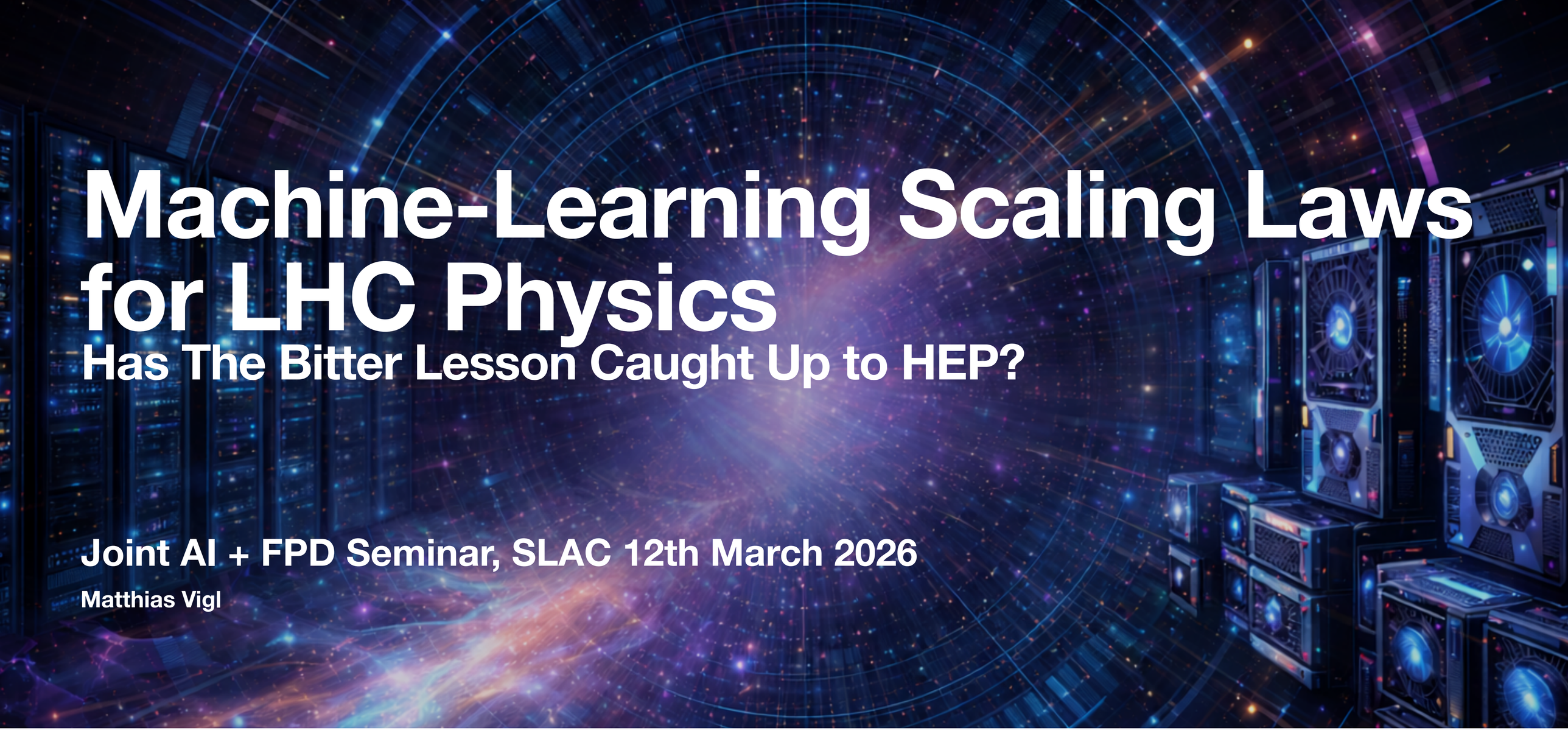


Machine-Learning Scaling Laws for LHC Physics

Has The Bitter Lesson Caught Up to HEP?

Joint AI + FPD Seminar, SLAC 12th March 2026

Matthias Vigl

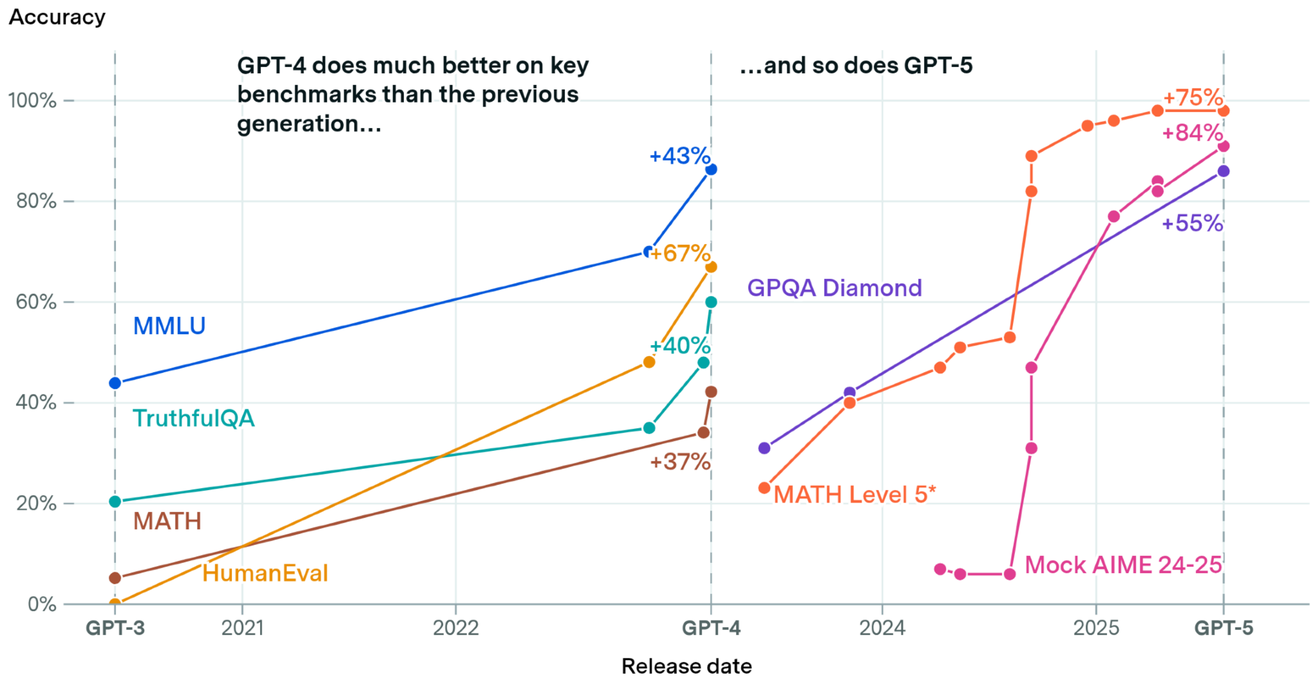
Technische Universität München

ORIGINS Excellence Cluster

ATLAS EXPERIMENT

MAX-PLANCK-INSTITUT FÜR PHYSIK

Industry models keep getting better (and bigger)



*MATH Level 5 is the most difficult subset of the original MATH benchmark
Figure only includes OpenAI models

Rate of improvement is quite impressive

Compute* is all you need

The Bitter Lesson

Rich Sutton

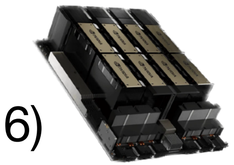
March 13, 2019

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. The ultimate reason for this is Moore's law, or rather its

O(1T) parameters

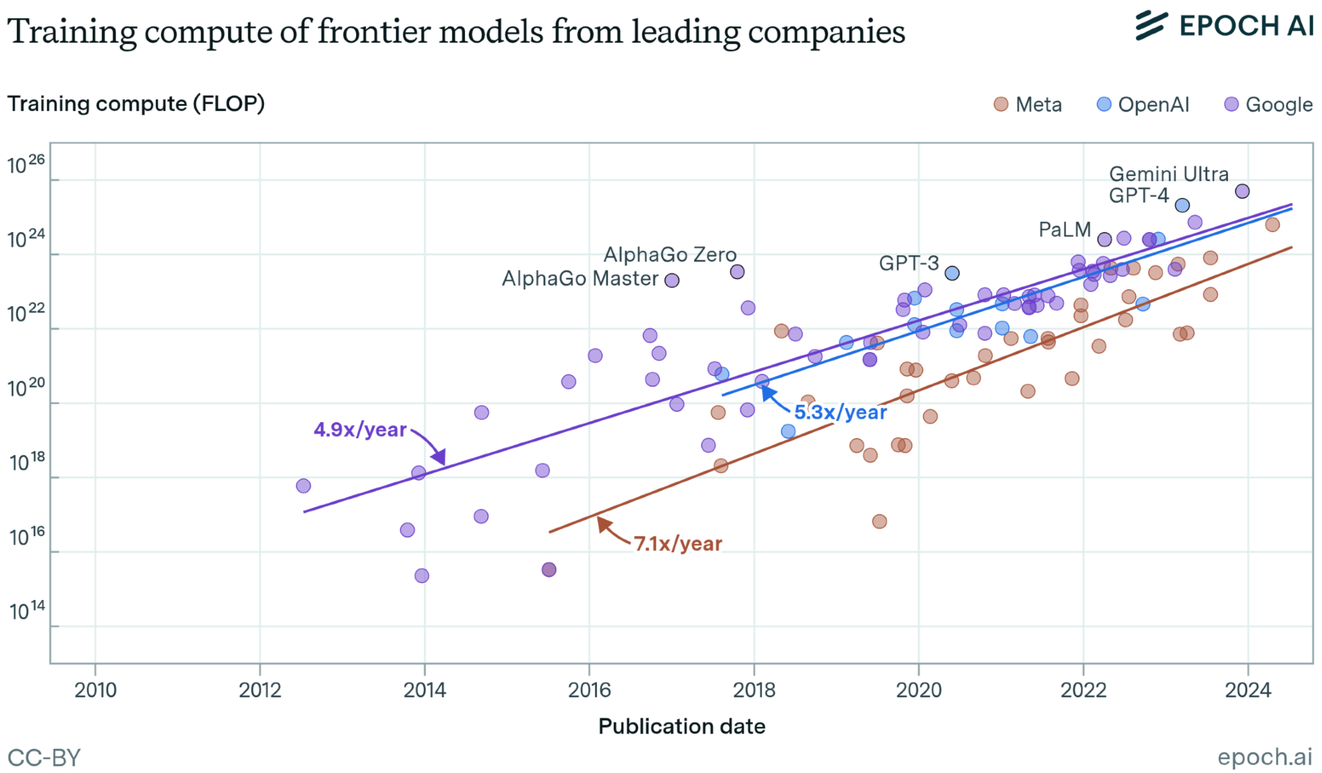
1 NVIDIA H200:

$\sim 10^{20}$ FLOPs/day (BF16)



***optimal use of compute**

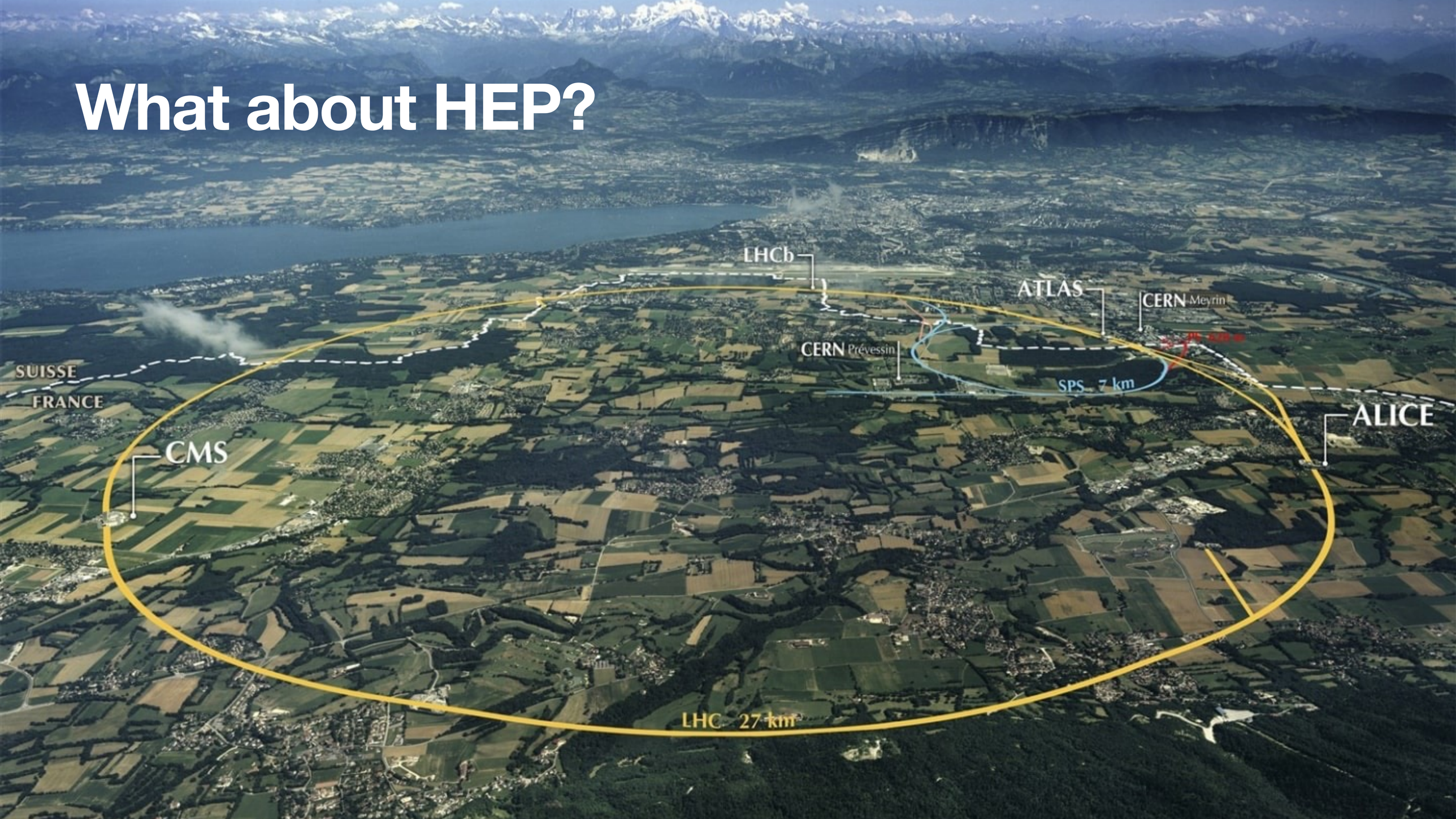
Delicate balance between dataset and model size



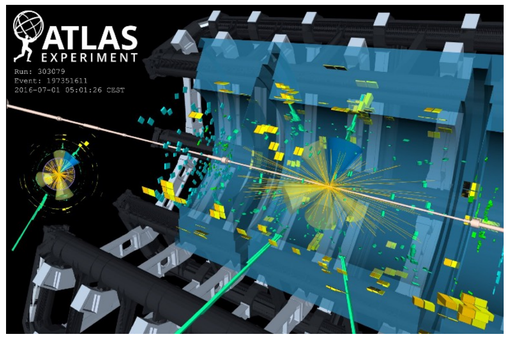
arXiv:2001.08361

Google DeepMind arXiv:2203.15556

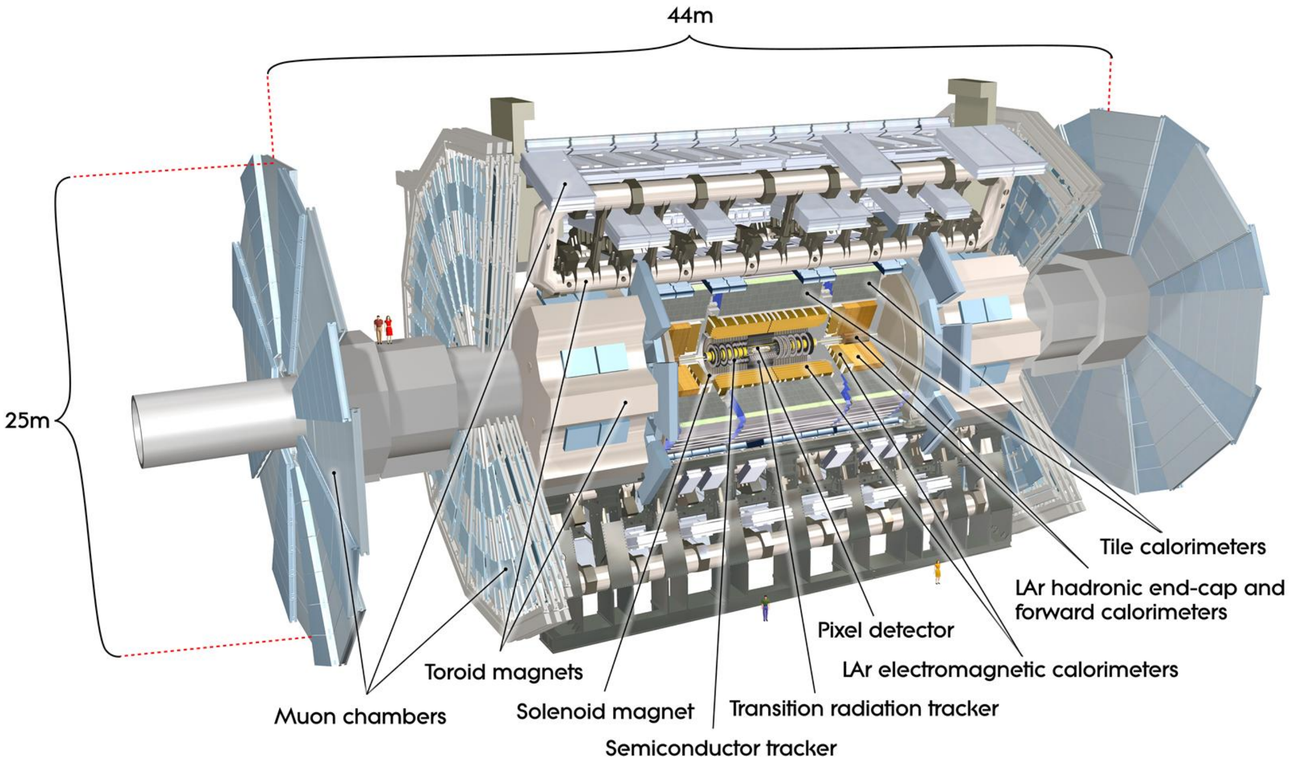
What about HEP?



The ATLAS detector

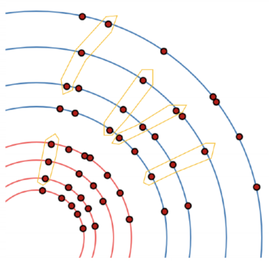


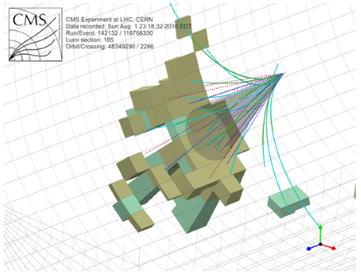
100 million read out channels

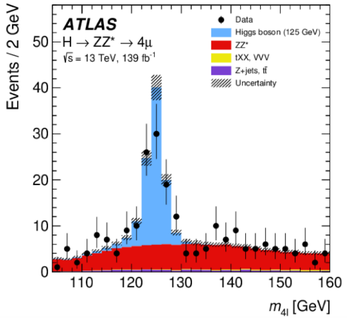


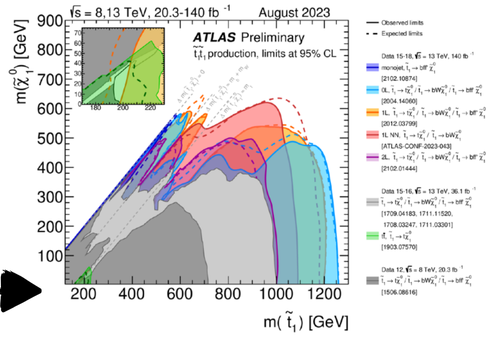
Making sense of the data

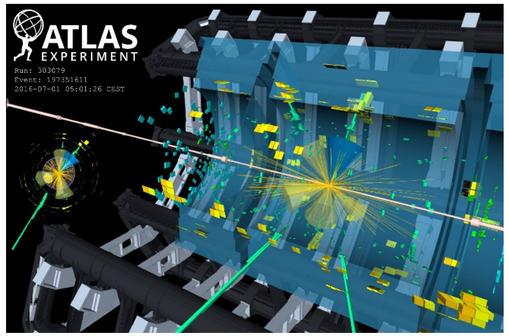
$\mathbb{R}^1$







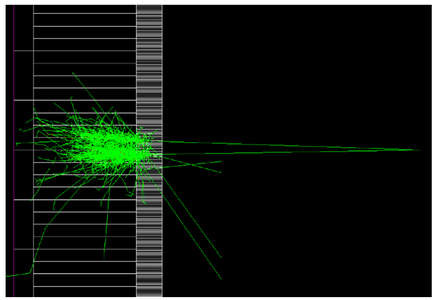


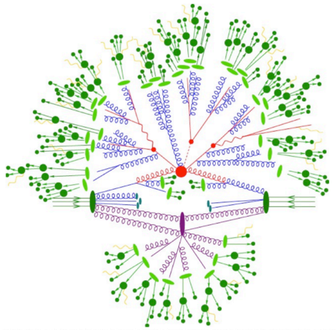


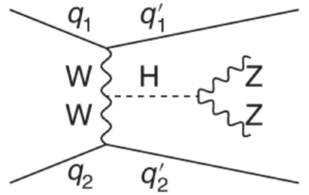
Tracking/Calo clusters **Particle/Jet reco** **Event observables "Analysis"**

Detector simulation **Parton evolution & Hadronization** **Event Generation**

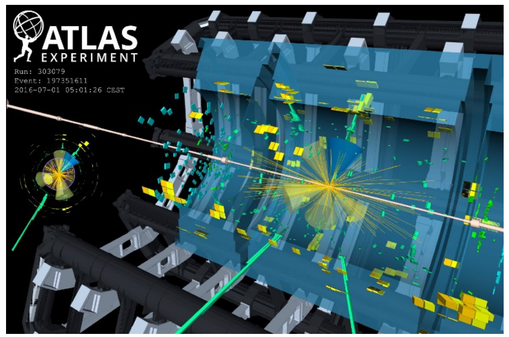
$$\mathcal{L} = -\frac{1}{4} F_{\mu\nu} F^{\mu\nu} + i\bar{\Psi}\not{D}\Psi + h.c. + \bar{\Psi}_i y_{ij} \Psi_j \phi + h.c. + |D_\mu \phi|^2 - V(\phi)$$

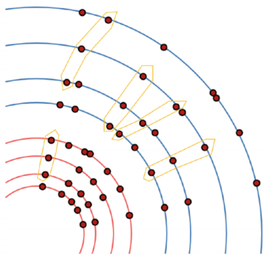


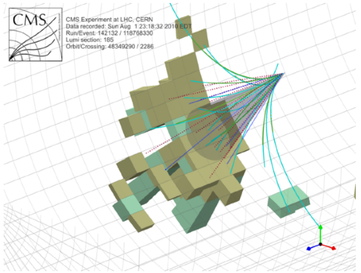




The role of reconstruction

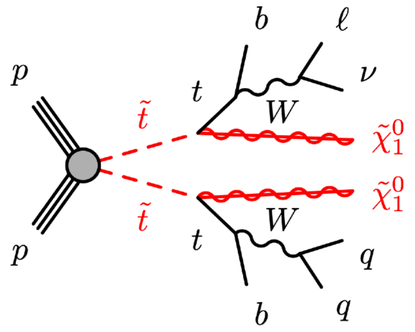


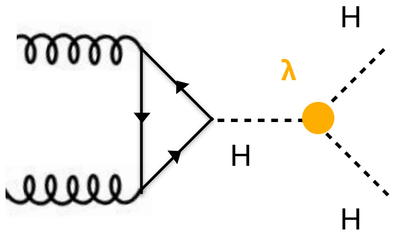




Tracking/Calo clusters Particle/Jet reco

Reco Event (particles, jets, MET)





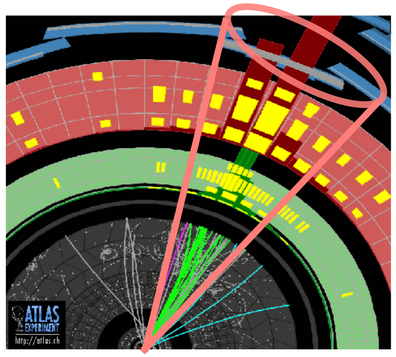
Foundation that serves all downstream physics analysis

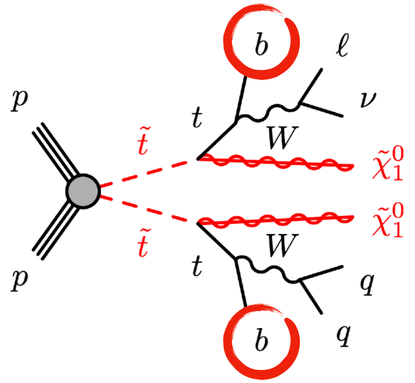
Low level features → Reconstructed Event

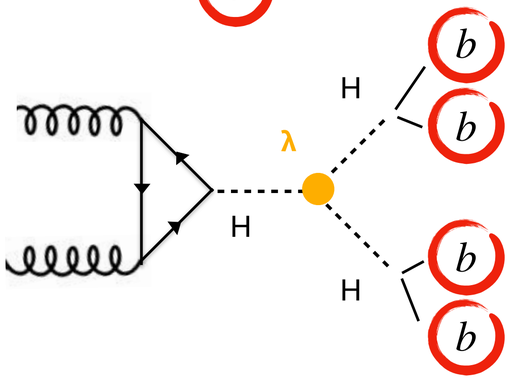
Lots of low level ML

Jets

In the detector, quarks & gluons create jets



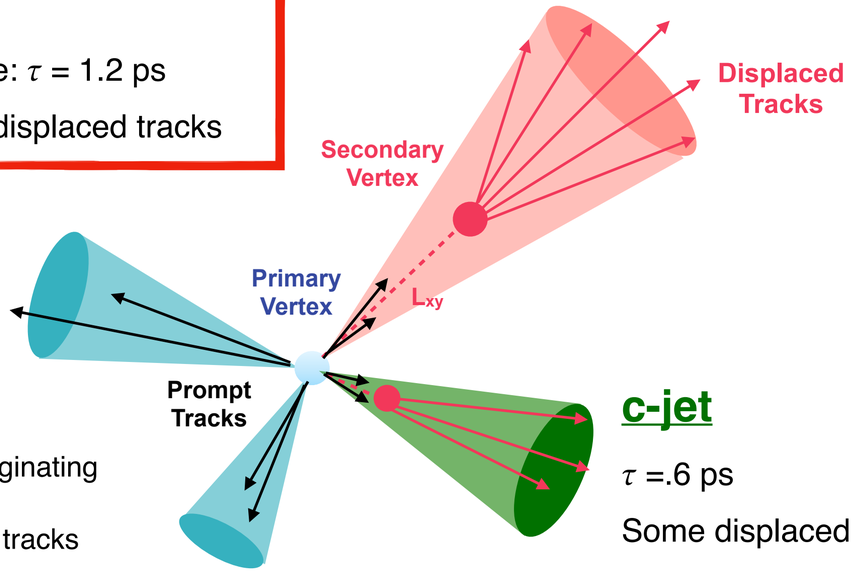




b-jet

"Long" lifetime: τ = 1.2 ps

Many (≈ 5) displaced tracks



light jet

u,d,s g

Most tracks originating from the PV

Few displaced tracks

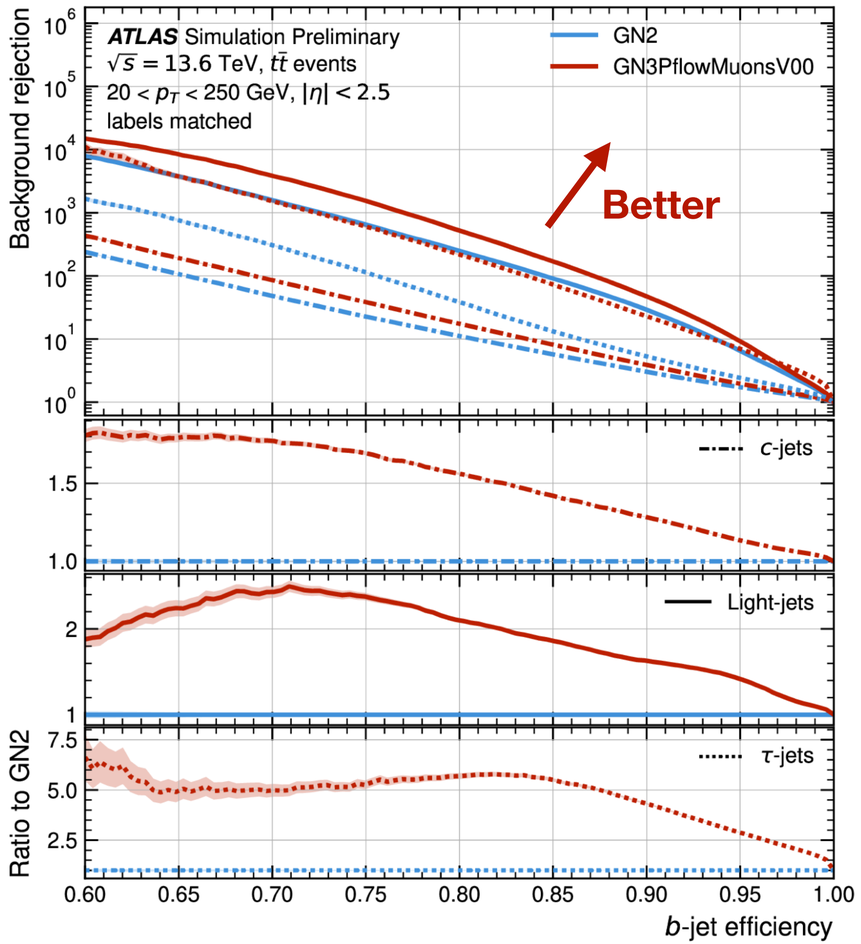
c-jet

τ =.6 ps

Some displaced tracks

Jet Flavor "Tagging" is crucial for many analyses

Jet Flavor Tagging in ATLAS

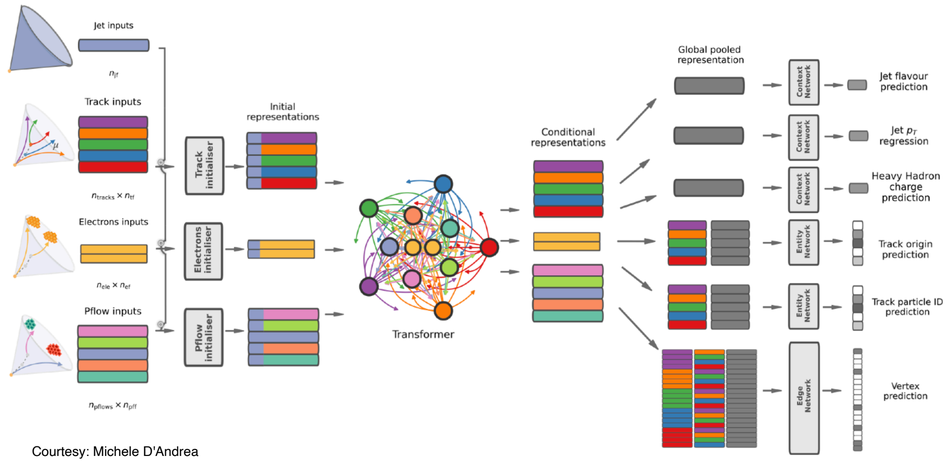


GN3 public plots

ATLAS TDR 1999

[T]he $H \to bb$ decay mode is dominant … [but] the extraction of a signal from $H \to bb$ decays in the WH channel will be very difficult at the LHC, even under the most optimistic assumptions for the b-tagging performance and calibration of the shape and magnitude of the various background sources from the data itself.

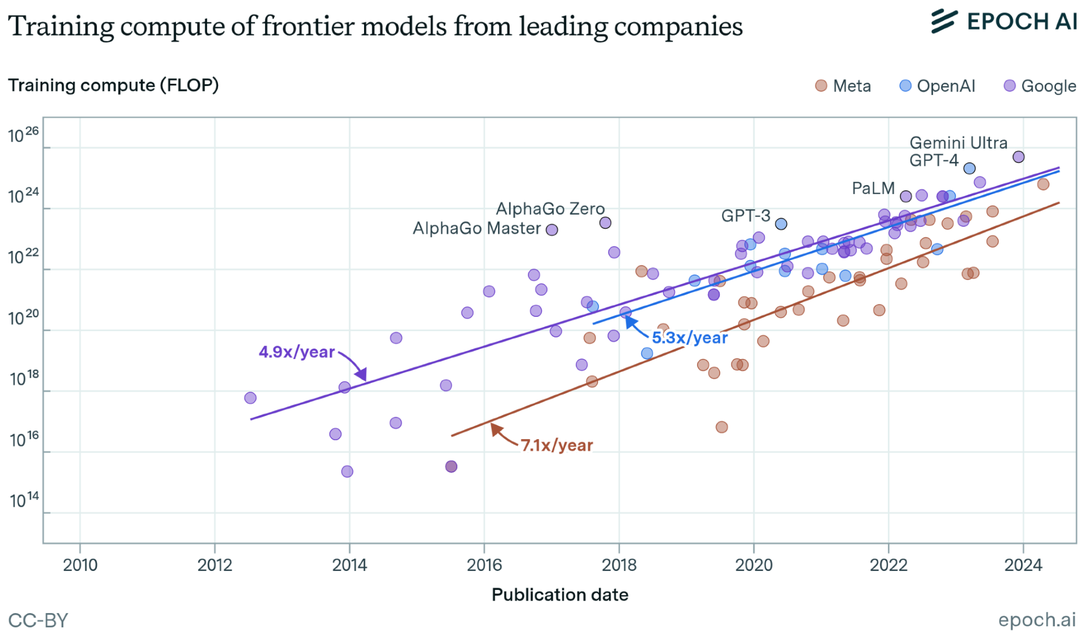
Long history of success with ML in ATLAS flavor tagging



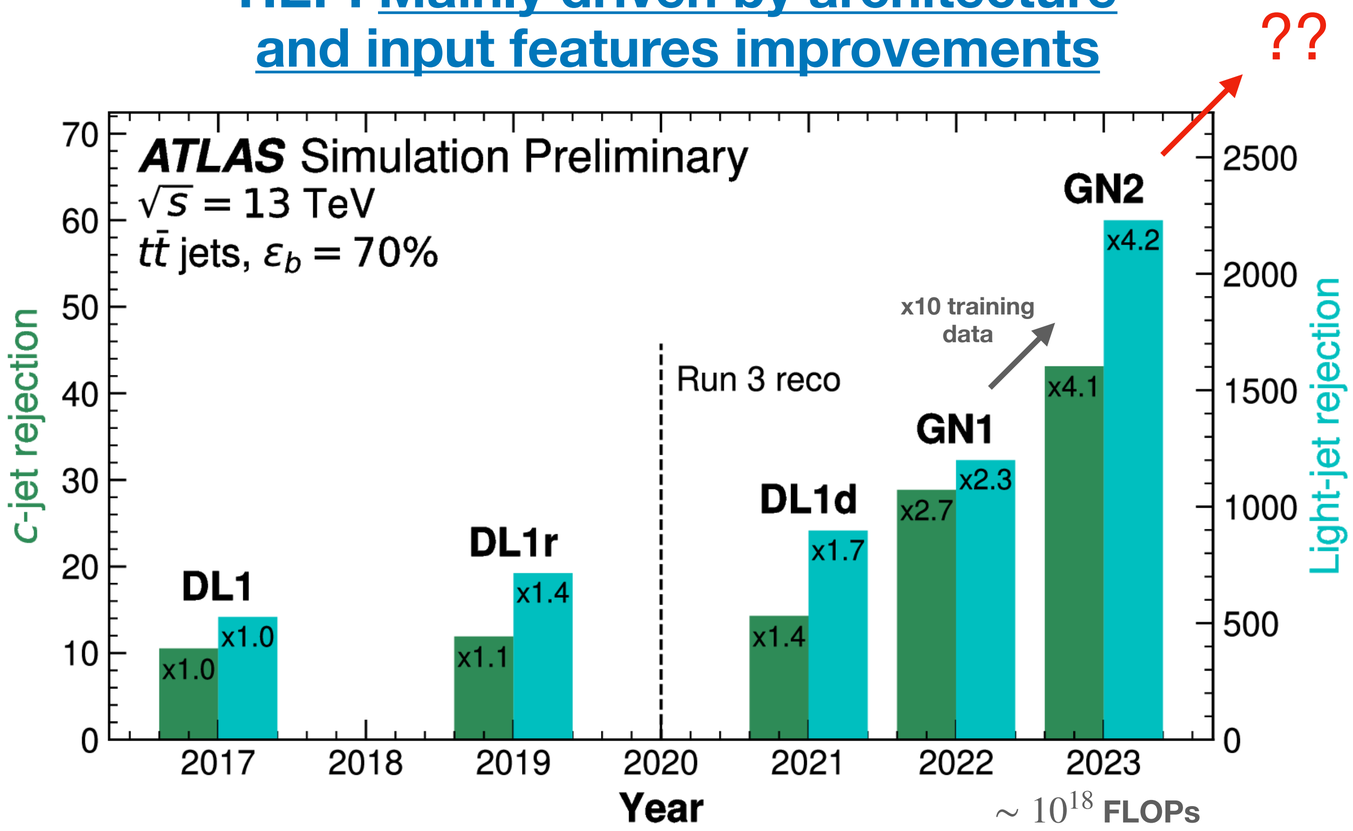
Courtesy: Michele D'Andrea

What's driving ML performance?

Industry: Mainly driven by scale

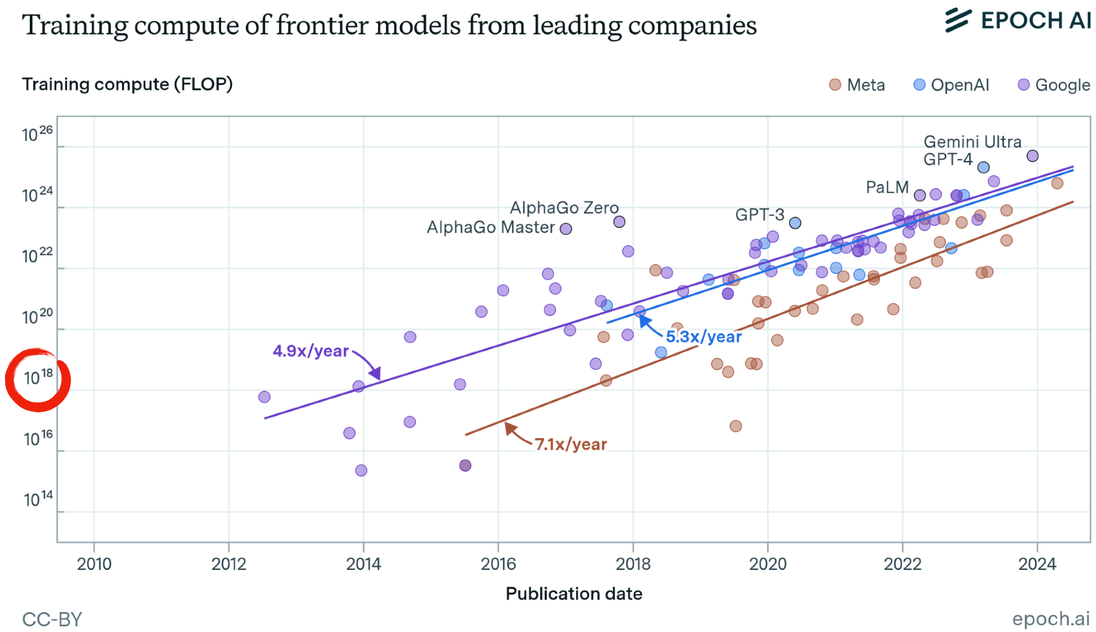


HEP: Mainly driven by architecture and input features improvements

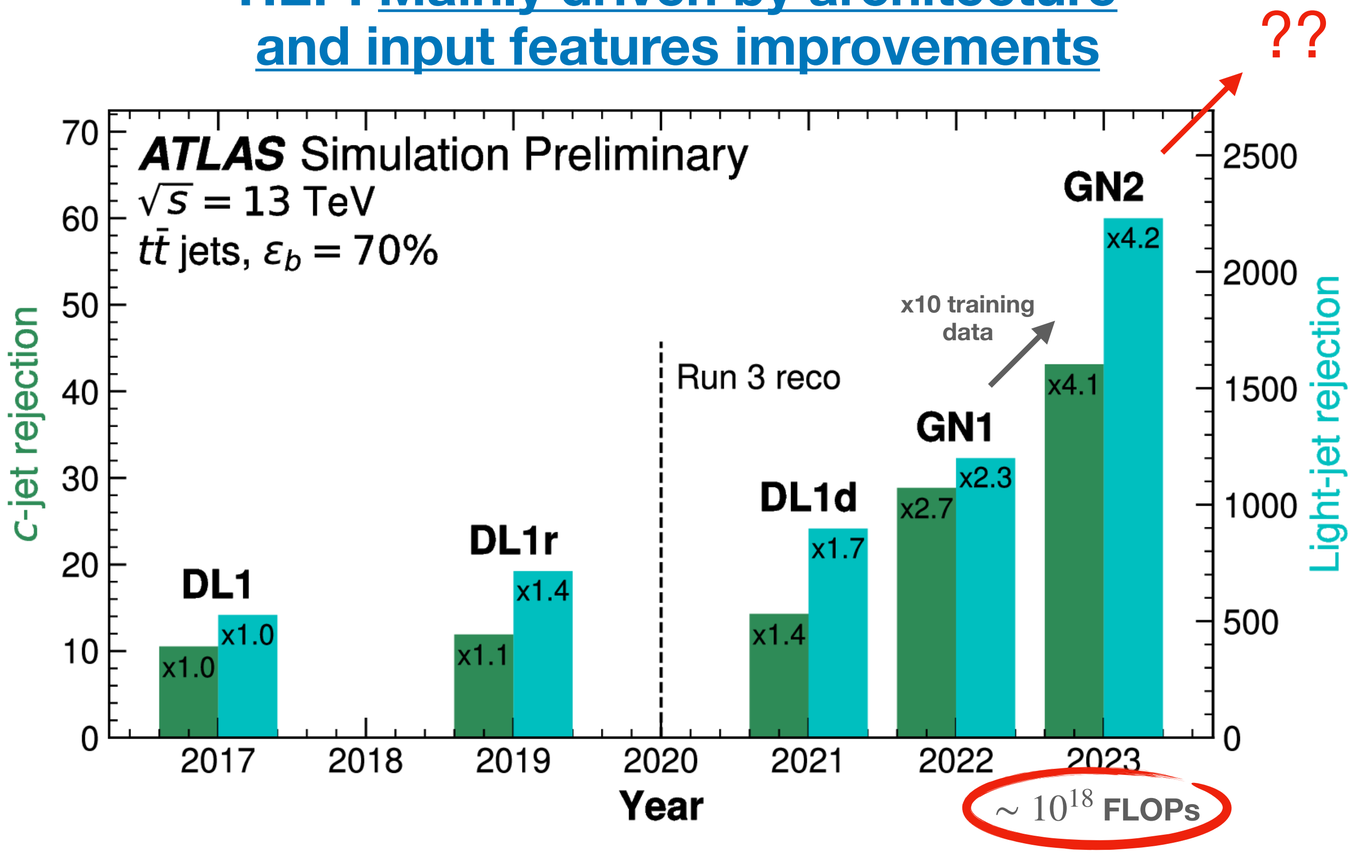


What's driving ML performance?

