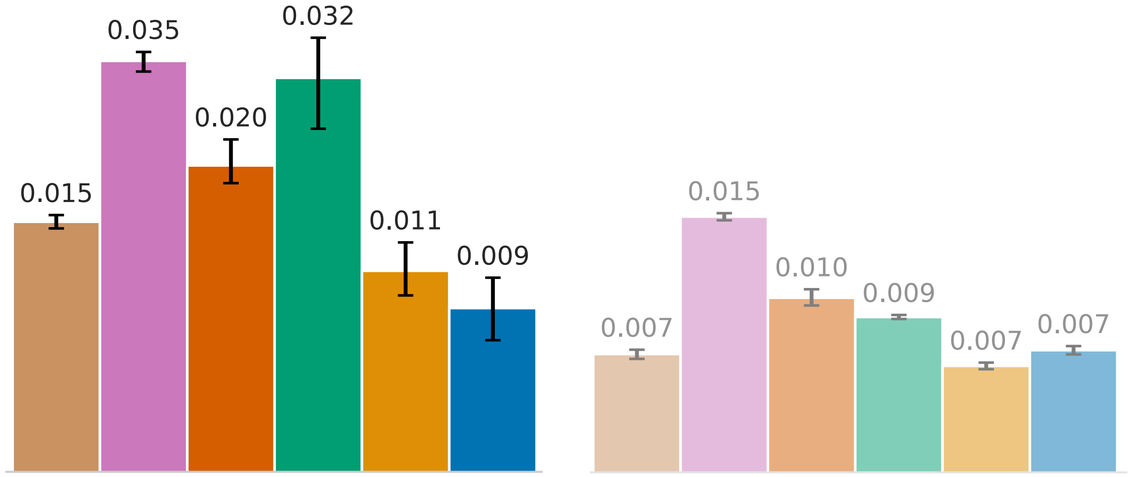
Task prediction loss



Pre-Trained

Supervised end-to-end

Accuracy prediction loss



Pre-Trained

Supervised end-to-end

Conclusion

Learning RNN Weight Representations

- Two RNN ‘Model Zoo’ datasets
- Emulation-based pre-training method
- Distinction between **mechanistic** and **functionalist** weight encoders
- Two novel functionalist encoder types
- Comparison of six different RNN weight encoder architectures

Functionalist encoders are superior at complex tasks

Thank you!

