# Post-training: Capabilities

## CSC2541H1 Topics in Machine Learning, Winter 2025, UToronto

**Chris J. Maddison**

# Announcements

- If you are assigned to **present on March 14, come talk** about presentations after class.
- The Project Proposal marks are up on MarkUs.
- Drop deadline is today - if you have concerns please discuss with me.

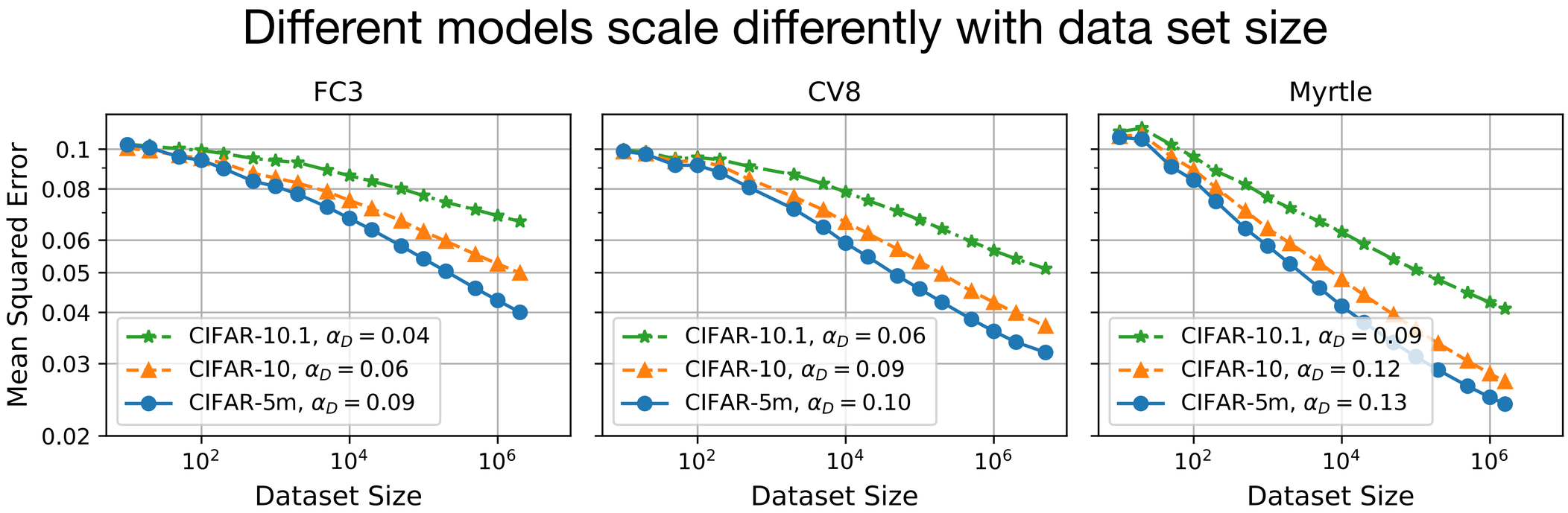
**Questions?**

# Today

- **Capability** is the ability to achieve a desired outcome
  - How do we specify desired outcomes?
  - How do we improve the ability of models to achieve those outcomes?
- But first, a detour into a question that has come up a few times.
  - Does the architecture change the scaling law?

# On the one hand

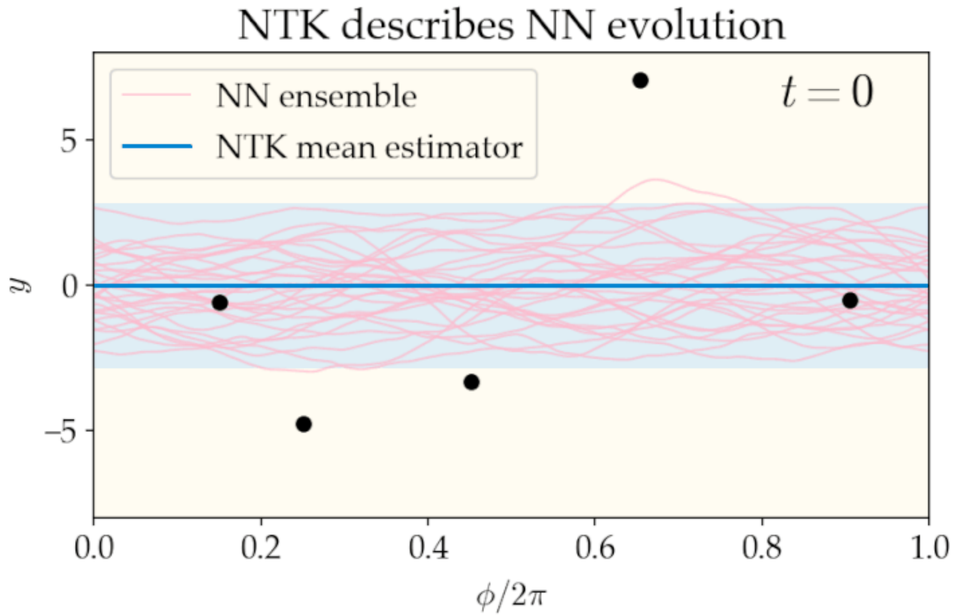
## it seems like the answer must be yes




Adlam et al, 2023, Kernel Regression with Infinite-Width Neural Networks on Millions of Examples