Industry: Mainly driven by scale



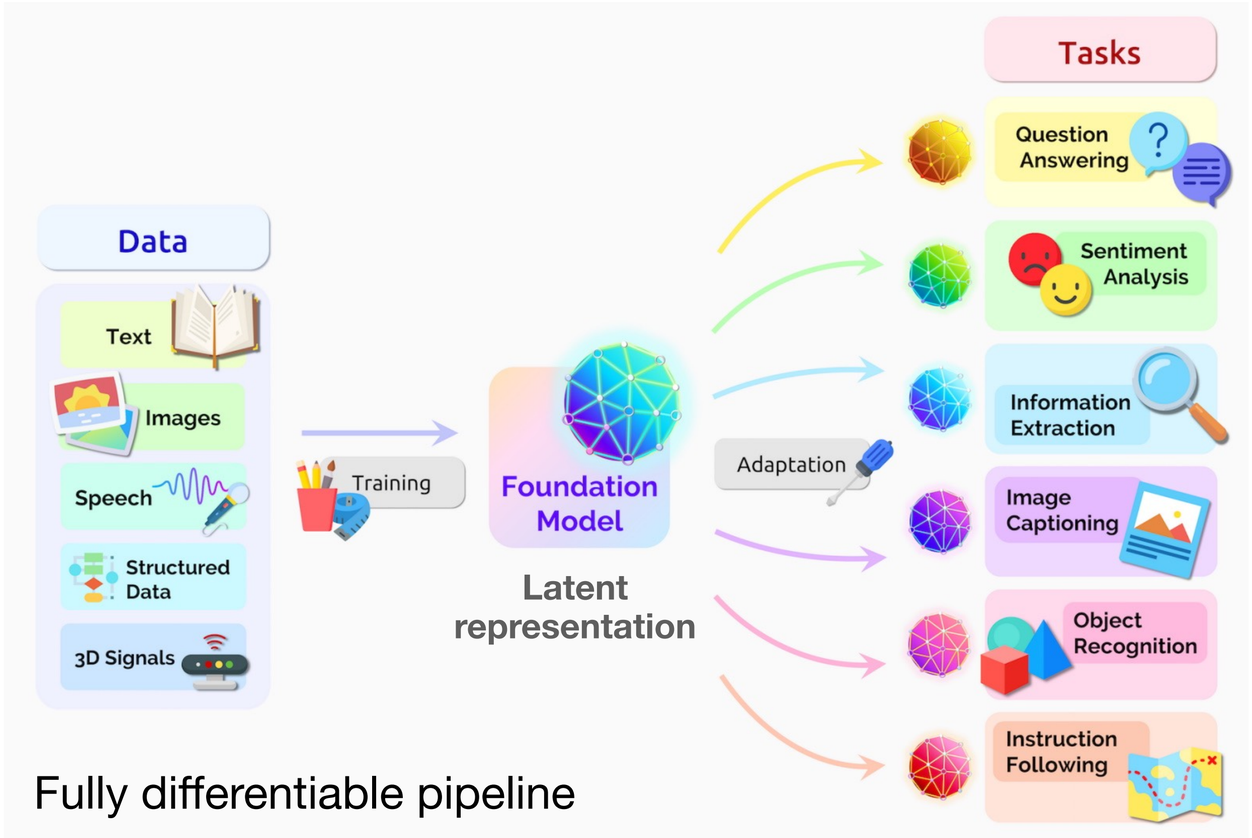
HEP: Mainly driven by architecture and input features improvements



~7 orders of magnitude away!

Industry: foundation models

arXiv:2108.07258



Fully differentiable pipeline

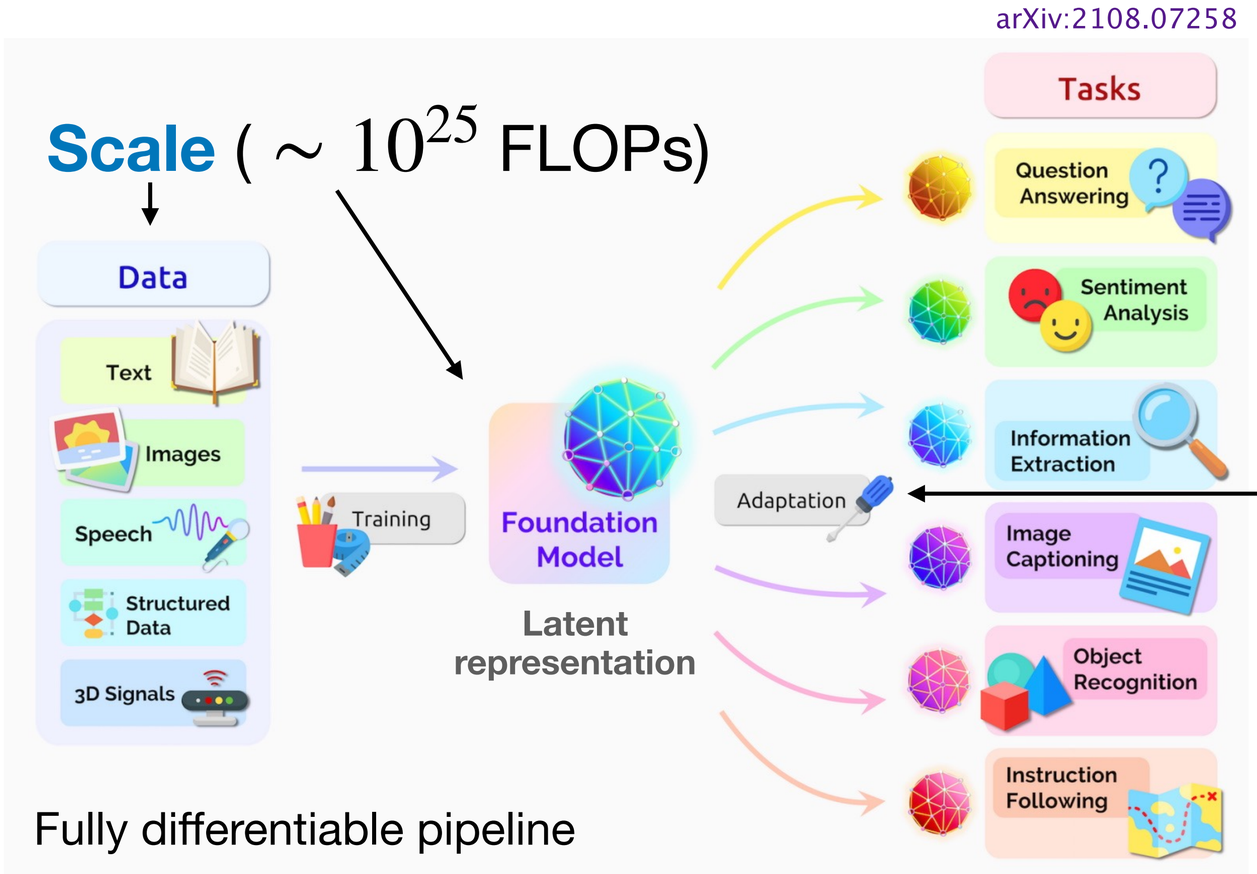
1) **Pre-training** on large dataset

Expensive, done only once

2) **Fine-tuning** on small dataset

- **End-to-end**
- Or from "**Frozen**" representation

Industry: foundation models



1) **Pre-training** on large dataset

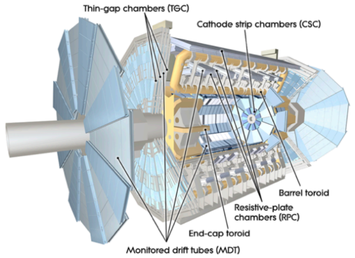
Expensive, done only once

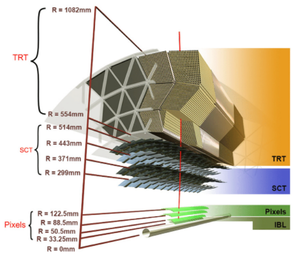
2) **Fine-tuning** on small dataset

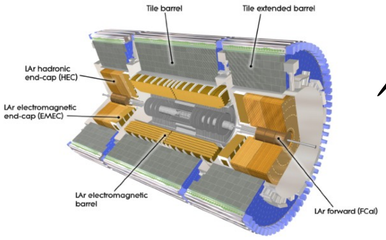
- **End-to-end**
- Or from "**Frozen**" representation

"Transfer learning is what makes foundation models possible,
but scale is what makes them powerful"

Reconstruction ≃ Foundation model







Reconstruction: Complex hierarchical pipeline of **frozen neural components**

Small analysis network

Muon modality

Tracking modality

Calorimeter modality

ML

ML

ML

ML

ϕ_1 ϕ_2 ϕ_3 ϕ_4

Small learnable blocks

Reco Event (particles, jets, MET)

Latent representation

Working points

ϕ

Analysis 1

Analysis 2

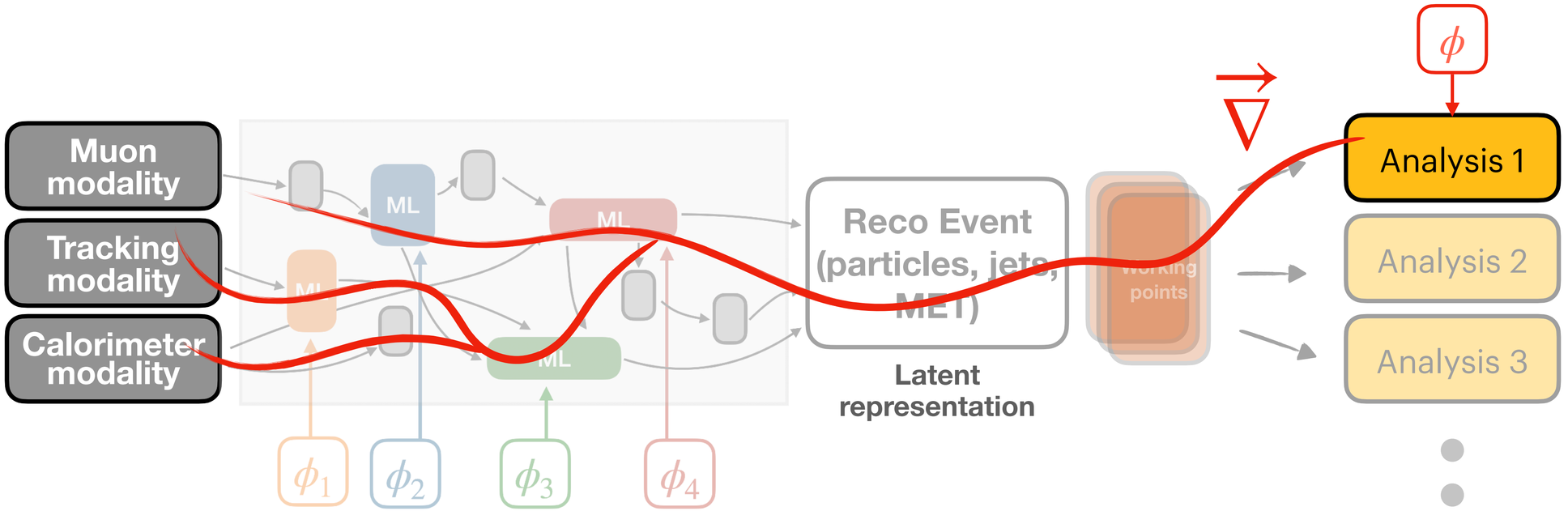
Analysis 3

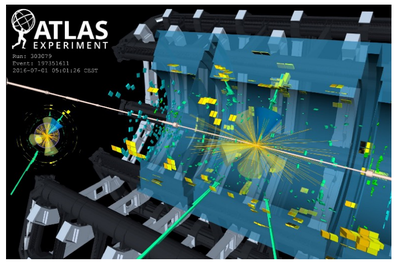
Scale ($\sim 10^{18}$ FLOPs)

Fine-tuning (discrete wp choices)

What are we missing?

Increase **Scale** and allow **Fine-tuning** (preserve gradients “end-to-end”)



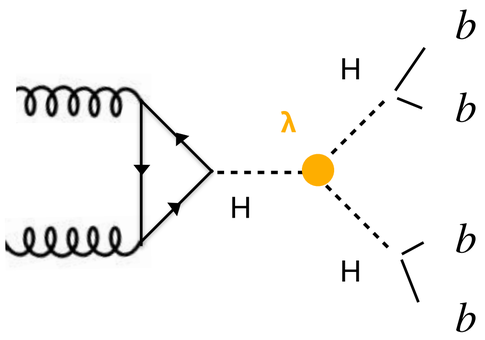


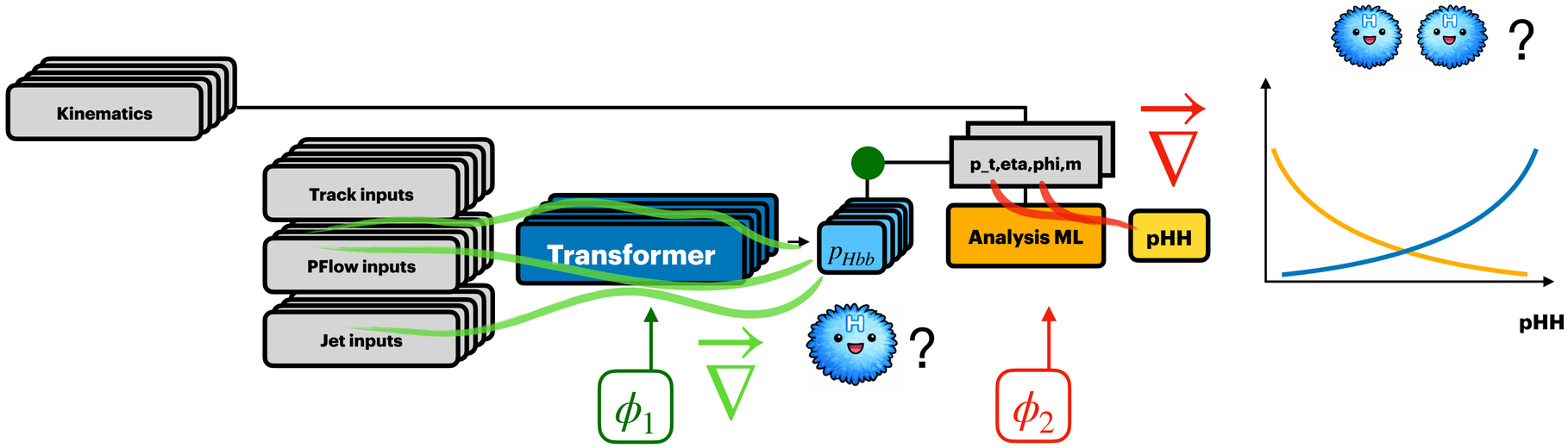
Data → Foundation model → Task 1, Task 2, Task 3, Task 4

Pre-training

Fine-tuning

Benchmark example: HH(4b)



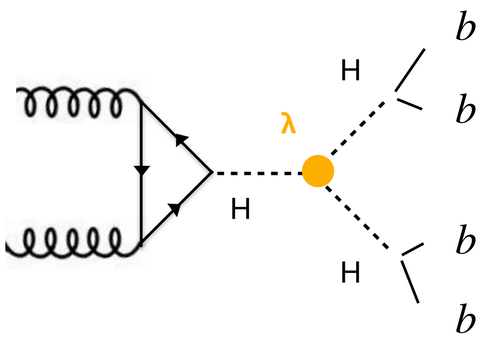


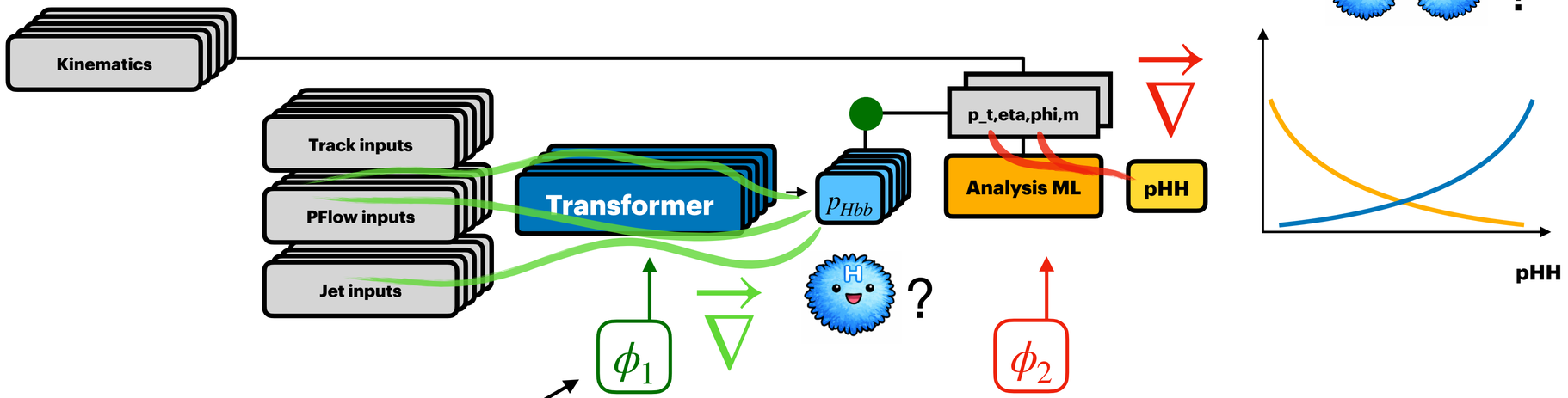
1) Jet flavors? — Low level constituents

2) Filter — Frozen "Representation"

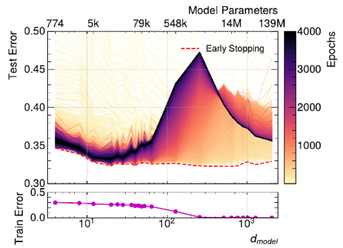
3) Is it a HH event? — High level features

Benchmark example: HH(4b)

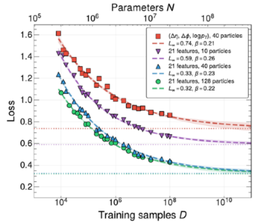




M. Vigl, L. Heinrich

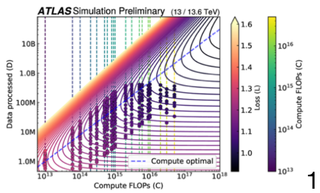


M. Vigl, N. Hartman, M. Kagan, L.Heinrich



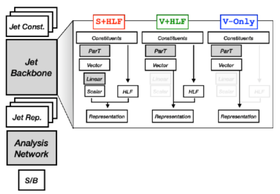
1. Scale

ATLAS collaboration



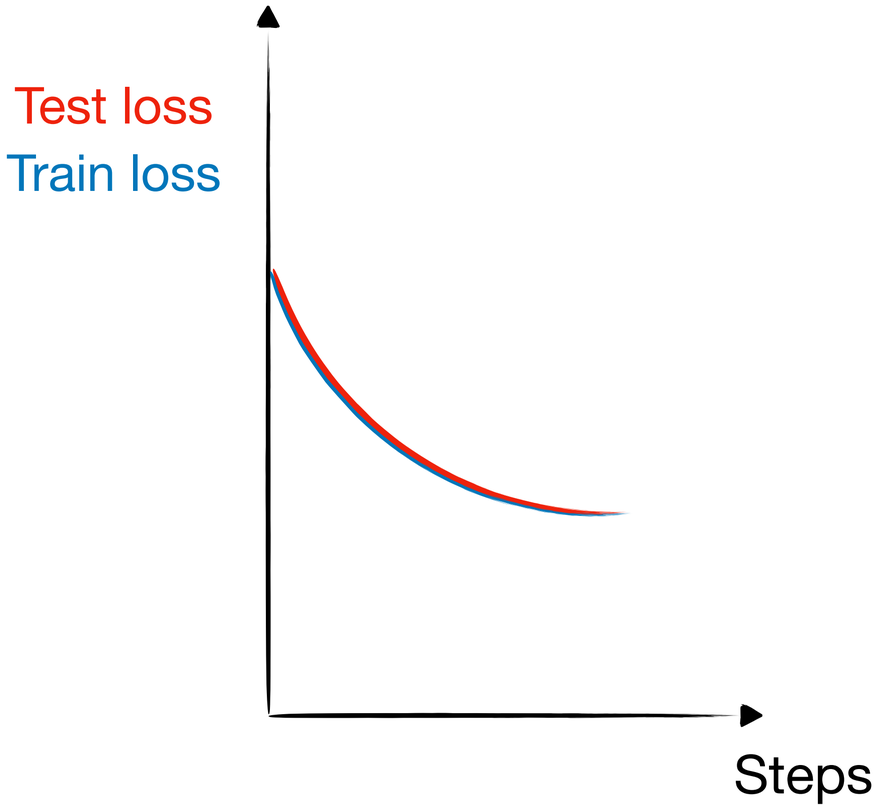
2. Fine-tuning

M. Vigl, N. Hartman, L. Heinrich



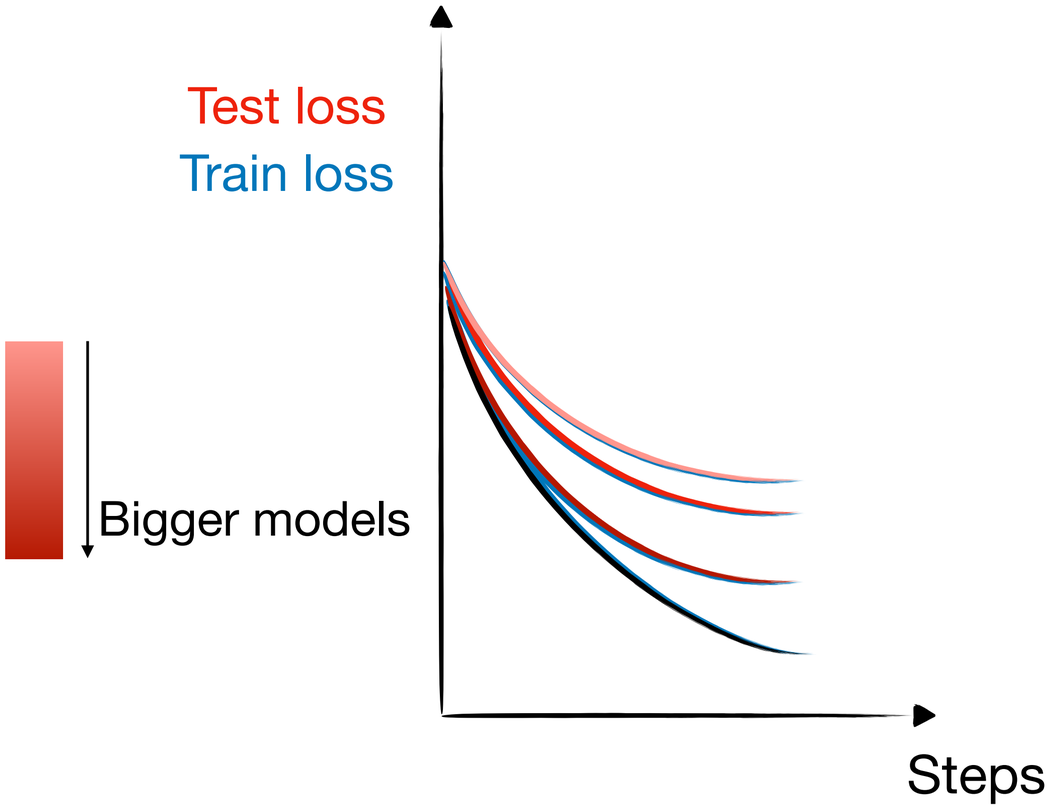
Neural Scaling laws (simplified)

No data repetition!



Neural Scaling laws (simplified)

No data repetition!

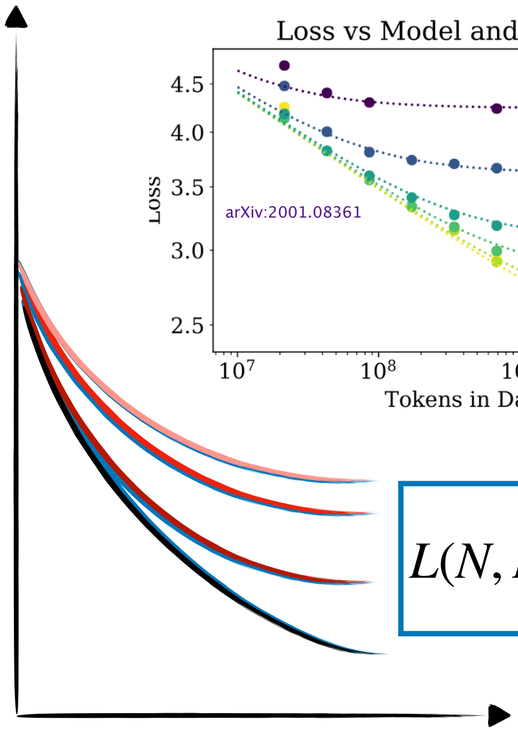


Neural Scaling laws (simplified)

Test loss

Train loss

Bigger models



$$L(N, D) = L_\infty + \frac{A}{N^\alpha} + \frac{B}{D^\beta}$$

arXiv:2203.15556

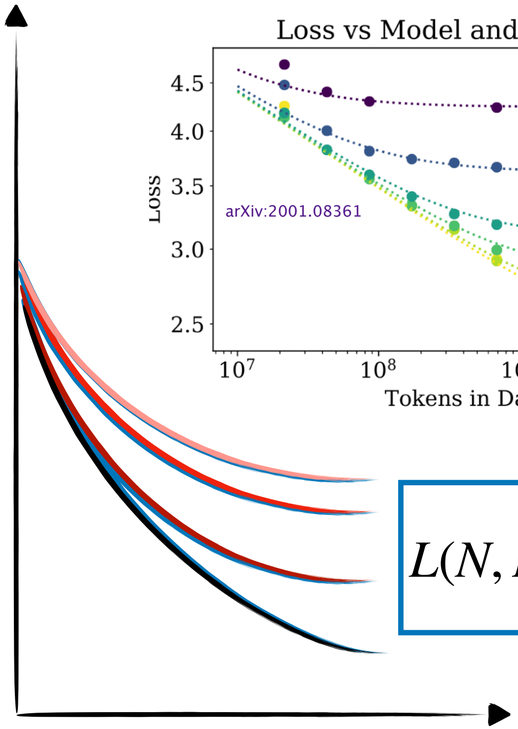
Steps

Neural Scaling laws (simplified)

Test loss

Train loss

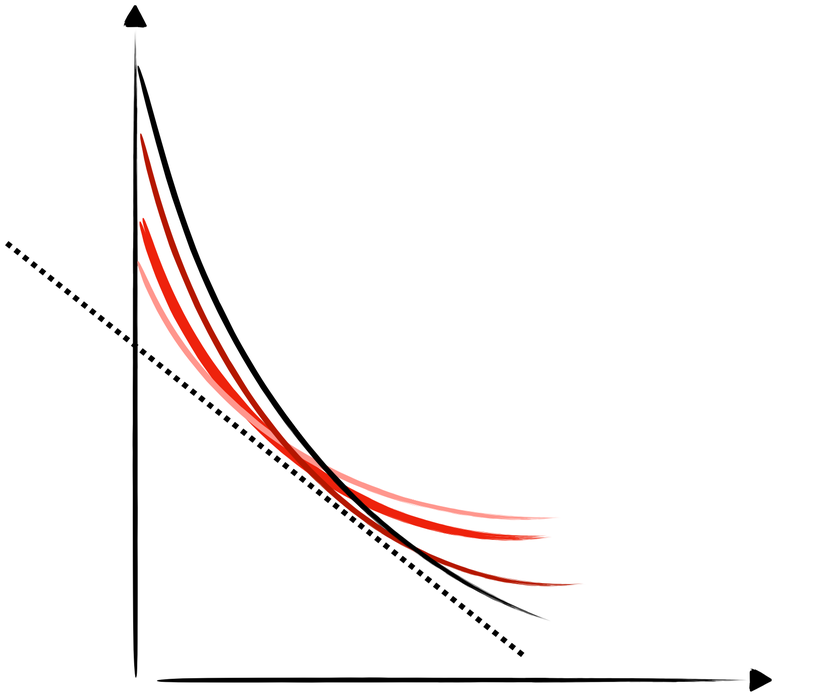
Bigger models



Steps

$$L(N, D) = L_\infty + \frac{A}{N^\alpha} + \frac{B}{D^\beta}$$

arXiv:2203.15556



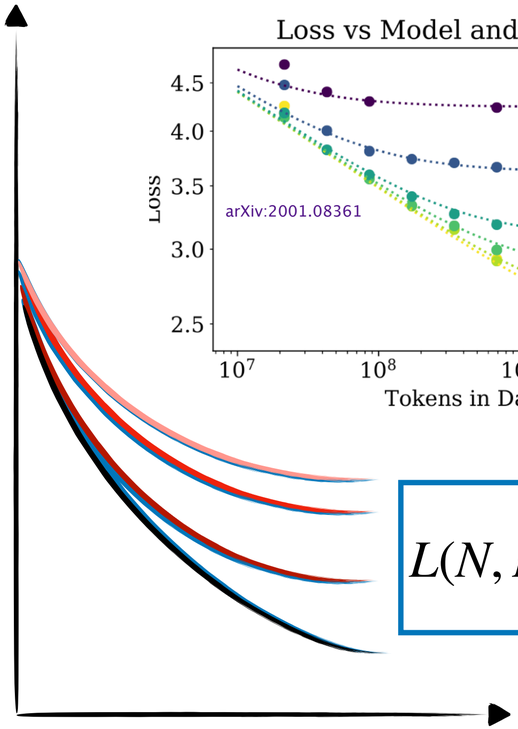
$$C = 6NBS$$

Neural Scaling laws (simplified)

Test loss

Train loss

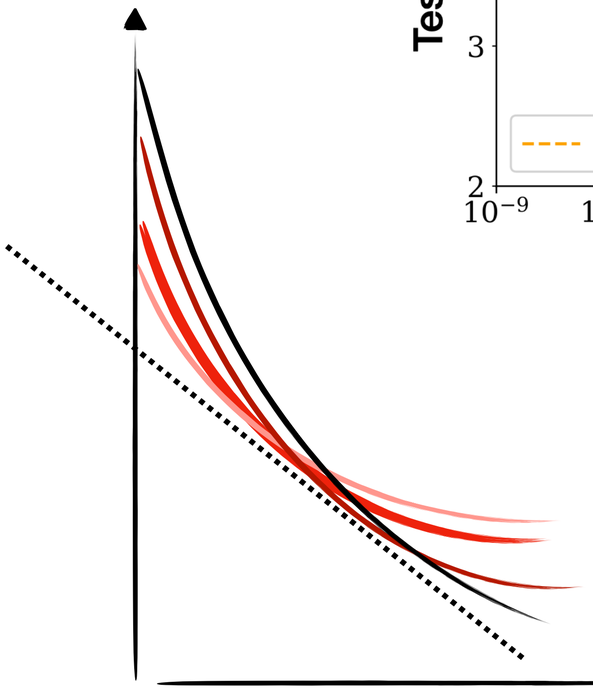
Bigger models



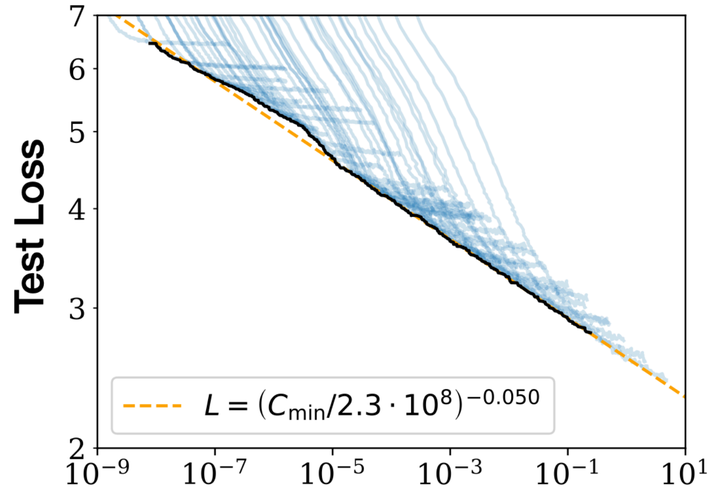
Steps

$$L(N, D) = L_\infty + \frac{A}{N^\alpha} + \frac{B}{D^\beta}$$

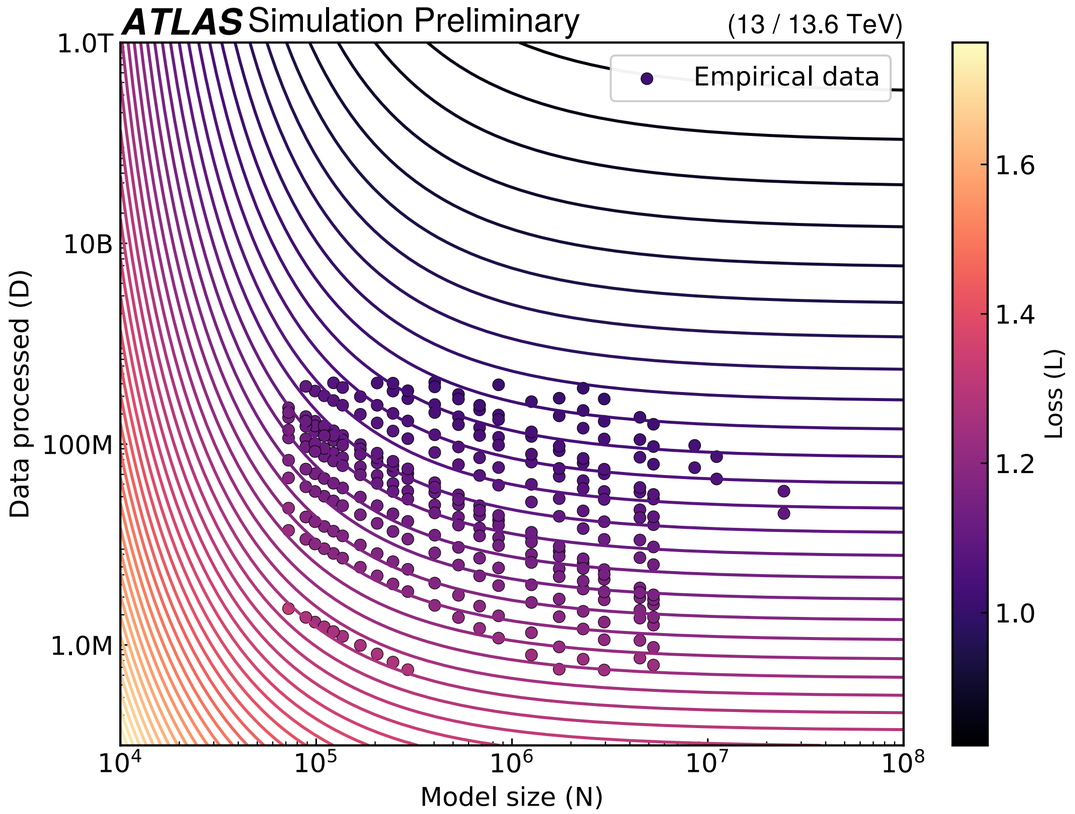
arXiv:2203.15556



$C = 6NBS$



Scaling GN3: Loss surface

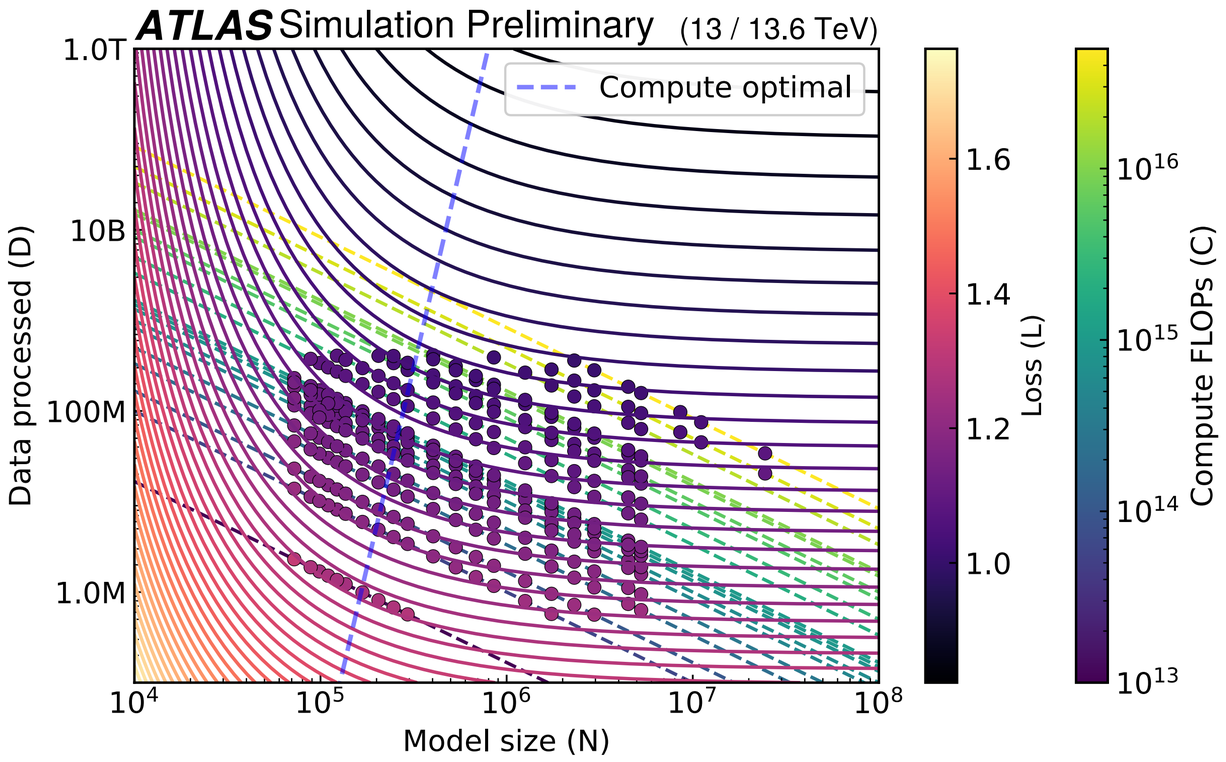


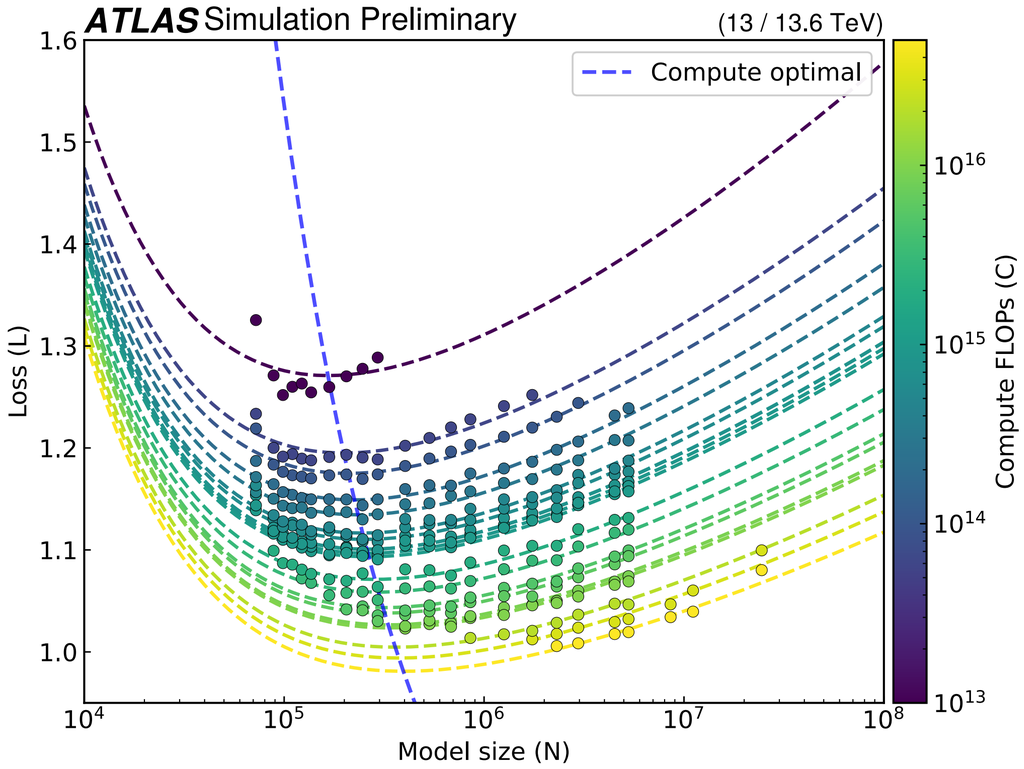
Highest achievable performance (on MC, with current inputs), we'll see later what this means in terms of physics reach potential

$$L(N, D) = L_\infty + \frac{A}{N^\alpha} + \frac{B}{D^\beta}$$

Quantity	Symbol	Description	Value
Irreducible loss	L_∞	$L(N_{\to\inf}, D_{\to\inf})$	0.619 (0.539, 0.671)
Model scaling exponent	α	$L_N \propto N^{-\alpha}$	0.677 (0.644, 0.764)
Data scaling exponent	β	$L_D \propto D^{-\beta}$	0.077 (0.065,0.086)