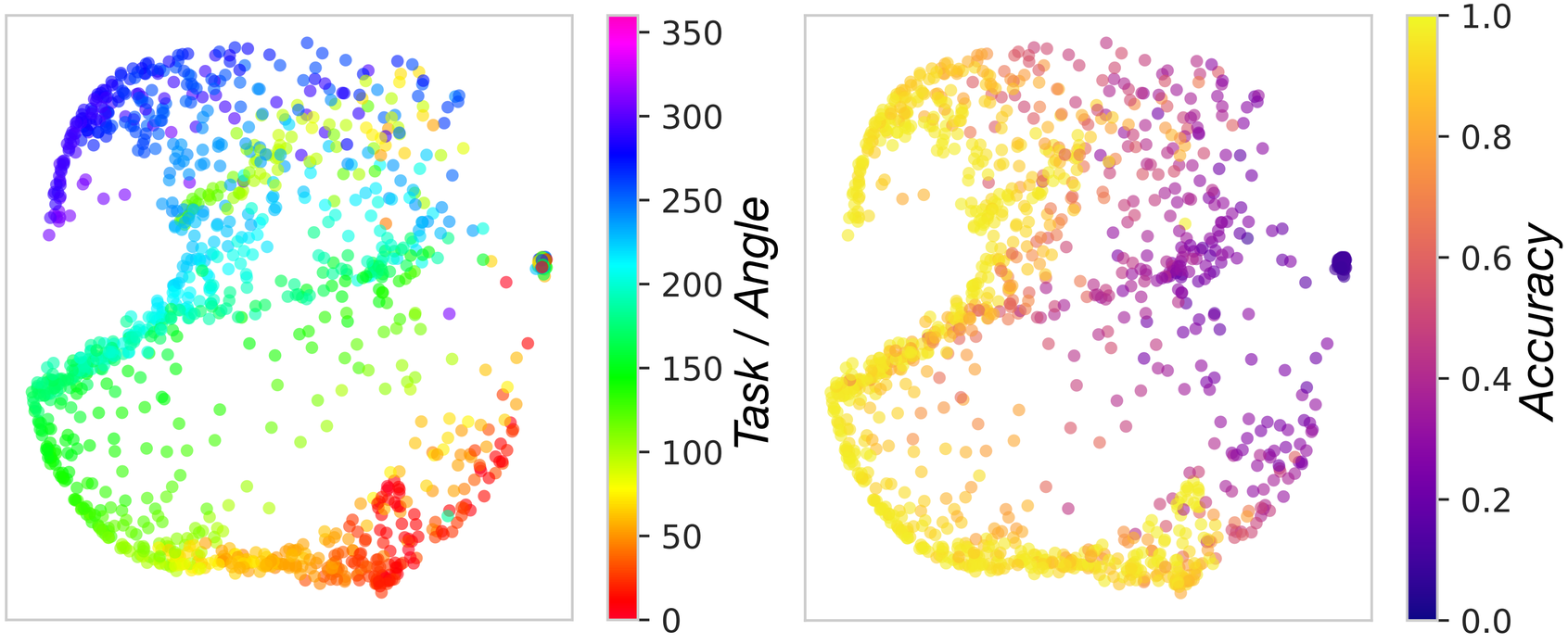
Paper

Code & Datasets

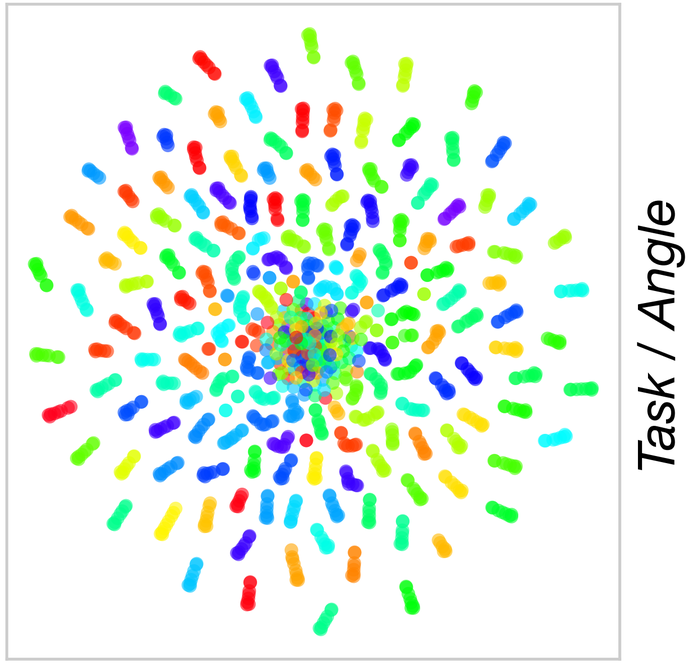
Learned Embedding Spaces

Sequential MNIST

Interactive Fingerprinting Embeddings (PCA)

