Scaling Laws for Autonomous Driving Models

How much and reasonable we can scale up models in Autonomy

Aleksandr Petiushko (apetiushko@nuro.ai)
ML Research
12/14/2023

ML4AD 2023 Symposium

Acknowledgements

Brian Yao

Zhuwen Li

Content

Motivation

0

Content

Use case 1: Behavior

Behavior Models: usually smaller compared to Vision ones.
Size vs Overfitting?

Use case 2: Perception

Perception Models: usually large enough.
Can we go bigger? Practically?

Approach

Let’s ablate:
Models Capacity and
Training Datasets size
Under the Fixed Computational Budget

Scaling Laws

1

Scaling Laws in AD: Main Goals

01

Goal 1
How better can we go with scaling up data/models?

Let's do it by ablation studies with different model capacities and training dataset sizes

Scaling Laws in AD: Main Goals

Let’s include in the ablations the conditioning on the number of training iterations that will help with a computation budget

(02)

Goal 2
How practical is the scaling?

Introduction

Computer vision[1,2] and Large language[3,4] models have shown great success by scaling to **billion/trillion-parameter** models and training with **web-scale data**.

However, can **self-driving** industry also benefit from this?

At Nuro, we hope to provide *statistical* answers to the following questions:

1. Do we need **larger models** onboard, and how large would it be?
2. Do we need **more training data**, and how many more do we need?
3. What's the best scale under **fixed budget**?

[1] Zhai X, Kolesnikov A, Houlsby N, Beyer L. Scaling vision transformers. InProceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition 2022 (pp. 12104-12113).
[2] Liu Z, Mao H, Wu CY, Feichtenhofer C, Darrell T, Xie S. A convnet for the 2020s. InProceedings of the IEEE/CVF conference on computer vision and pattern recognition 2022 (pp. 11976-11986).
[3] Kaplan J, McCandlish S, Henighan T, Brown TB, Chess B, Child R, Gray S, Radford A, Wu J, Amodei D. Scaling laws for neural language models. arXiv preprint arXiv:2001.08361. 2020 Jan 23.
[4] Hoffmann J, Borgeaud S, Mensch A, Buchatskaya E, Cai T, Rutherford E, Casas DD, Hendricks LA, Welbl J, Clark A, Hennigan T. Training compute-optimal large language models. arXiv preprint arXiv:2203.15556. 2022 Mar 29.

Nomenclature

- ***N*** - Model capacity: number of parameters
- ***D*** - Training dataset size: number of samples
- ***S*** - Number of training iterations
- ***C*** - Training budget: cost in FLOPs

$$C = FLOPs(N, S) \propto N \times S$$

- $L_{train/eval}$ - Training or Eval loss metrics
  - We assume the loss is a function of *N*, *D*, and *S*

$$L^*(N^*, D^*, S^*)$$

Problem Formulation

Two types of **scaling laws**:

- Model **performance** scaling law:
  - How does model performance improve with <u>model</u> and <u>data</u> size scales ↑ / ↓?
- **Optimal** model scaling law:
  - What is the optimal scale at different <u>costs</u>?

$$\hat{N}, \hat{D}, \hat{S} = \underset{FLOPs(N,S)=C}{\arg\min} L(N, D, S) \quad \Longrightarrow \quad \hat{N} = A_N C^{\alpha_N} \text{ , and } \hat{S} = A_S C^{\alpha_S}$$

Let's try these simple patterns…

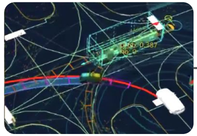
* Note that it will be *argmax* if metrics L is the higher the better

Behavior Encoder Preliminaries & Experiment Details

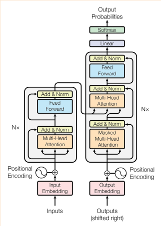
2

Behavior Encoder

We study the scaling laws for some standard tf-based behavior **encoder** model.



Behavior Encoder (BE)



Task-specific decoders

- Prediction
- Planning
- Motion Selection
- Simulation
- ...
- Scene understanding

Experiments Details

Model capacities *(N)*: 1X, 4X, 8X, 16X, 32X
Typical dataset sizes *(D)*: 1X, 2X, 5X, 7X

Id	BE capacity (*N*)	Batch size (K_{GPU} * B_{GPU})
1	**1X**	1K * 4B
2	**4X**	1K * 4B
3	**8X**	2K * 2B
4	**16X**	2K * 2B
5	**32X**	4K * 1B

Baseline model config

Experiment details:

- Fixed total **batch size** = 4*K*B
- **LR**: Same Scheduler
- **Training steps** *(S)*:
  - All models are trained for *S* epochs, using K/2K/4K 40G GPUs A100
  - Training time varies from ~1 day to ~14 days
- **Metrics**: BE train and evaluation losses + prediction and planning metrics

Scaling Laws: Behavior

3.1

- ➔ **Model Performance Scaling Law**
- ➔ Optimal Model Scaling Law Method 1
- ➔ Optimal Model Scaling Law Method 2
- ➔ Scaling Law with Behavior Eval

Model Performance Scaling Law

Let's use as an **optimal** model the best checkpoint of every experiment:

$$L(N, D) = \min_{\forall S \leq \max(S)} L(N, D, S)$$

Loss function is modeled as[3]:

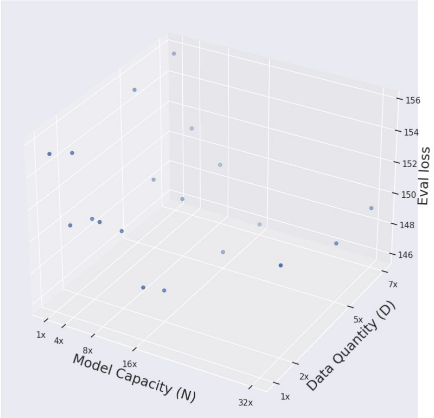
$$L(N, D) = E + A * N^{-\alpha} + B * D^{-\beta}$$

- E - Irreducible error for theoretically infinite model capacity and data
- A - Weight of imperfect model caused by insufficient capacity
- B - Weight of imperfect model caused by insufficient data
- α, β - Model's power law parameters

* Limitation: above formulation assumes N and D scales independently, which is not necessarily true. For example if model is too small, increasing data may result in worse performance.

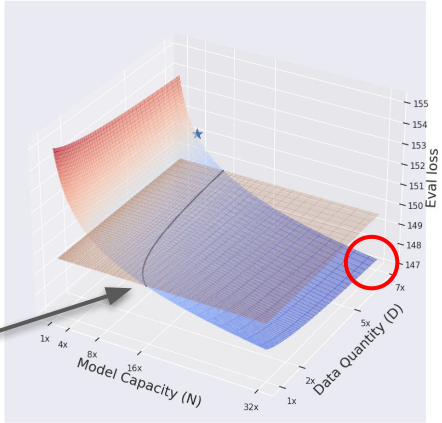
Model Performance Scaling Law

Collected the *(L, N, D)* triplets from experiments



Experiment points (L, N, D)

Fitted the model performance scaling function *L(N, D)*



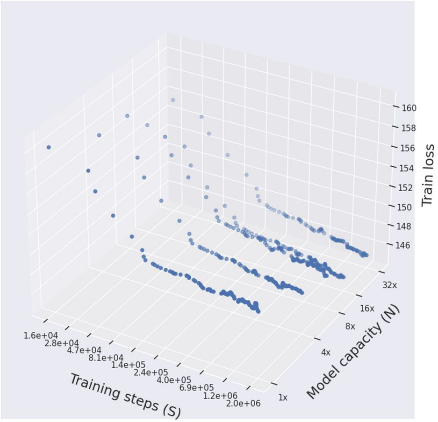
Fitted Performance Scaling Function L(N, D), and profile curve (black) at 1% loss improvement

Conclusion:

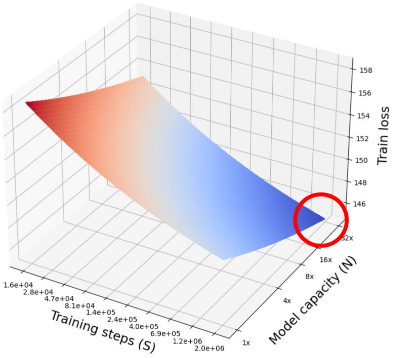
- Better performance: with the larger *N* and *D*
- Fixing *L,* one can find the needed *(N,D)* profile