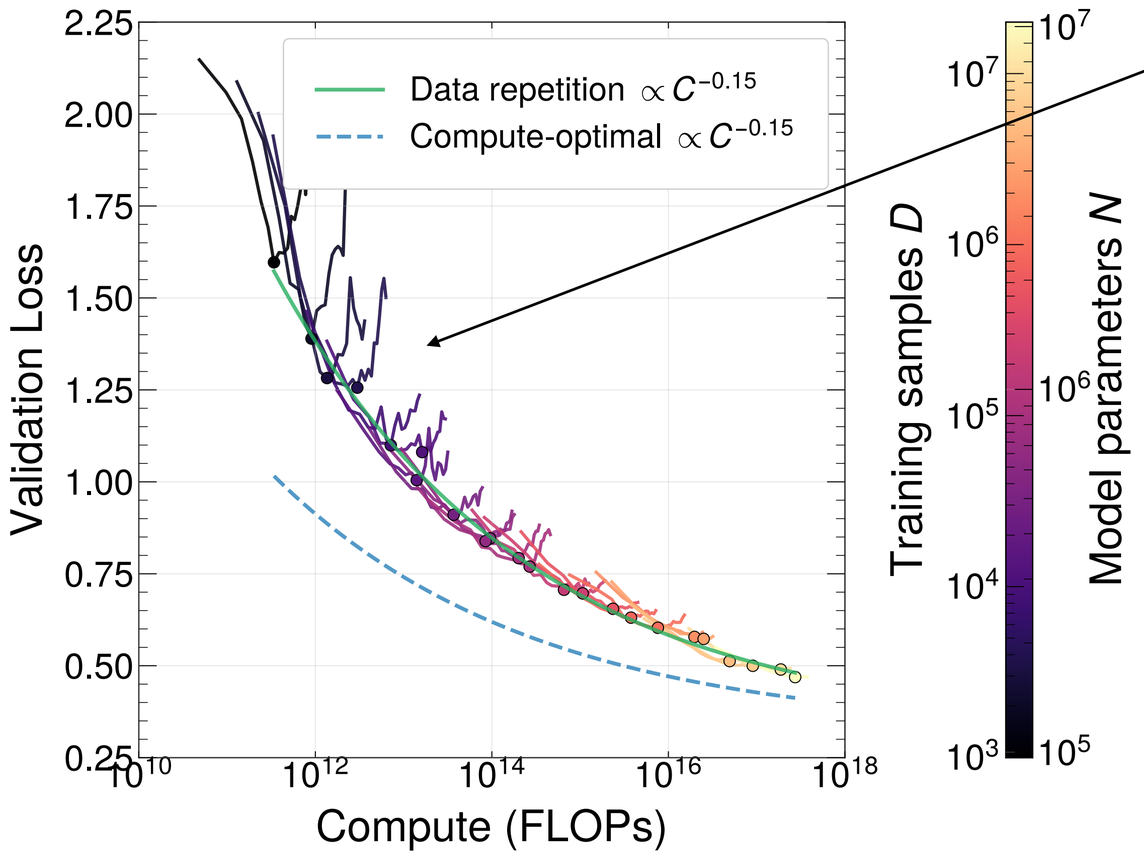
Compute optimal trajectories





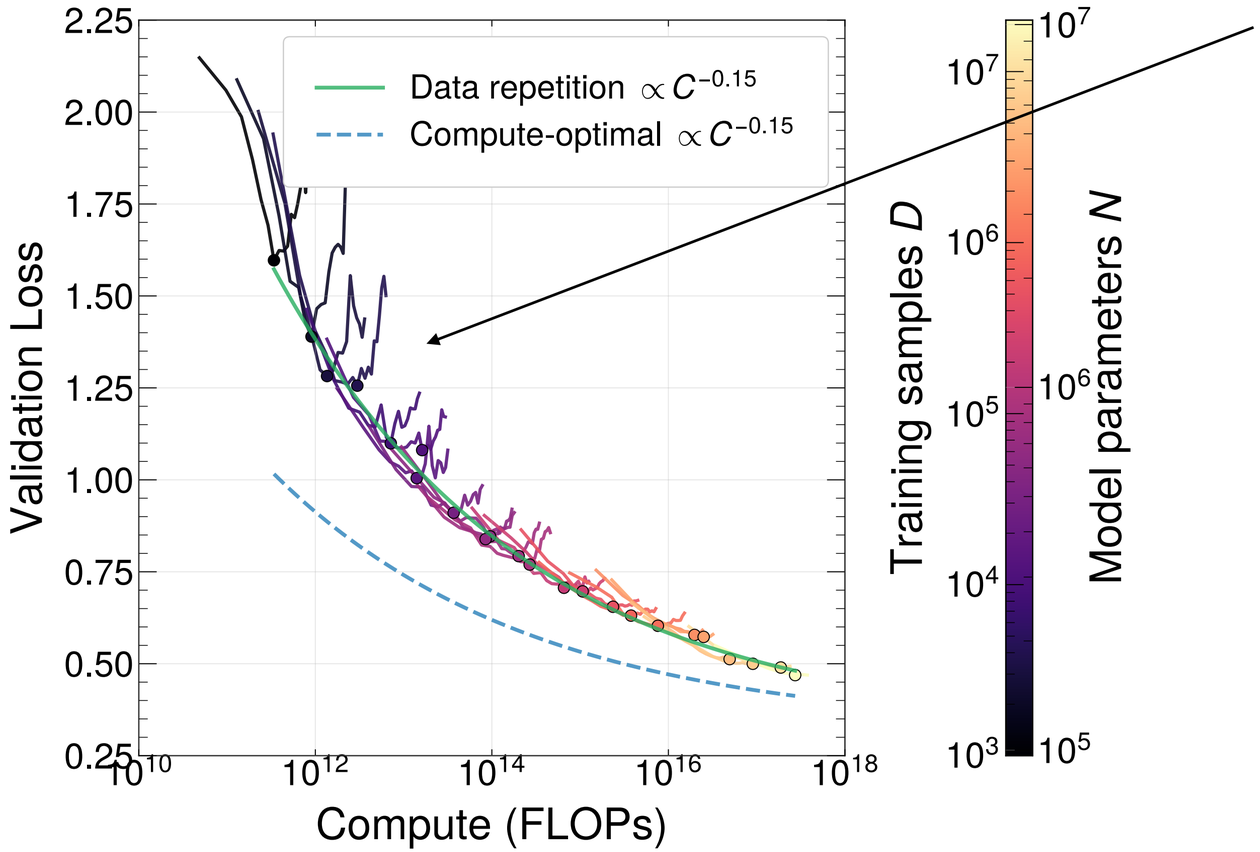
$$\min_{N,D} L(N, D) \qquad \text{s.t.} \quad C = 6ND = \text{const.}\, C_{\text{budget}}$$

Data limited scaling

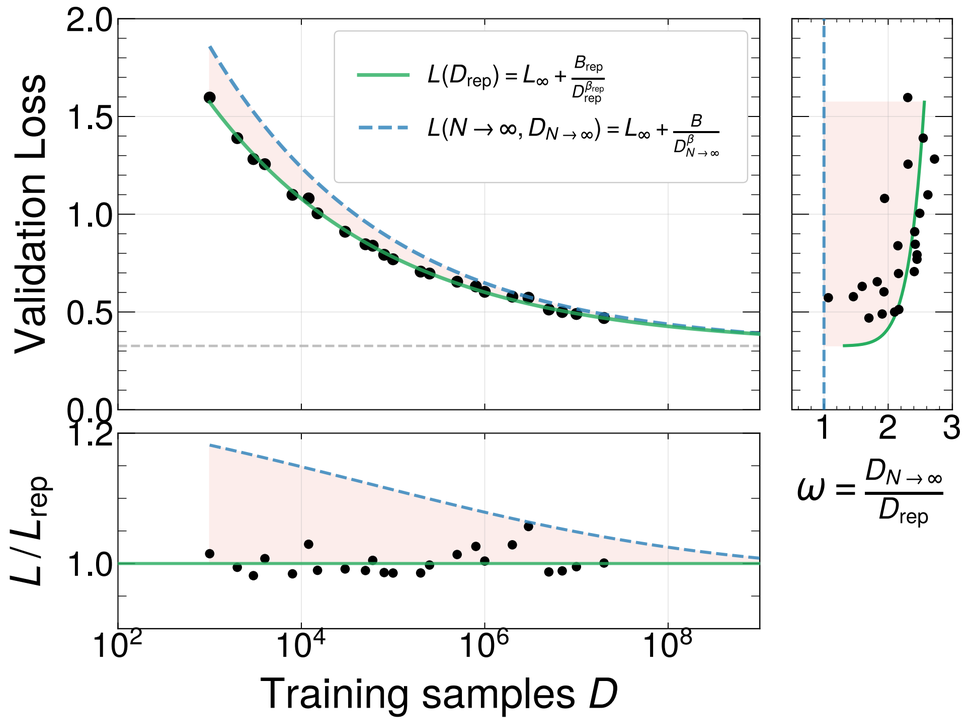


Data repetition is always compute sub-optimal

Data limited scaling

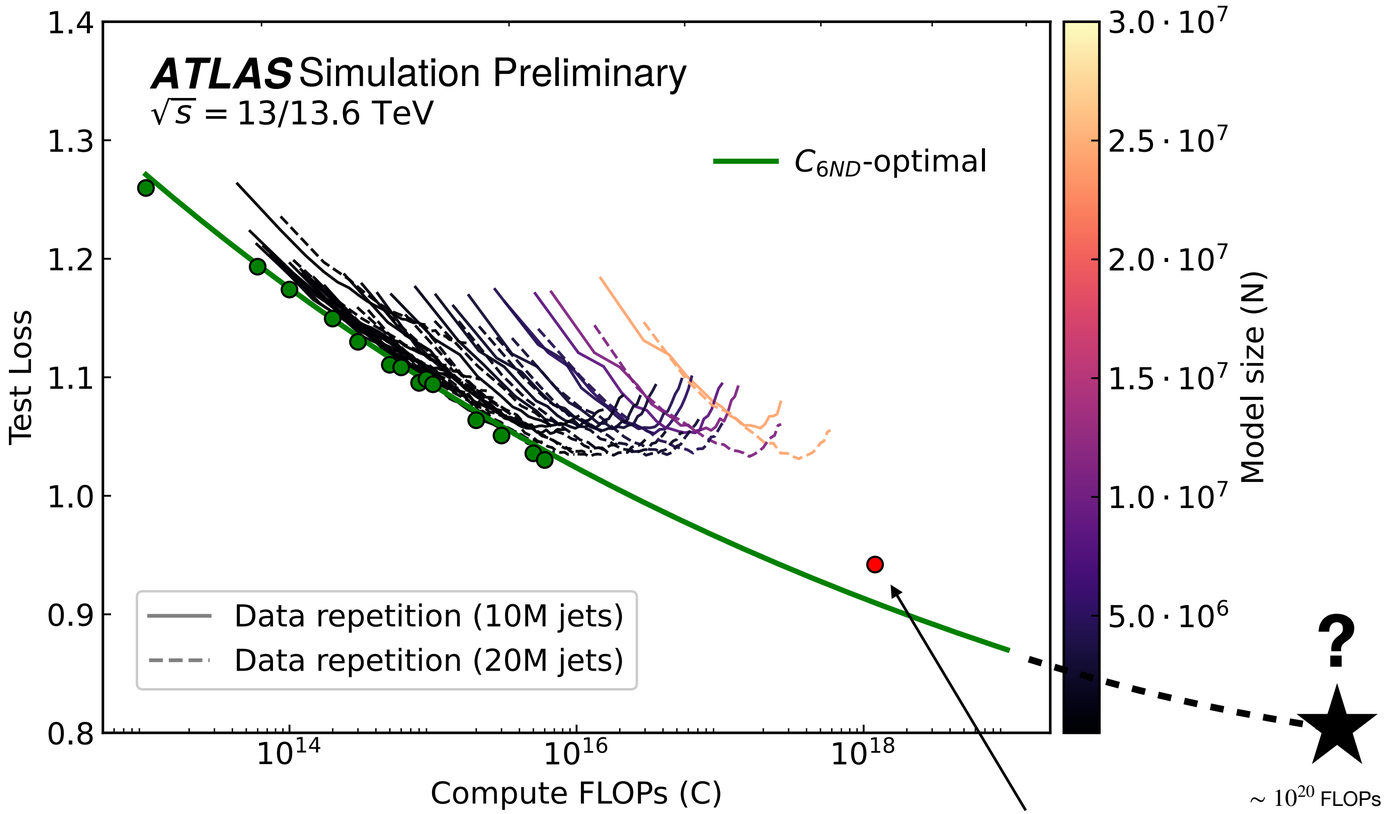


Data repetition is always compute sub-optimal



10x compute spent on data repetition can get us up to x3 effective Dataset size

Where are we now?



?

~ 10^{20} FLOPs

GPT-3 scale

~ 10^{23} FLOPs

GPT-5 scale

~ 10^{25} FLOPs

Current state-of-the-art (GN3)

We do have a lot of compute

Can reasonably get access to O(100-1000) GPUs for a few days ($\sim 10^{22} - 10^{23}$ FLOPs)

MAX PLANCK COMPUTING & DATA FACILITY MPCDF

Let's put the scaling laws to test →

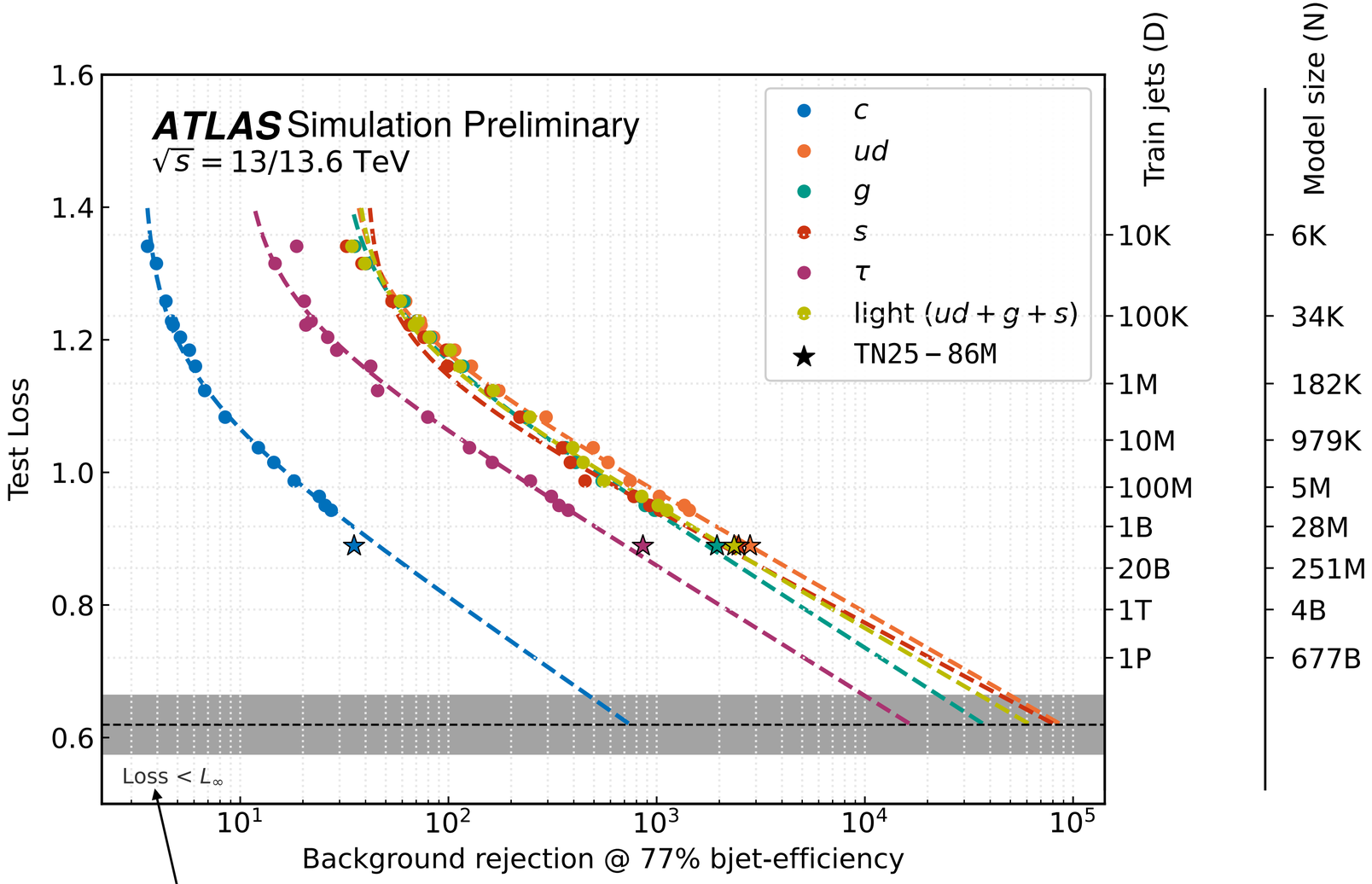
★ TN25-86M

~ 7.7 B Jets

$\sim 4 \times 10^{20}$ FLOPs

160 A100s

It works!



$$L(N, D) = L_\infty + \frac{A}{N^\alpha} + \frac{B}{D^\beta}$$

Seems like we were far away from solving jet tagging, **still x10-50 to improve upon sota!**

- Predictable improvements: can get there with scale alone

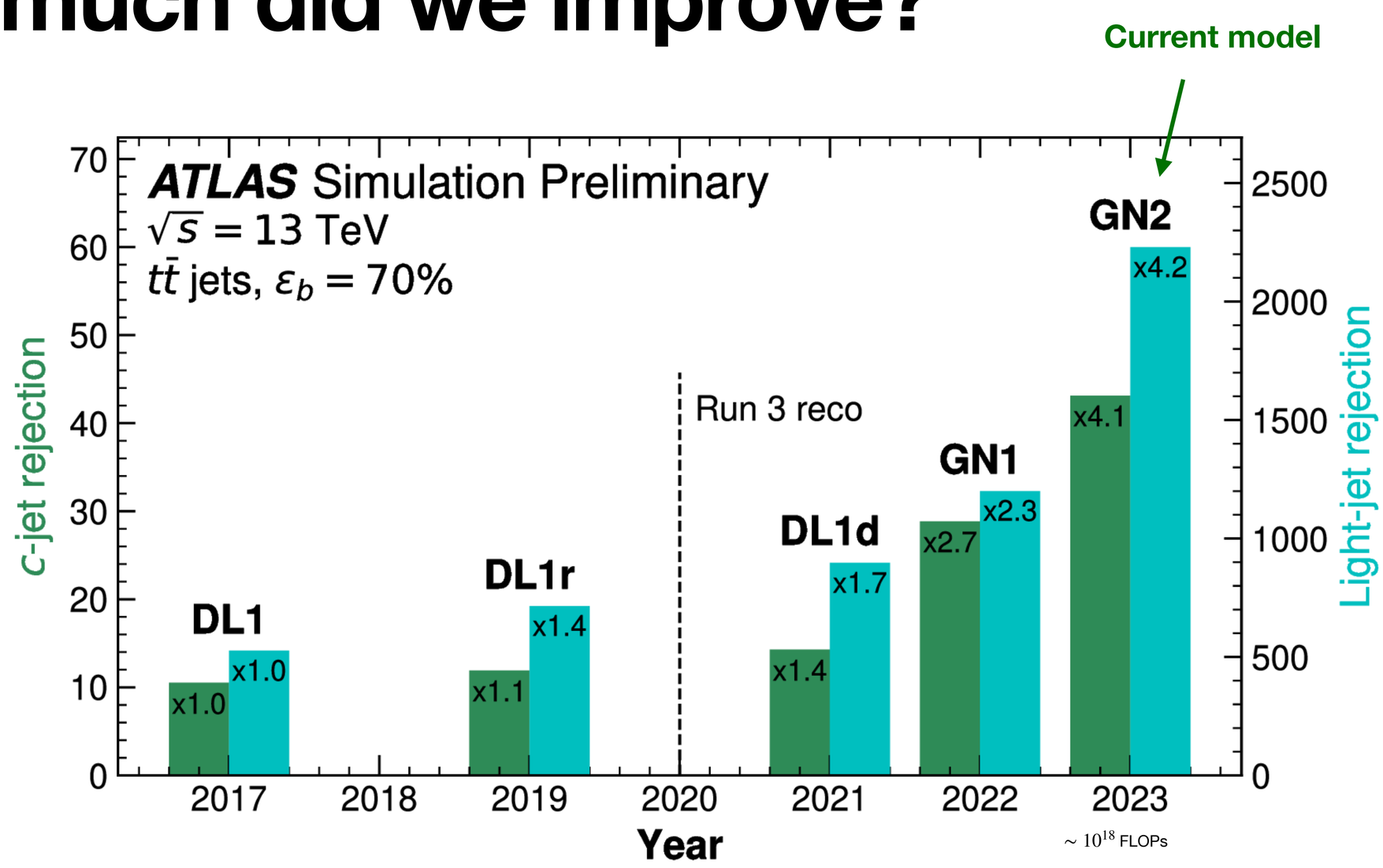
★ TN25-86M

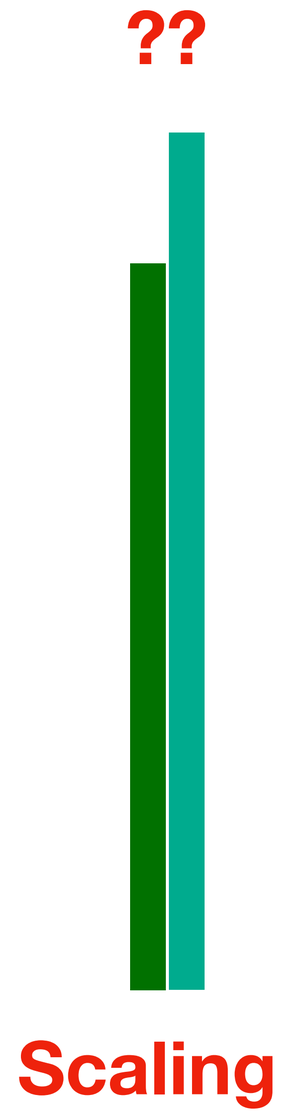
~ 7.7 B Jets

$\sim 4 \times 10^{20}$ FLOPs

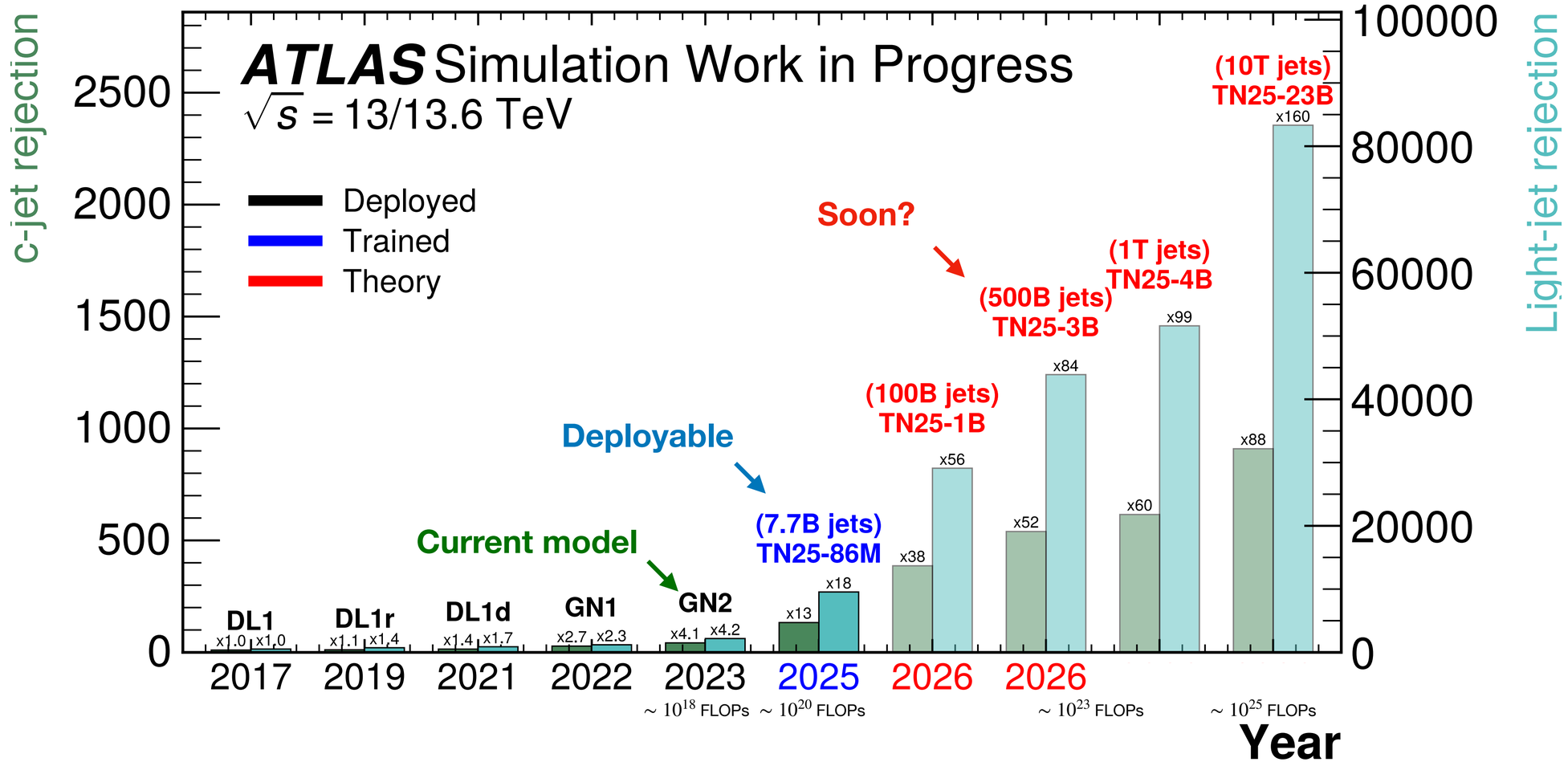
160 A100s

How much did we improve?





How much did we improve?



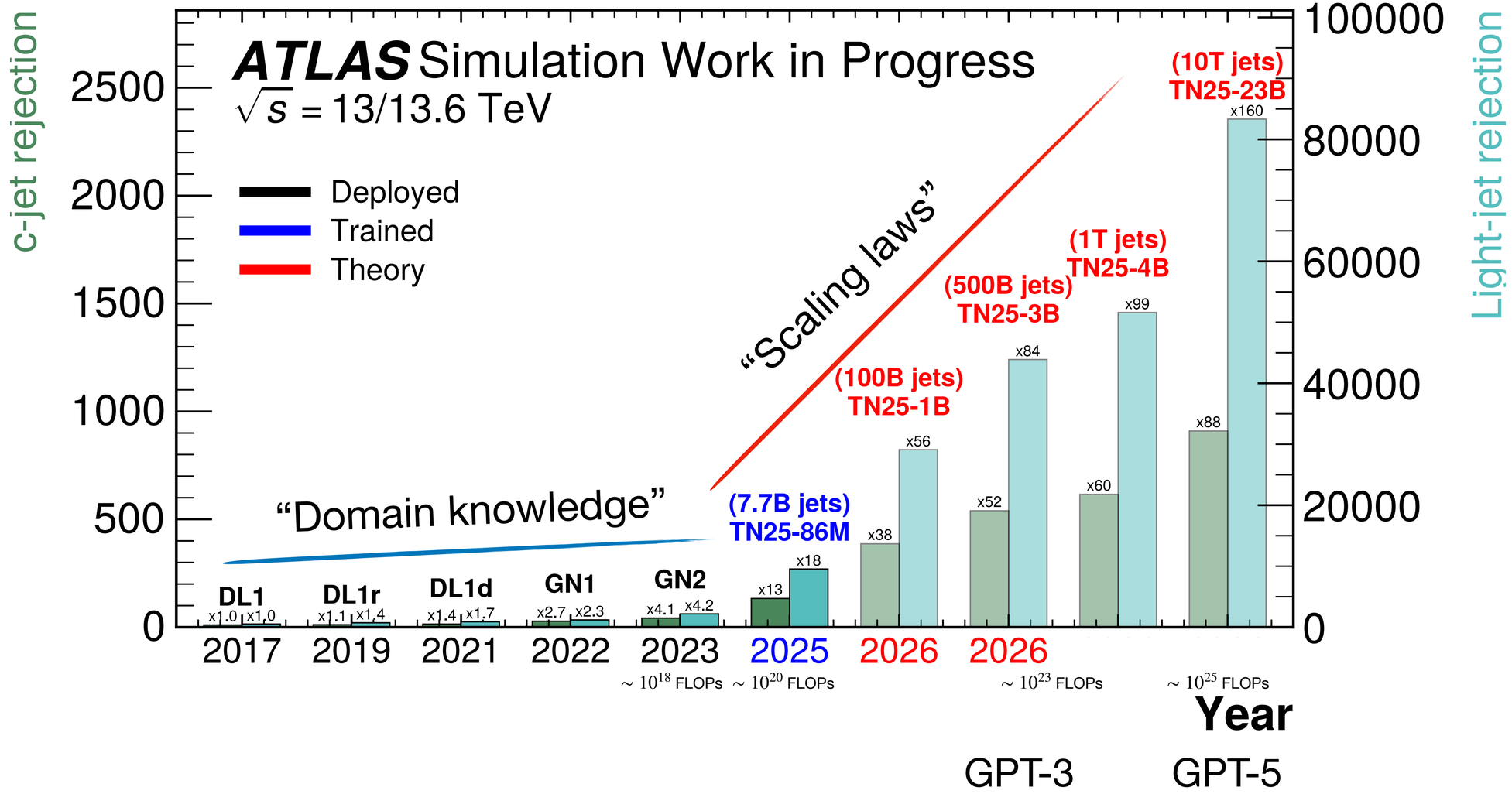
The bitter lesson

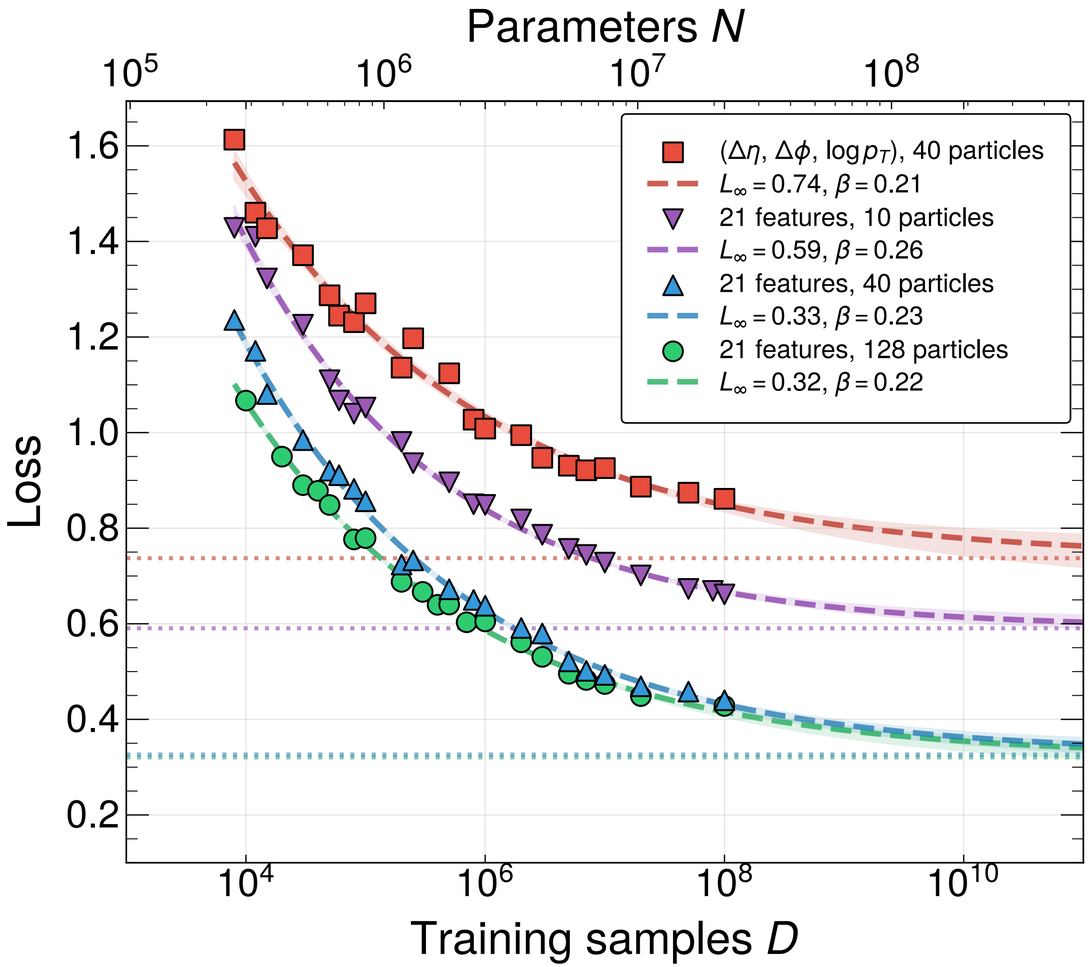
The Bitter Lesson

Rich Sutton

March 13, 2019

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. The ultimate reason for this is Moore's law, or rather its



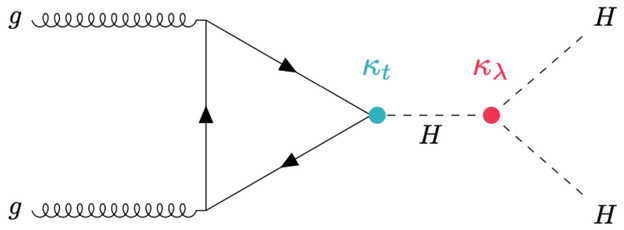


More low level data increases the performance ceiling (L_∞ floor) which can consistently be reached by scaling compute

$$L(N, D) = L_\infty + \frac{A}{N^\alpha} + \frac{B}{D^\beta}$$

Impact of scale on HH(4b)

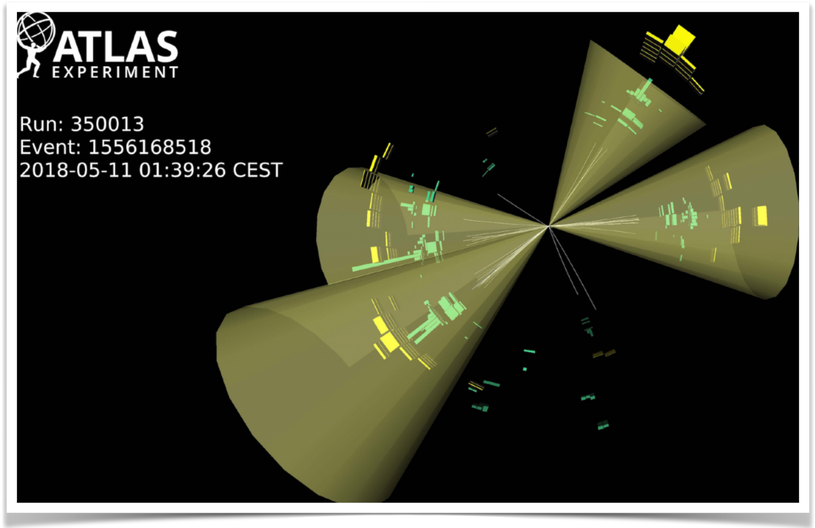
$$Z(L,\epsilon_b) = \frac{S(L,\epsilon_b)}{\sqrt{B(L,\epsilon_b)}} = \frac{s\,\epsilon_b^4\,L}{\sqrt{\left(f\,b_r + b_{ir}\,\epsilon_b^4\right)\,L}} = \frac{s\,\epsilon_b^4}{\sqrt{f\,b_r + b_{ir}\,\epsilon_b^4}}\sqrt{L}\,, \qquad b_{ir}\,\epsilon_b^4 \gg f\,b_r,$$



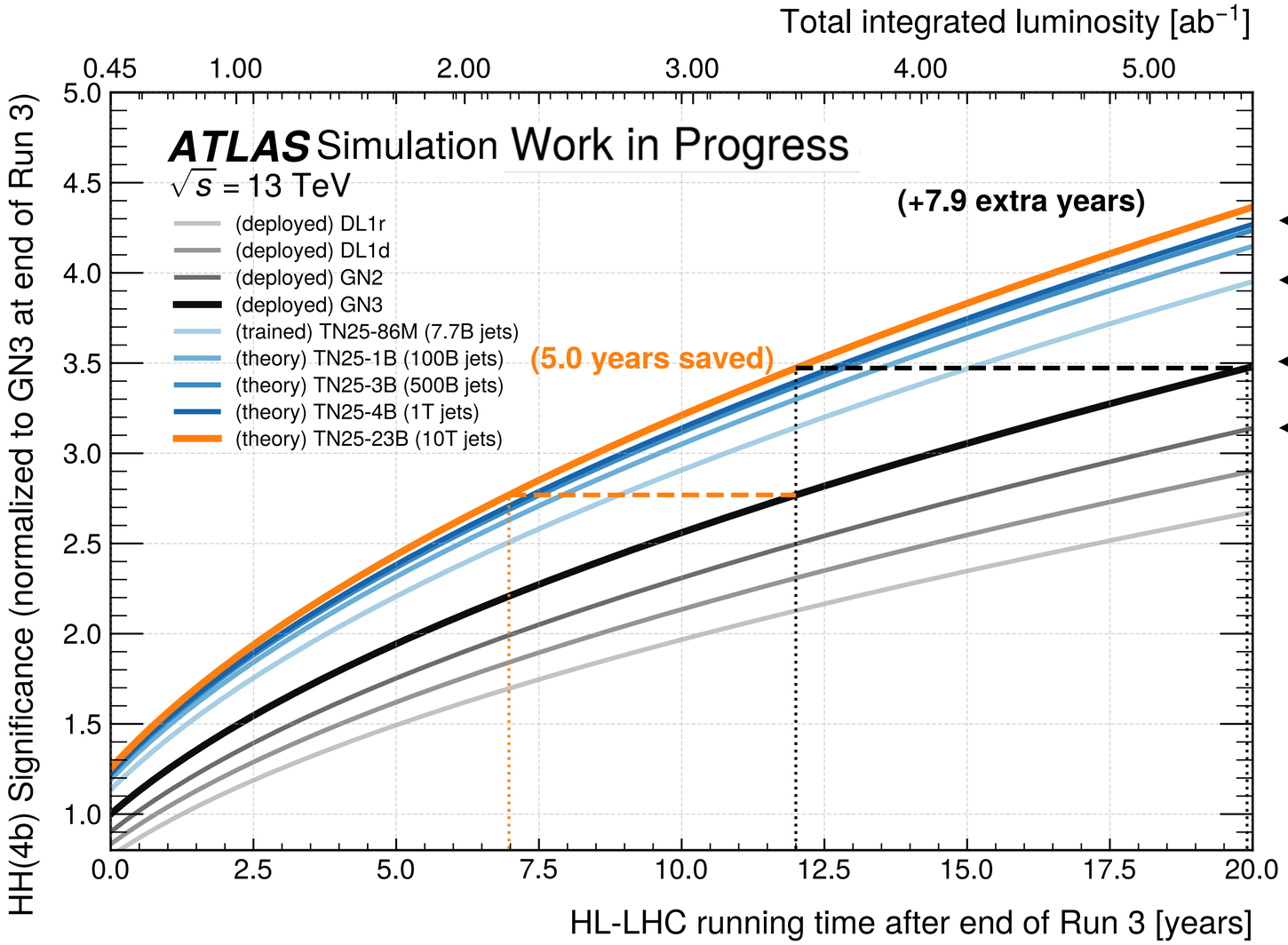
ϵ_b = **Tagger b-jet efficiency at fixed bkg. rejection**

$$Z(L,\epsilon_b) \approx \frac{s\,\epsilon_b^4}{\sqrt{b_{ir}\,\epsilon_b^4}}\sqrt{L} = \frac{s}{\sqrt{b_{ir}}}\,\epsilon_b^2\,\sqrt{L}$$

$$\frac{Z(L,\epsilon_{b,2})}{Z(L,\epsilon_{b,1})} = \frac{\epsilon_{b,2}^4}{\epsilon_{b,1}^4}\sqrt{\frac{f\,b_r + b_{ir}\,\epsilon_{b,1}^4}{f\,b_r + b_{ir}\,\epsilon_{b,2}^4}} \implies \frac{Z_2}{Z_1} \approx \left(\frac{\epsilon_{b,2}}{\epsilon_{b,1}}\right)^2$$



Impact of scale on HH(4b)



Soon?