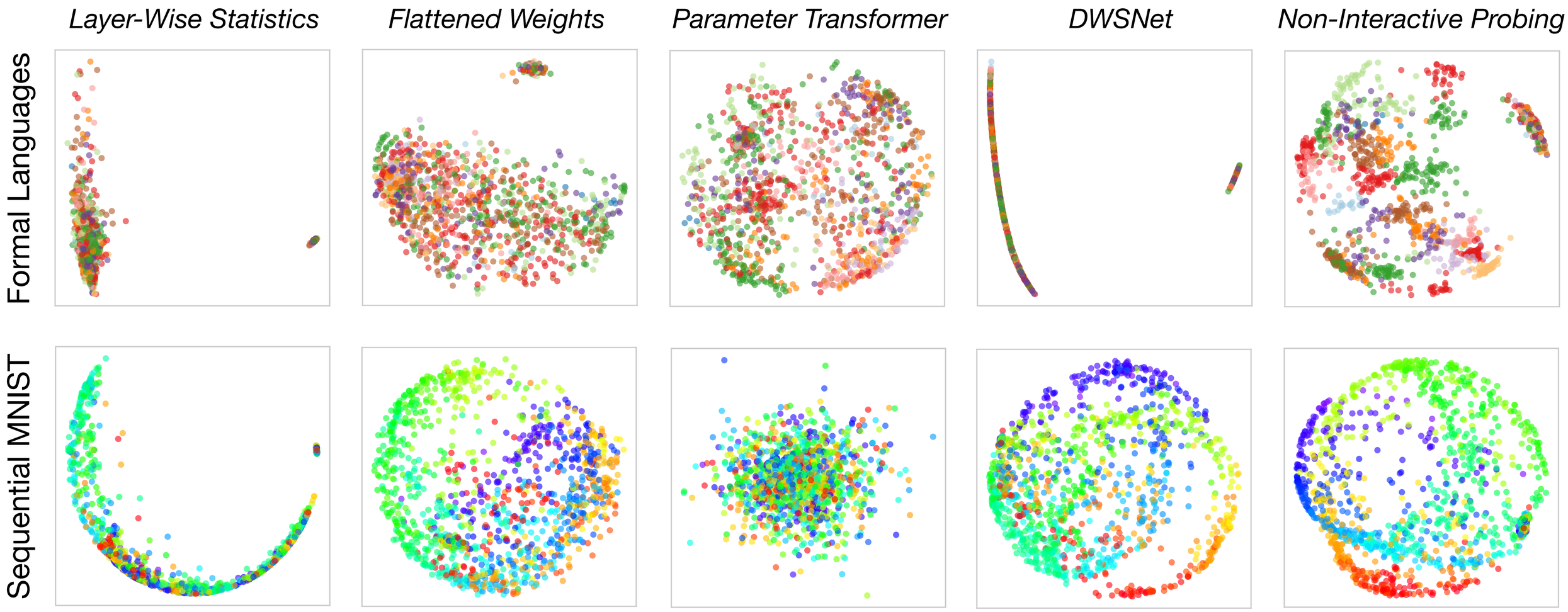


Comparison:
t-SNE of weights



Learned Embedding Spaces

Other encoder architectures, coloured by task

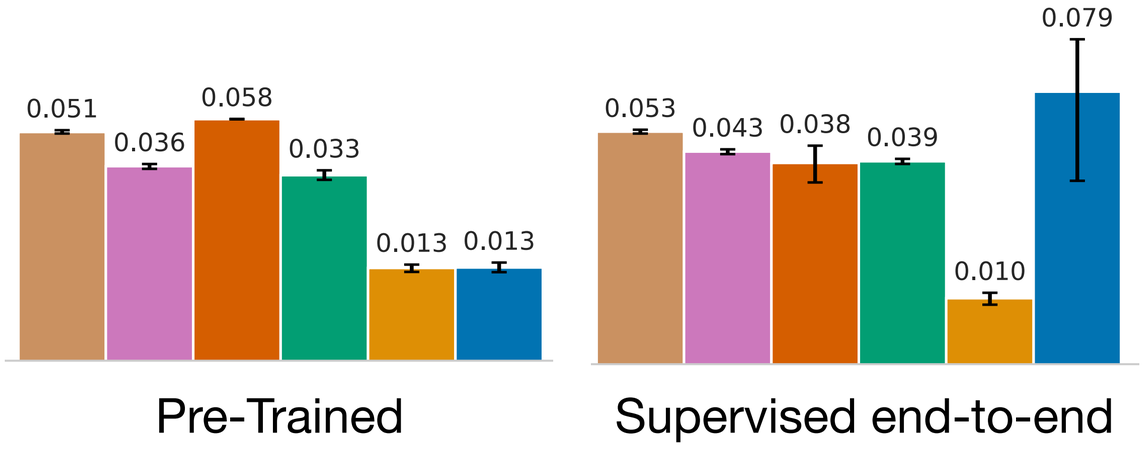


Downstream Results

Predictor MLP on top of learned representations

Sequential MNIST

Task prediction loss



Accuracy prediction loss