Model Performance Scaling Law

Collected the *(L, N, S)* triplets from experiments



Fitted the model performance scaling function *L(N, S)*



Fitted Performance Scaling Function L(N, S)

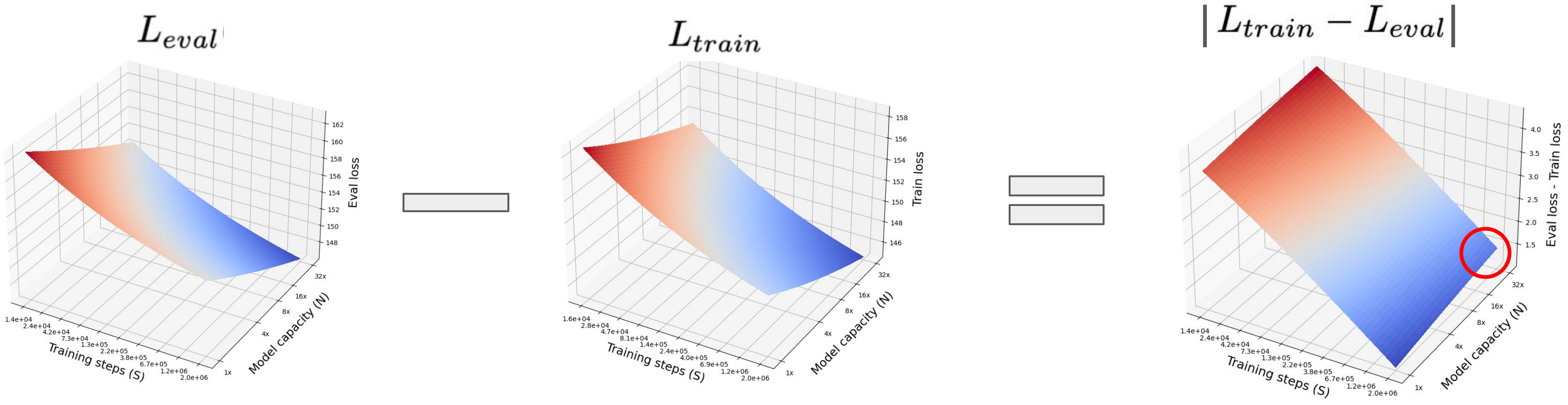
Conclusion:

- Better performance: with the larger *N* and *S*

Model Performance Scaling Law

We can even **compare** *L(N, S)* for the Train and Eval losses!

Conclusion: with larger model and longer training the train/eval **loss gap** is **smaller**.



Ideal setting is nice but…

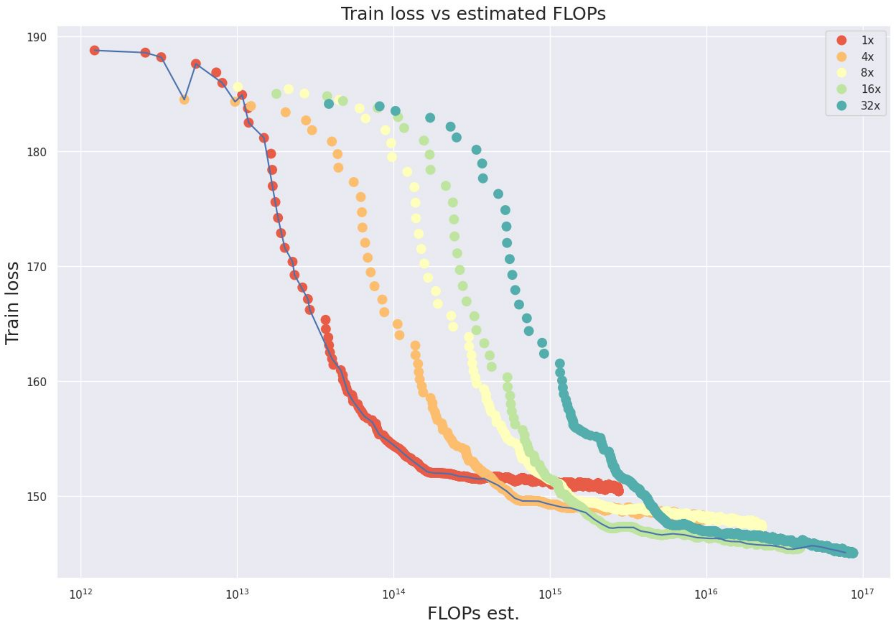
what about **practical limitations**?

Scaling Laws: Behavior

3.2

Fixed Model Size with Varying FLOPs - Train Loss

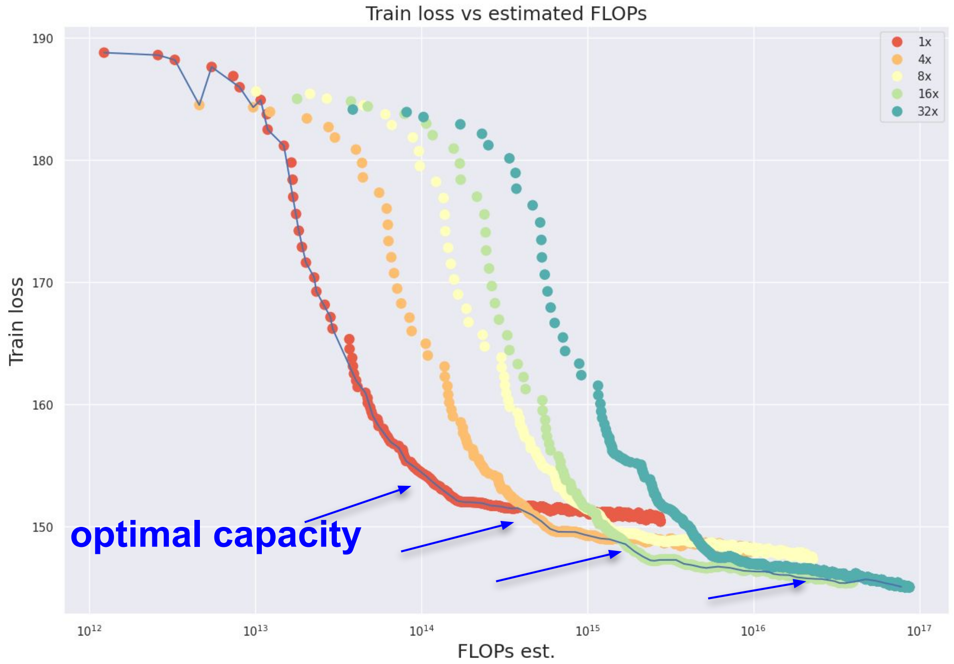
- Most **realistic** setting: the largest dataset, **BE 7X data** (D)
- Start with the **training loss** at varying estimated FLOPs C



Fixed Model Size with Varying FLOPs - Train Loss

Then: at each FLOPs point, get the lower envelope of the training loss, as the "**optimal capacity**".

- Note 1: for different C the **optimal capacity** can belong to **different models**
- Note 2: every **point** corresponds to some # of **training steps** S, and **color** to **capacity** N



Fixed Model Size with Varying FLOPs - Train Loss

Approach: to use the lower envelope to fit a power law between N and C

- Linear regression in **log** scale:

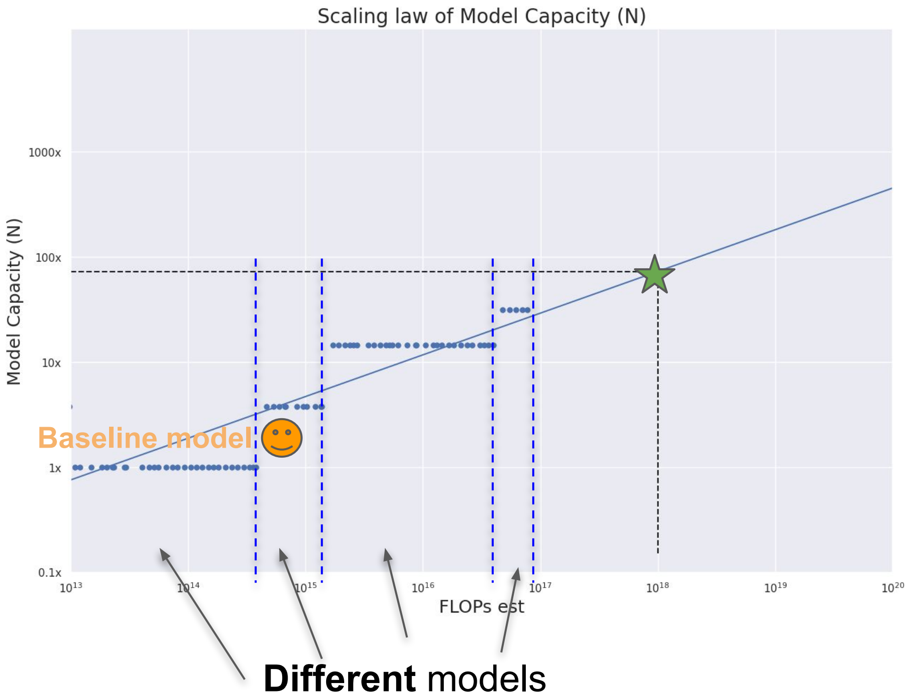
$$\log(N) = 0.397 * \log(C) + 1.253$$

- In **linear** scale:

$$N = 3.501 * C^{0.397}$$

Then: estimate best model capacity $N*$ at a **given cost budget** $C*$, e.g.:

- $C^* = 1e18 \longrightarrow N^* \approx 72X$



Note: the bigger C, the higher N of the optimal model

Fixed Model Size with Varying FLOPs - Train Loss

Approach: to use the lower envelope to fit a power law between S and C

- Linear regression in **log** scale:

$$\log(S) = 0.603 * \log(C) - 8.877$$

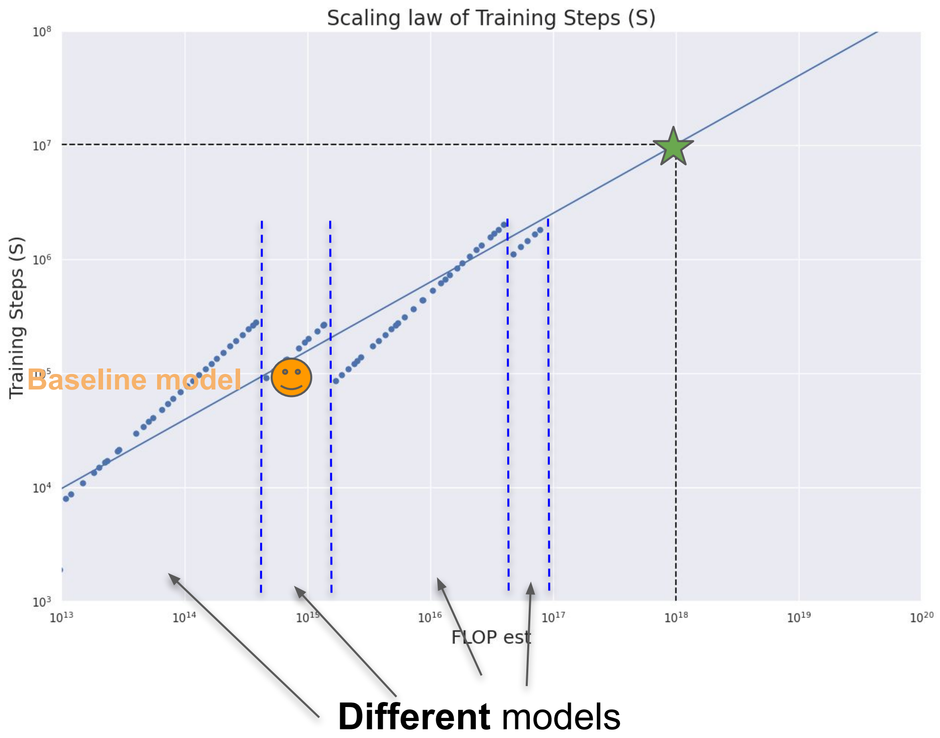
- In **linear** scale:

$$S = 1.4\text{e}{-4} * C^{0.603}$$

Then: estimate best training steps $S*$ at a **given cost budget** $C*$, e.g.:

- $C^* = 1\text{e}18 \longrightarrow S^* \approx 10.06M$

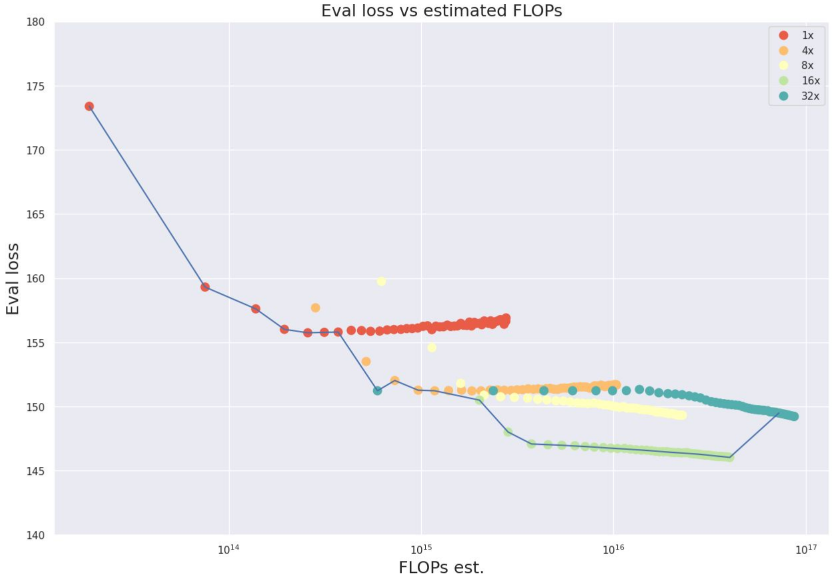
Note: S can be the same for different C



Can similar **scaling laws** be applied to different setting,

e.g. **Eval** Loss?

Fixed Model Size with Varying FLOPs - Eval Loss



Observations:

- **Smaller** models **diverge** after training for a while, while **larger** models are not **even fully converged**
  - May relate to the Double Descent / Broken Neural Scaling Laws[5].
- **Small** models (1x and 4x) **converge fast but unstable**, **larger** models (32x) **converge too slow**.

[5] Caballero E, Gupta K, Rish I, Krueger D. Broken neural scaling laws. arXiv preprint arXiv:2210.14891. 2022 Oct 26.

Fixed Model Size with Varying FLOPs - Eval Loss

Approach: to use the lower envelope to fit a power law between N and C

- Linear regression in **log** scale:

$$\log(N) = 0.469 * \log(C) - 1.314$$

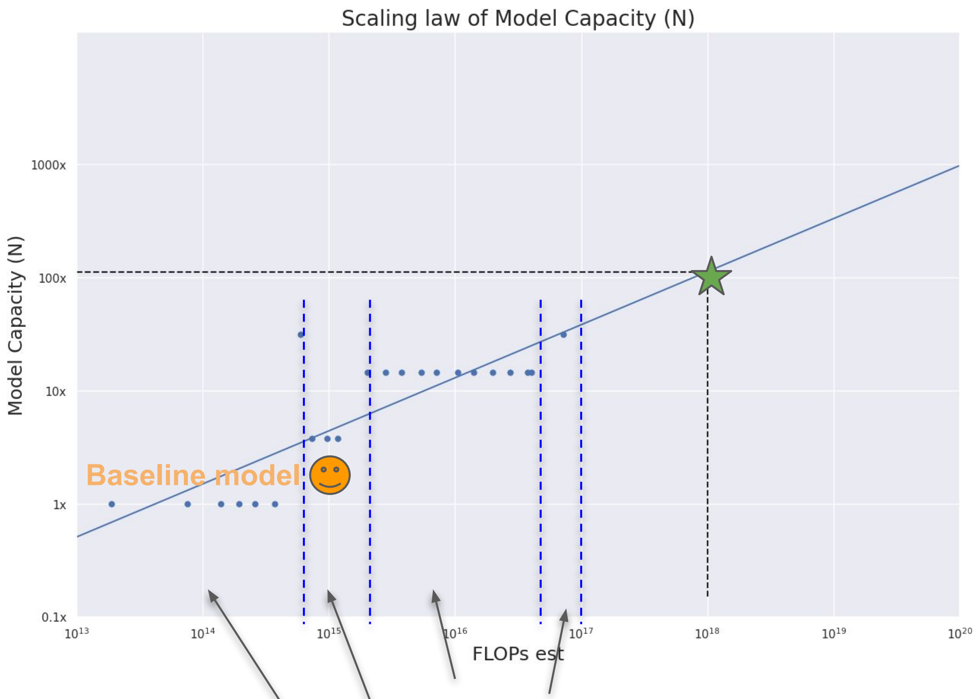
- In **linear** scale:

$$N = 0.269 * C^{0.469}$$

Then: estimate best model capacity $N*$ at a **given cost budget** $C*$, e.g.:

- $C^* = 1e18 \longrightarrow N^* \approx 112X$

Note: compare with train loss: $N^* \approx 72X$



Different models

Fixed Model Size with Varying FLOPs - Eval Loss

Approach: to use the lower envelope to fit a power law between S and C

- Linear regression in **log** scale:

$$\log(S) = 0.531 * \log(C) - 6.311$$

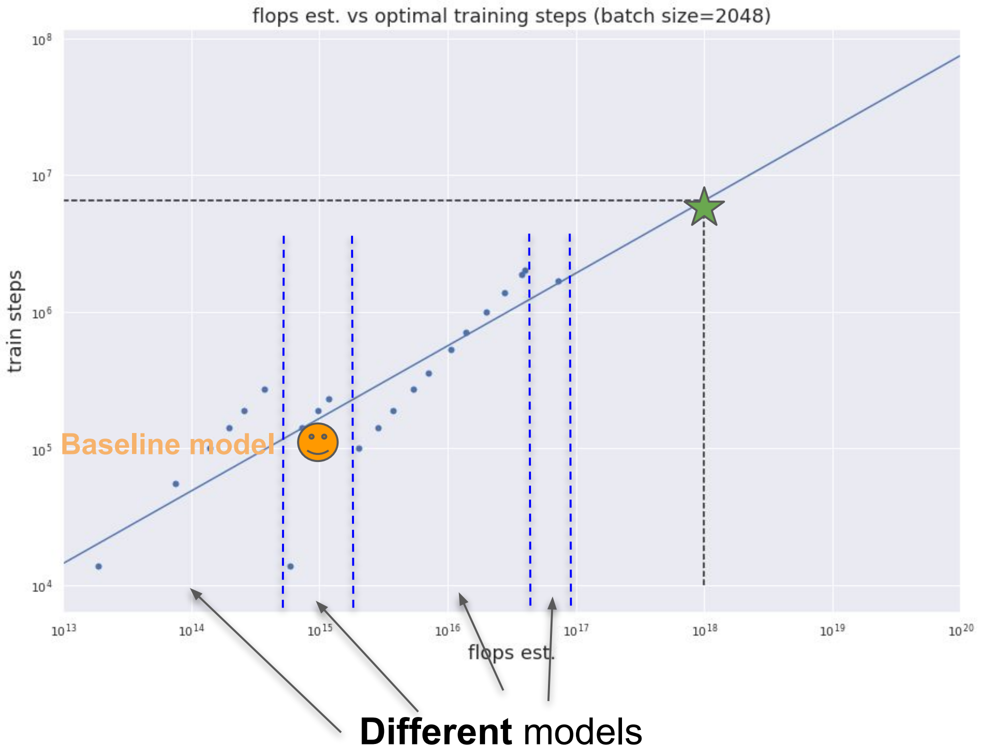
- In **linear** scale:

$$S = 1.8\text{e}{-3} * C^{0.531}$$

Then: estimate best training steps $S*$ at a **given cost budget** $C*$, e.g.:

- $C^* = 1\text{e}18 \longrightarrow S^* \approx 6.5M$

Note: compare with train loss: $S^* \approx 10.06M$

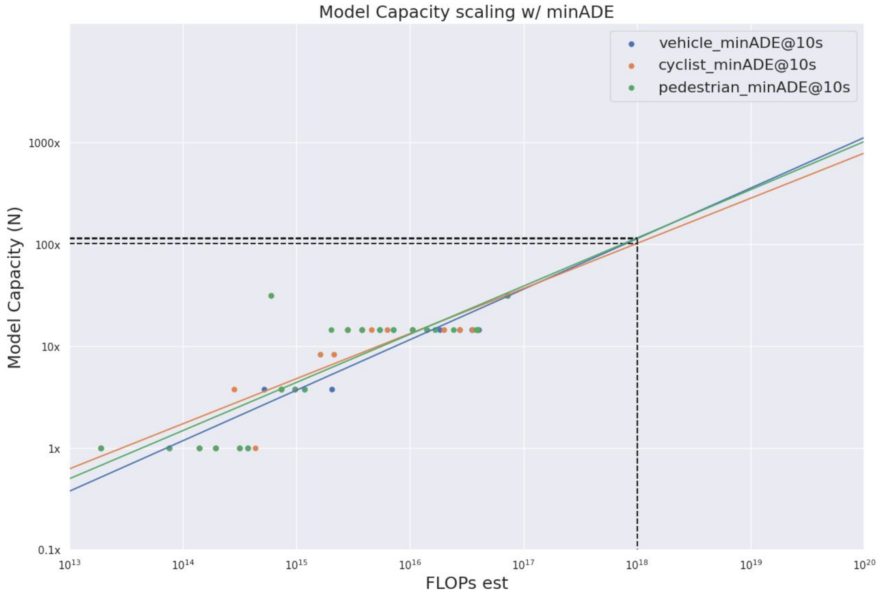


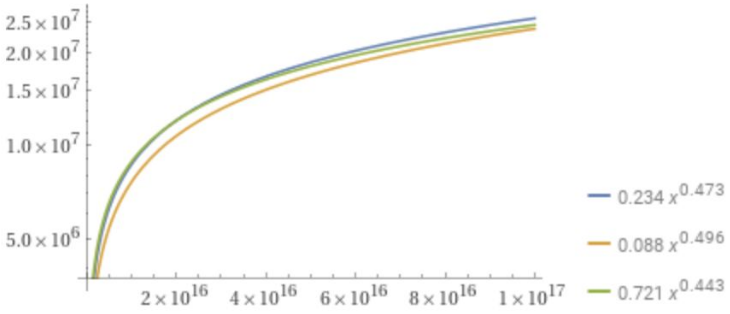
Can similar **scaling laws** be applied to **downstream** metrics?

Optimal Model Scaling Laws with Agent Trajectory Prediction

Metric: Min Average Displacement Error (minADE) @10 seconds.

Track types: Vehicle, Pedestrian, Cyclist.





Vehicle	Cyclist	Pedestrian
$N = 0.234 * C^{0.473}$	$N = 0.088 * C^{0.496}$	$N = 0.721 * C^{0.443}$

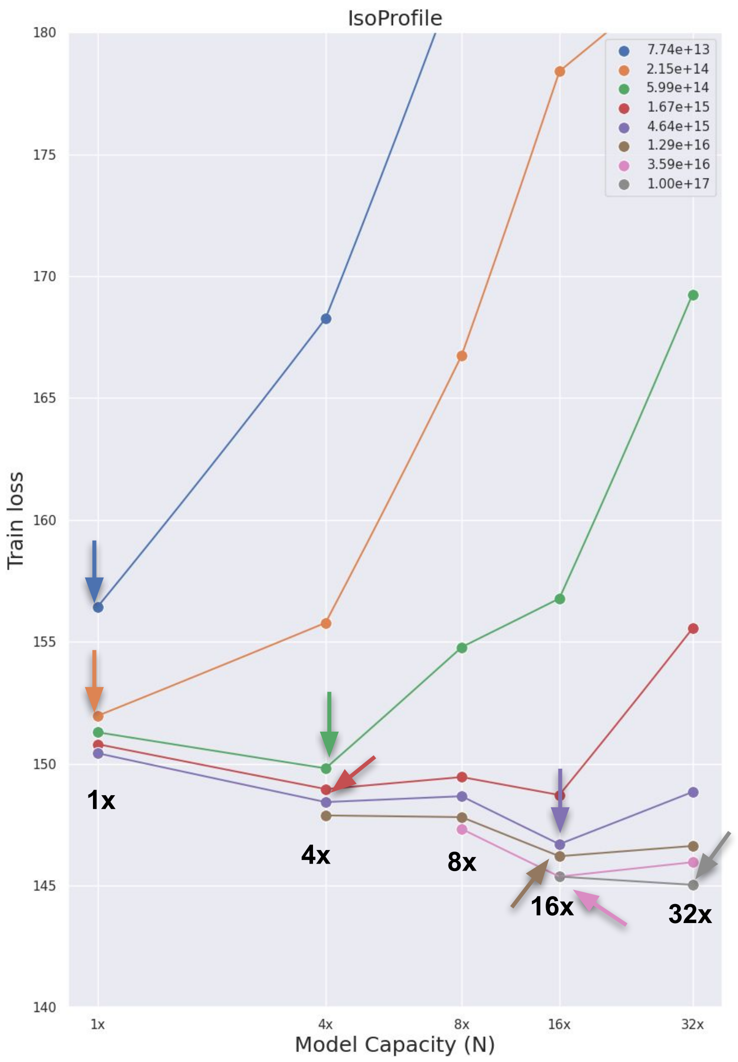
Scaling Laws: Behavior

3.3

IsoFLOP Profiles - Train Loss

Method 1: Varying FLOPs along x-axis
Method 2: Fix FLOPs to find optimal model
<u>Algorithm</u>:

1. Define multiple C levels (e.g. 8)
2. For each C level, plot the profile curves
3. Find the model capacity N that has lowest training loss L (shown by the arrows)
4. Fit the power law between optimal N and C



IsoFLOP Profiles - Train Loss

Approach: to use the lower envelope to fit a power law between N and C

- Linear regression in **log** scale:

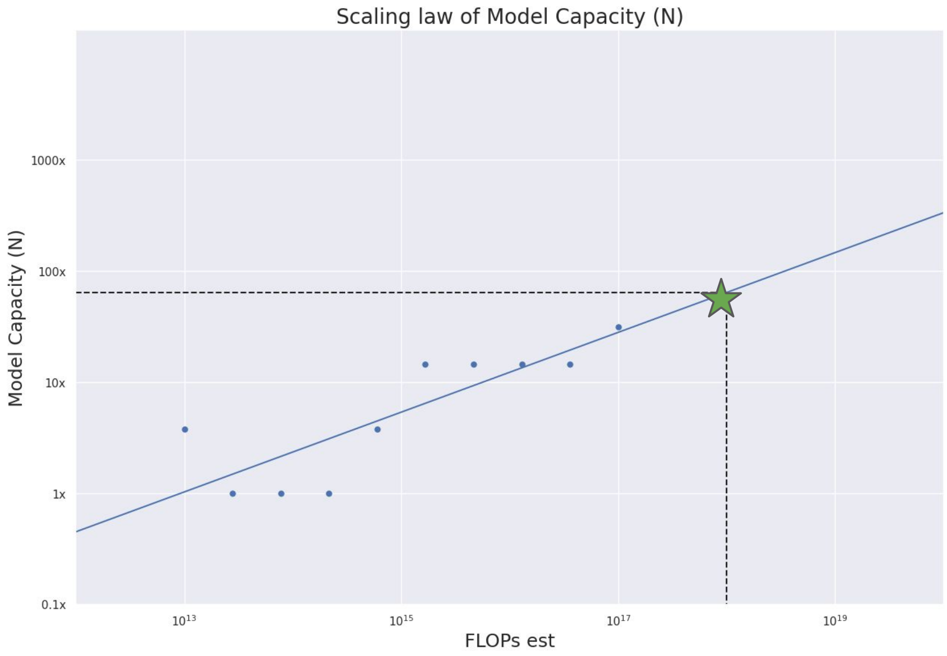
$$\log(N) = 0.422 * \log(C) + 0.259$$

- In **linear** scale:

$$N = 1.296 * C^{0.422}$$

Then: estimate best model capacity N* at a **given cost budget** C*, e.g.:

- $C^* = 1e18 \longrightarrow N^* \approx 76X$



IsoFLOP Profiles - Train Loss

Approach: to use the lower envelope to fit a power law between S and C

- Linear regression in **log** scale:

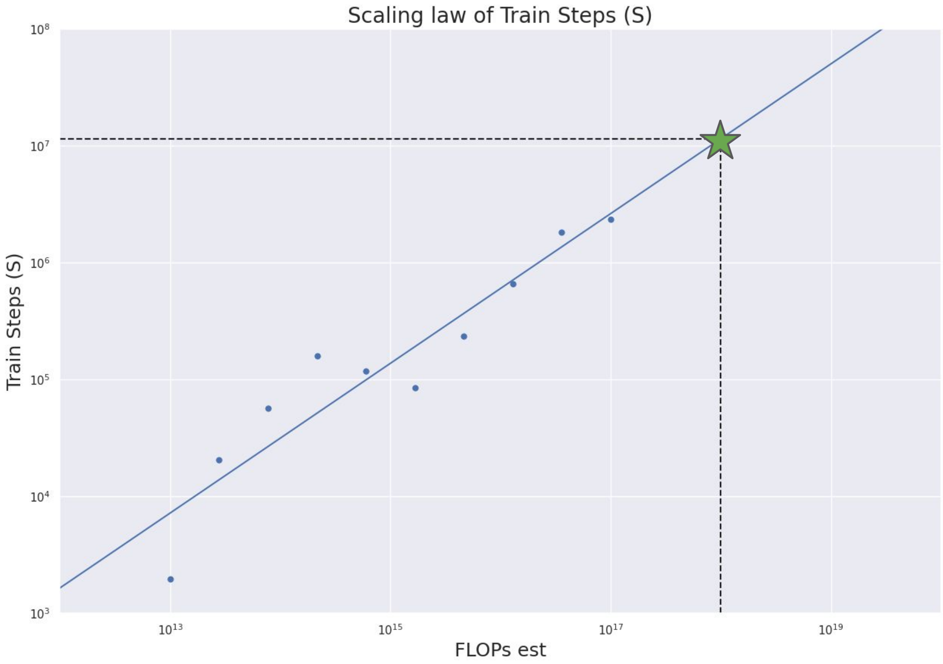
$$\log(S) = 0.578 * \log(C) - 7.883$$

- In **linear** scale:

$$S = 3.8\text{e}{-4} * C^{0.578}$$

Then: estimate best training steps $S*$ at a **given cost budget** $C*$, e.g.:

- $C^* = 1e18 \longrightarrow S^* \approx 9.69M$



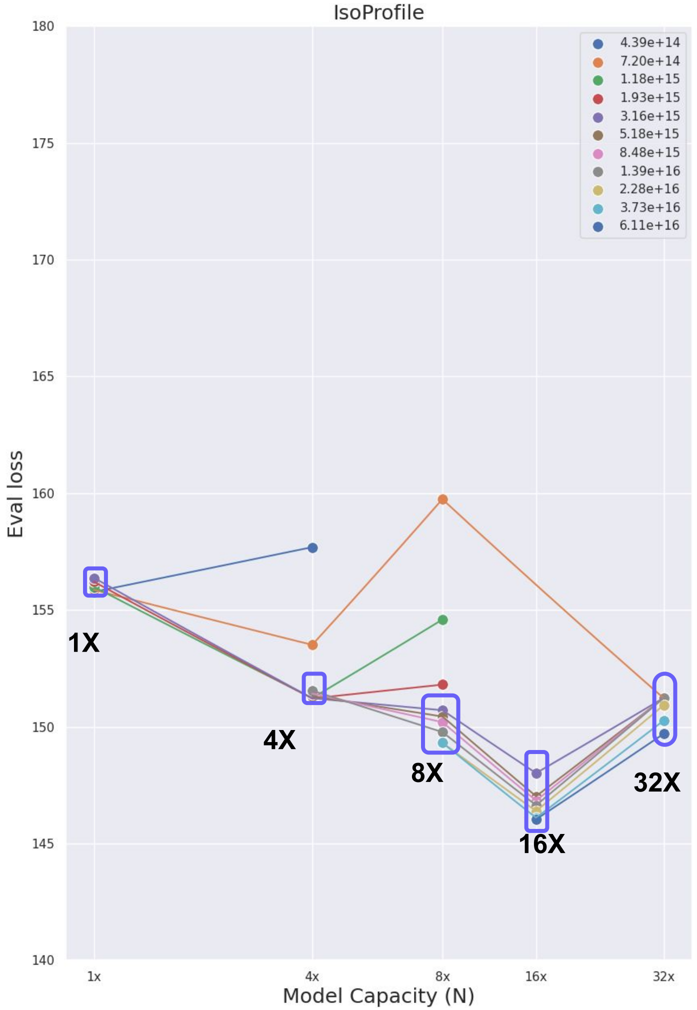
IsoFLOP Profiles - Eval Loss

Similarly we can get the IsoFLOP Profiles for **eval** loss.