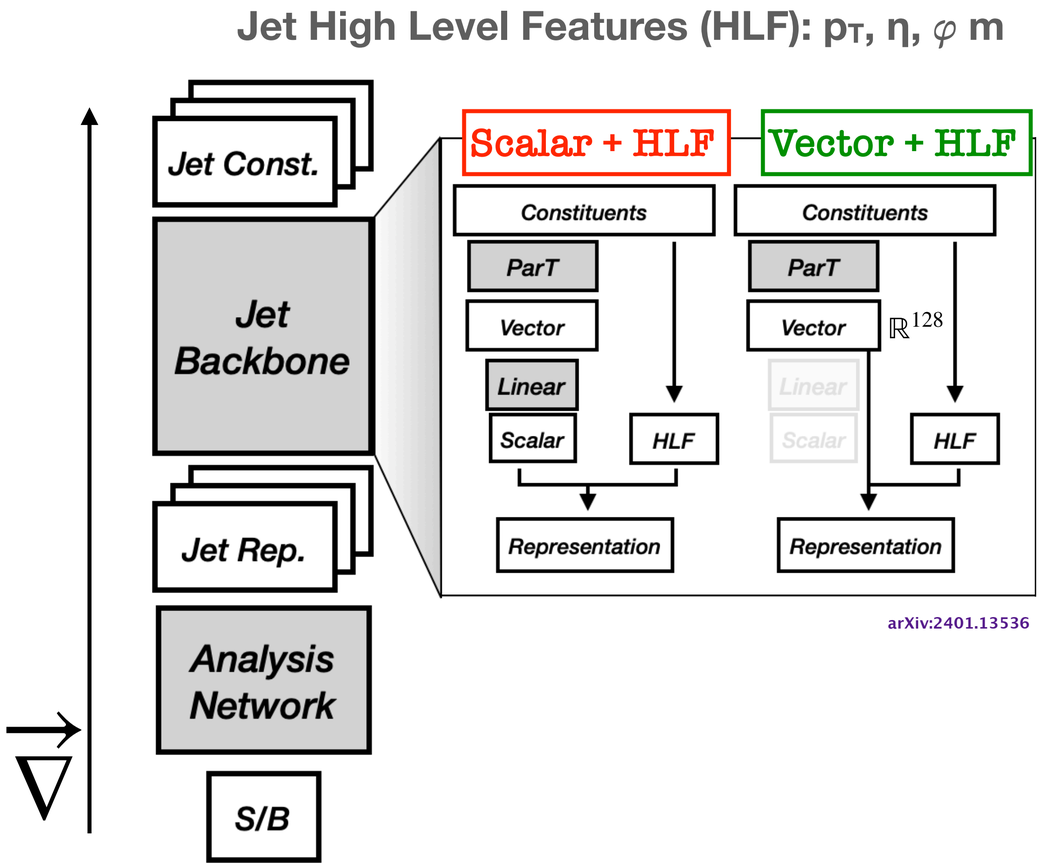
Already Trained

Baseline (GN3)

Current model (GN2)

5-8 years of data taking!

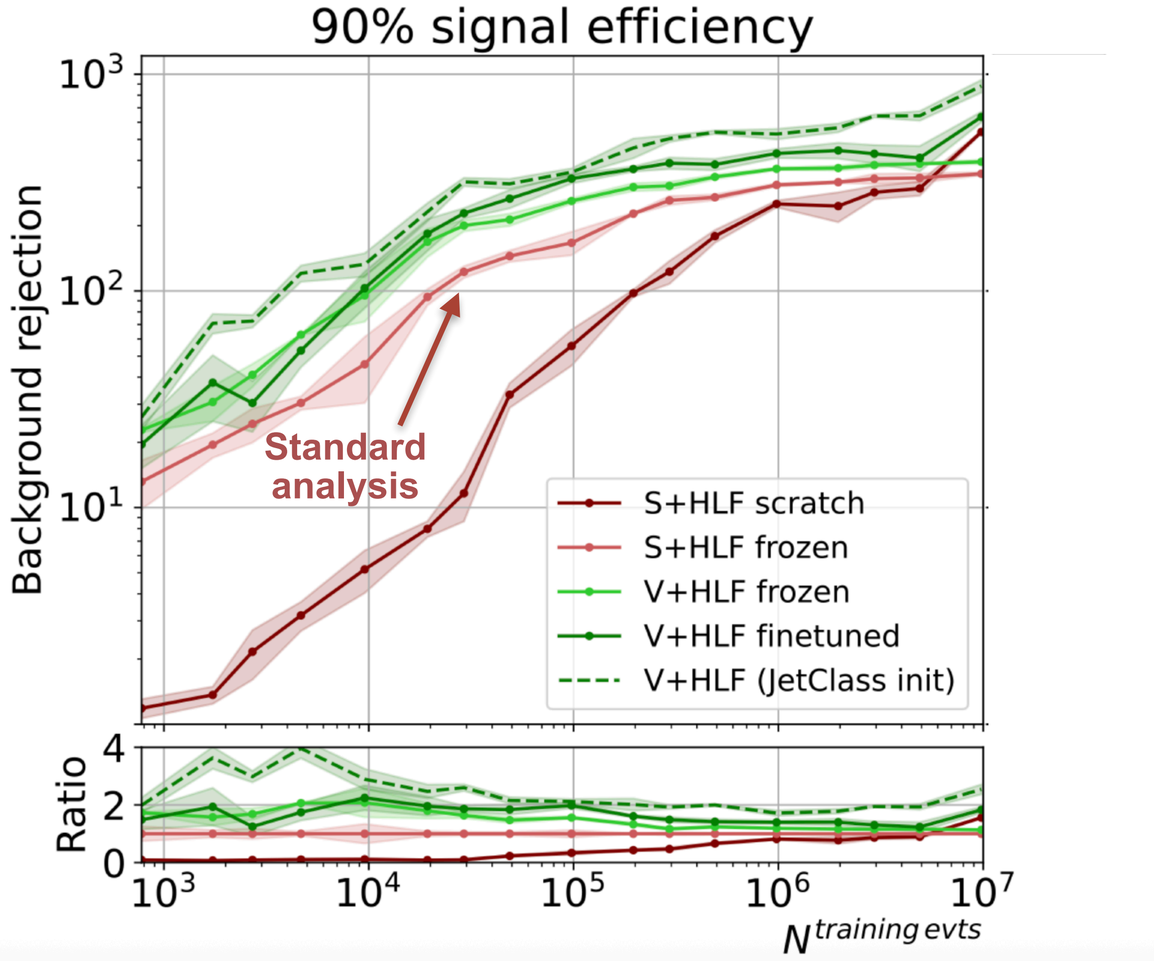
“End-to-end” HH(4b)



“Foundation model” Jet Backbone

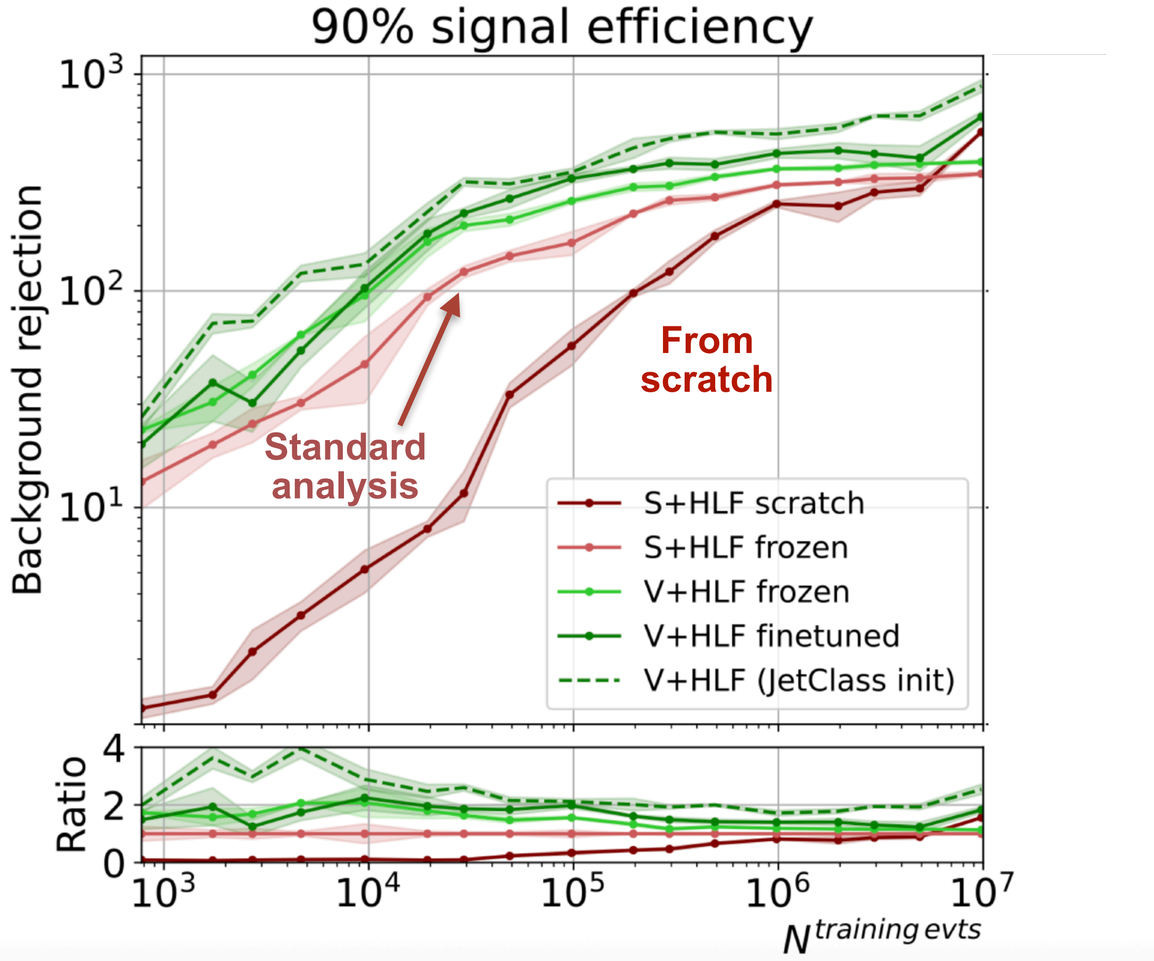
- Do we simply train the full pipeline **from scratch** on the analysis objective?
- Do we pre-train on **jet tagging?**
  - Is this **frozen representation** all we need?
  - Does **fine-tuning** help?
  - Are **high dimensional per-jet representations** better?

What do we gain by preserving gradients?



Standard analysis: “frozen jet tagging”
plus analysis network

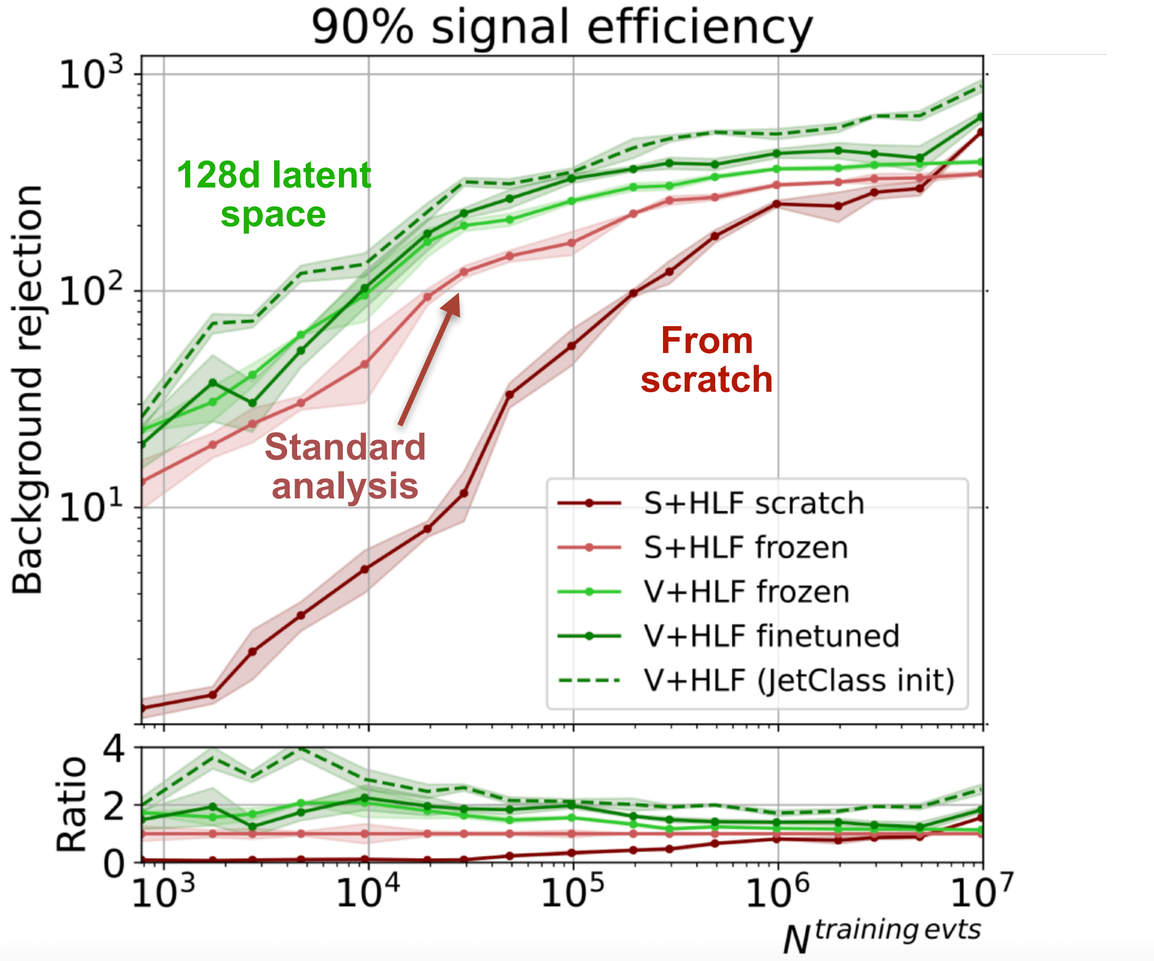
What do we gain by preserving gradients?



Standard analysis: “**frozen** jet tagging”
plus analysis network

From scratch (end-to-end): slower (of course)
but eventually surpasses frozen backbone:
there’s more than just jet tagging!

What do we gain by preserving gradients?

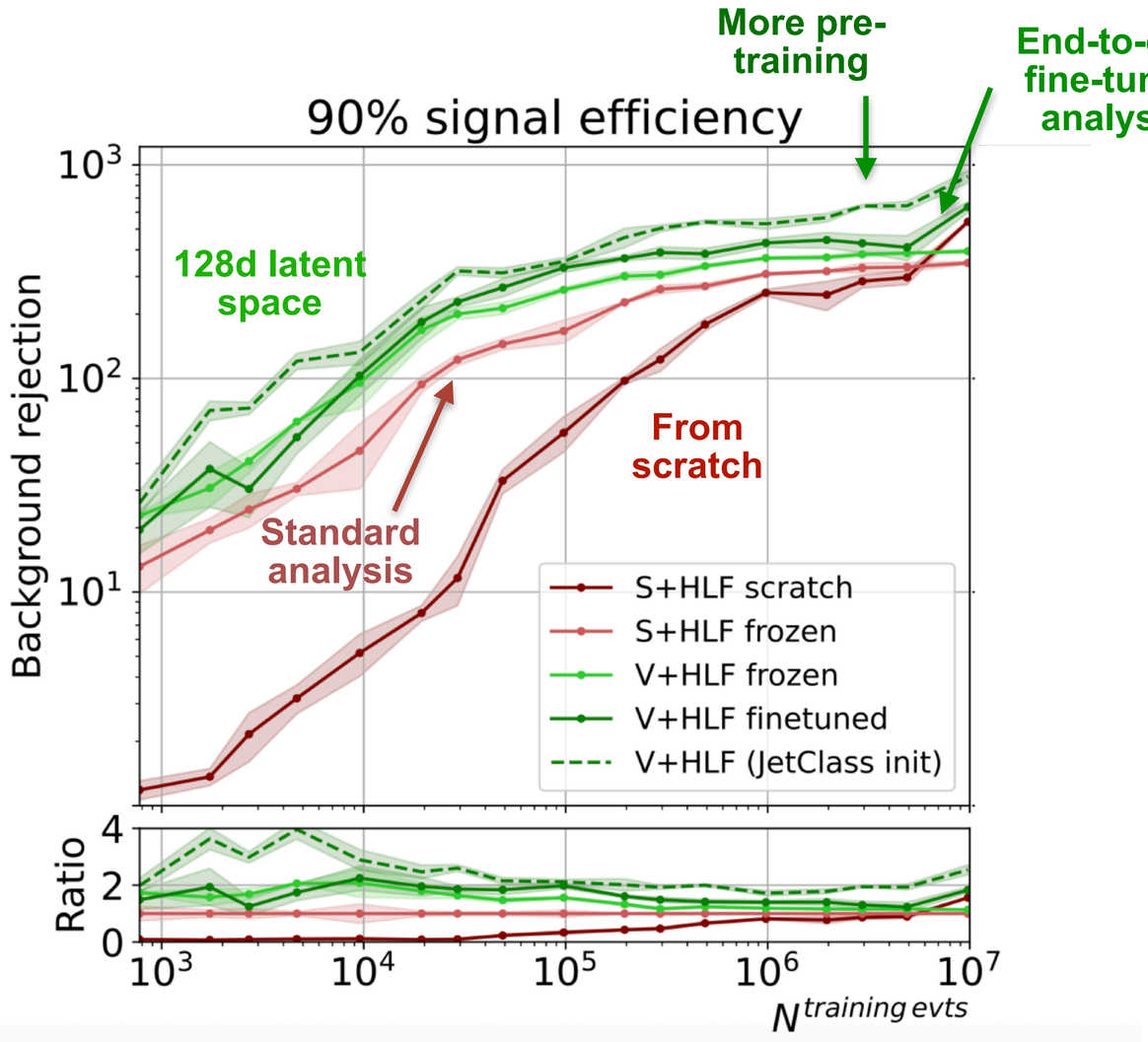


Standard analysis: “**frozen** jet tagging” plus analysis network

From scratch (end-to-end): slower (of course) but eventually surpasses frozen backbone: there’s more than just jet tagging!

High dimensional representations are more expressive

What do we gain by preserving gradients?



Standard analysis: "frozen jet tagging"
plus analysis network

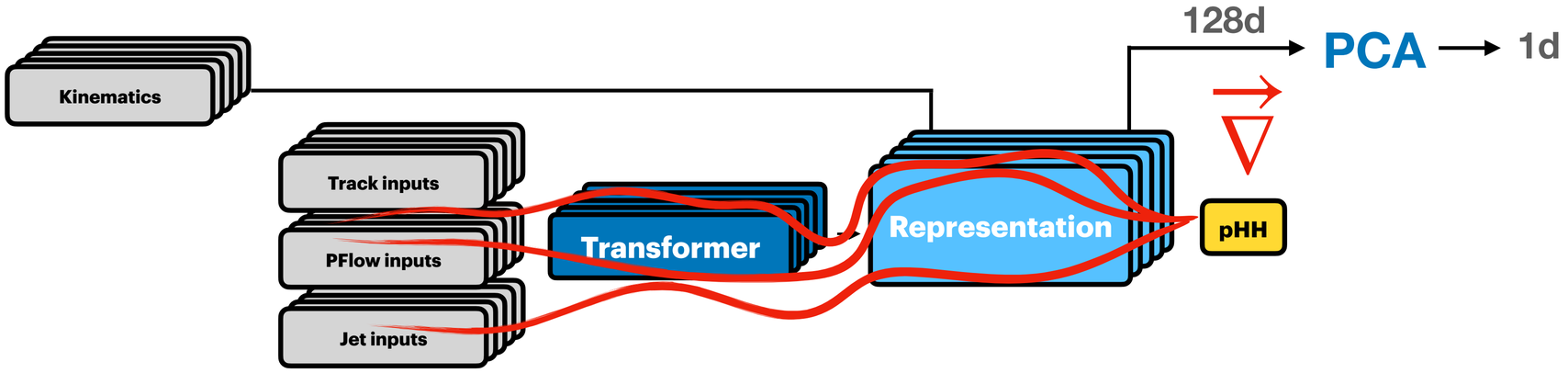
From scratch (end-to-end): slower (of course)
but eventually surpasses frozen backbone:
there's more than just jet tagging!

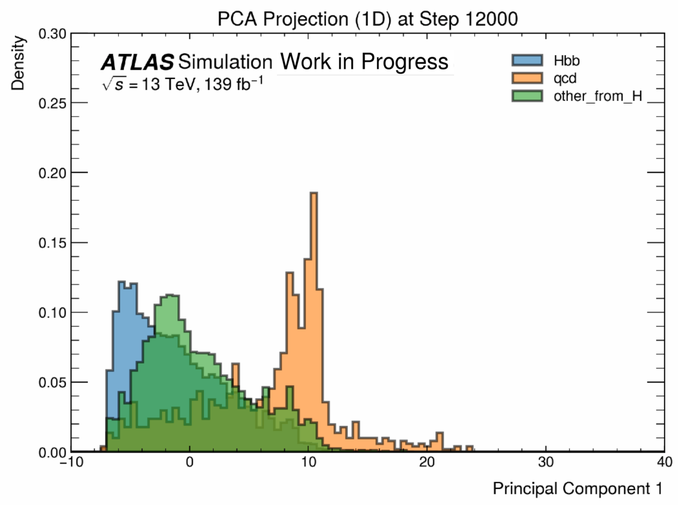
High dimensional representations are
more expressive

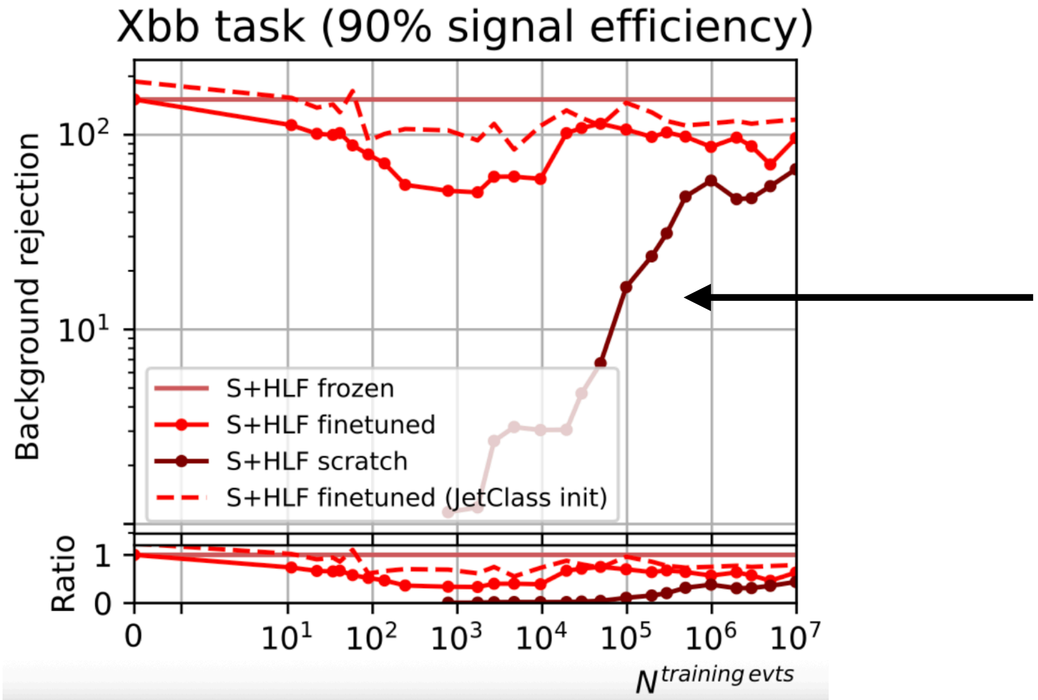
Fine-tuning and **more pre-training** helps

$S/\sqrt{B}$ **: increased significance by 40%**

Bonus: Jet tagging as emergent feature



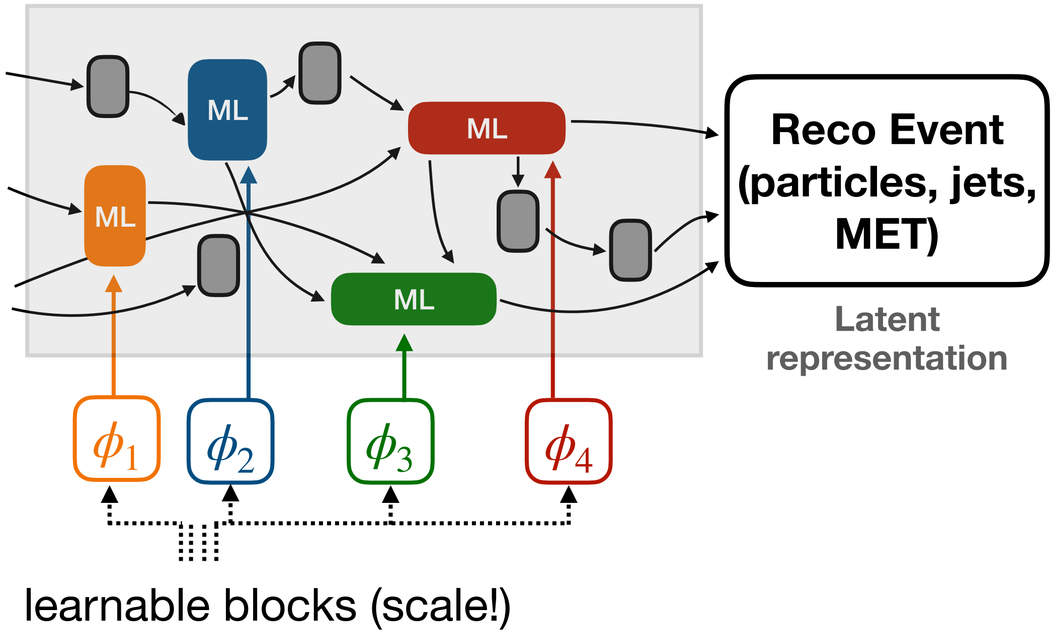




With enough data, the network figures out what sub-task (Higgs tagging) is crucial to solve the downstream physics goal

How far can we take this?

- Hierarchical set of reconstruction tasks in HEP pipeline, plus the analysis objective
  - We use ML for most of them
  - Each task will saturate (L_∞) at increasing scale as we go back in the pipeline



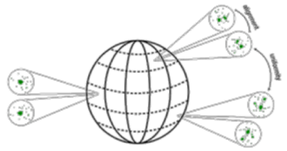
Can scale all of these tasks individually, but a unified approach is preferable

Are foundation models the future of HEP?

Towards Foundation Models in HEP

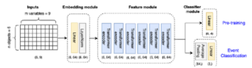
M. Kagan slides 20

Contrastive Learning: Symmetry Augmentation



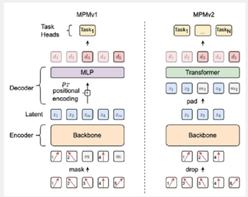
Dillon, Kasieczka, Olischlager
Plehn, Sorrenson, Vogel, 2108.04253

Masked Particle Type Prediction



Kishimoto, Morinaga, Saito
Tanaka, 2312.06909

Masked Particle Modeling



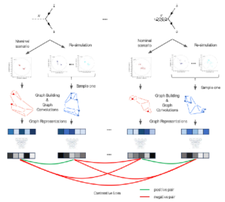
Golling, Heinrich, **MK**, Klein, Leigh, Osadchy, Raine, 2401.13537

Leigh, Klein, Charton, Golling, Heinrich, **MK**, Ochoa, Osadchy, 2409.12589

Next Token Predictoin

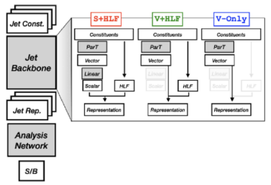
Birk, Hallin, Kasieczka, 2403.05618

Contrastive Learning: Re-Simulation



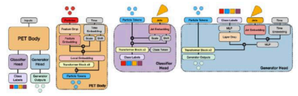
Harris, **MK**, Krupa, Maier, Woodward, 2403.07066

Supervised Pre-training and Joint Optimization



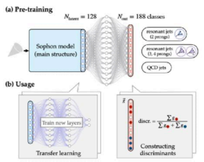
Vigl, Hartman, Heinrich, 2401.13536

Supervised Classification and Generation



Mikuni, Nachman 2404.16091

Large-Scale Fine-Grained Classification



Li, Li, et al. 2405.12972

- **Scalable (and high performance ceiling)**

  1. Low level inputs, complex task, scalable model

- **Justifies Simulation/Compute allocation**

  2. Useful for many downstream tasks

- **Scale alone is as important as architecture choice**

  3. Need to (Pre)-Train on 3-4 orders of magnitude more data

Fine-tuning machine-learned particle-flow reconstruction for new detector geometries in future colliders

Farouk Mokhtar,[1,*] Joosep Pata,[2,†] Dolores Garcia,[3] Eric Wulff,[3] Mengke Zhang,[4] Michael Kagan,[5] and Javier Duarte[4]

Towards a foundation model for heavy-ion collision experiments through point cloud diffusion

Manjunath Omana Kuttan,[1,2,*] Kai Zhou,[3,1,†] Jan Steinheimer,[4,1] and Horst Stoecker[1,4,5]

Aspen Open Jets: Unlocking LHC Data for Foundation Models in Particle Physics

Oz Amram,[1,*] Luca Anzalone,[2,3,†] Joschka Birk,[4,‡] Darius A. Faroughy,[5,§] Anna Hallin,[4,¶] Gregor Kasieczka,[4,6,‖] Michael Krämer,[7,**] Ian Pang,[5,††] Humberto Reyes-Gonzalez,[7,‡‡] and David Shih[5,§§]

Large Physics Models: Towards a collaborative approach with Large Language Models and Foundation Models

Kristian G. Barman*[1], Sascha Caron*[2], Emily Sullivan[3], Henk W. de Regt[4], Roberto Ruiz de Austri[5], Mieke Boon[6], Michael Färber[7], Stefan Fröse[8], Faegheh Hasibi[9], Andreas Ipp[10], Rukshak Kapoor[11], Gregor Kasieczka[12], Daniel Kostić[13], Michael Krämer[14], Tobias Golling[15], Luis G. Lopez[16], Jesus Marco[17], Sydney Otten[18,19], Pawel Pawlowski[1], Pietro Vischia[20], Erik Weber[1], and Christoph Weniger[21]

Agents of Discovery

Sascha Diefenbacher[1], Anna Hallin[2], Gregor Kasieczka[2], Michael Krämer[3], Anne Lauscher[4], Tim Lukas[2],

1 Physics Division, Lawrence Berkeley National Laboratory, Berkeley, USA
2 Institut für Experimentalphysik, Universität Hamburg, Germany
3 Institute for Theoretical Particle Physics and Cosmology, RWTH Aachen University, Germany
4 Data Science Group, Universität Hamburg, Germany

September 11, 2025

Particle Trajectory Representation Learning with Masked Point Modeling

Sam Young[1,2] Yeon-jae Jwa[2] Kazuhiro Terao[2]

Reconstructing hadronically decaying tau leptons with a jet foundation model

Laurits Tani[1,2*], Joosep Pata[1†] and Joschka Birk[3‡]

Pretrained Event Classification Model for High Energy Physics Analysis

Joshua Ho, Ryan Roberts, Shuo Han, and Haichen Wang
Department of Physics, University of California, Berkeley, CA 94720
Physics Division, Lawrence Berkeley National Laboratory, Berkeley CA 94720

Bumblebee: Foundation Model for Particle Physics Discovery

Andrew J. Wildridge — Department of Physics and Astronomy, Purdue University, West Lafayette, IN 47907, awildrid@purdue.edu
Jack P. Rodgers — Department of Physics and Astronomy, Purdue University, West Lafayette, IN 47907, jprodger@purdue.edu
Ethan M. Colbert — Department of Physics and Astronomy, Purdue University, West Lafayette, IN 47907, colberte@purdue.edu
Yao Yao — Department of Physics and Astronomy, Purdue University, West Lafayette, IN 47907, yao317@purdue.edu
Andreas W. Jung — Department of Physics and Astronomy, Purdue University, West Lafayette, IN 47907, anjung@purdue.edu
Miaoyuan Liu — Department of Physics and Astronomy, Purdue University, West Lafayette, IN 47907, liu3173@purdue.edu

Self-Supervised Learning Strategies for Jet Physics

Patrick Rieck[a,1], Kyle Cranmer[b], Etienne Dreyer[c,2], Eilam Gross[c], Nilotpal Kakati[c], Dmitrii Kobylanskii[c], Garrett W. Merz[b], Nathalie Soybelman[c]

FM4NPP: A Scaling Foundation Model for Nuclear and Particle Physics

David Park*[1], Shuhang Li*[2], Yi Huang*[1], Xihaier Luo[1], Haiwang Yu[2], Yeonju Go[2], Christopher Pinkenburg[2], Yuewei Lin[1], Shinjae Yoo[1], Joseph Osborn[2], Jin Huang[2], Yihui Ren[1]

OmniLearned: A Foundation Model Framework for All Tasks Involving Jet Physics

Wahid Bhimji,[1] Chris Harris,[1] Vinicius Mikuni,[2,*] and Benjamin Nachman[3,4,†]
[1] *NERSC, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA*
[2] *Nagoya University, Kobayashi-Maskawa Institute, Aichi 464-8602, Japan*
[3] *Department of Particle Physics and Astrophysics, Stanford University, Stanford, CA 94305, USA*
[4] *Fundamental Physics Directorate, SLAC National Accelerator Laboratory, Menlo Park, CA 94025, USA*

Conclusions

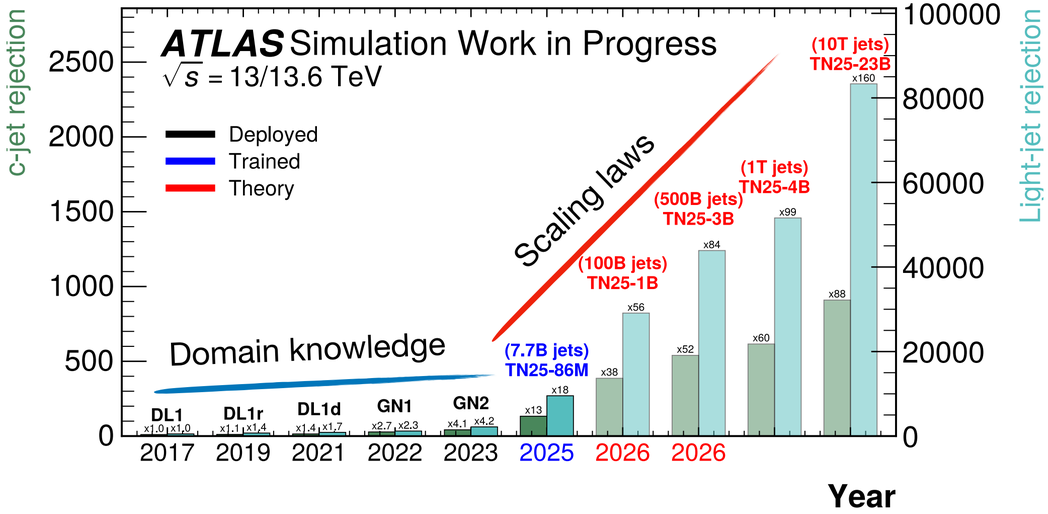
Large scale ML (as in industry frontier compute) is under explored in HEP

Clear potential from scaling current models, no need to change anything else

- We have more compute than we think
- Paves the way for large scale model deployment (IaaS) in LHC exp. frameworks

More ambitious:

- Drift towards **foundation-model pipeline**
- Brings scale to all downstream tasks, cross-experiment pre-training, end-to-end fine-tuning, fast analysis turnaround



Scale is the main driver of performance

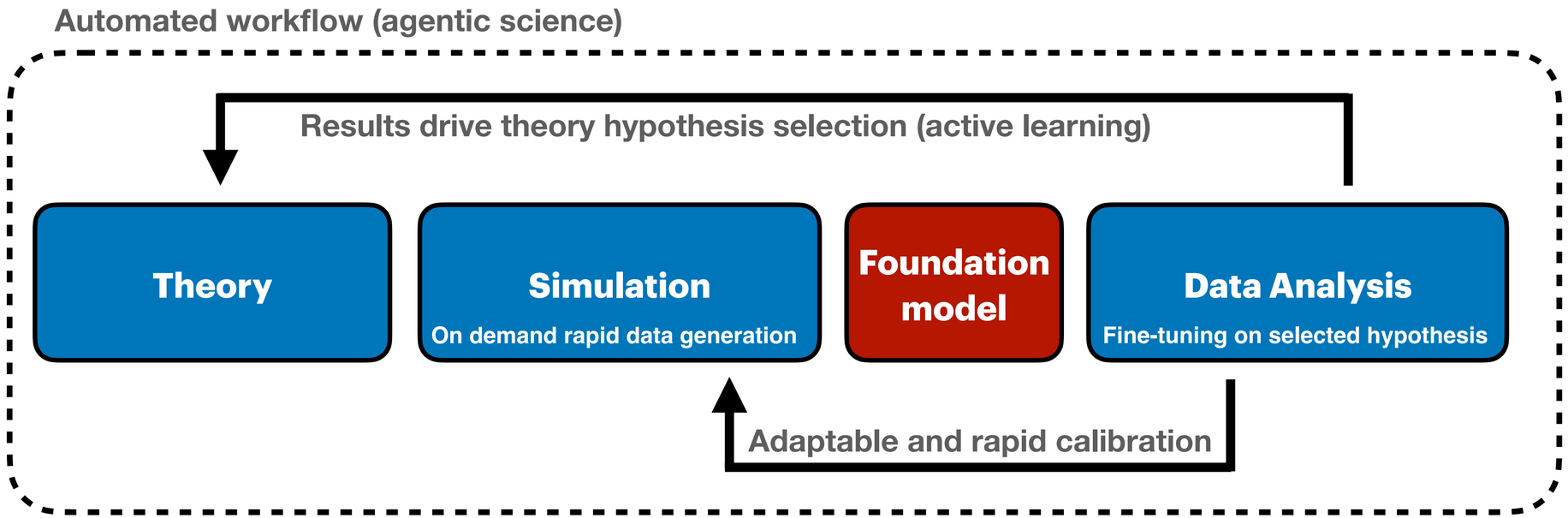
BACKUP

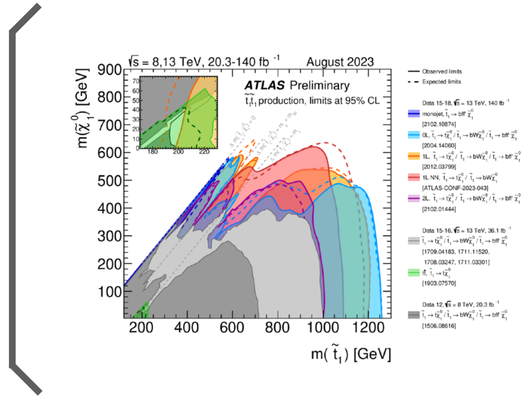
Are foundation models the future of HEP?