# How does architecture affect scaling?

- If you make neural architectures bigger, their training dynamics look more and more like linear regression in the limit.
- Can we study scaling laws via linear regression?
- Some people are trying!



# Linear regression scaling laws

**high M-dim linear regression**

$$y = x^\top w^* + \epsilon$$

**power law structure**

$$\lambda_k(Cov(x)) \propto k^{-b}$$

$$(w_k^*)^2 \lambda_k(Cov(x)) \propto k^{-a}$$

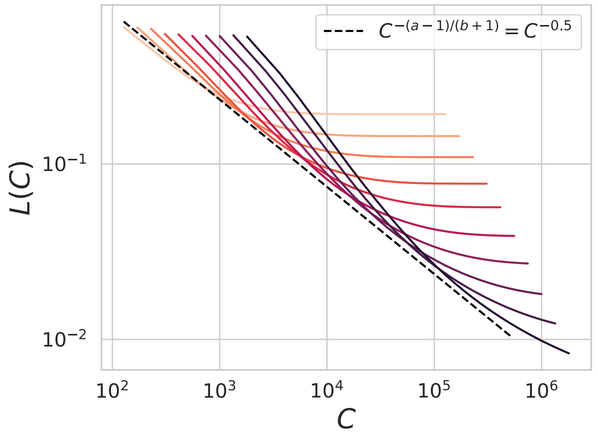
**the architecture influences this**

**these are like the finite NTK features**

**observe N-dim projections**

$$\{Px_i, y_i\}_{i=1}^{D} \text{ where } P \in \mathbb{R}^{N \times M}$$

**scaling laws in N and D**

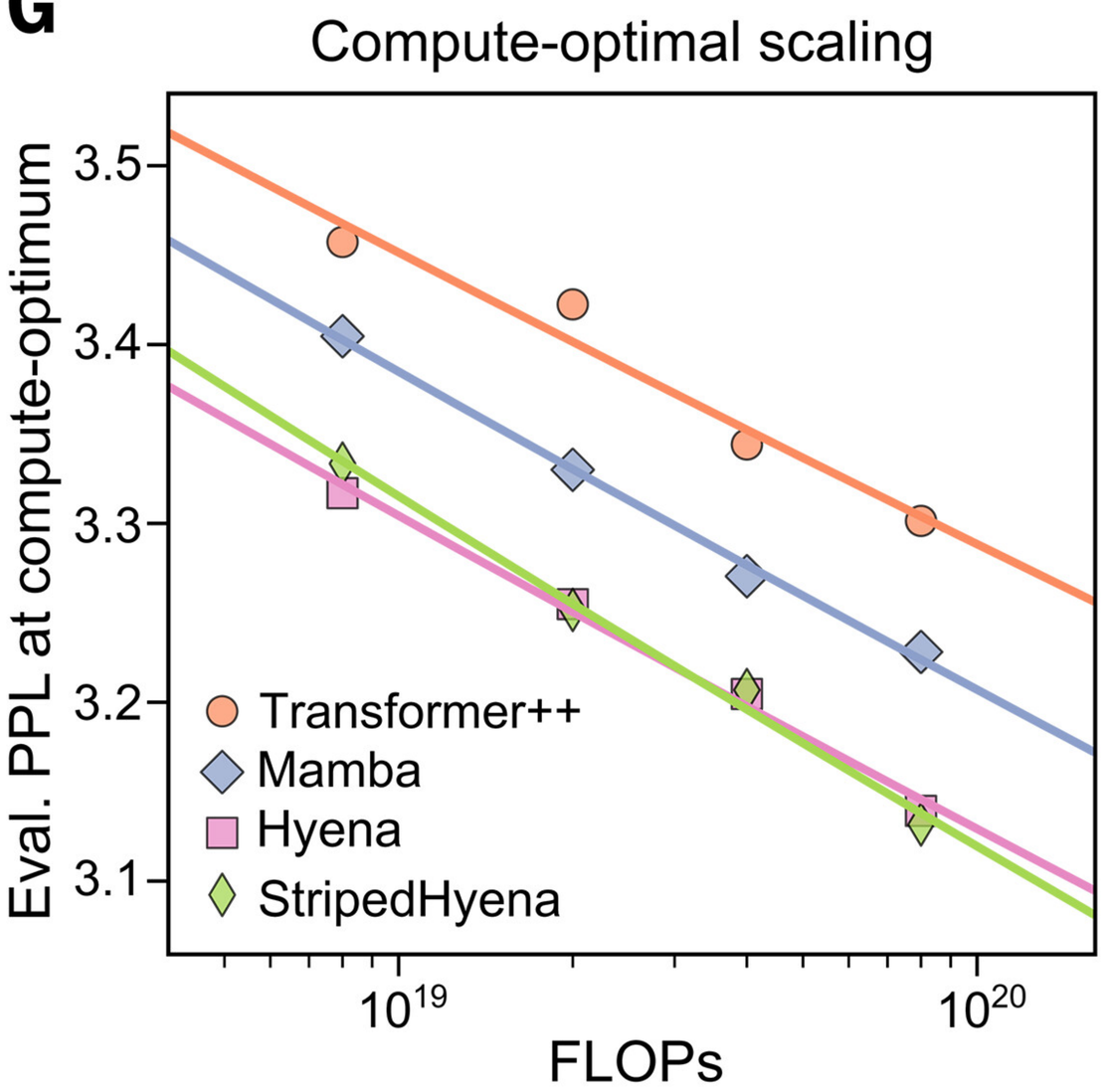



Bordelon et al, 2024, A Dynamical Model of Neural Scaling Laws

# On the other hand

## In practice…

- Very different architectures often have the similar scaling rates.
  - Different constant offsets, but similar rates.
- Why??? I don’t know.
- It seems hard to design the rate.




Nguyen et al, 2024, Sequence modeling and design from molecular to genome scale with Evo

# Some interesting directions

## That try to improve the scaling rate via architecture

- https://arxiv.org/abs/2412.09871
- https://arxiv.org/abs/2406.06248

# Post-training for capabilities

## Back to today's topic…

- The question that we're considering today is: **how do we ensure that our models are capable enough to solve problems we care about?**
  - Pre-training encourages models to be strong next-token predictors
  - Alignment methods encourage models to maximize our preferences

# Post-training for capabilities

- Post-training for capabilities is a very diverse set of methods.
- To classify post-training methods, you can as **two questions:**
  - What information does it have access to?
  - How does it use that information?

# For example…

| | | | |
|---|---|---|---|
| **Information source** | Python interpreter | Supervised data with binary label (e.g., correct / not correct) | Base LLM capabilities & unsupervised data |