Note that some points are missed because we didn't have a evaluation run of that model capacity at that FLOPs level.

Some **findings**:

- The **32x** model performed **better** on **low FLOPs** side, because it learned fast at the beginning of training
- It in general **takes longer** to **train larger models**
- The **32x** model is likely **not trained to its best**



IsoFLOP Profiles - Eval Loss

Approach: to use the lower envelope to fit a power law between N and C

- Linear regression in **log** scale:

$$\log(N) = 0.478 * \log(C) - 1.768$$

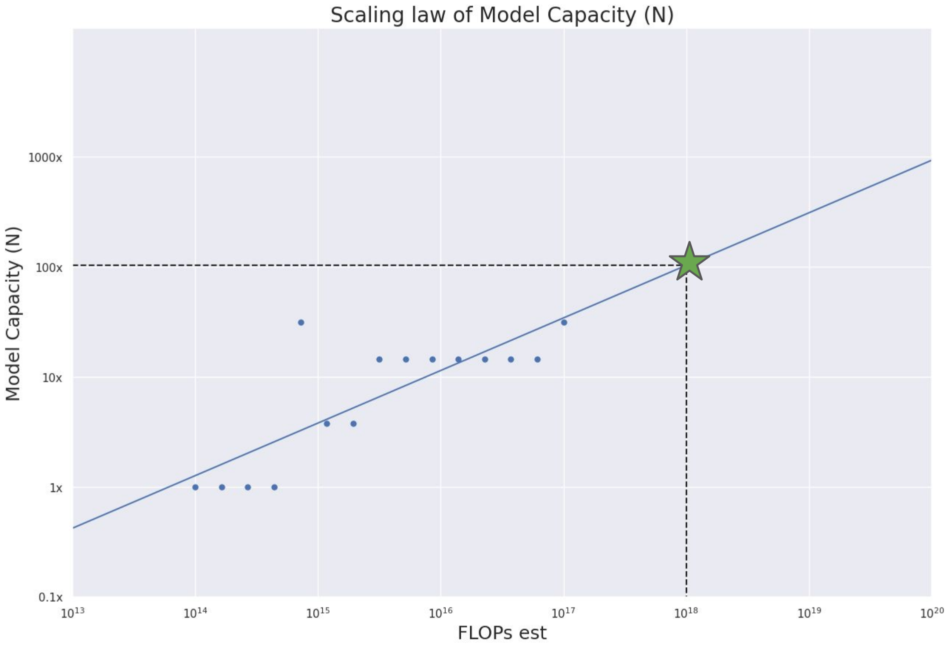
- In **linear** scale:

$$N = 0.171 * C^{0.478}$$

Then: estimate best model capacity $N*$ at a **given cost budget** $C*$, e.g.:

- $C^* = 1e18 \longrightarrow N^* \approx 103X$

Note: compare with train loss: $N^* \approx 76X$



IsoFLOP Profiles - Eval Loss

Approach: to use the lower envelope to fit a power law between S and C

- Linear regression in **log** scale:

$$\log(S) = 0.522 * \log(C) - 5.857$$

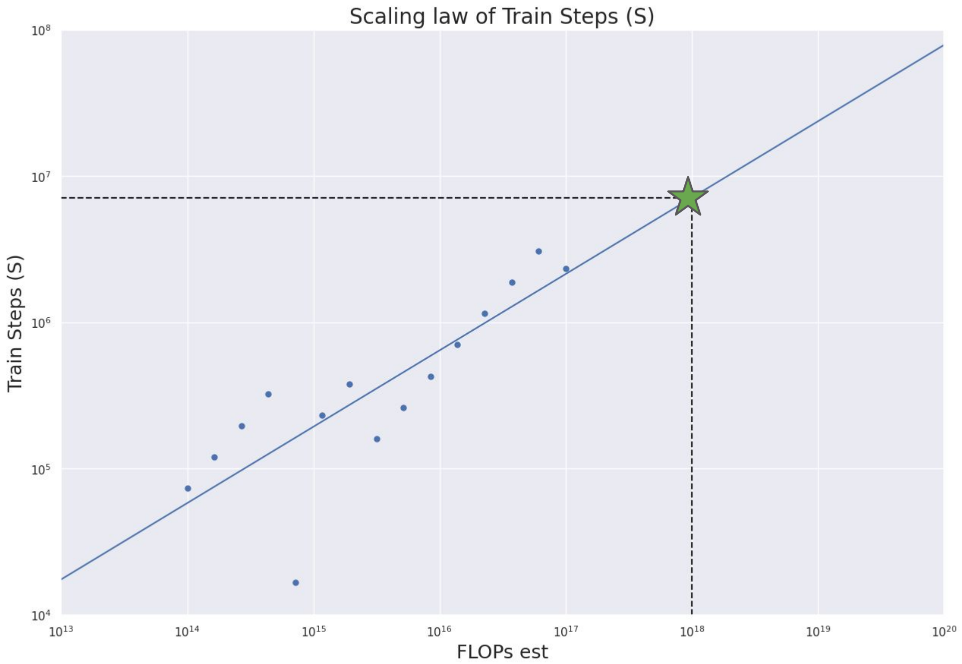
- In **linear** scale:

$$S = 2.9\text{e}{-3} * C^{0.522}$$

Then: estimate best training steps $S*$ at a **given cost budget** $C*$, e.g.:

- $C^* = 1e18 \longrightarrow S^* \approx 7.09M$

Note: compare with train loss: $S^* \approx 9.69M$



Method 1 vs Method 2

Both: find the best model at different C and yield **similar** results

- **Method 2** directly shows the profiles at specified **cost level**
- **Method 2** has **better correspondence** between **train** and **eval** predictions

	Capacity	Steps	Estimated Capacity @ *C=1e18*	Estimated steps @ *C=1e18*
Method 1, train loss	$N = 3.501 * C^{0.397}$	$S = 1.4\text{e}{-4} * C^{0.603}$	72X	10.06M
Method 2, train loss	$N = 0.269 * C^{0.469}$	$S = 1.8\text{e}{-3} * C^{0.531}$	76X	9.69M
Method 2, eval loss	$N = 1.296 * C^{0.422}$	$S = 3.8\text{e}{-4} * C^{0.578}$	103X	7.09M
Method 1, eval loss	$N = 0.171 * C^{0.478}$	$S = 2.9\text{e}{-3} * C^{0.522}$	112X	6.5M

Note: for $C^* = 1e18$

- Best model **capacity** is **~72X – 112X**, and the best training **steps** is **~6.5-10 Million**
- As a comparison, the **baseline configuration** is a **4X** encoder trained for **~270K** steps at a level of C ≈1.4e15

Scaling Laws: Behavior

3.4

Behavior Eval Tasks

The **BE** is **shared** by many behavior **tasks**.

Prediction task:

- ~175km (~15 hrs) total driving
- Major metrics: **minADE**, minFDE, Miss Rate, Overlap Rate, Long/Lat Errors…

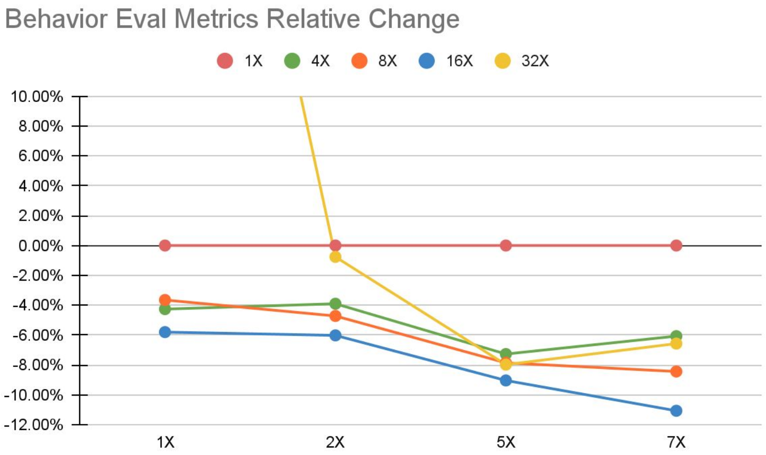
Planning task:

- ~125km (~10 hrs) total driving
- Major metrics: Rigorously defined “**Passes**”

Experiment details:

1. For each experiment, we export the **final** checkpoint
2. Initialize the Prediction and Planning models from it
3. Freeze the BE and finetune model decoders

Behavior Eval Tasks

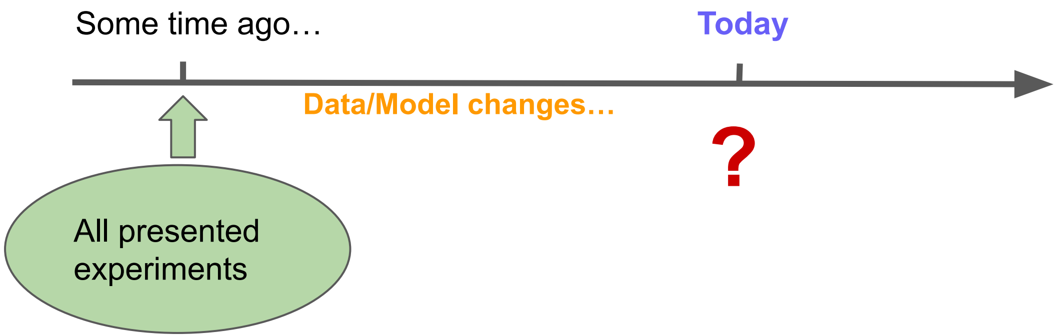


Training Dataset size

Observations:

- **Smaller** models (<8x) **scale** well when **dataset is small** (<5X)
  - Then start to overfit
- **16x** model **scales** almost **linearly** with dataset size
- **32x** model significantly **overfits** when dataset is **small**, it also **didn't fully converge** within limited training epochs when dataset is larger

Behavior Eval With the Model Change



Baseline model:

- ~1.4e15 estimated FLOPs, trained for ~2 days

Scaled model:

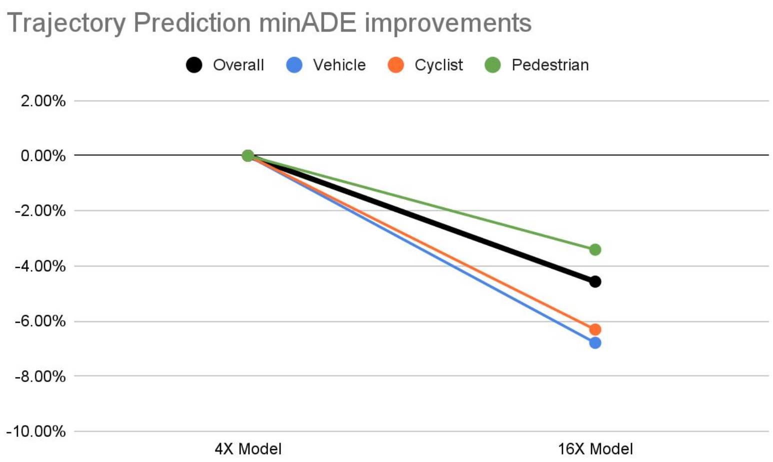
- ~10X FLOPs*, trained for ~4 days, use scaling laws to select the best model:

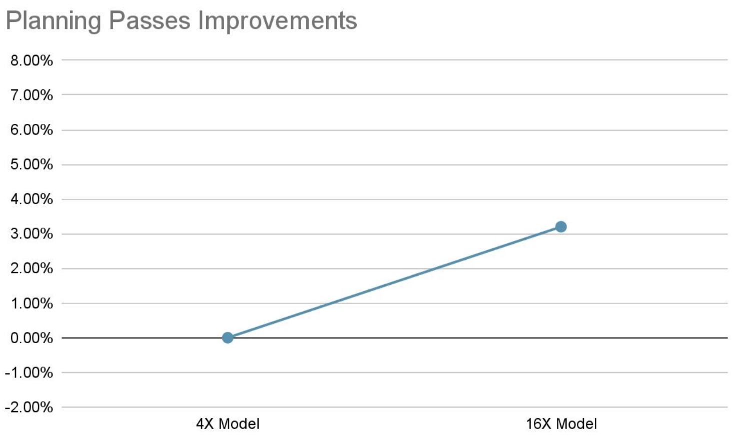
Target flops	Method 1 w/ train loss	Method 1 w/ eval loss	Method 2 w/ train loss	Method 2 w/ eval loss
1e16	11.8X	12.4X	11.0X	11.9X
1.4e16	13.5X	14.5X	12.7X	13.9X
2e16	15.5X	17.2X	14.7M	16.3X

- Let's select the nearest candidate - **16X** BE model

*10X FLOPs is some design choice that scales model with reasonable cost.

Behavior Eval With the Model Change





* w/o optimization, the onboard latency ~2X with the 16X model.

Key **observations**:

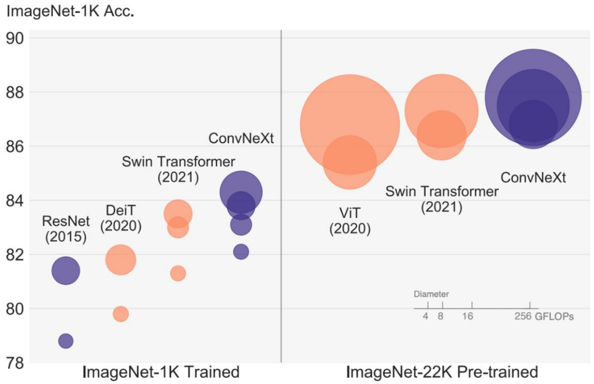
- Scaling law can be transferred to different behavior model architectures
- **Advanced model** (16x vs 4x) **benefits more** (from 2% to 5%) from scaling

Perception Model Preliminaries & Experiment Details

4

Scaling up Convolutional Neural Network

Perception task in Autonomous Driving: **2D object detection**
Experiment: ConvNeXt as backbone in an object detection module



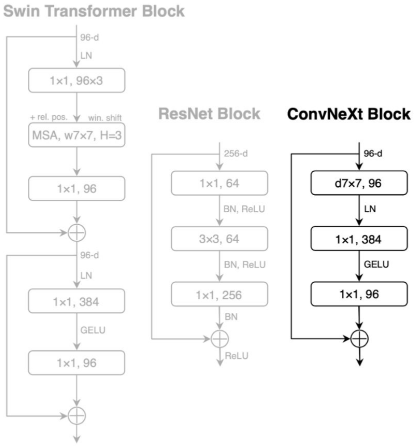
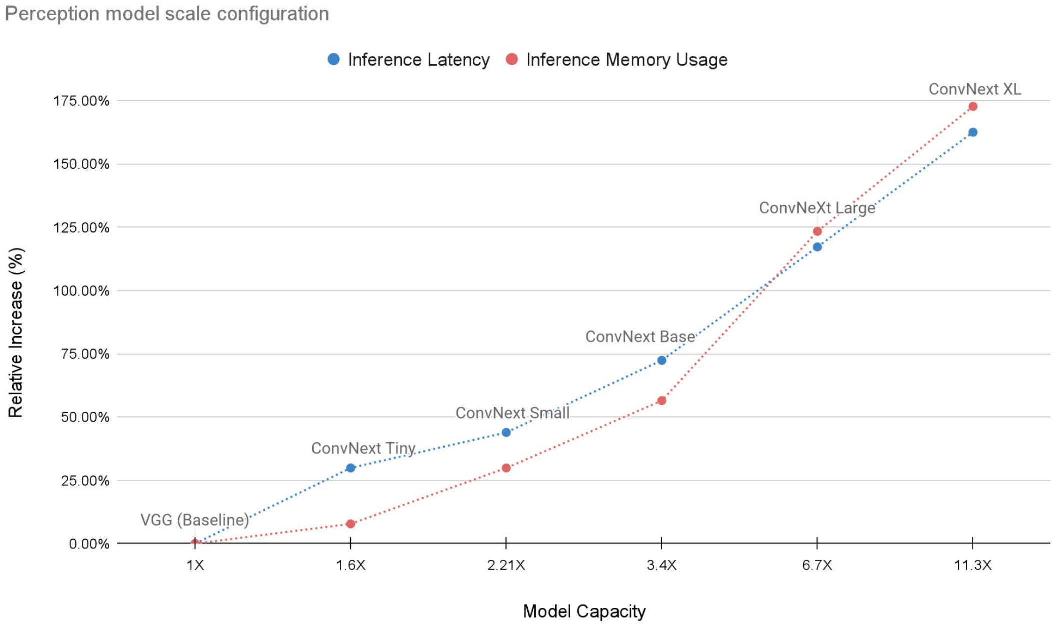


Figure cited from the ConvNeXt paper[6]

[6] Liu Z, Mao H, Wu CY, Feichtenhofer C, Darrell T, Xie S. A convnet for the 2020s. InProceedings of the IEEE/CVF conference on computer vision and pattern recognition 2022 (pp. 11976-11986).