System to do rapid, optimal, and automated searches through lots of data and theory space

How would this look like? 🤔

Are foundation models the future of HEP?





System to do rapid, optimal, and automated searches through lots of data and theory space

Brute force: large-scale pre-trained Foundation Model → **Analysis as fine-tuning** → **Result** → **New hypothesis**

Reusable, adaptable and generalised ML tool

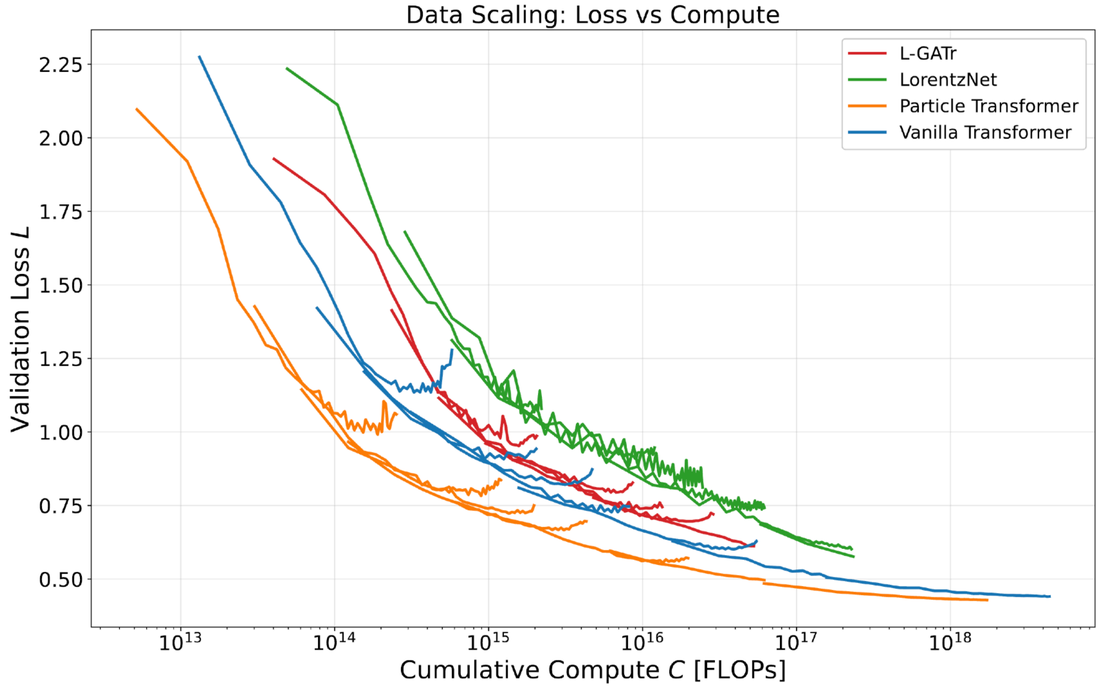
Fast turnaround

Stolen from M. Kagan

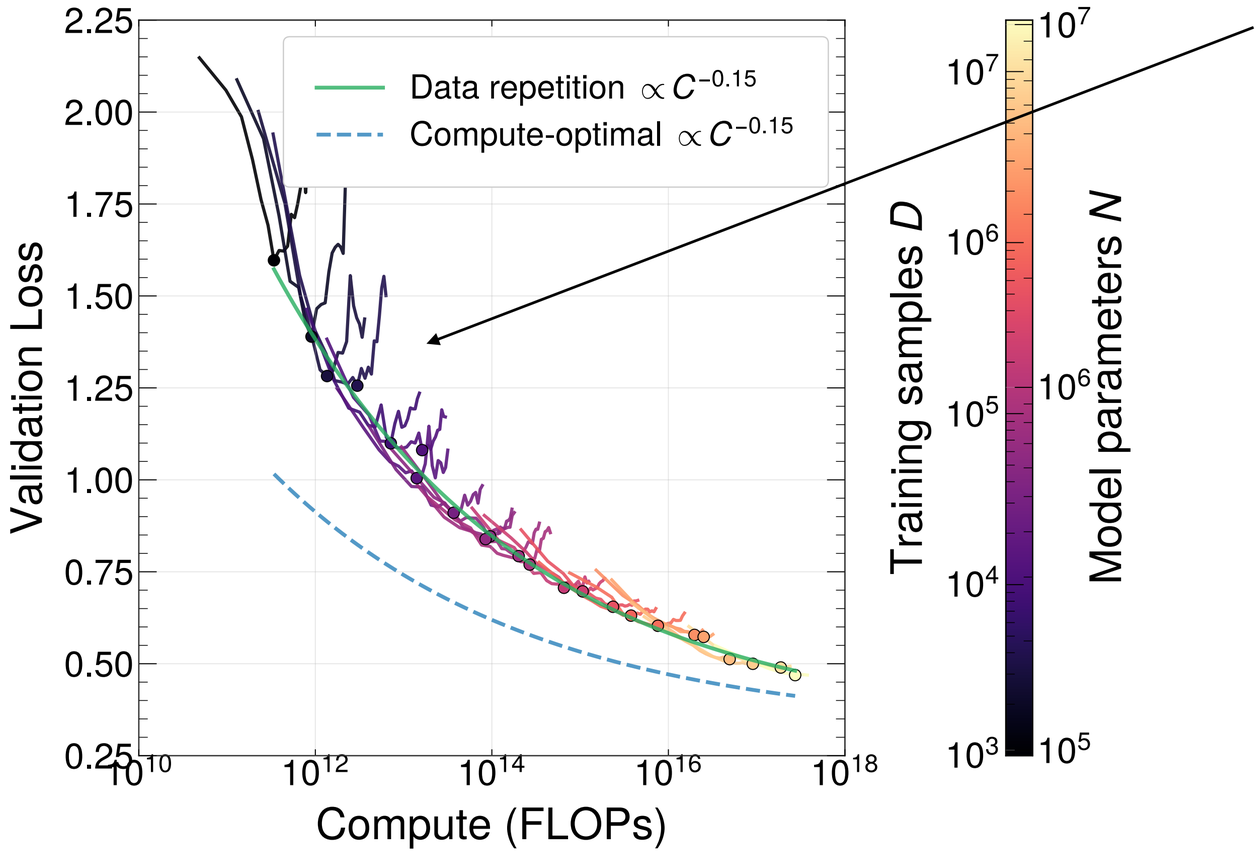
Agentic science: maybe not so far away?

I want to know which FTAG architecture I should deploy in ATLAS if I have a training budget of ~10^23 FLOPs. Here (/ftag_papers/*.pdf) are a few papers for models that enforce e.g. symmetries to gain performance at fixed dataset size, but they might be more compute expensive. Can you derive scaling laws as a function of compute and compare to a vanilla transformer, using the Jetclass dataset? Search the web for existing GitHub code, work autonomously and validate experiment runs when they're finished, if something breaks, fix it and continue. You have 8 H200 gpus at your disposable and 1 week to get results

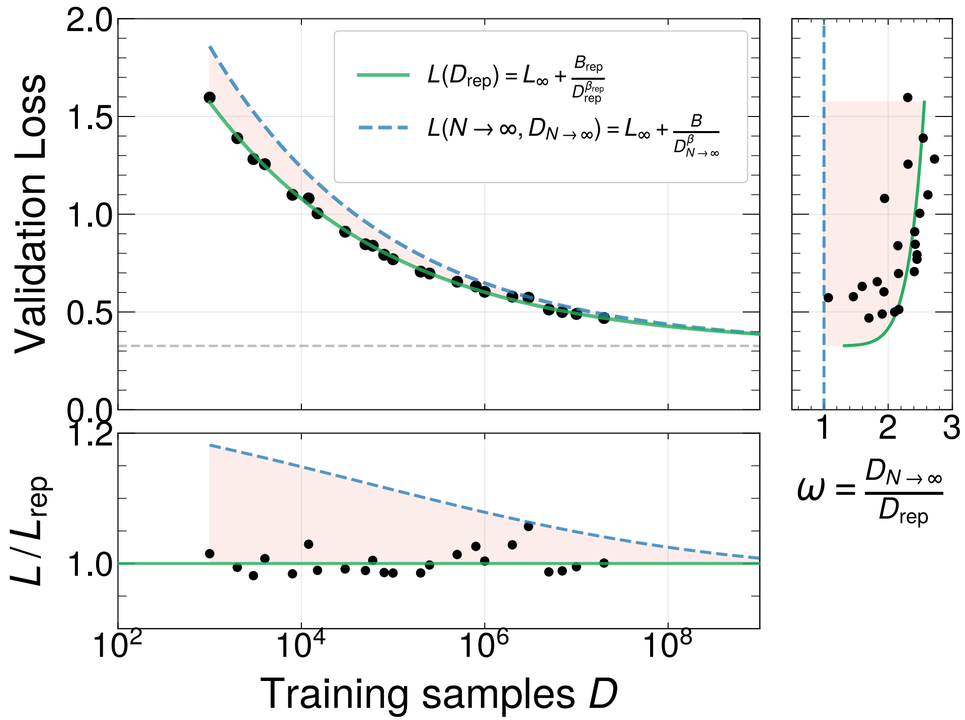
x8 NVIDIA H200
for ~1 week



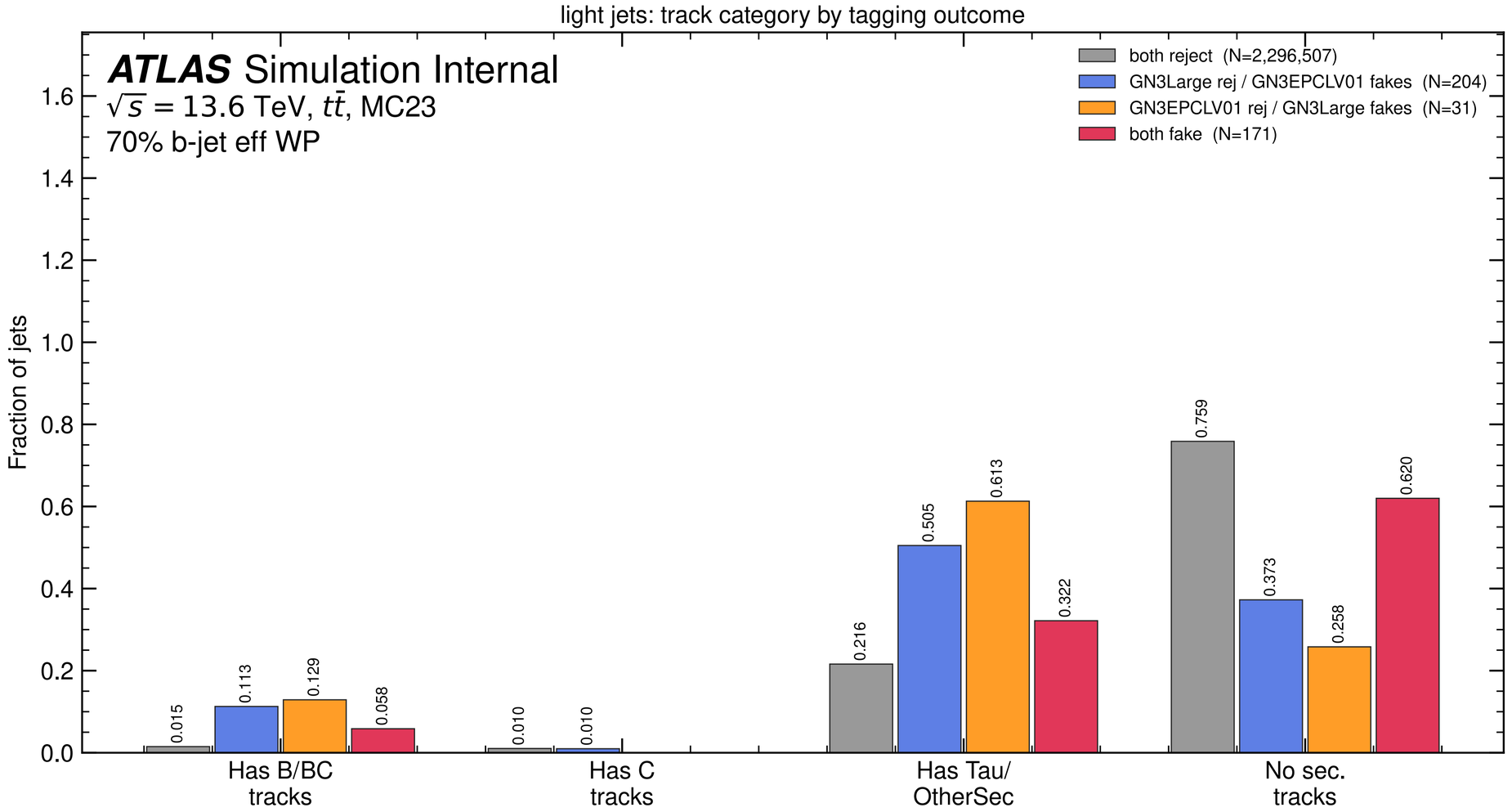
Data limited scaling



Data repetition is always compute sub-optimal



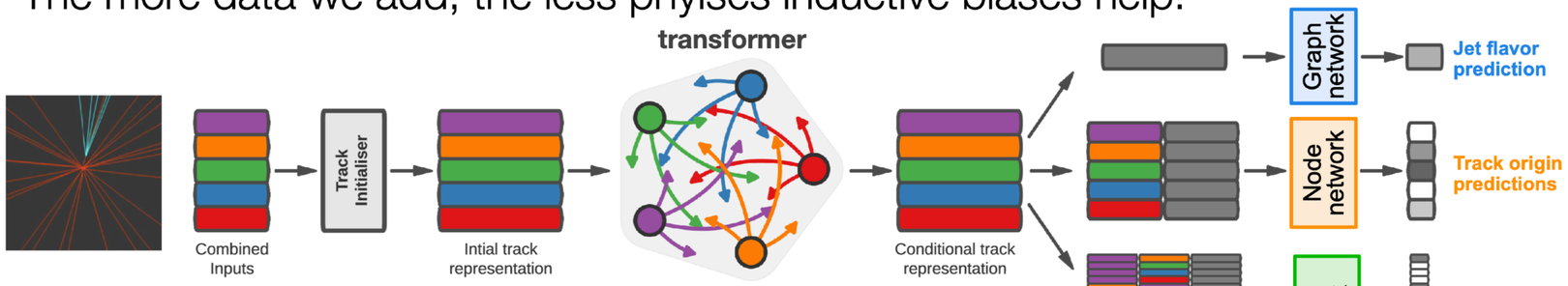
10x compute spent on data repetition can get us up to x3 effective Dataset size


light jets: track category by tagging outcome
ATLAS Simulation Internal
√s = 13.6 TeV, tt̄, MC23
70% b-jet eff WP
both reject (N=2,296,507)
GN3Large rej / GN3EPCLV01 fakes (N=204)
GN3EPCLV01 rej / GN3Large fakes (N=31)
both fake (N=171)
Fraction of jets
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
0.015
0.113
0.129
0.058
0.010
0.010
0.216
0.505
0.613
0.322
0.759
0.373
0.258
0.620
Has B/BC tracks
Has C tracks
Has Tau/ OtherSec
No sec. tracks

The bitter lesson

More data, less physics

The more data we add, the less phyiscs inductive biases help.



$$\mathscr{L}_{tot} = \mathscr{L}_{jet} + \alpha \mathscr{L}_{trk} + \beta \mathscr{L}_{vtx}$$

GN1 (2021): helped a lot

- 30m training jets
- 100% improvement

GN2 (2023): help a little

- 300m training jets
- 15% improvement

Alphafold 1

CNN + geometric modeling

High inductive bias

~20–30 M parameters

Alphafold 2

Transformer (Evoformer) + structure module

Medium inductive bias

~90–100 M parameters

Alphafold 3

Diffusion Transformer (multi-molecule)

Low inductive bias

~500–700 M parameters

Matthias Vigl

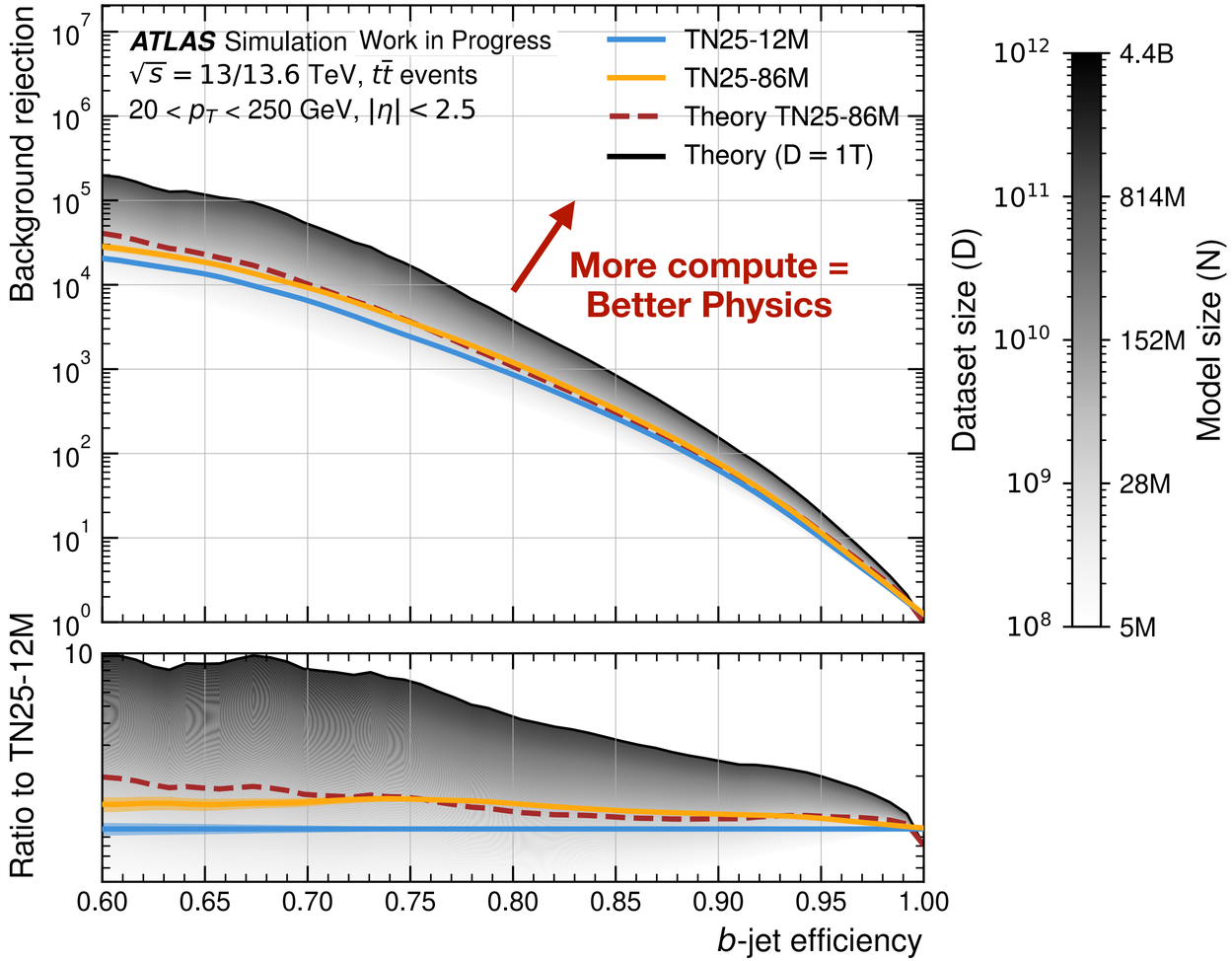
The bitter lesson

The Bitter Lesson

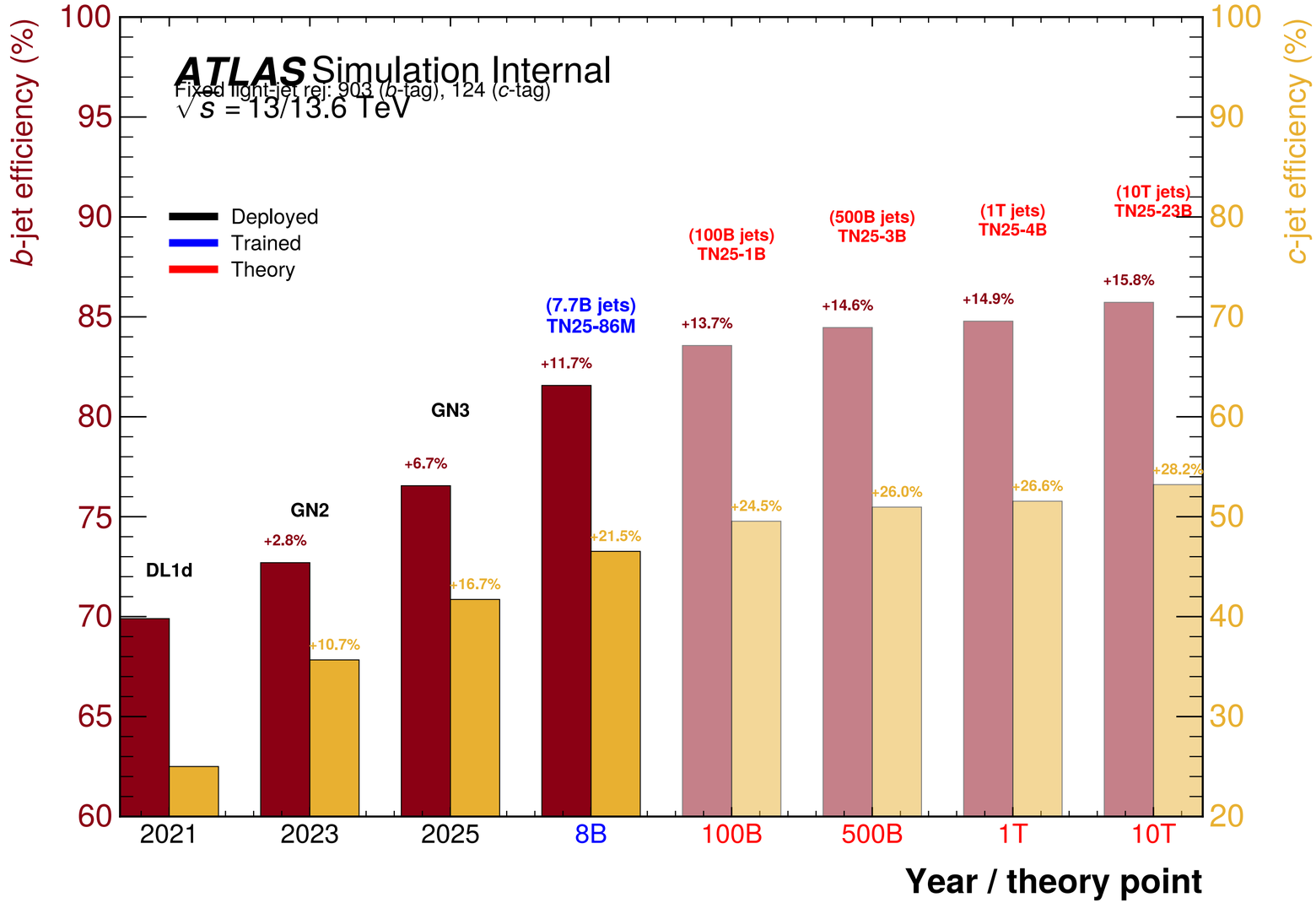
Rich Sutton

March 13, 2019

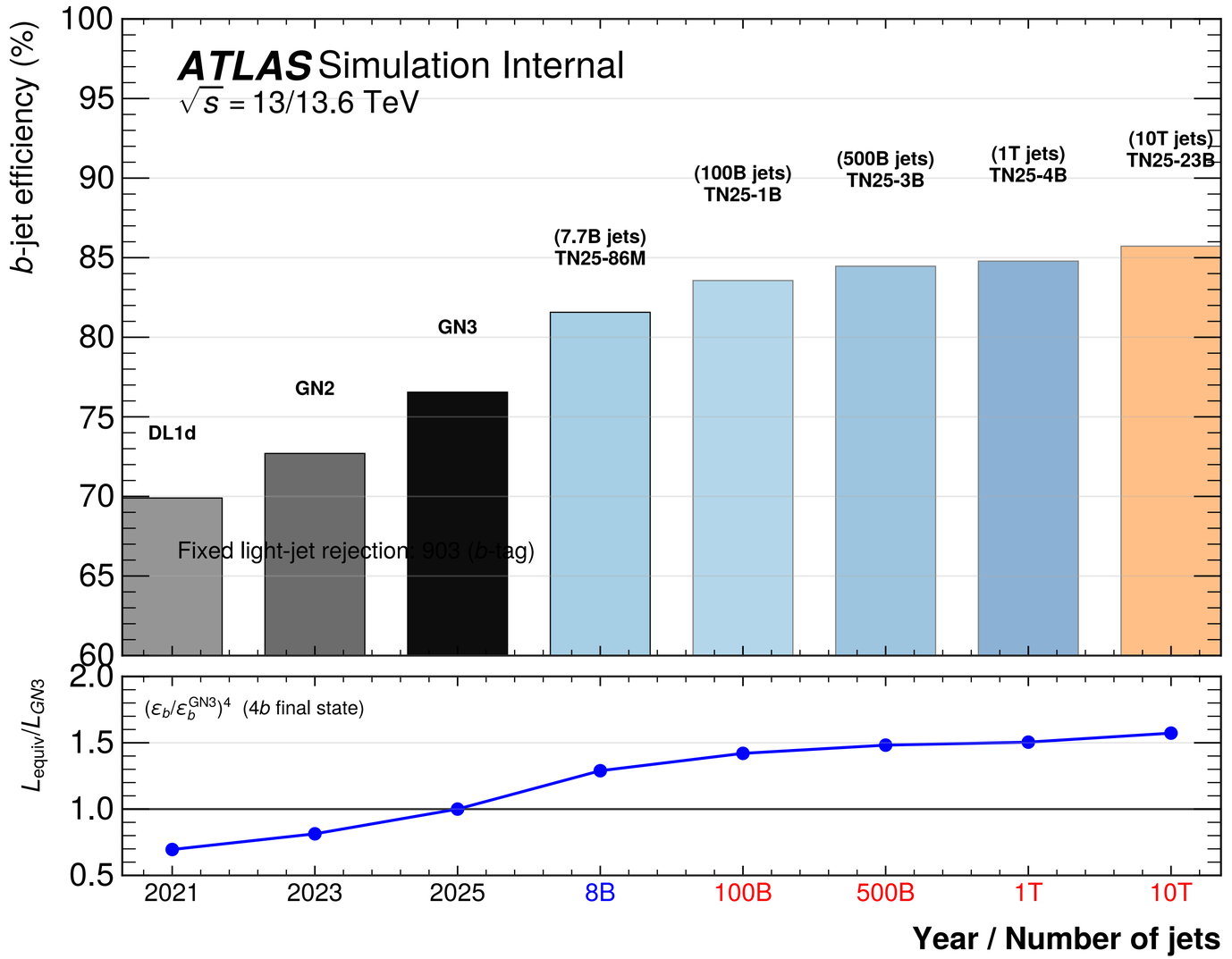
The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. The ultimate reason for this is Moore's law, or rather its



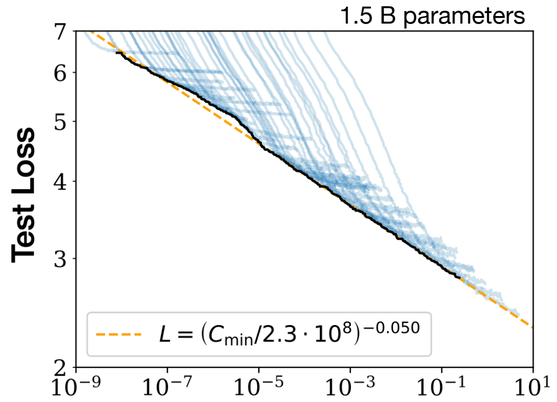
Impact of scale on HH(4b)



Impact of scale on HH(4b)



Validating neural scaling laws: a brief history



$\sim 10^8$ FLOPs

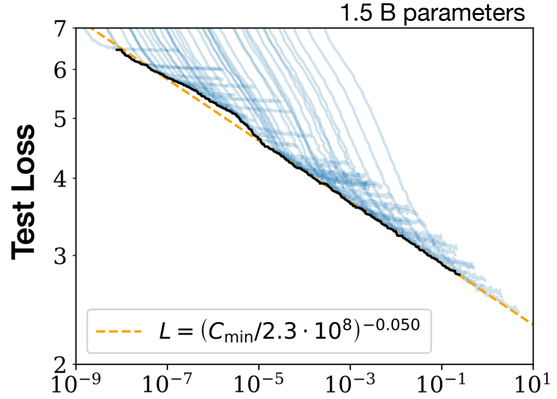
$\sim 8.6 \times 10^{19}$ FLOPs

33 NVIDIA Tesla V100

Jan 2020: 1st scaling laws paper

Validating neural scaling laws: a brief history

1.5 B parameters



$\sim 10^8$ FLOPs

$\sim 8.6 \times 10^{19}$ FLOPs

33 NVIDIA Tesla V100

Jan 2020: 1st scaling laws paper

🤝 Microsoft

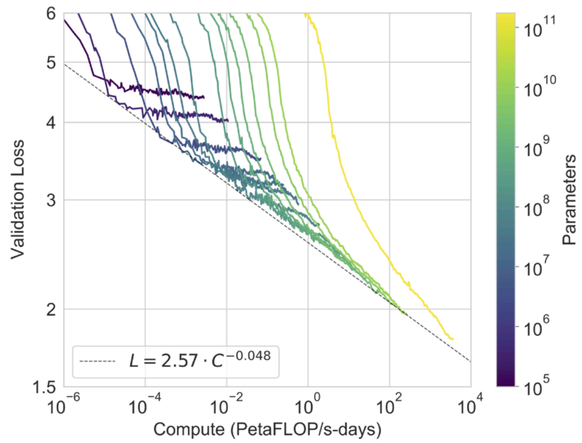
Bet: is it going to hold?

$\sim 10^{23}$ FLOPs

10000 NVIDIA Tesla V100

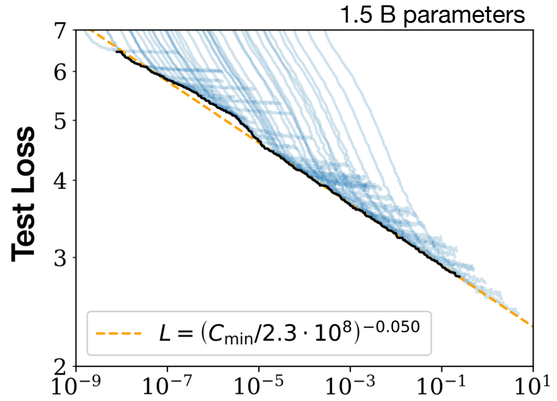
July 2020: GPT-3

175 B parameters



Still no flattening!

Validating neural scaling laws: a brief history



$\sim 10^8$ FLOPs

$\sim 8.6 \times 10^{19}$ FLOPs

33 NVIDIA Tesla V100

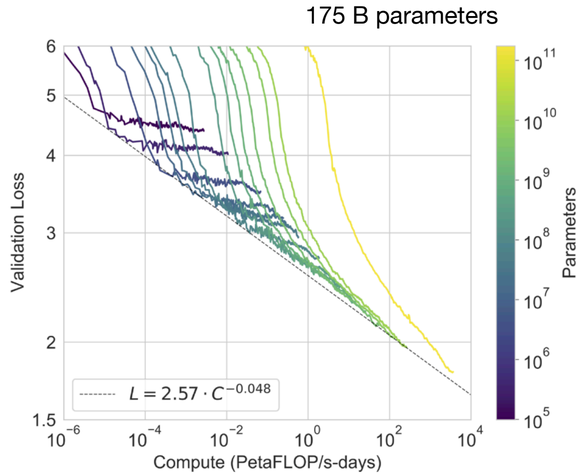
Jan 2020: 1st scaling laws paper

Microsoft

Bet: is it going to hold?

$\sim 10^{23}$ FLOPs

10000 NVIDIA Tesla V100



July 2020: GPT-3

Still no flattening!

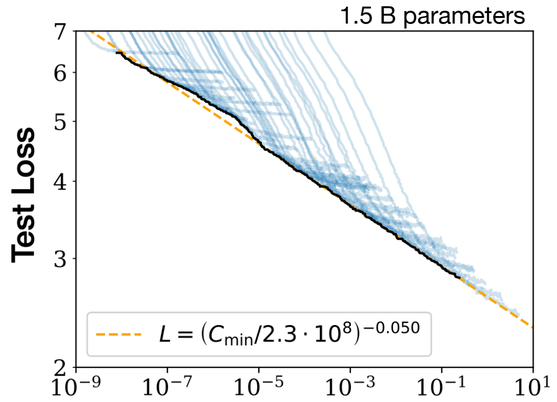
Google DeepMind

March 2022: Chinchilla compute-optimal scaling

$$L(N, D) = \boxed{L_\infty} + \frac{A}{N^\alpha} + \frac{B}{D^\beta}$$

Entropy (of language) as irreducible limit

Validating neural scaling laws: a brief history



$\sim 10^8$ FLOPs

Jan 2020: **1st scaling laws paper**

$\sim 8.6 \times 10^{19}$ FLOPs

33 NVIDIA Tesla V100

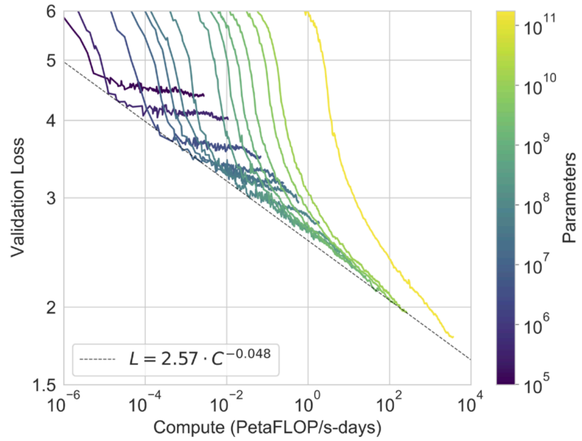
Microsoft

$\sim 10^{23}$ FLOPs

10000 NVIDIA Tesla V100

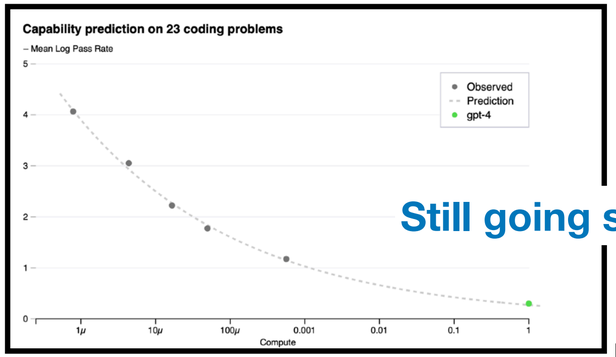
Bet: is it going to hold?

July 2020: **GPT-3**



Still no flattening!

March 2024: **GPT-4** $\sim 10^{25}$ FLOPs



Still going strong!

Google DeepMind

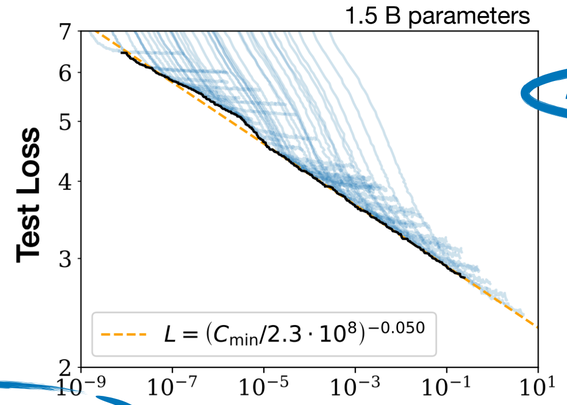
March 2022: **Chinchilla compute-optimal scaling**

$$L(N, D) = \boxed{L_\infty} + \frac{A}{N^\alpha} + \frac{B}{D^\beta}$$

Entropy (of language) as irreducible limit

Matthias Vigl

Validating neural scaling laws: a brief history



$\sim 10^8$ FLOPs

Jan 2020: 1st scaling laws paper

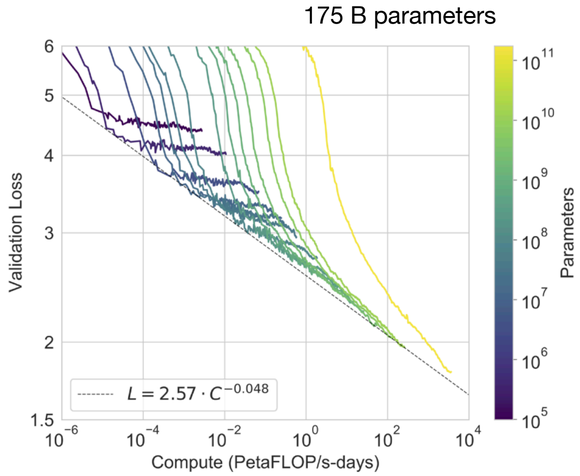
$\sim 8.6 \times 10^{19}$ FLOPs

33 NVIDIA Tesla V100

Bet: is it going to hold?

$\sim 10^{23}$ FLOPs

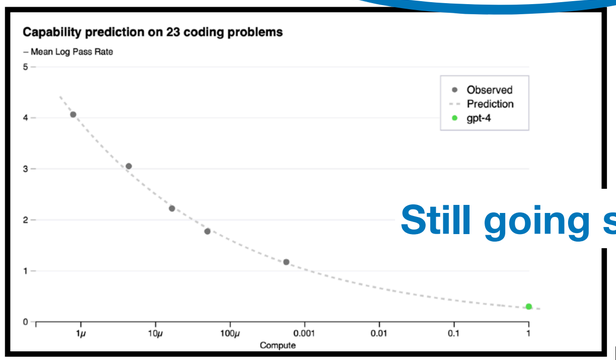
10000 NVIDIA Tesla V100



July 2020: GPT-3

Still no flattening!

March 2024: GPT-4 $\sim 10^{25}$ FLOPs



Still going strong!

