Investigating ICL Beyond Self-Attention

Rohan Gulati, Alena Chao, Natalie Wei, Andrew Choy, Minjune Kim

Abstract

In Context Learning (ICL) is a key capability of large language models, yet the mechanisms that enable it remain poorly understood. This project investigates whether self-attention is a necessary component for ICL on a lower scale. We will implement several similarly-sized minimal sequence models, including Recurrent Neural Networks (RNNs), Long-Short Term Memory Networks (LSTMs), and State Space Models (SSMs) on simple language tasks that Transformers can accomplish. We expect to show that this ability emerges across architectures and is not limited to self-attention.

Introduction and Background

Our investigation is built upon a body of work that seeks to understand ICL and explore its presence in other architectures:

- **Language Models are Few-Shot Learners**: This paper introduced in-context learning in large-scale Transformer models.
- **What can transformers learn in-context? A case study of linear models**: This paper provides a key “simple task” testbed, demonstrating that Transformers can learn simple function classes like linear regression in-context.
- **A Theoretical View on In-Context Learning: Transformers as Algorithm Learners**: This work provides a theoretical framework suggesting that Transformers performing ICL are effectively learning and executing algorithms specified by the in-context examples.
- **In-context Learning and Induction Heads**: This paper offers a mechanistic explanation for how self-attention enables ICL, identifying “induction heads” that copy and re-use information from context.
- **What determines in-context learning? Investigating the role of architecture and training**: This finds that standard LSTMs struggle with ICL on complex tasks where Transformers excel.
- **Transformers as Statisticians: Provable In-Context Learning with In-Context Algorithm Selection**: Proves that transformers can implement a broad class of statistical learning algorithms and even select among them in-context without parameter updates.
- **What and How does In-Context Learning Learn? Bayesian Model Averaging, Parameterization, and Generalization**: Adopts a Bayesian

perspective and shows that ICL with transformers approximates Bayesian model averaging and gives bounds on error rates and generalization.

- **From Markov to Laplace: How Mamba In-Context Learns Markov Chains**: This work shows that the state-space model architecture Mamba (an SSM) can perform in-context learning of Markov chain transition dynamics.

Key Questions

1. Is the self-attention mechanism required for ICL?
2. Can simpler architectures, like RNNs or LSTMs, demonstrate ICL on simple algorithmic tasks?
3. Do modern sequence models without attention mechanisms, such as SSMs, demonstrate ICL capabilities comparable to Transformers?

Hypothesis

We hypothesize that ICL is a property of any sequence model with the architecture to learn an algorithm based on the provided in-context sequence. We predict that SSMs will demonstrate ICL performance similar to Transformers on simple algorithmic tasks. We further hypothesize that while standard RNNs/LSTMs will initially fail, models trained on specific long-sequence prompts can also show ICL emergence, showing that attention is not crucial.

Methods

We will develop a “simple task” setting. This would involve creating small-scale text pattern recognition tasks, synonym matching, and logic tasks to be trained separately.

Architectures: We will implement 4 “minimal” models in PyTorch, matching parameter counts as closely as possible: a 2-4 layer decoder-only Transformer with self-attention, a standard RNN and LSTM, and a SSM.

Training: All models will be trained from scratch on the same training data distribution. The task will be standard next-token prediction. We will vary the sequence length and the number of in-context examples (k) during training.

Evaluation: The primary metric will be the model’s test loss on held-out tasks, plotted as a function of the number of examples (k) provided at inference time. The emergence of ICL is defined as a “scaling law” where the model’s error decreases as k increases, approaching the error of the optimal algorithm.

Expected Compute

We expect training for each model to take 4-8 hours on a single GPU. The primary cost will be the hyperparameter search and training runs for each of the architectures. We estimate a total compute budget of $\sim 100 - 150$ GPU hours.