| **Use of information** | Execution feedback for RL | Feedback signal for EM algorithm | Bootstrap data generation (generate, feedback, refine) |
| **Examples** | PAL (Gao et al, 2022) used an interpreter as a tool, Llama 3 used an interpreter to | ReST$^{EM}$ (Singh et al, 2024) | Self-Refine (Madaan et al, 2023) used this to improve the completion distribution, Llama 3 seems to have used this to self-refine data for SFT |

# PAL

## Gao et al, 2022

**Chain-of-Thought (Wei et al., 2022)**

Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 tennis balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Q: The bakers at the Beverly Hills Bakery baked 200 loaves of bread on Monday morning. They sold 93 loaves in the morning and 39 loaves in the afternoon. A grocery store returned 6 unsold loaves. How many loaves of bread did they have left?

Model Output

A: The bakers started with 200 loaves. They sold 93 in the morning and 39 in the afternoon. So they sold 93 + 39 = 132 loaves. The grocery store returned 6 loaves. So they had 200 - 132 - 6 = 62 loaves left.
The answer is 62.

**Program-aided Language models (this work)**

Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 tennis balls.
```
tennis_balls = 5
```
2 cans of 3 tennis balls each is
```
bought_balls = 2 * 3
```
tennis balls. The answer is
```
answer = tennis_balls + bought_balls
```

Q: The bakers at the Beverly Hills Bakery baked 200 loaves of bread on Monday morning. They sold 93 loaves in the morning and 39 loaves in the afternoon. A grocery store returned 6 unsold loaves. How many loaves of bread did they have left?

Model Output

A: The bakers started with 200 loaves
```
loaves_baked = 200
```
They sold 93 in the morning and 39 in the afternoon
```
loaves_sold_morning = 93
loaves_sold_afternoon = 39
```
The grocery store returned 6 loaves.
```
loaves_returned = 6
```
The answer is
```
answer = loaves_baked - loaves_sold_morning
  - loaves_sold_afternoon + loaves_returned
```

```
>>> print(answer)
74
```

Python variables carry semantic meaning in their names

Run program to get answer

In-context examples of Python code that solves the question, with CoT reasoning as comments