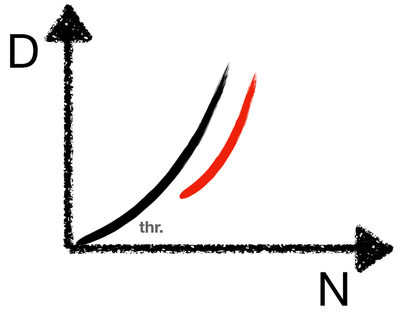
March 2022: Chinchilla compute-optimal scaling

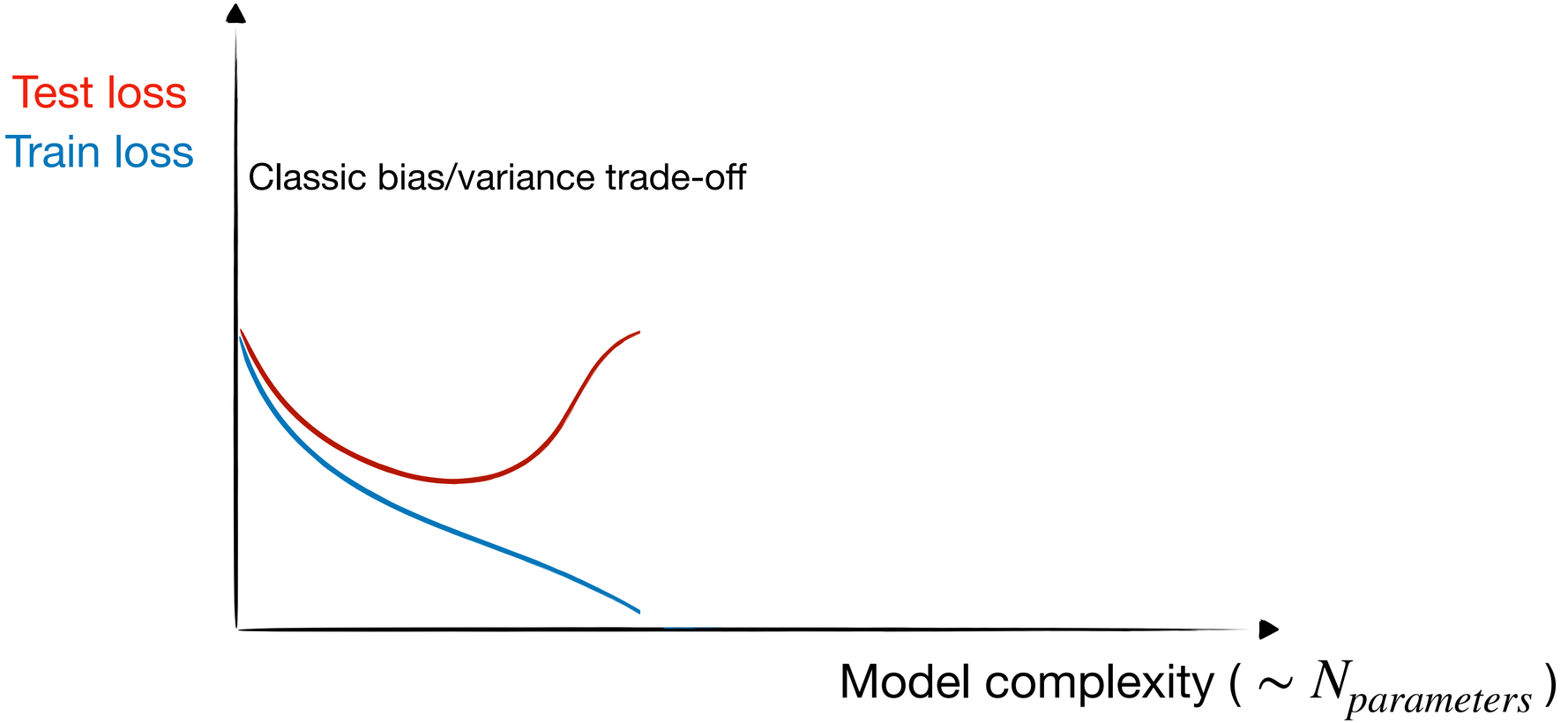
$$L(N, D) = \boxed{L_\infty} + \frac{A}{N^\alpha} + \frac{B}{D^\beta}$$

Entropy (of language) as irreducible limit

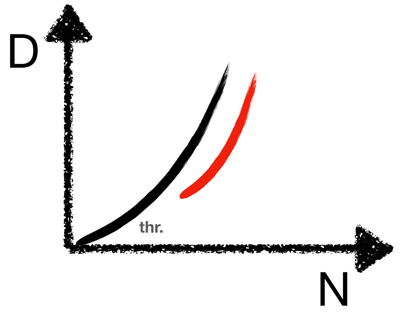
Matthias Vigl

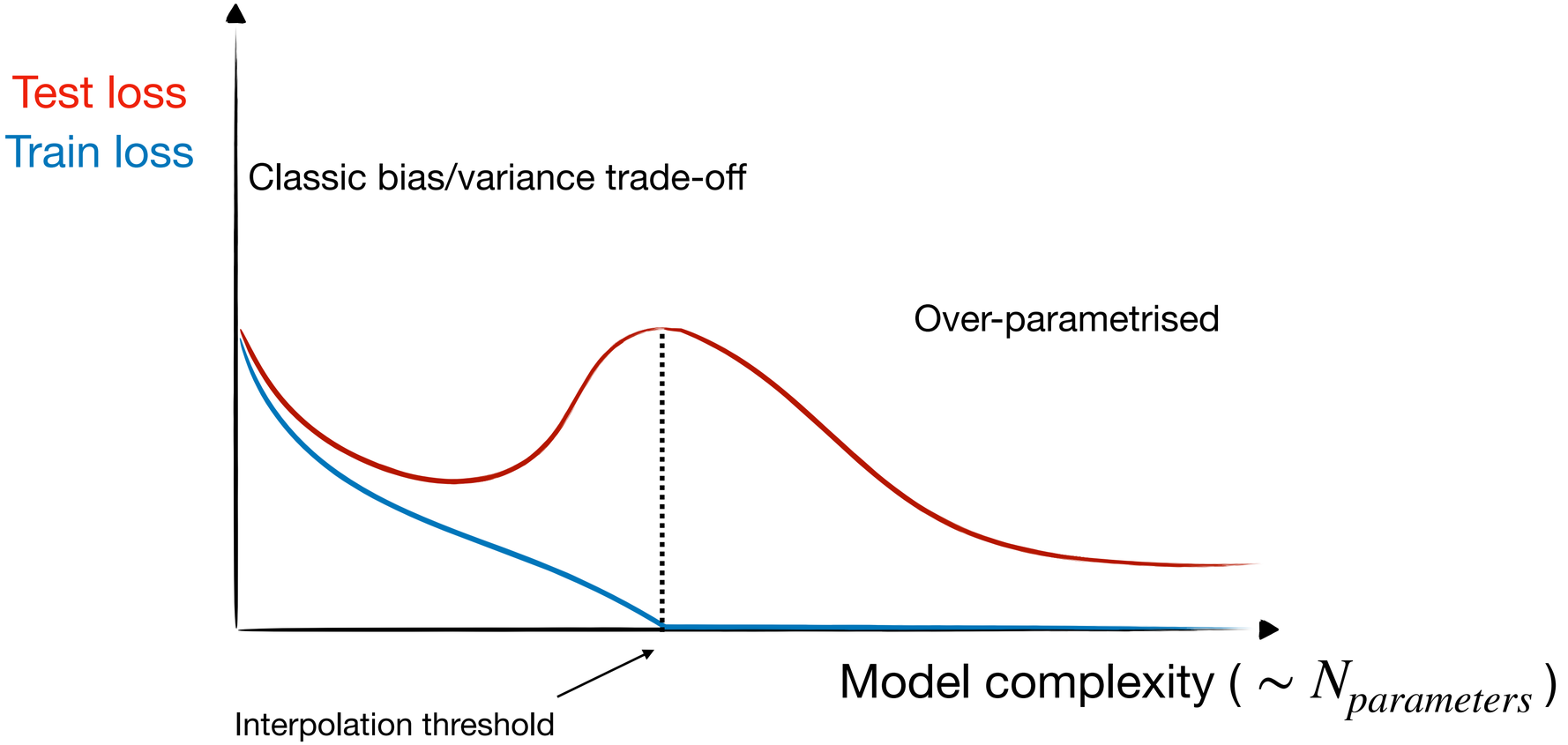
Double descent



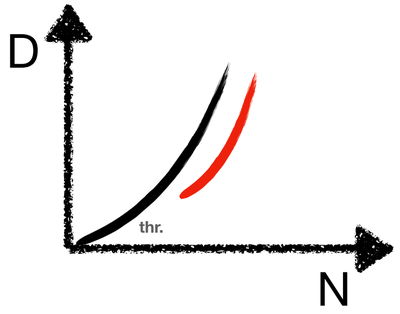


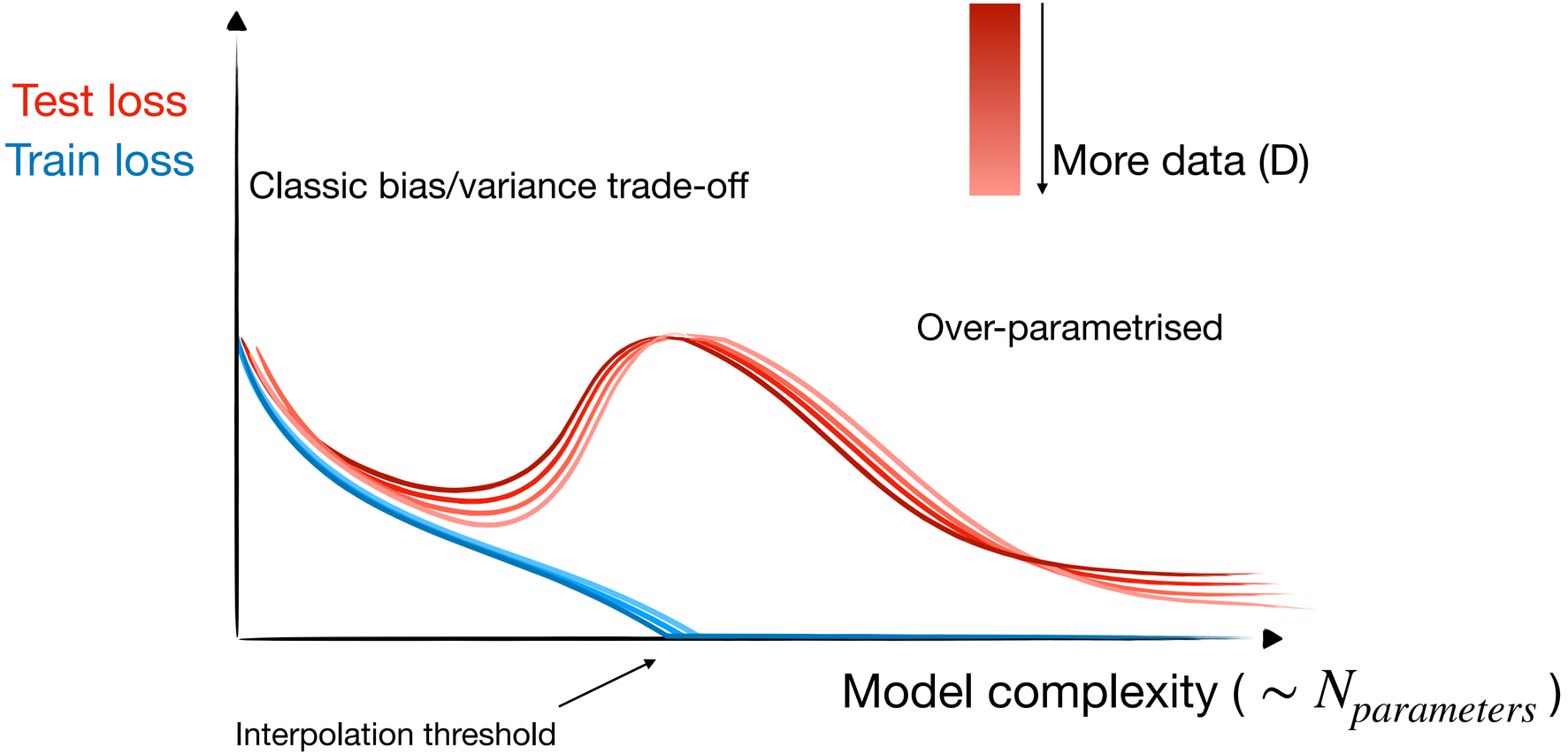
Double descent



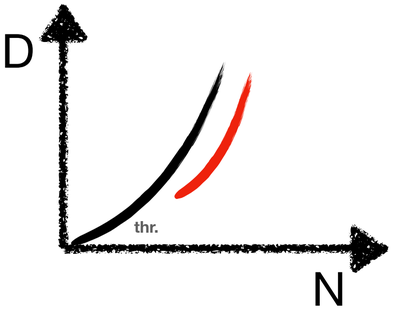


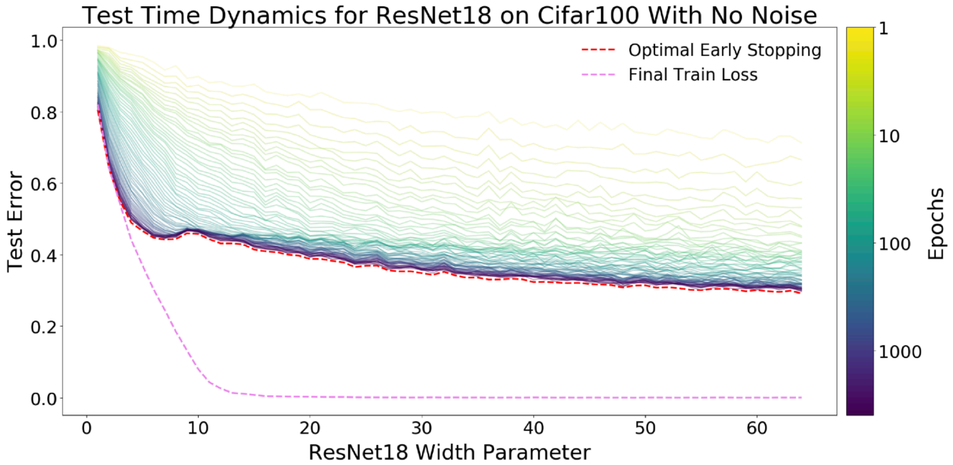
Double descent



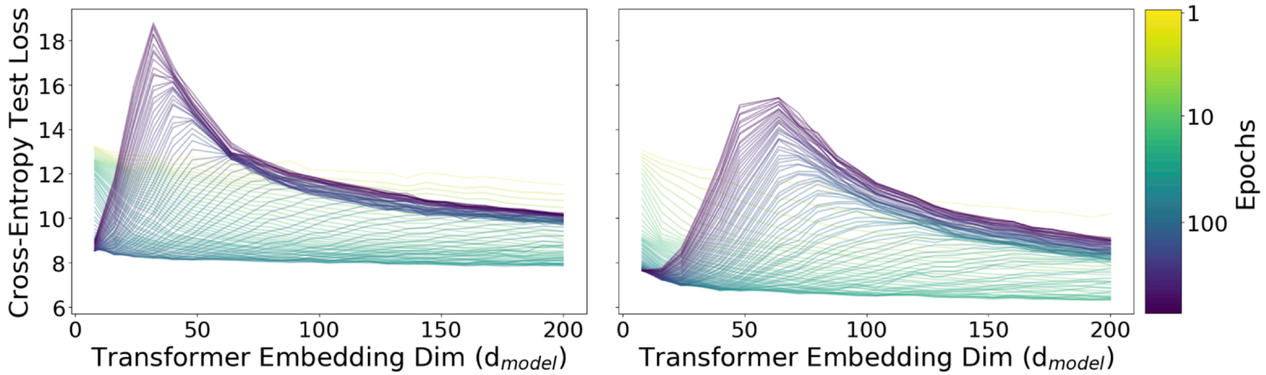


Double descent in modern ML

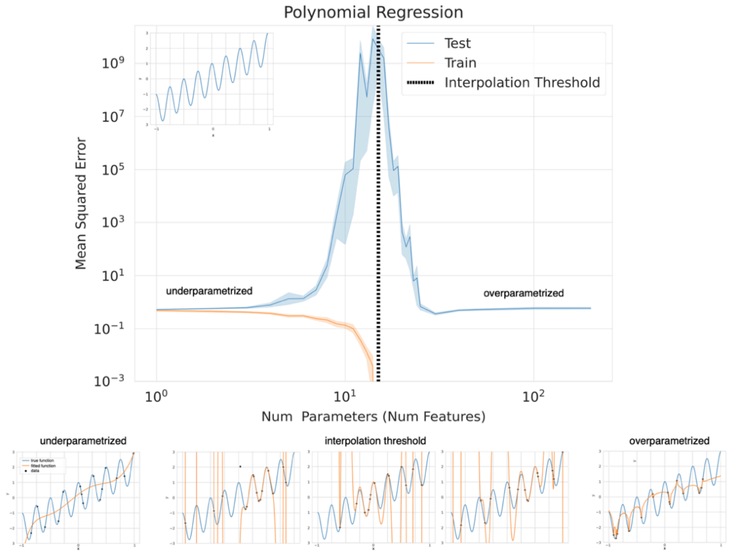




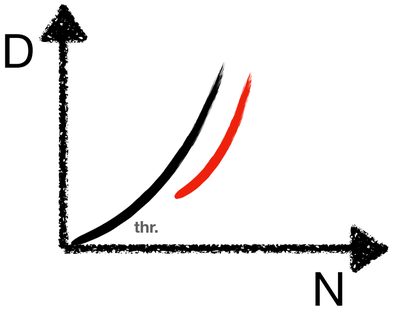
arXiv:1912.02292

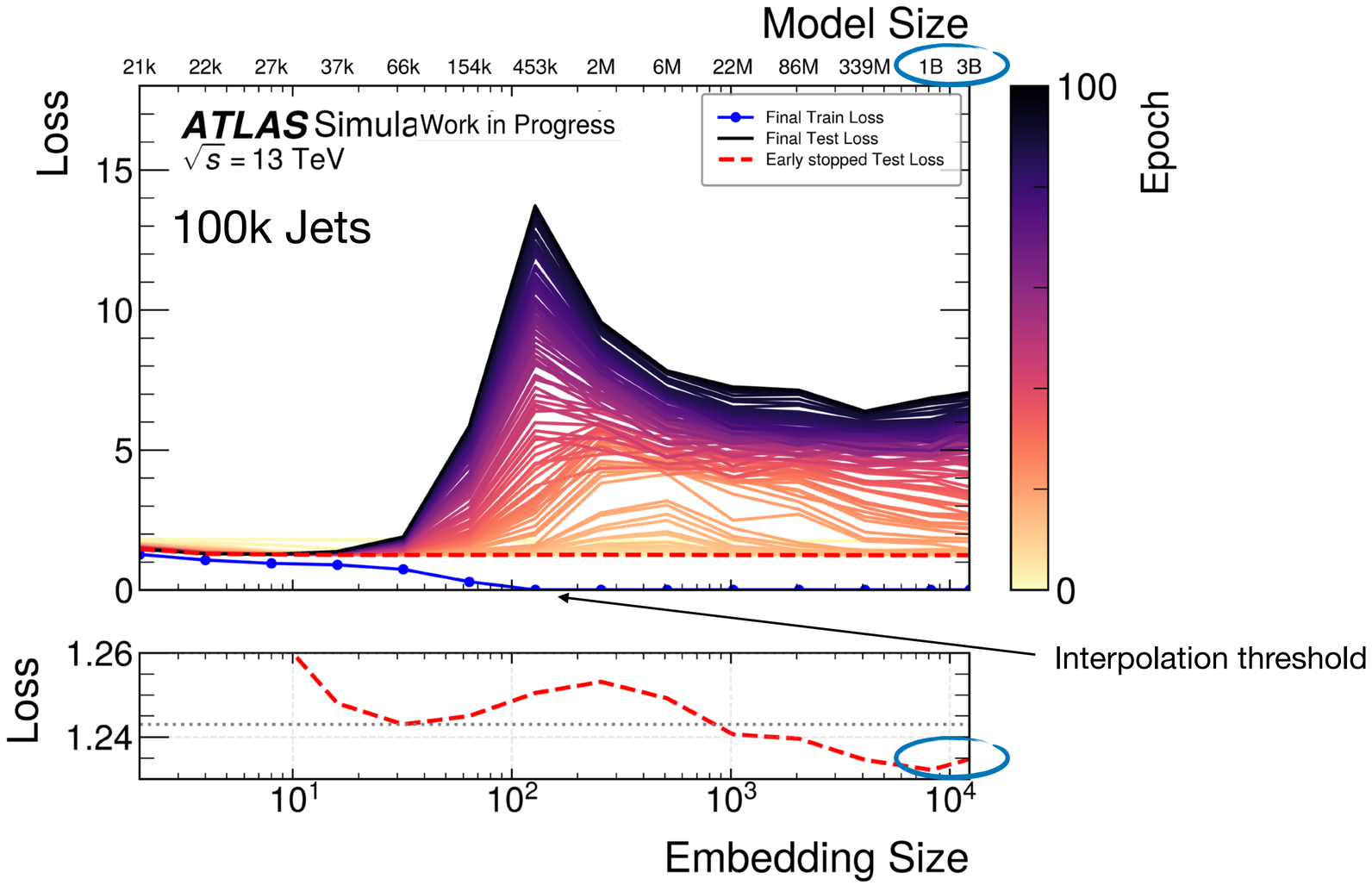


arXiv:2303.14151



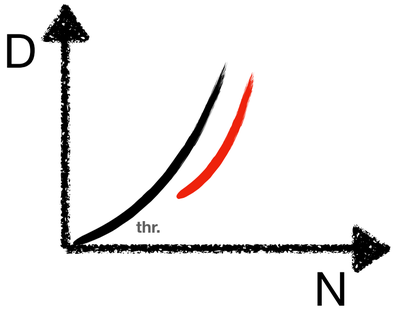
Double descent in jet tagging





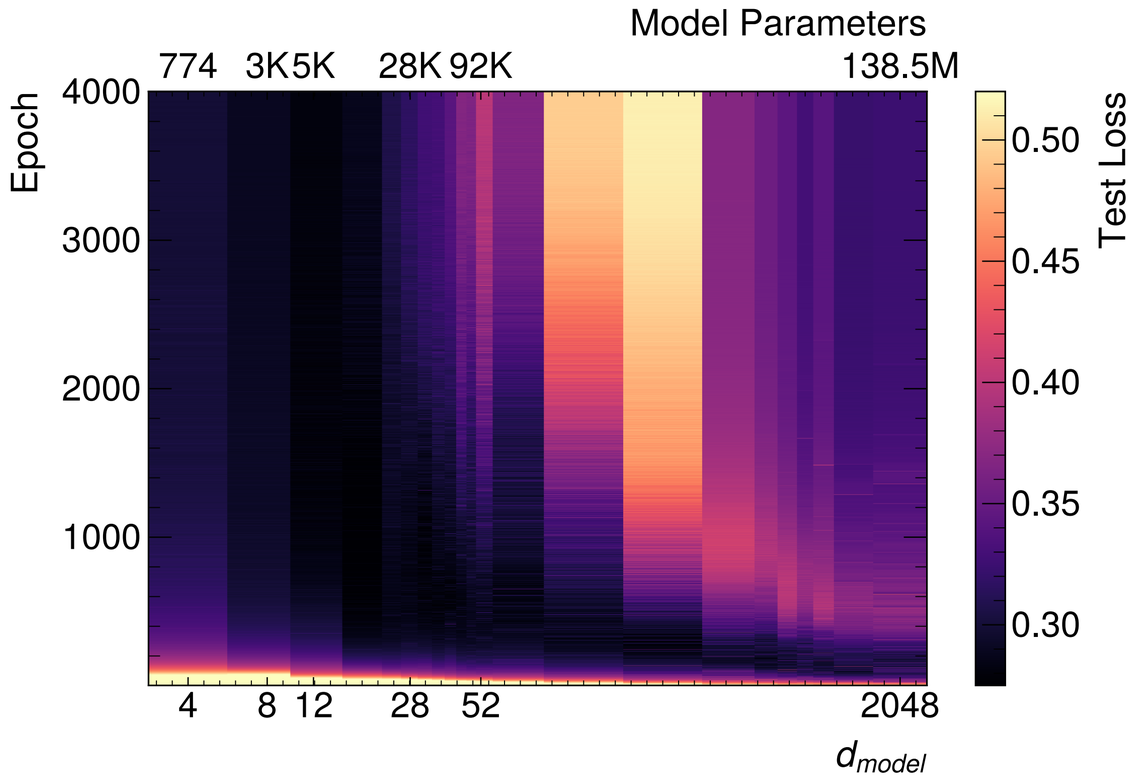
Jet pt regression

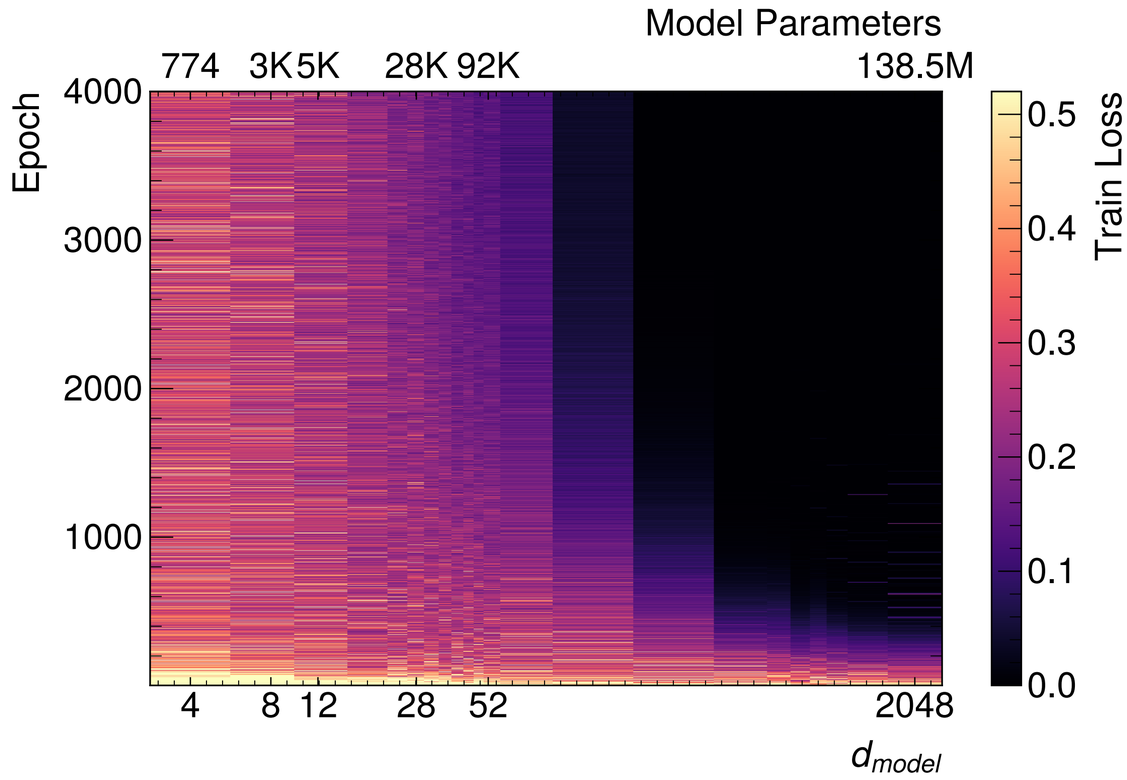
Spin-off



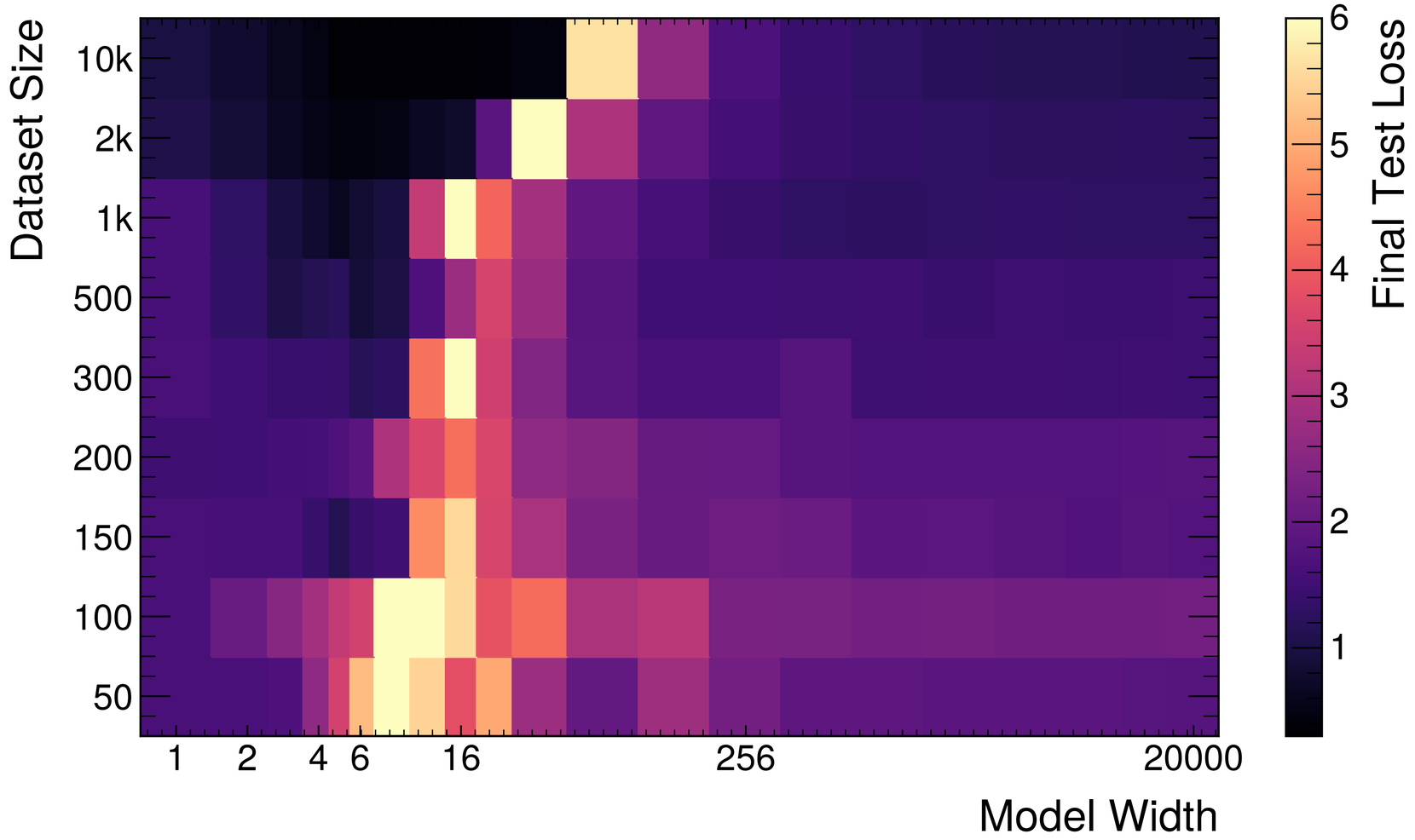
Model wise DD

Epoch wise DD

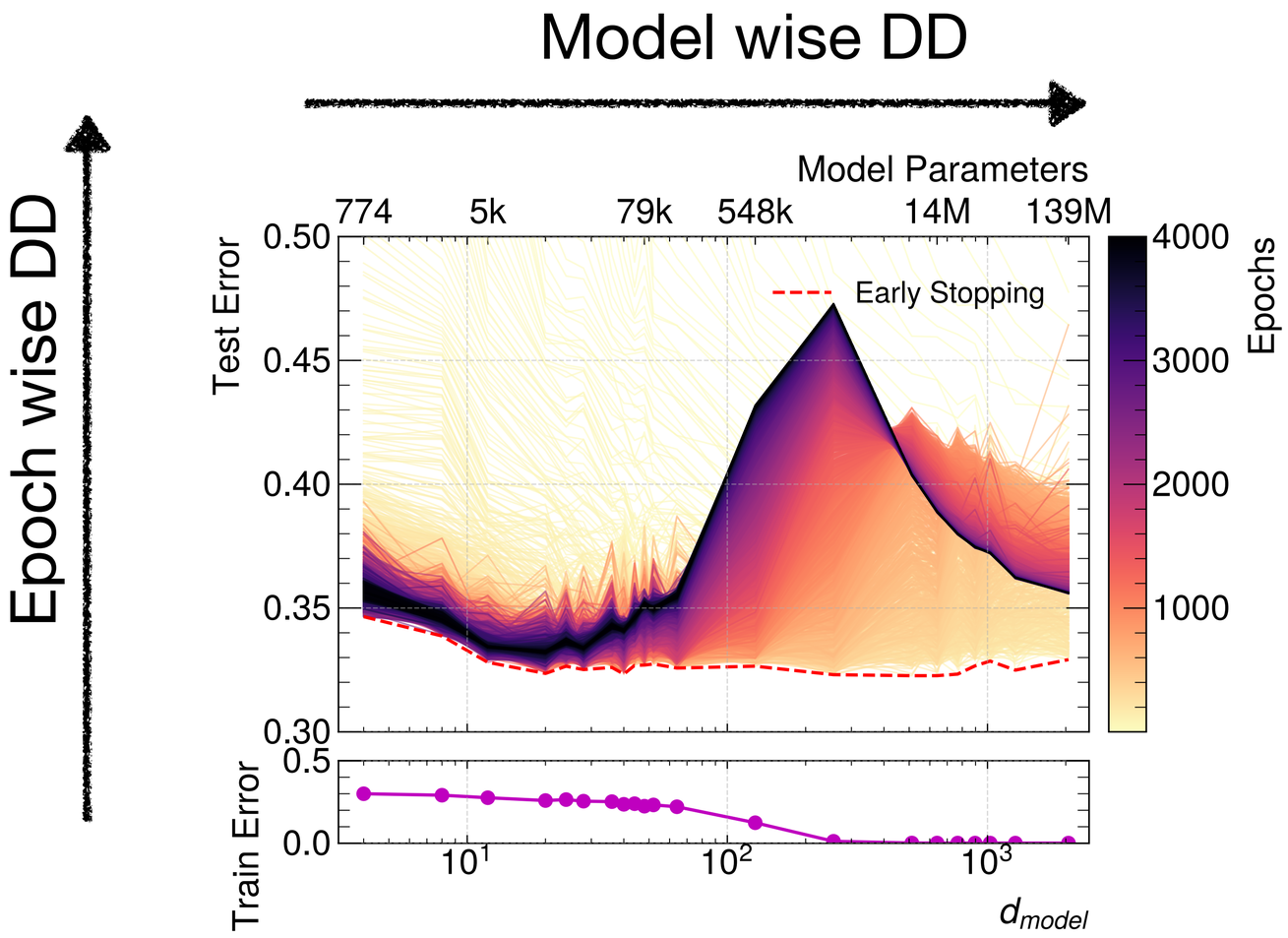


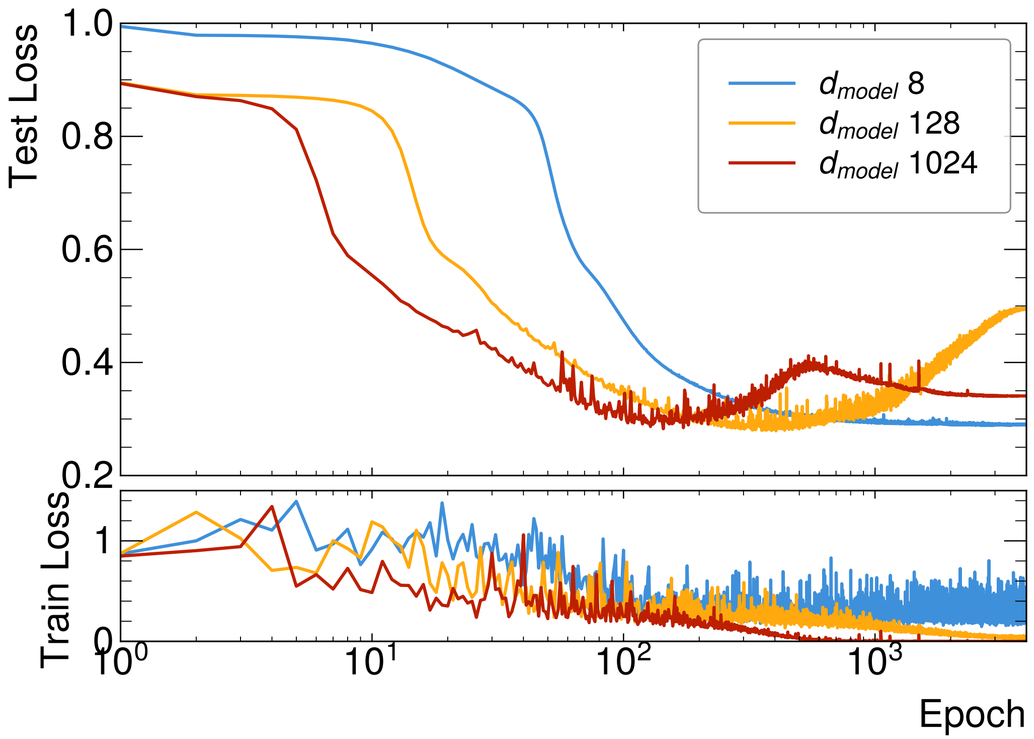


SUSY vs SM classification



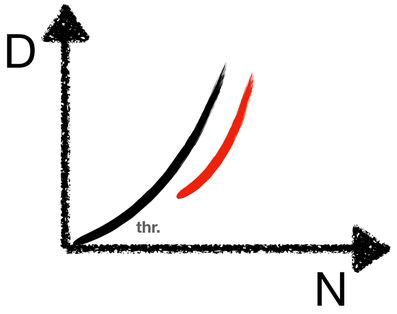
Jet pt regression



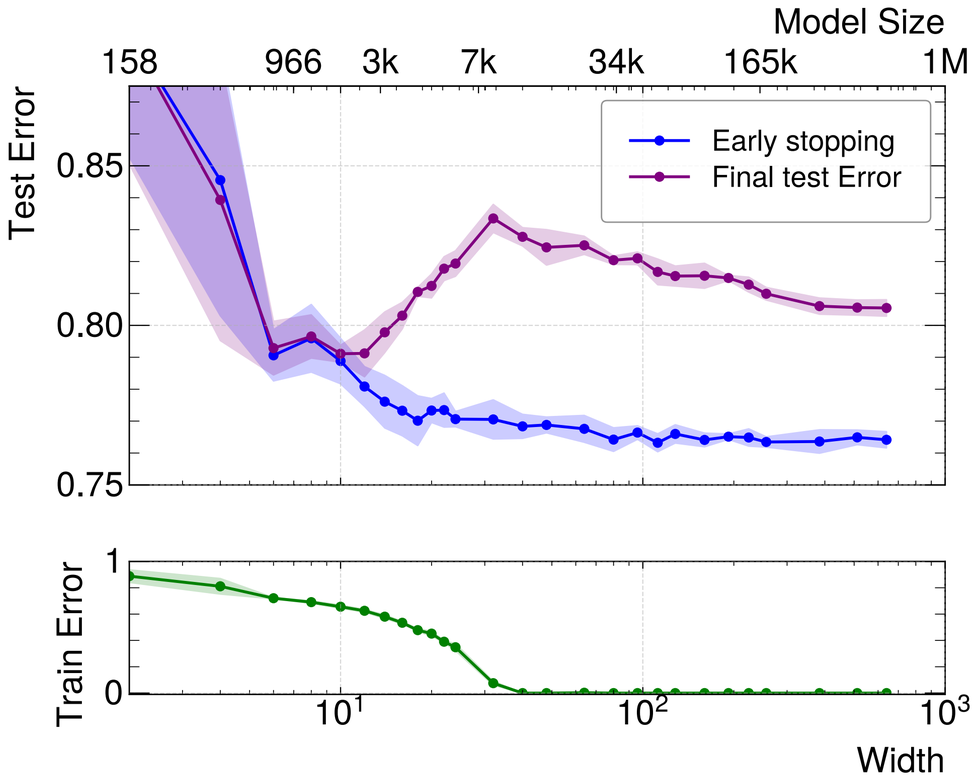


SUSY vs SM classification

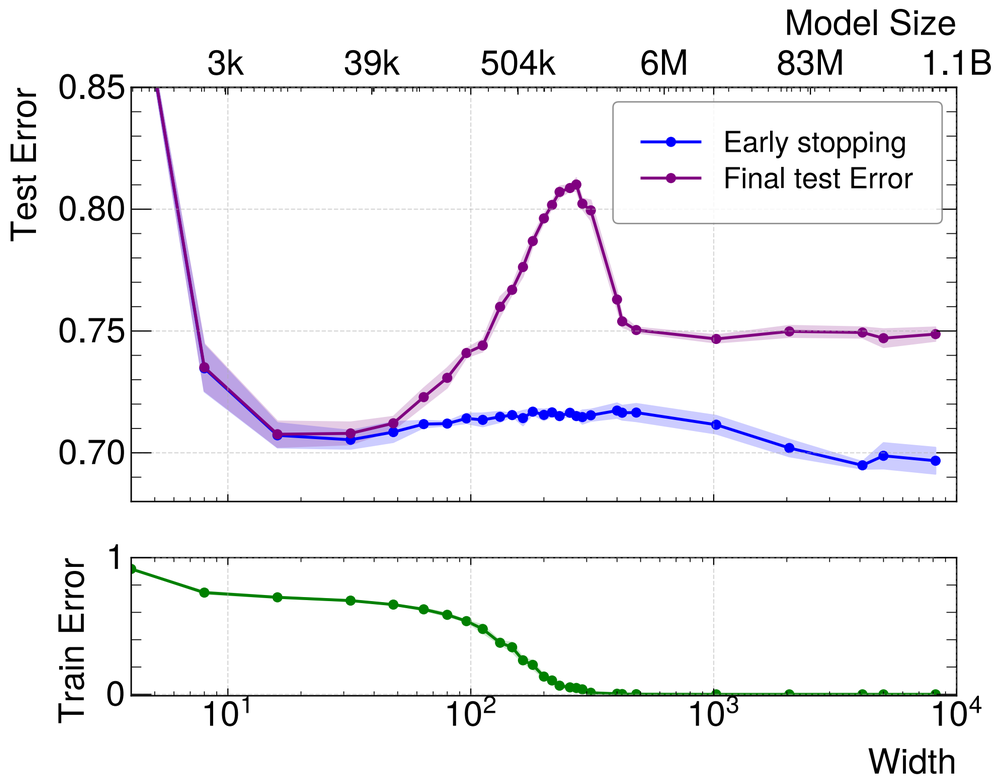
Spin-off



Early stopping can show double descent

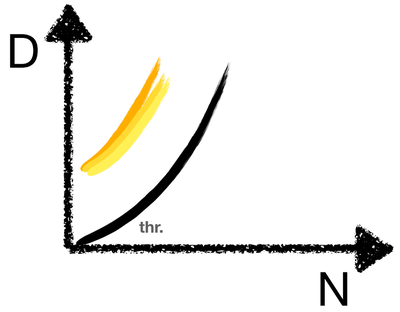


3k events



150k events

Complete cost model



$$C = 6ND + kD + mN$$

Training cost

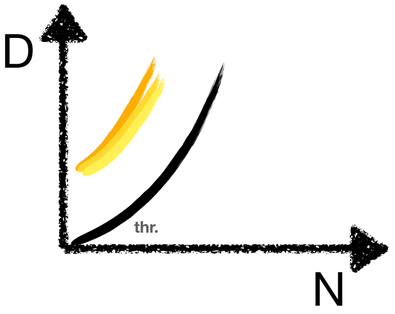
Dataset creation cost

+ preprocessing / validation

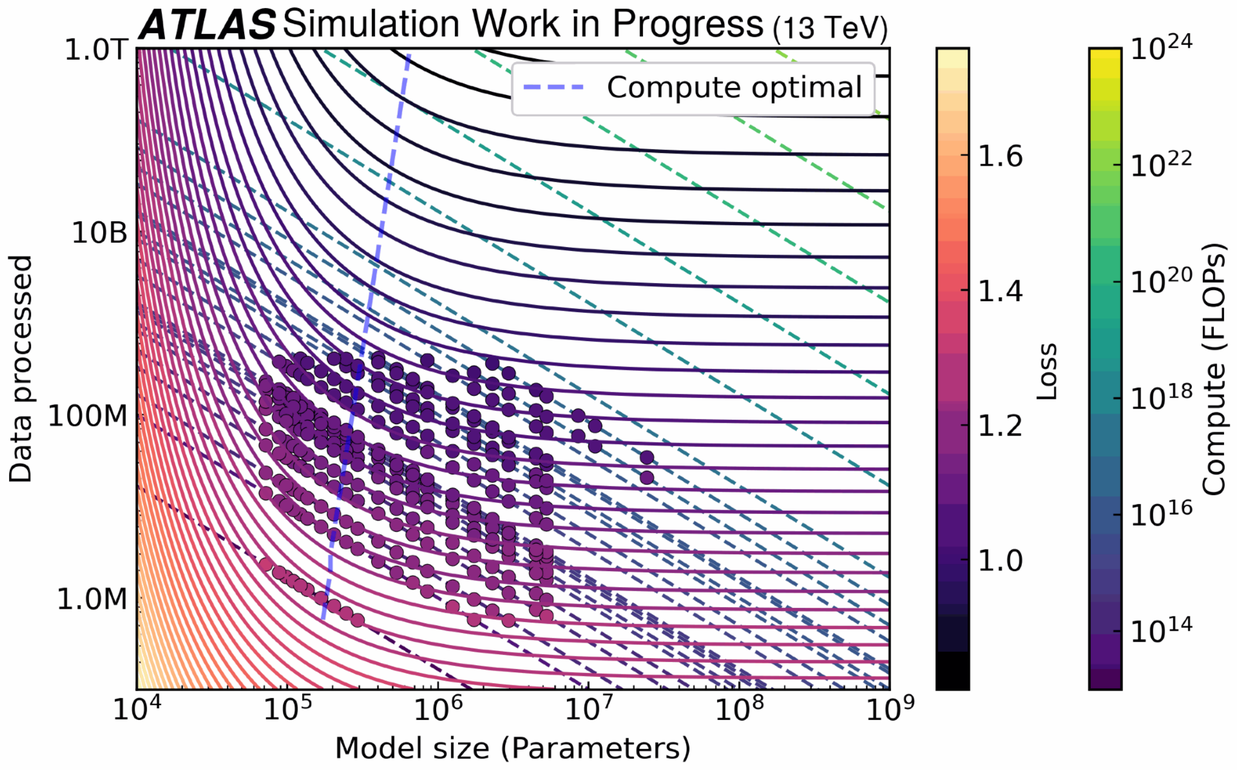
Inference cost

$$\min_{N,D} L(N, D) \qquad \text{s.t.} \quad C = 6ND + kD + mN = \text{const.}$$