Experiments Settings



Note: Memory usage is mostly aligned with latency

Scaling Laws: Perception

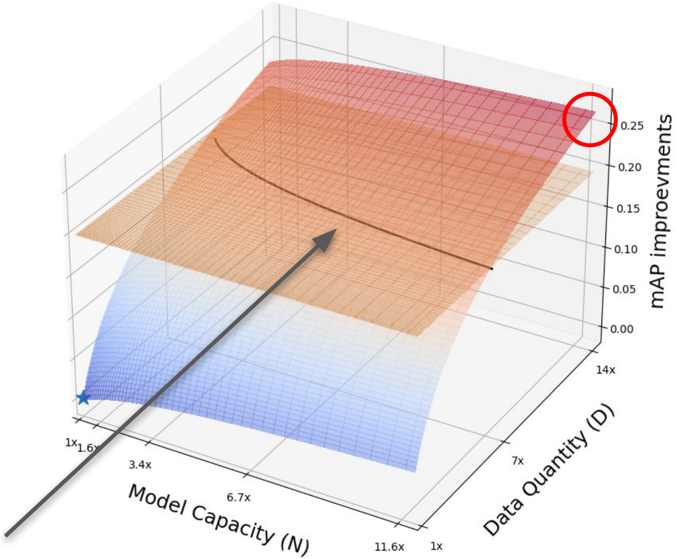
5

Model Performance Scaling Law

Collected the *(L, N, D)* triplets

Fitted the model performance scaling function $L(N, D)$

mAP is used as the model performance metric

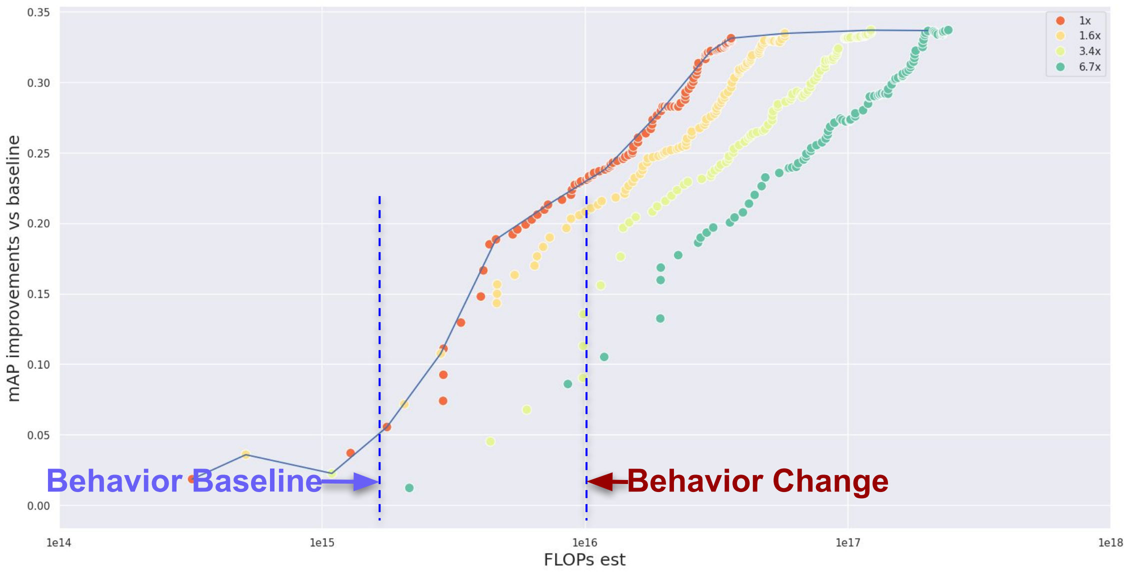


Conclusion:

- Better performance: with the larger N and D
- Fixing L, one can find the needed (N,D) profile

Exactly the **same conclusions** as for Behavior!

Optimal Model Scaling Law



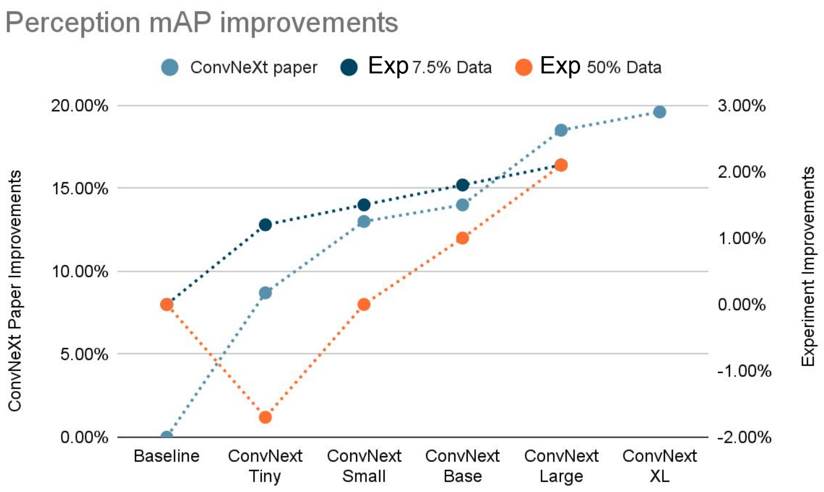
The upper envelope of performance:

- **1X** model: **superior** performance in a **low FLOPs** regime
- Further experiment is needed (**massive cost!!!**) in order to benefit from larger model

Similar conclusions as for Behavior

Data Scale: Replicating ConvNext Paper

- Model **scaling works** – but requires **large data scale**
  - At **7.5%**: improvement is **smaller** and **plateaued**
  - At **50%**: improvement is **similar**, but it was costly to complete training for all of the models
- Latency, model size, and **cost increase extremely fast**



Conclusion

6

Next Steps

01

Better understand width vs depth difference, LR impact

02

Explore double descent and broken neural scaling laws

03

Verify the predicting power of Scaling Laws in the >1e20 computing budget regime

Limitations

01

Scaling laws are biased towards the **extremum** models: can **lose** similar “good” but an **order less complex** model

02

Scaling laws **may not generalize** to different evaluation datasets and metrics, especially with data scales up[7]

03

Improvement: **additional**, but cost: **multiplicative**

Need really **reliable infrastructure** and **fast experimentation pipeline**

[7] Diaz F, Madaio M. Scaling Laws Do Not Scale. arXiv preprint arXiv:2307.03201. 2023 Jul 5.

Conclusions

01

Scaling laws are **applicable** in Autonomous Driving

02

Different extrapolation methods have their own pros and cons, but mostly **coincide in the order**

03

Even **w/o** significant **model chang**e, we can do **better** with longer training / more capacity[8]

[8] E.g., by **8x more cost**

- ~**7% less minADE** (Prediction)
- ~**3% more Pass Rate** (Planner)

Main Conclusion

Scaling

A very **trivial way** of adding a handful of percents (like ensembling in kaggle competitions) that can be even **forecasted** in advance.

And, it is **not a game changer**: slight marginal improvements by the extremely high cost / orders more data (having 1K times bigger models/GPUs/data would not give you the clear preference over other companies).

Still need **better ideas** :)

Visit Our Poster Today!

https://ml4ad.github.io/#papers

Multi-Constraint Safe RL with Objective Suppression for Safety-Critical Applications

Zihan Zhou
University of Toronto
Vector Institute
footoredo@gmail.com

Jonathan Booher
Nuro Inc.
jbooher@nuro.ai

Wei Liu
Nuro Inc.
w@nuro.ai

Aleksandr Petiushko
Nuro Inc.
apetiushko@nuro.ai

Animesh Garg
University of Toronto
Vector Institute
Nvidia
garg@cs.toronto.edu

Q&A


Thank
You.
nuro