# PAL - strong empirical performance

## Gao et al, 2022

| | GSM8K | GSM-HARD | SVAMP | ASDIV | SINGLEEQ | SINGLEOP | ADDSUB | MULTIARITH |
|---|---|---|---|---|---|---|---|---|
| DIRECT Codex | 19.7 | 5.0 | 69.9 | 74.0 | 86.8 | 93.1 | 90.9 | 44.0 |
| COT UL2-20B | 4.1 | - | 12.6 | 16.9 | - | - | 18.2 | 10.7 |
| COT LaMDA-137B | 17.1 | - | 39.9 | 49.0 | - | - | 52.9 | 51.8 |
| COT Codex | 65.6 | 23.1 | 74.8 | 76.9 | 89.1 | 91.9 | 86.0 | 95.9 |
| COT PaLM-540B | 56.9 | - | 79.0 | 73.9 | 92.3 | 94.1 | 91.9 | 94.7 |
| COT Minerva 540B | 58.8 | - | - | - | - | - | - | - |
| PAL | **72.0** | **61.2** | **79.4** | **79.6** | **96.1** | **94.6** | **92.5** | **99.2** |

Table 1: Problem solve rate (%) on mathematical reasoning datasets. The highest number on each task is in **bold**. The results for DIRECT and PaLM-540B are from Wei et al. (2022), the results for LaMDA and UL2 are from Wang et al. (2022b), and the results for Minerva are from Lewkowycz et al. (2022). We ran PAL on each benchmark 3 times and report the average; the standard deviation is provided in Table 7.

# PAL - ablation study of prompting format

## Gao et al, 2022

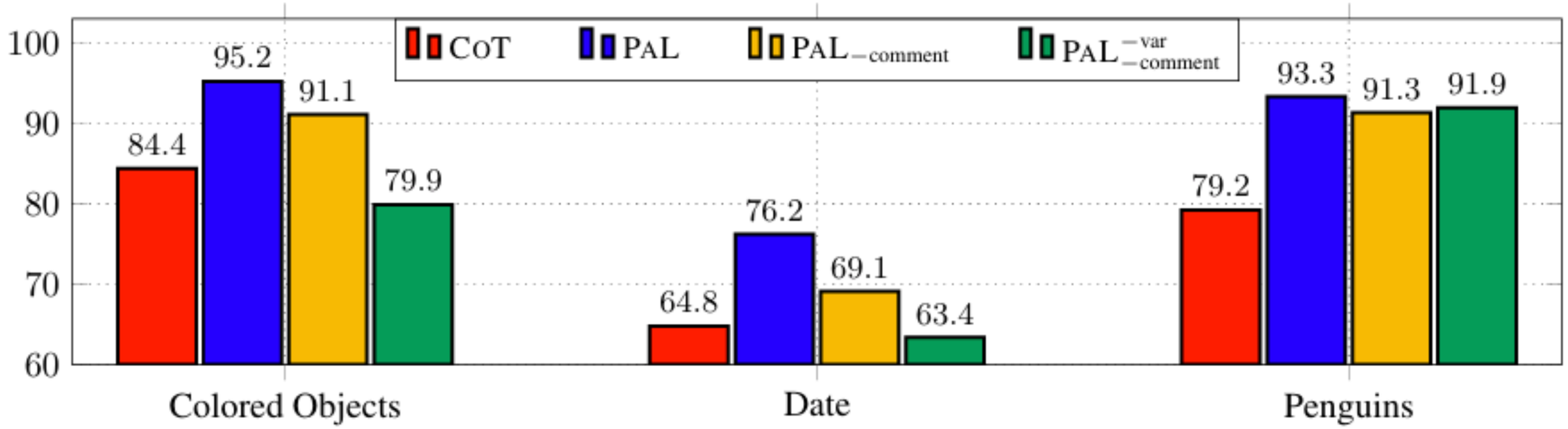



Figure 9: Ablation study of PAL prompt formats. We consider the original PAL prompt, it with natural language comments removed ($\text{PAL}_{-\text{comment}}$), and further variable names replaced with random character ($\text{PAL}^{-\text{var}}_{-\text{comment}}$). As a reference, we also show the CoT performance (blue).

# Llama 3

## PAL is a prompting method. Can we use interpreter for learning?

- Llama 3 used the interpreter / compiler for learning.
- **Synthetic data generation with execution feedback:**
  - **Problem generation:** Sampled diverse code snippets and ask Llama 3 to create programming challenges
  - **Solution generation:** Prompt Llama 3 to solve the problem
  - **Correctness verification:** Run solution through
    - static analysis (parsing/linting) and unit tests (generated by Llama 3!!)
  - **Self-correction loop:** When solutions failed, model received execution feedback and revised
- About 20% of solutions were initially incorrect but successfully self-corrected
- Only dialogues passing all correctness checks were included in the final training dataset

# Take-homes

## Post-training for capabilities

- Post-training for capabilities is a very active area of inquiry.
- There is a lot of diversity and not just “one” method.