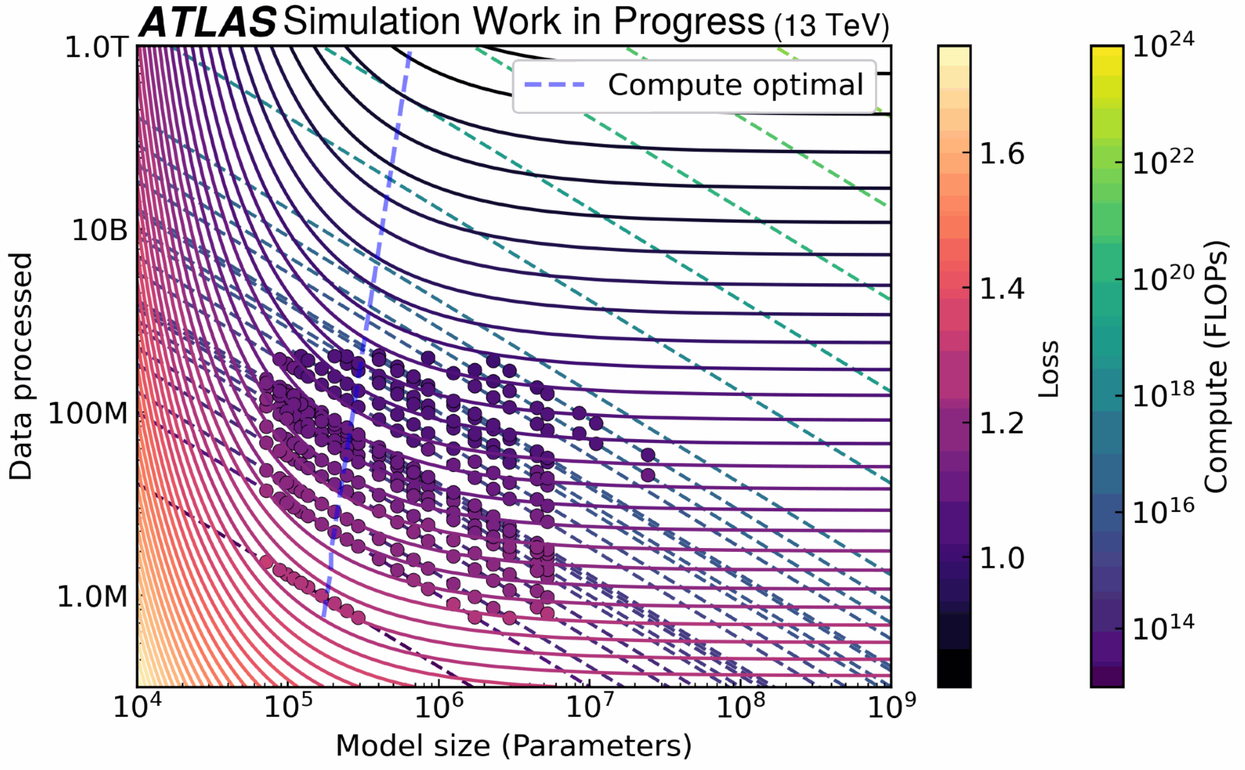
Compute optimal trajectories



$k \uparrow$

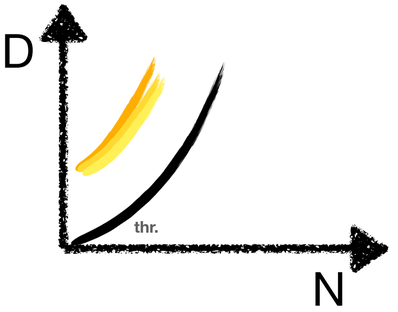


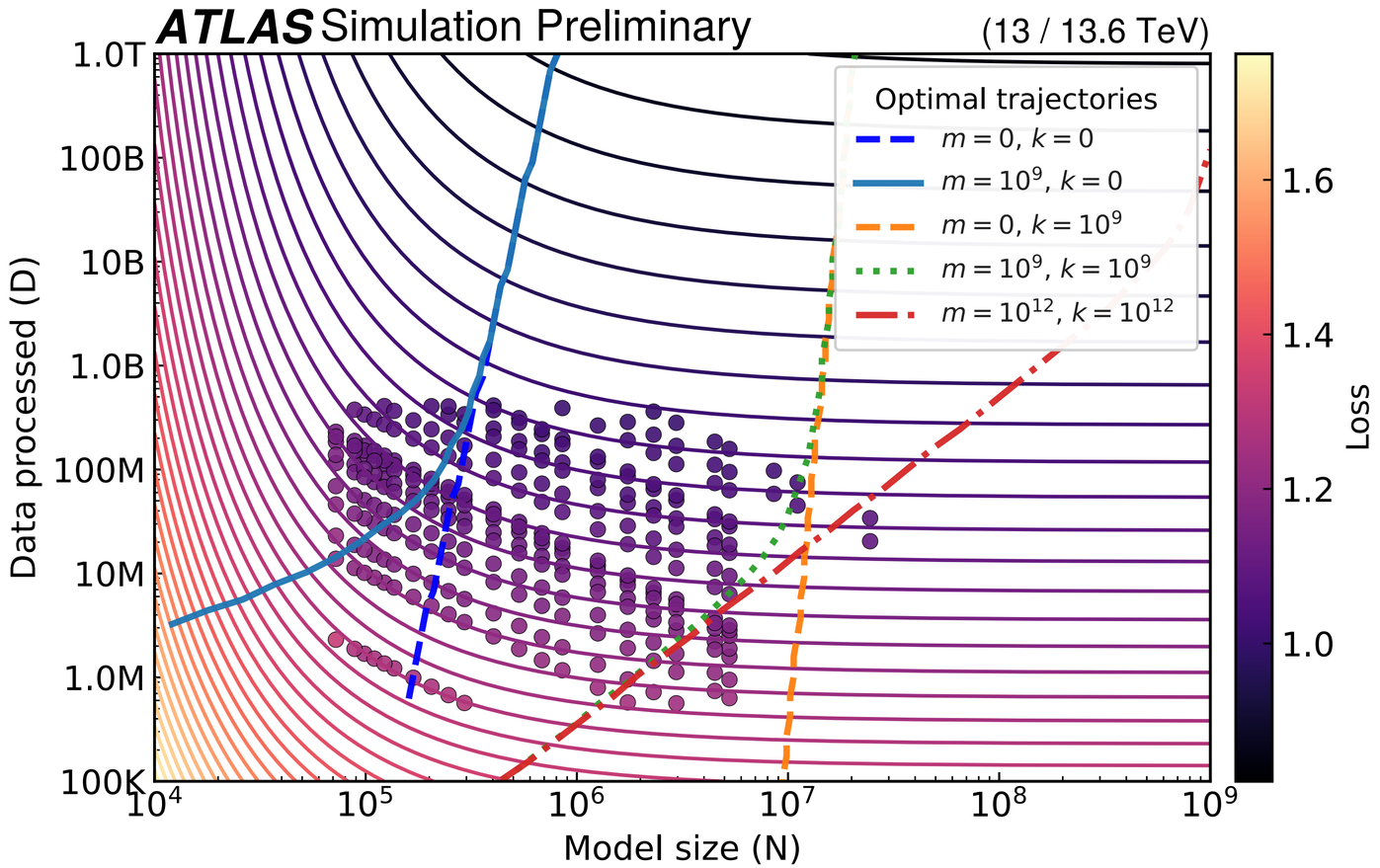
$m \uparrow$



$$C = 6ND + kD + mN$$

Compute optimal trajectories

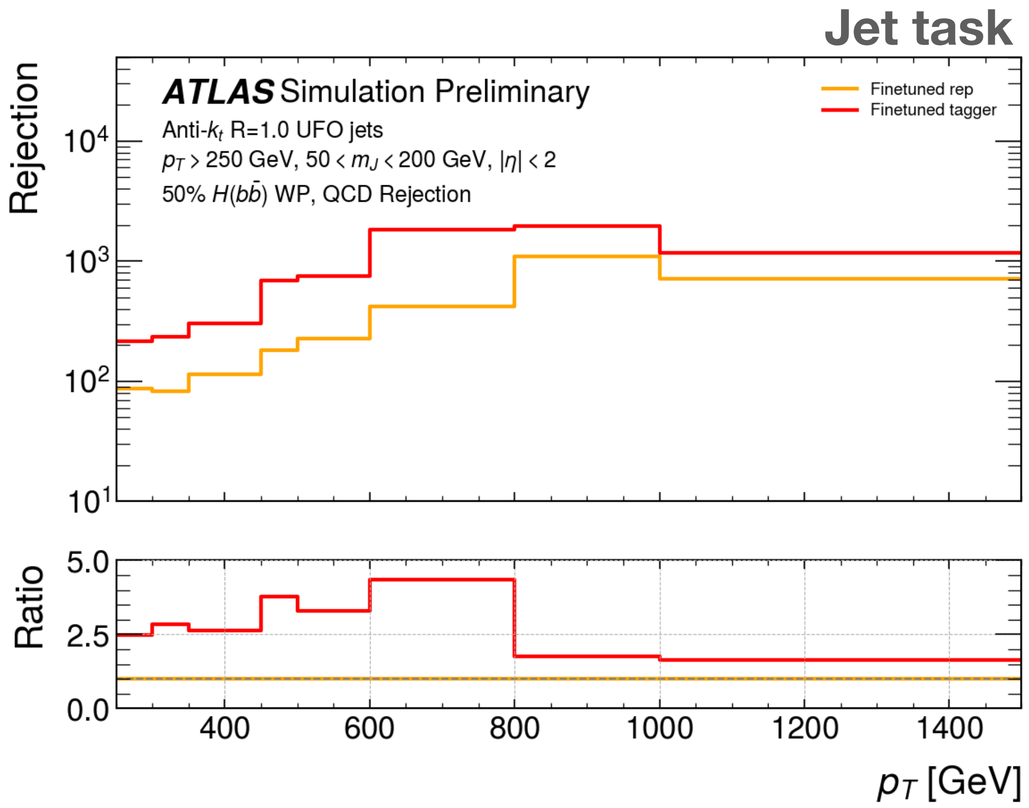


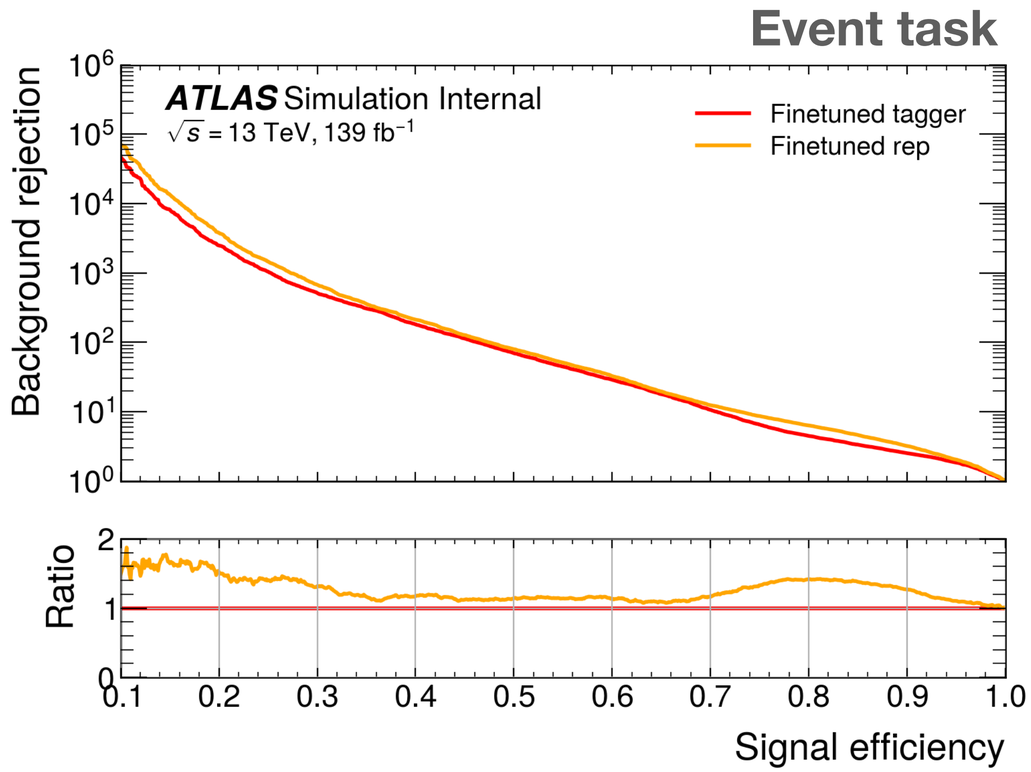


$$C = 6ND + kD + mN$$

Any jet representation is allowed during event-level finetuning, not obvious this would be so easily mapped to jet probabilities

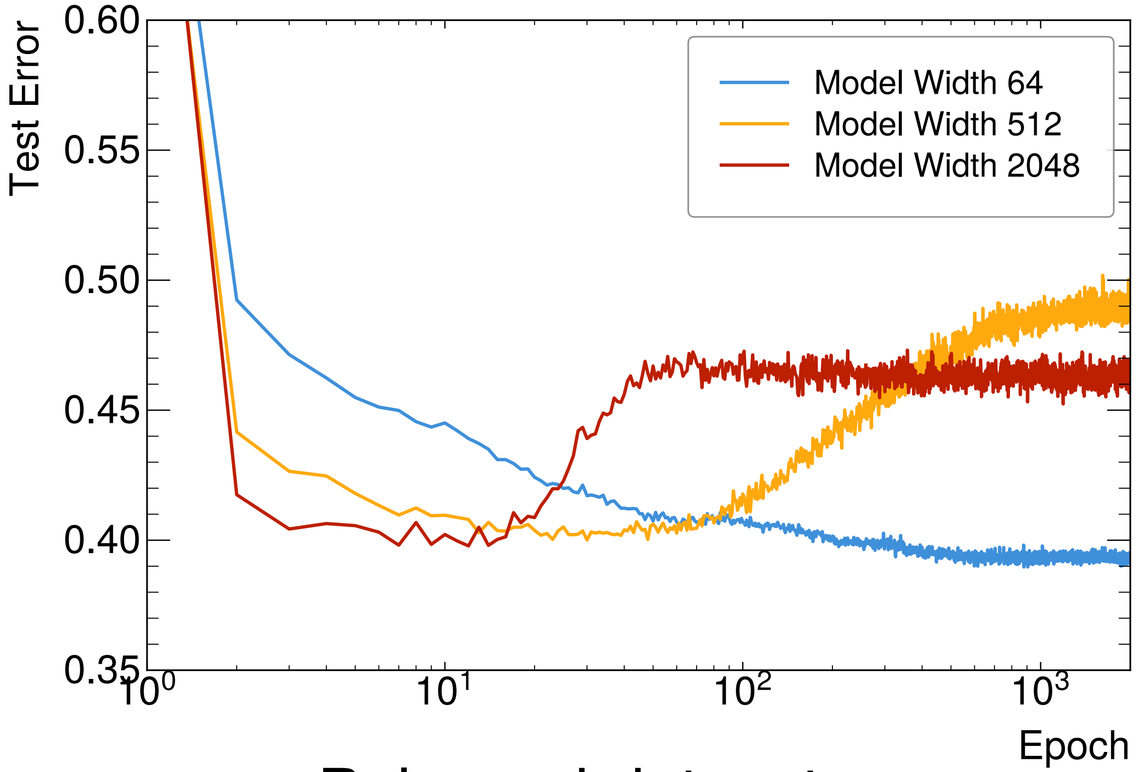
- Finetuning on even-level doesn't necessarily find the best possible tagger, but finds the best possible representation!
- Finetuning a tagger for HH (only jet ID task) yields better tagging performance but lower event-level discrimination



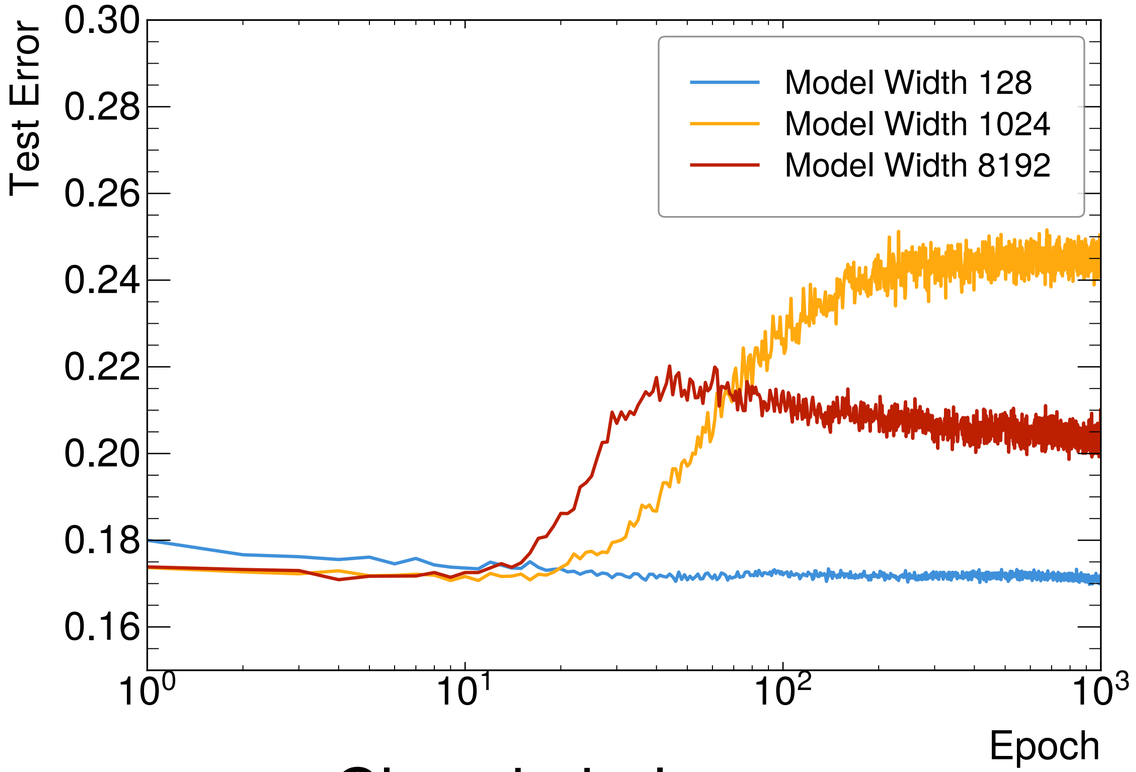


SUSY vs SM classification

Epoch-wise double descent doesn't always happen



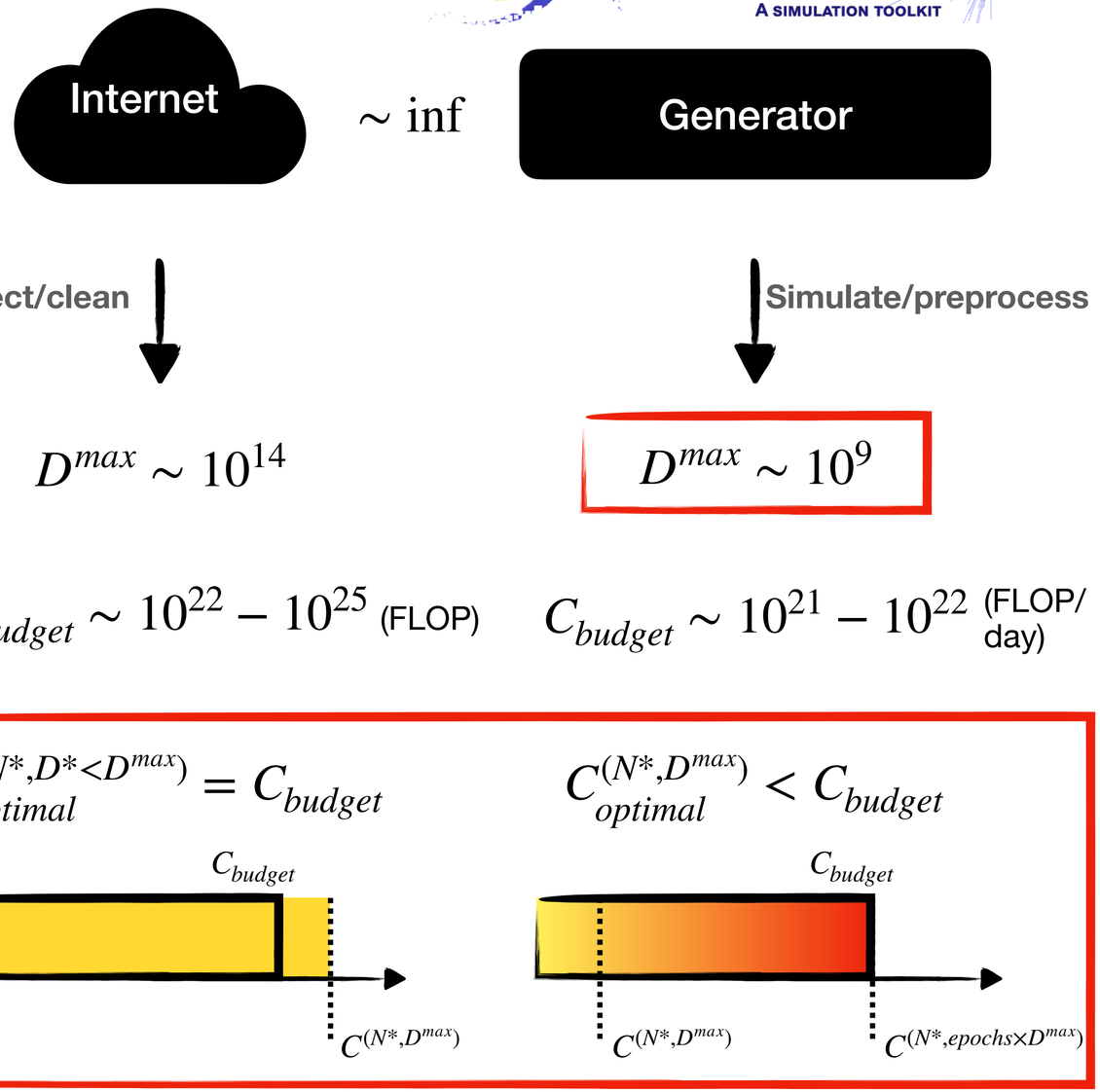
Balanced dataset



Class imbalance

What are we missing?

$$C = 6NBS$$



LLMs typically never see data more than once before running out of compute budget.

- We limit ourself to (few) simulated data, quickly **running out of optimal compute** - **excess compute spent sub-optimally** (e.g. train for multiple epochs, train large models with some degree of overfitting). $BS = epochs \times D^{max}$

How do we scale?

1) Compute optimal scaling (LLM-style)

Training Compute-Optimal Large Language Models,

Training Compute-Optimal Protein Language Models,

Compute-Optimal LLMs Probably Generalize Better With Scale

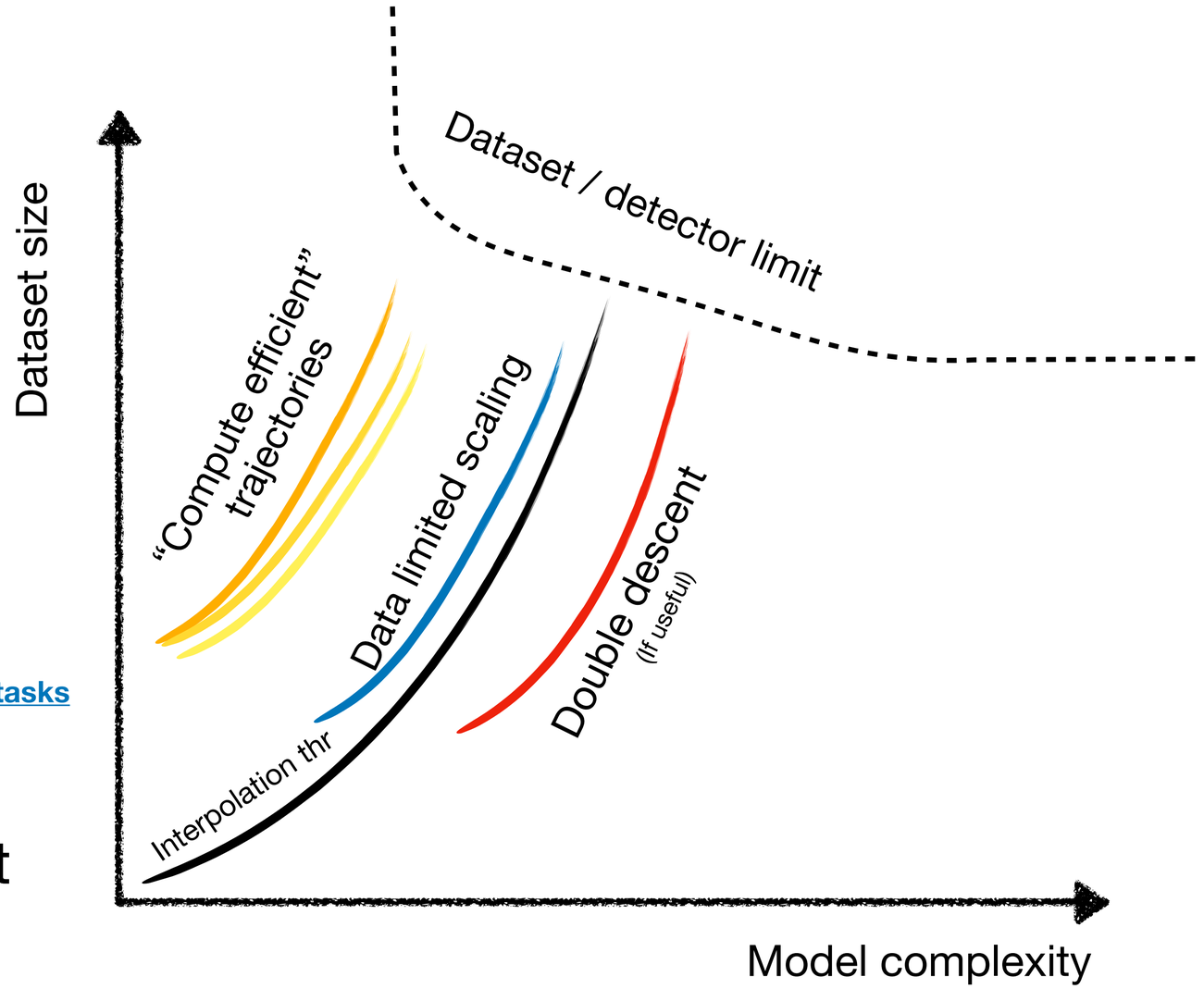
2) Data limited (or loss-optimal) scaling

Scaling Data-Constrained Language Models,

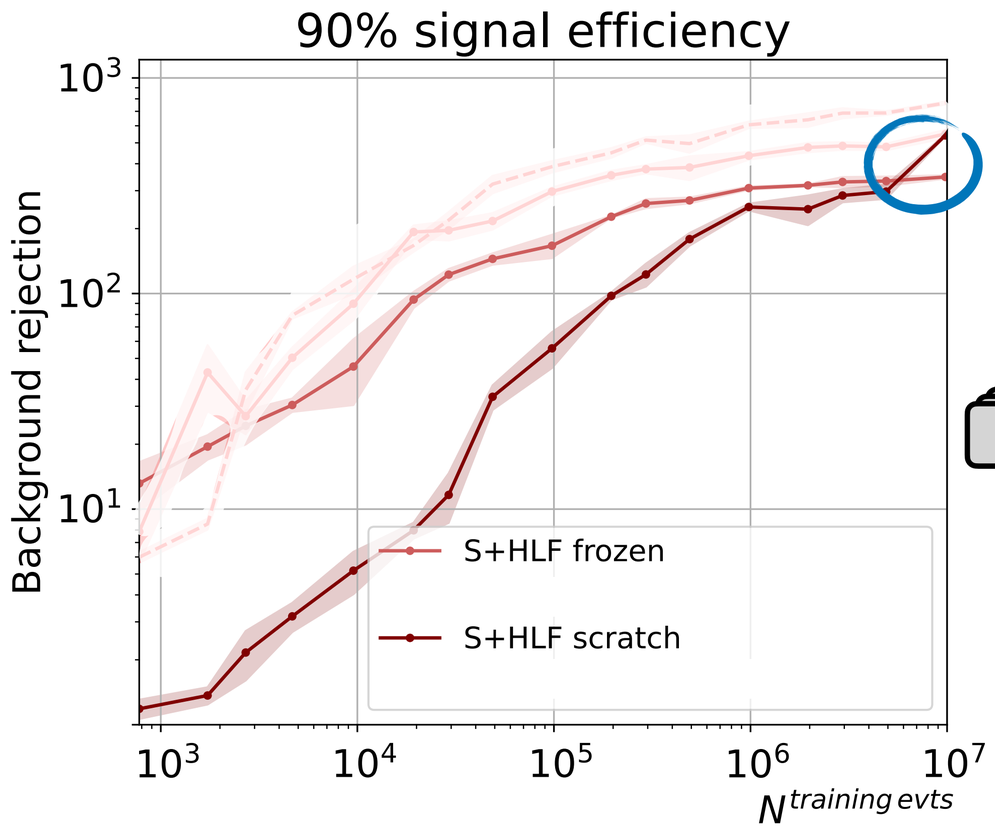
Language models scale reliably with over-training and on downstream tasks

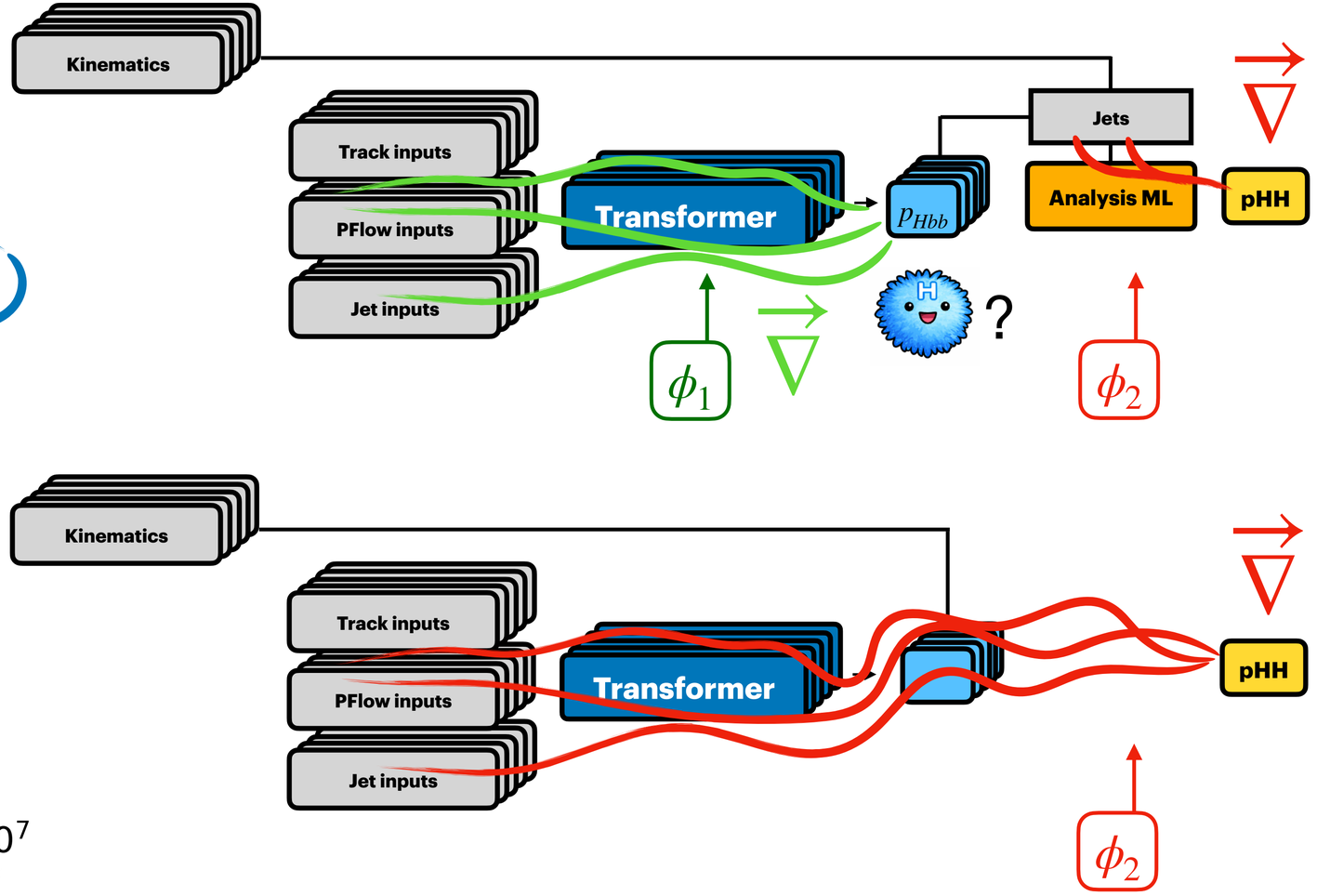
3) Data limited scaling -> double descent

Deep Double Descent: Where Bigger Models and More Data Hurt

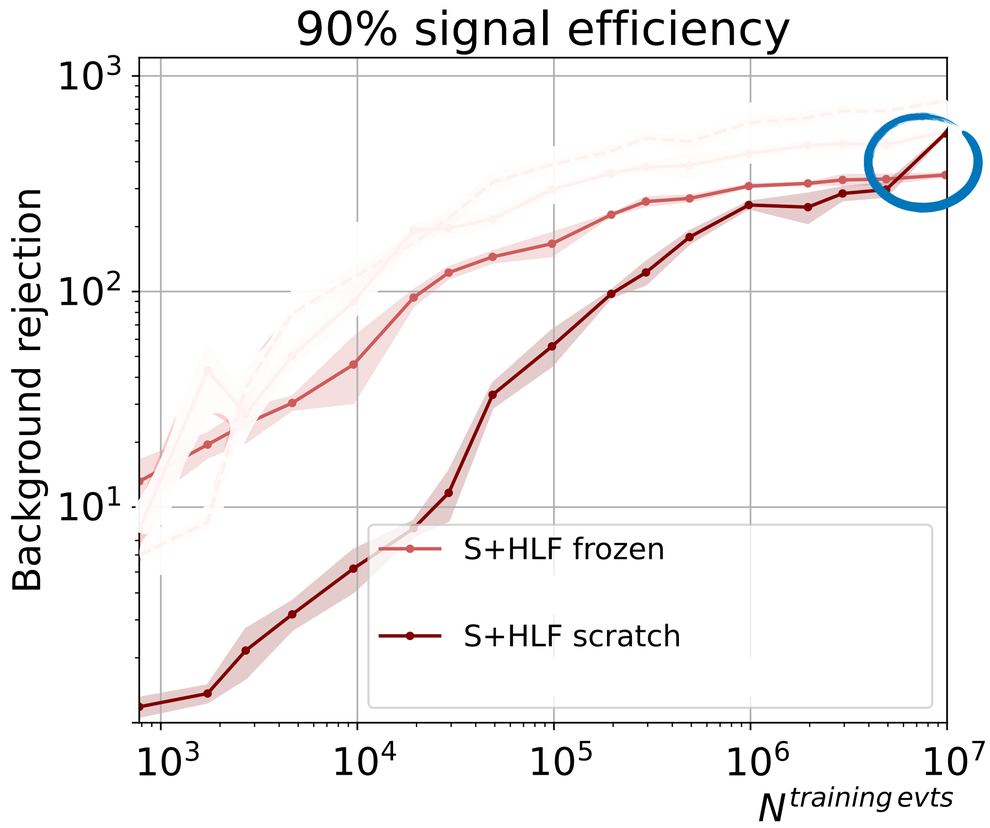


More than just Higgs tagging





More than just Higgs tagging



Training the full pipeline from scratch vs on a frozen backbone (trained on Higgs tagging)

- Much slower (of course) but eventually does better

There must be more (useful) information to be extracted from the jet constituents other than jet label

Compute optimal trajectories

*what defines the optimal trajectory is the assumption on compute cost

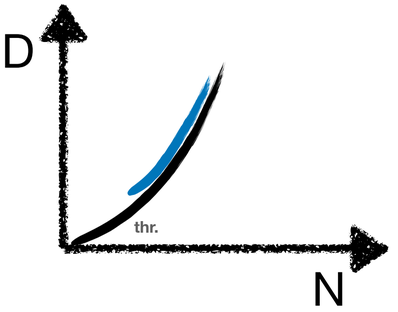
$$L(N, D) = L_\infty + \frac{A}{N^\alpha} + \frac{B}{D^\beta}$$

$$\min_{N,D} L(N, D) \qquad \text{s.t.} \quad C = 6ND = \text{const.}\, C_{\text{budget}}$$

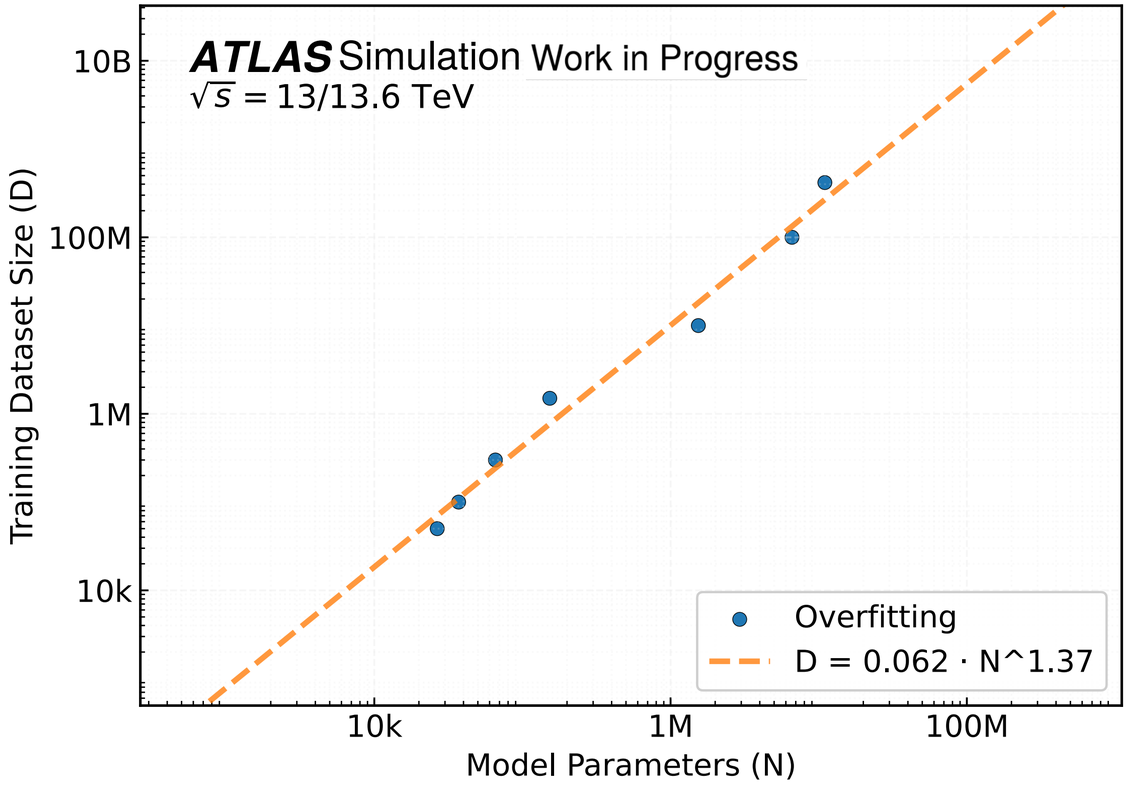
$$N^\star \propto C_{\text{budget}}^{a}, \quad D^\star \propto C_{\text{budget}}^{1-a}, \quad L^\star \propto C_{\text{budget}}^{-\gamma}$$

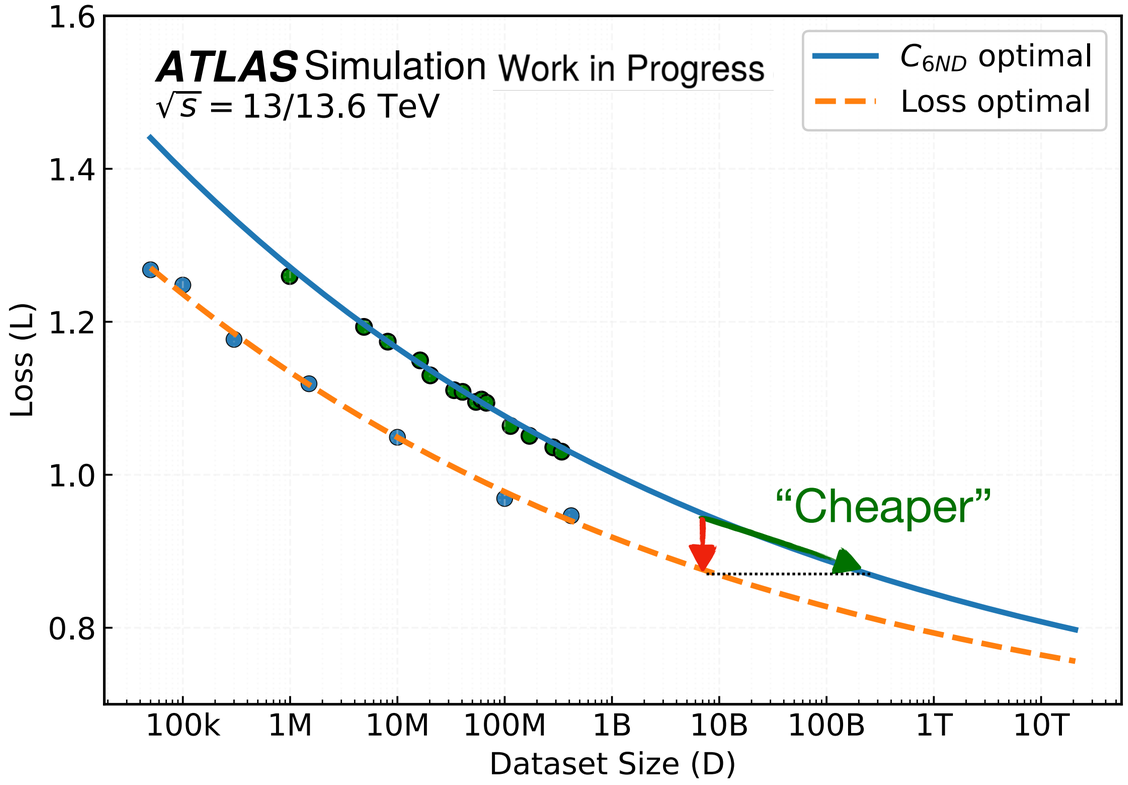
$$a = \frac{\beta}{\alpha + \beta}, \quad \gamma = \frac{\alpha\beta}{\alpha + \beta}$$

Data-constrained scaling
(or loss-optimal)

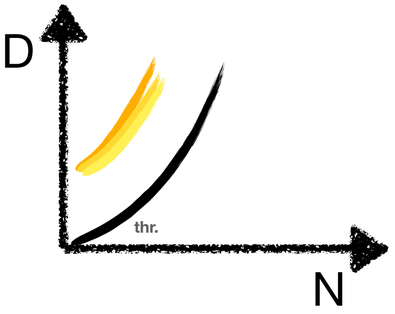


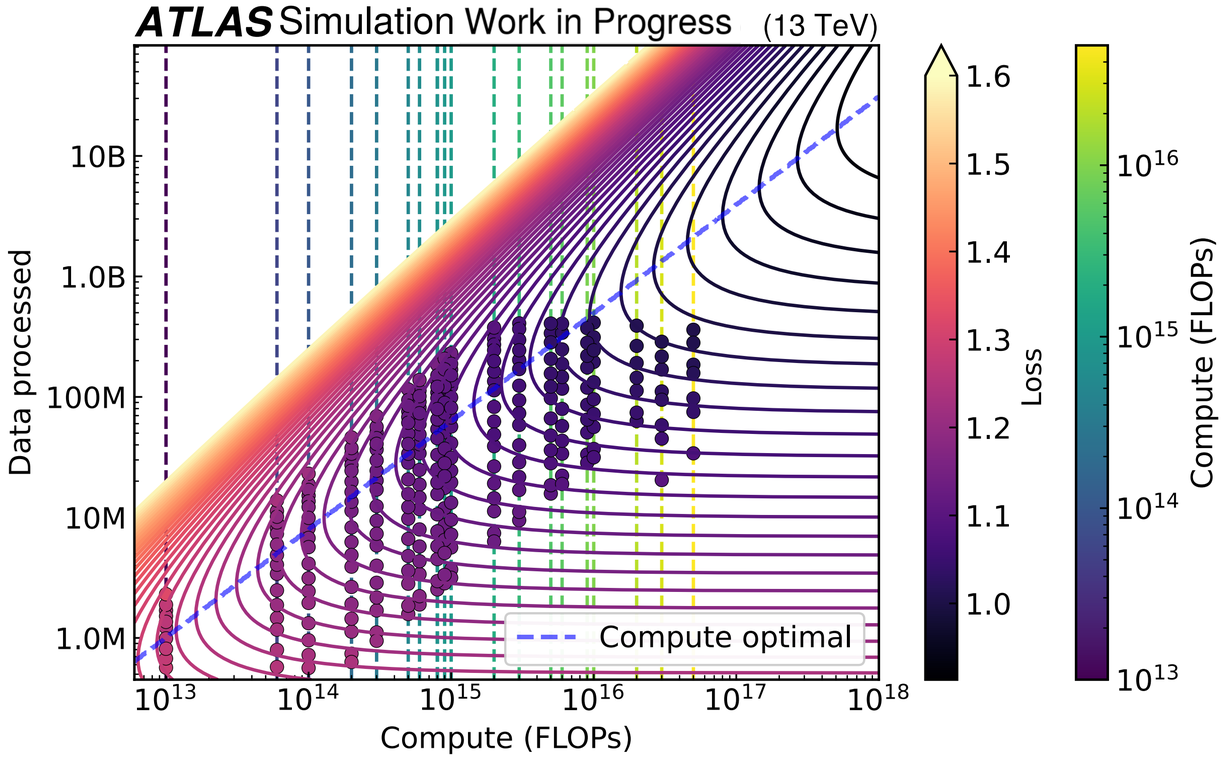
Want to have enough capacity to see a minimum in val loss ↔ stay approximately around the overfitting threshold

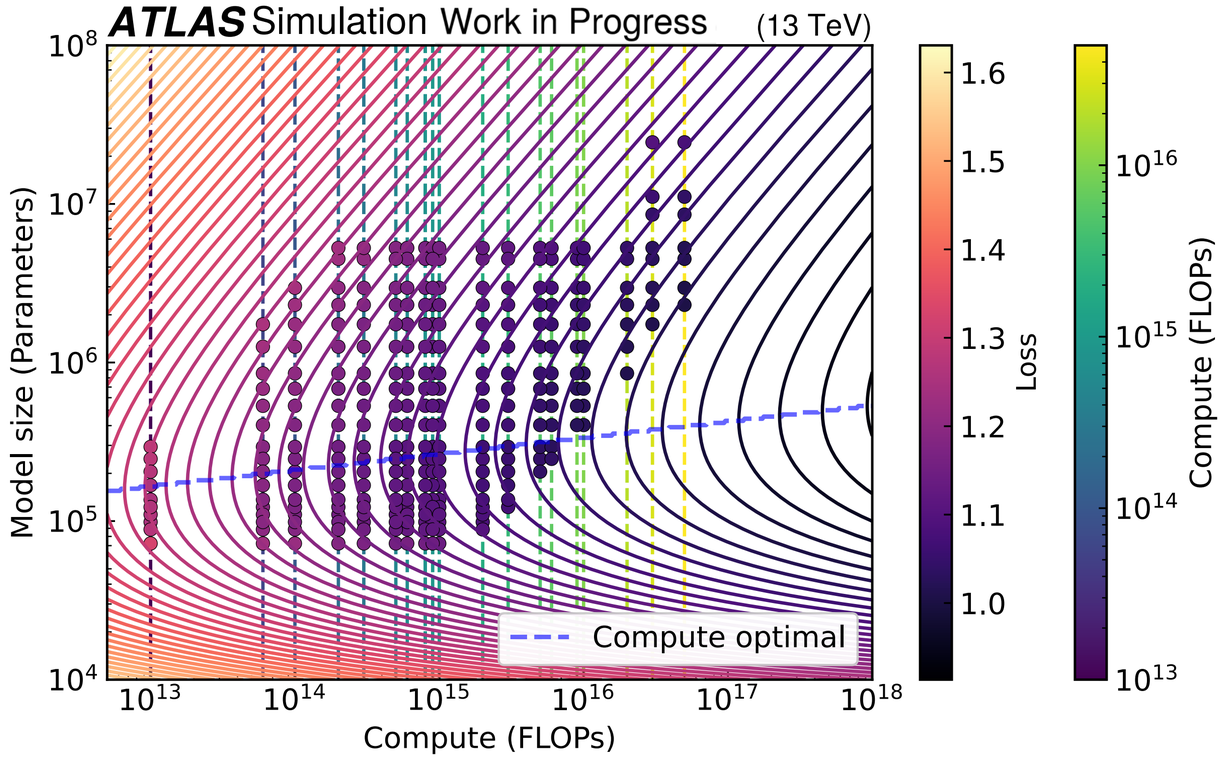




Compute optimal trajectories

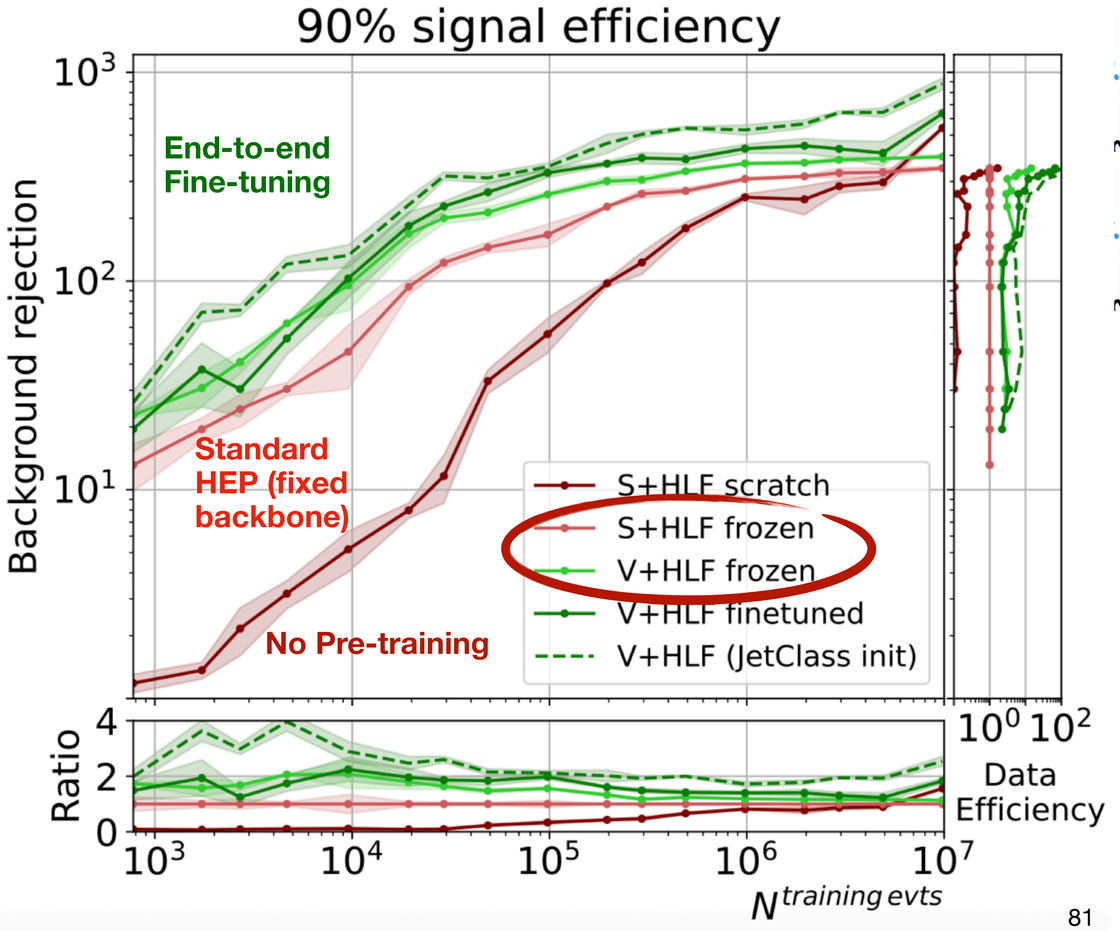


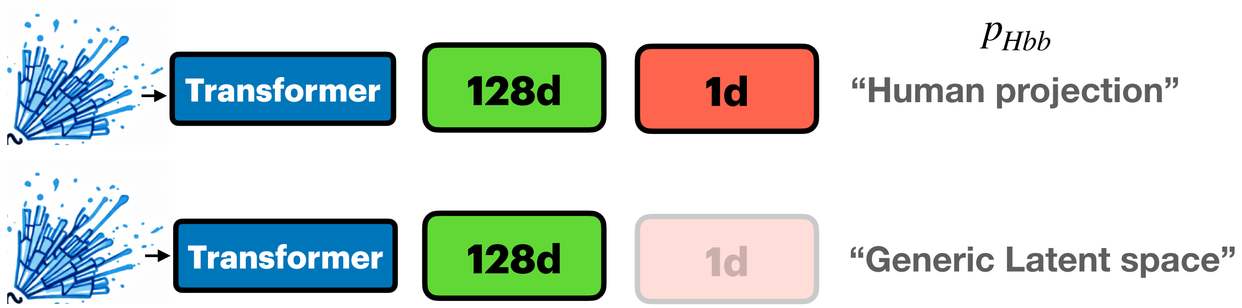




$$C = 6ND$$

Latent representations: bitter lesson?

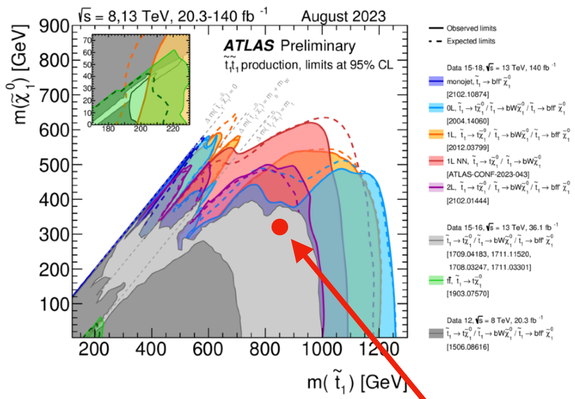


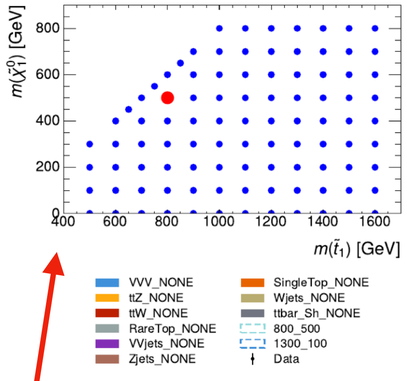


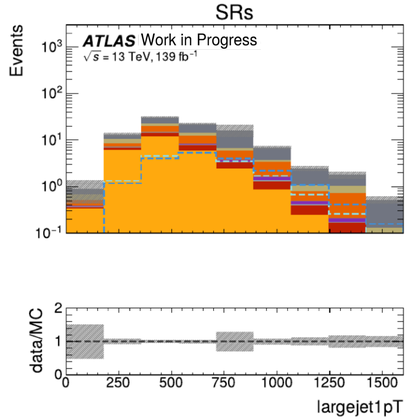
“human interpretable” projections = bottleneck

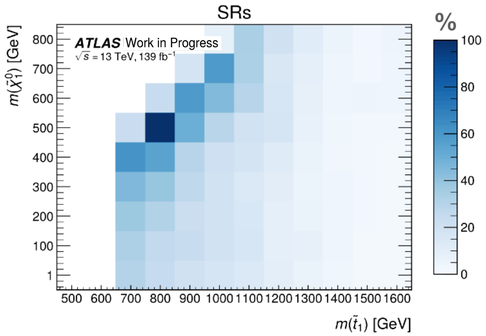
The final step: statistical “Analysis”

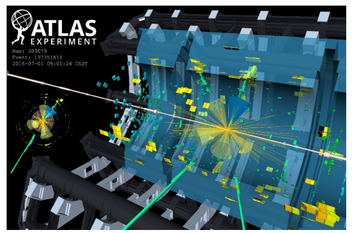
e.g. SUSY $\tilde{t}$ 1L search in ATLAS

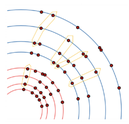


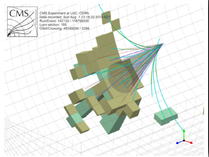




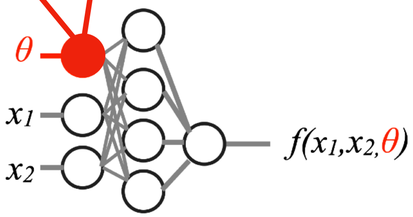




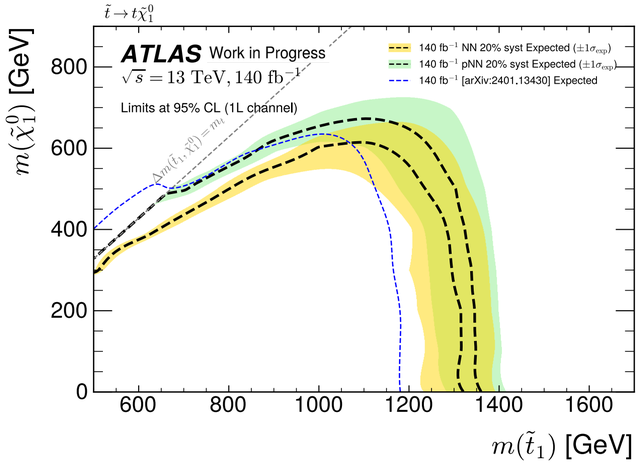




Reco Event (particles, jets, MET)



pyhf
differentiable
Likelihoods



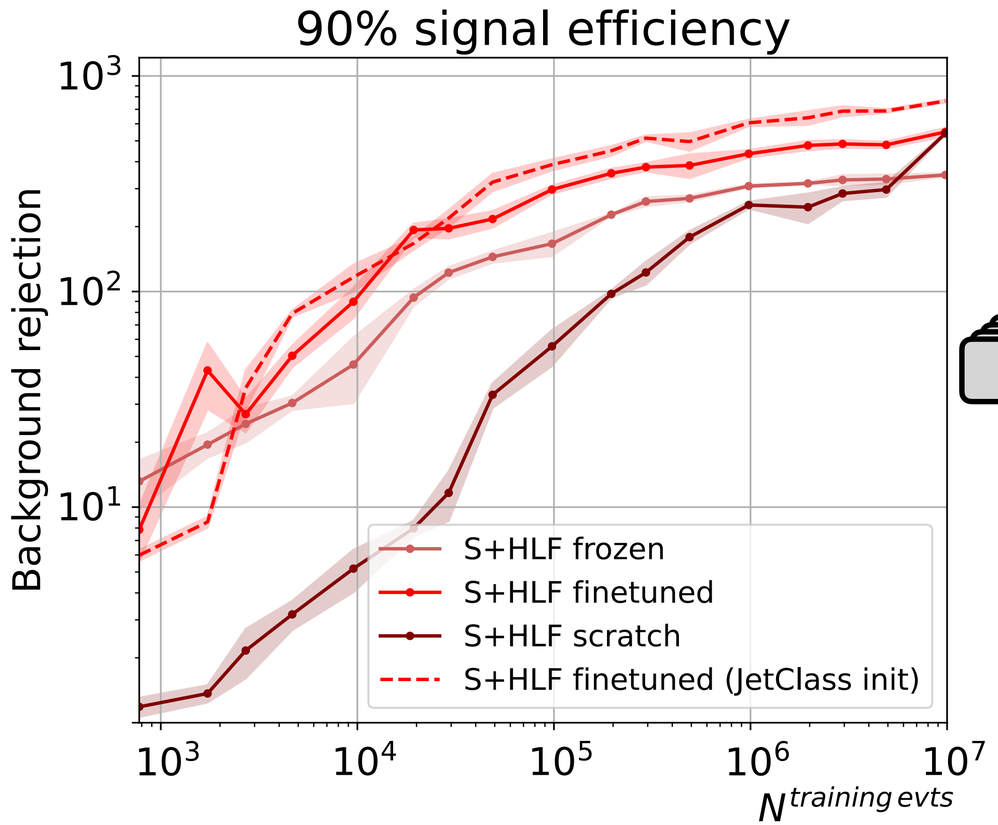
Theory point (hypothesis) drives the analysis definition

SBI

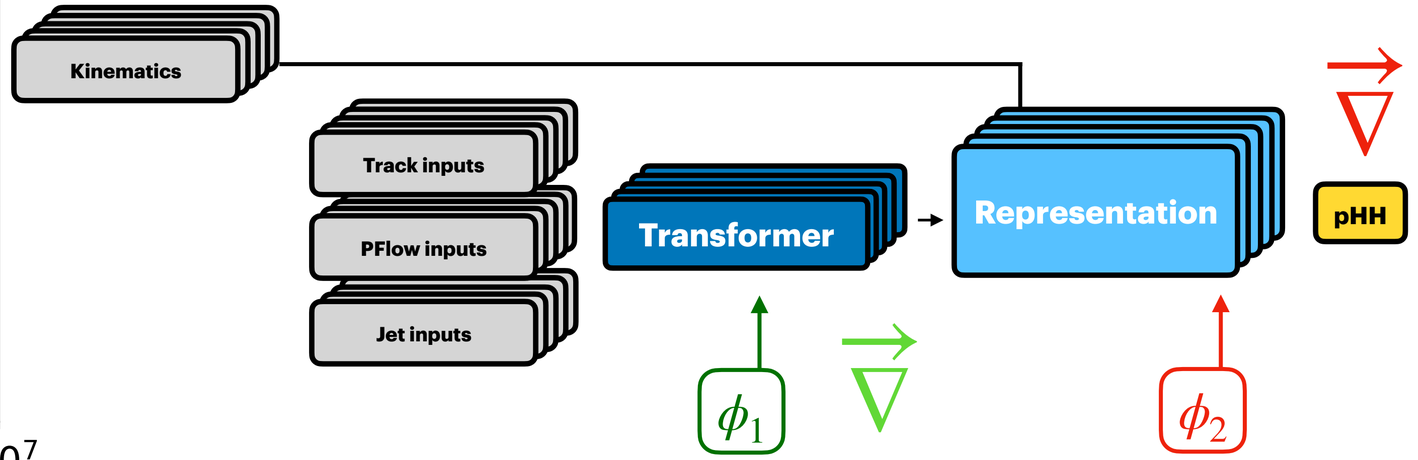
Optimal test statistics

Best of both worlds

Differentiability: we can do both



Pre-training supervised on what we think is useful, let the machine fineteune later to get optimal performance



The Bitter Lesson

Rich Sutton

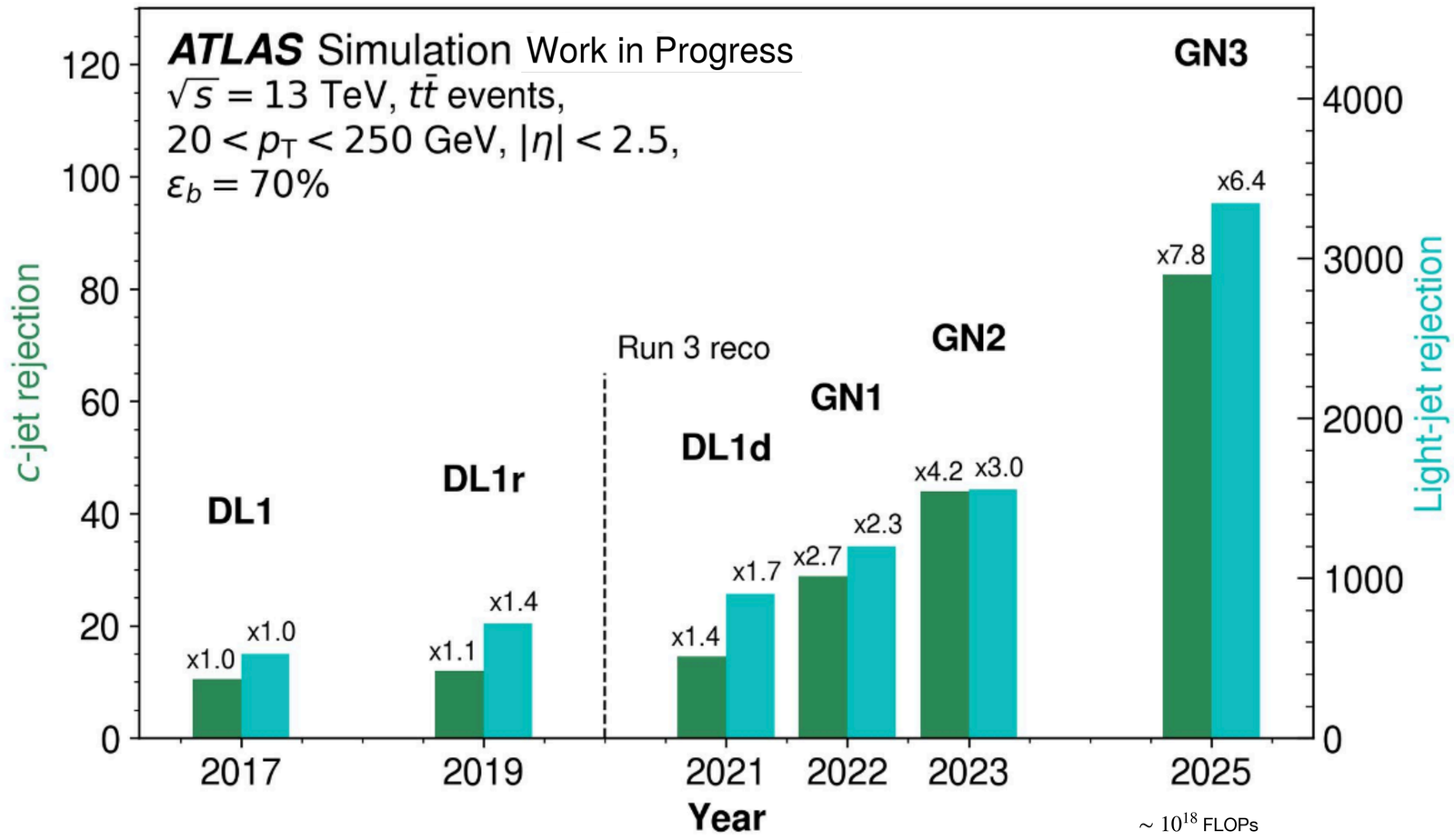
March 13, 2019

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. The ultimate reason for this is Moore's law, or rather its generalization of continued exponentially falling cost per unit of computation. Most AI research has been conducted as if the computation available to the agent were constant (in which case leveraging human knowledge would be one of the only ways to improve performance) but, over a slightly longer time than a typical research project, massively more computation inevitably becomes available. Seeking an improvement that makes a difference in the shorter term, researchers seek to leverage their human knowledge of the domain, but the only thing that matters in the long run is the leveraging of computation. These two need not run counter to each other, but in practice they tend to. Time spent on one is time not spent on the other. There are psychological commitments to investment in one approach or the other. And the human-knowledge approach tends to complicate methods in ways that make them less suited to taking advantage of general methods leveraging computation. There were many examples of AI researchers' belated learning of this bitter lesson, and it is instructive to review some of the most prominent.

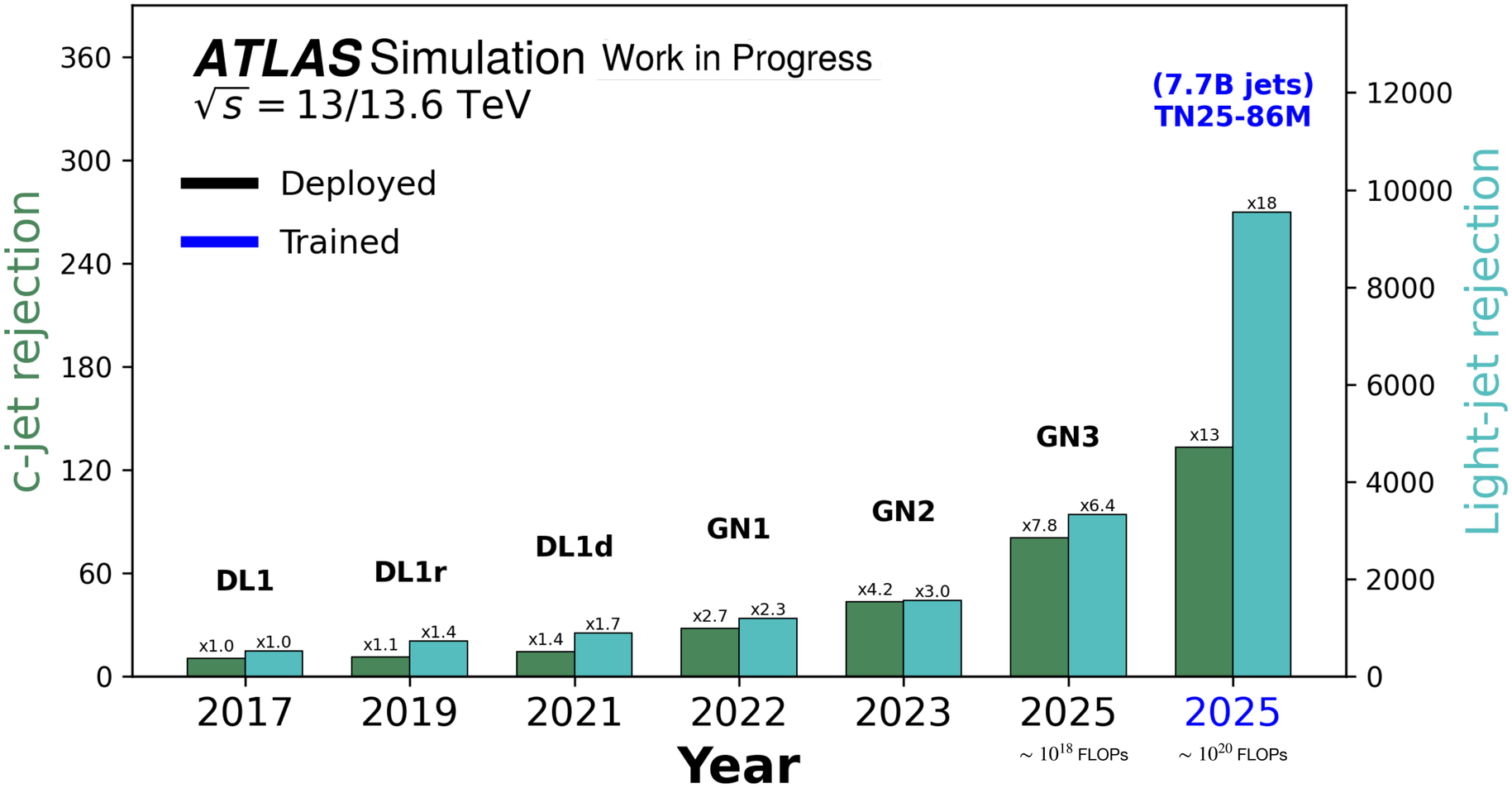
TLDR: “Seeking an improvement that makes a difference in the shorter term, researchers seek to leverage their human knowledge of the domain, but the only thing that matters in the long run is the leveraging of computation.”

Personal take: “Use scalable methods”. If your inductive bias meshes well with scale-up that’s great, use it, but don’t overcomplicate your setup to add inductive bias in a way that hinders scaling

How much are we talking about

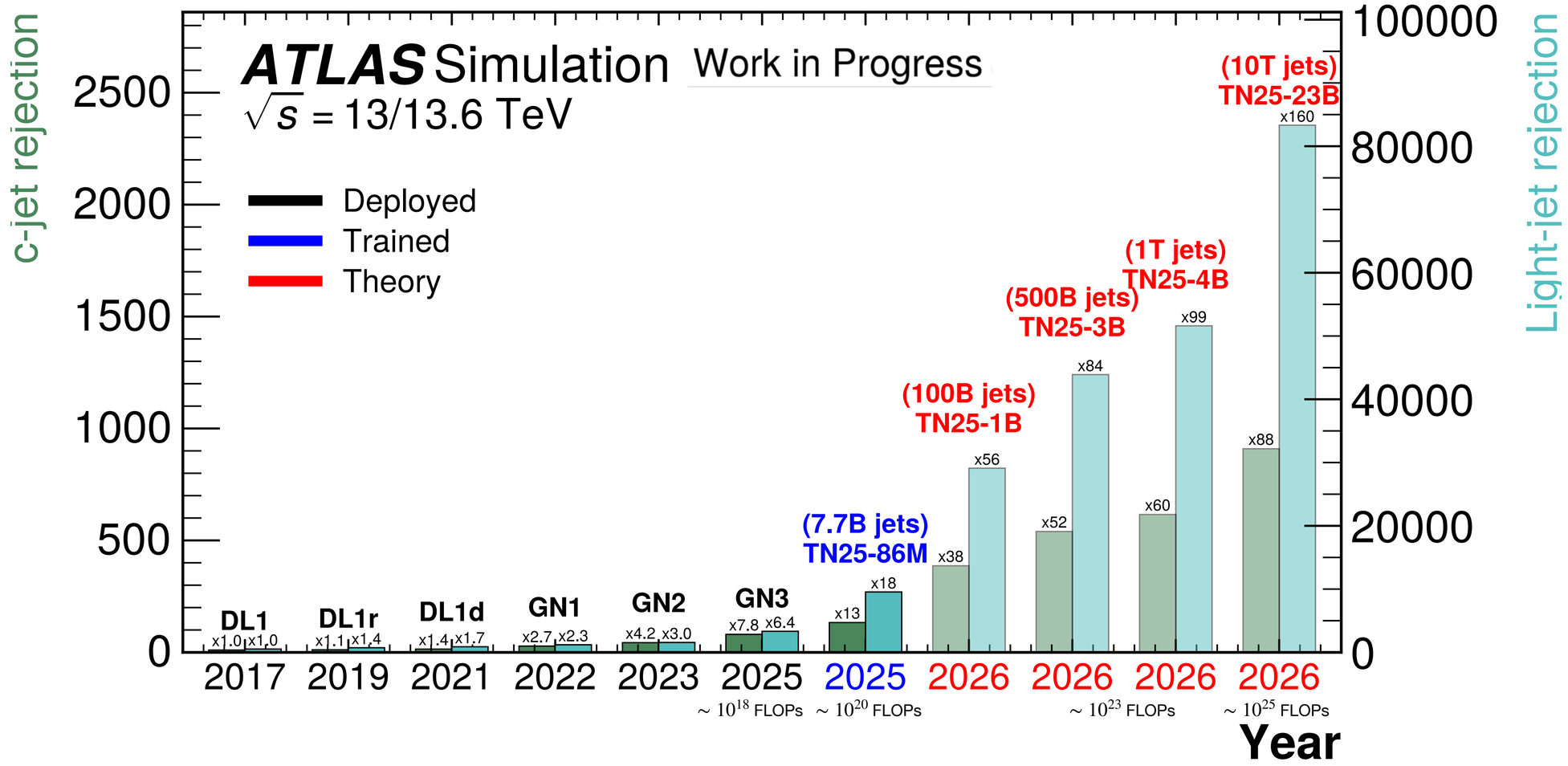


How much are we talking about



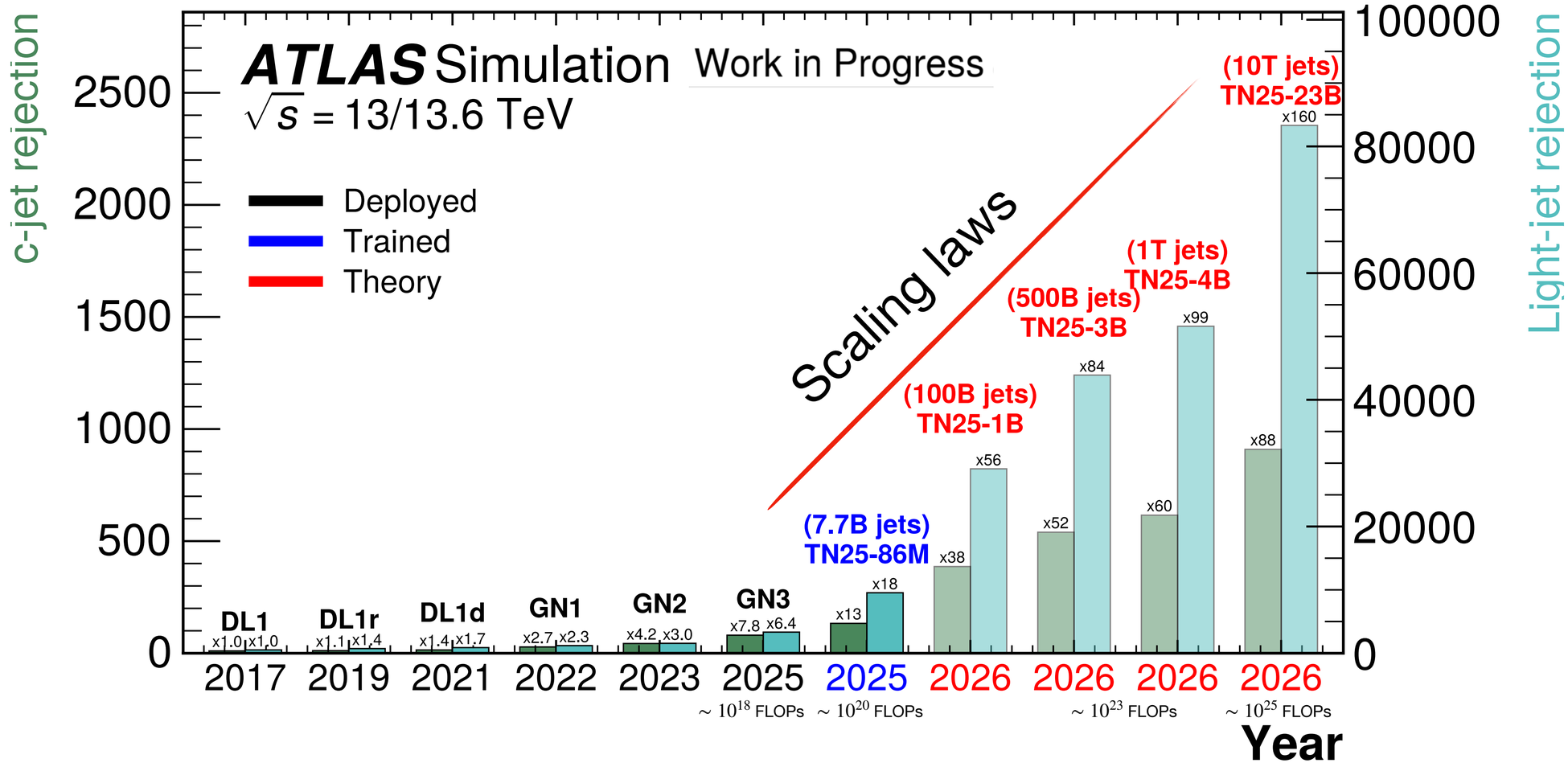
(Deployment on the grid through IaaS)

How much are we talking about

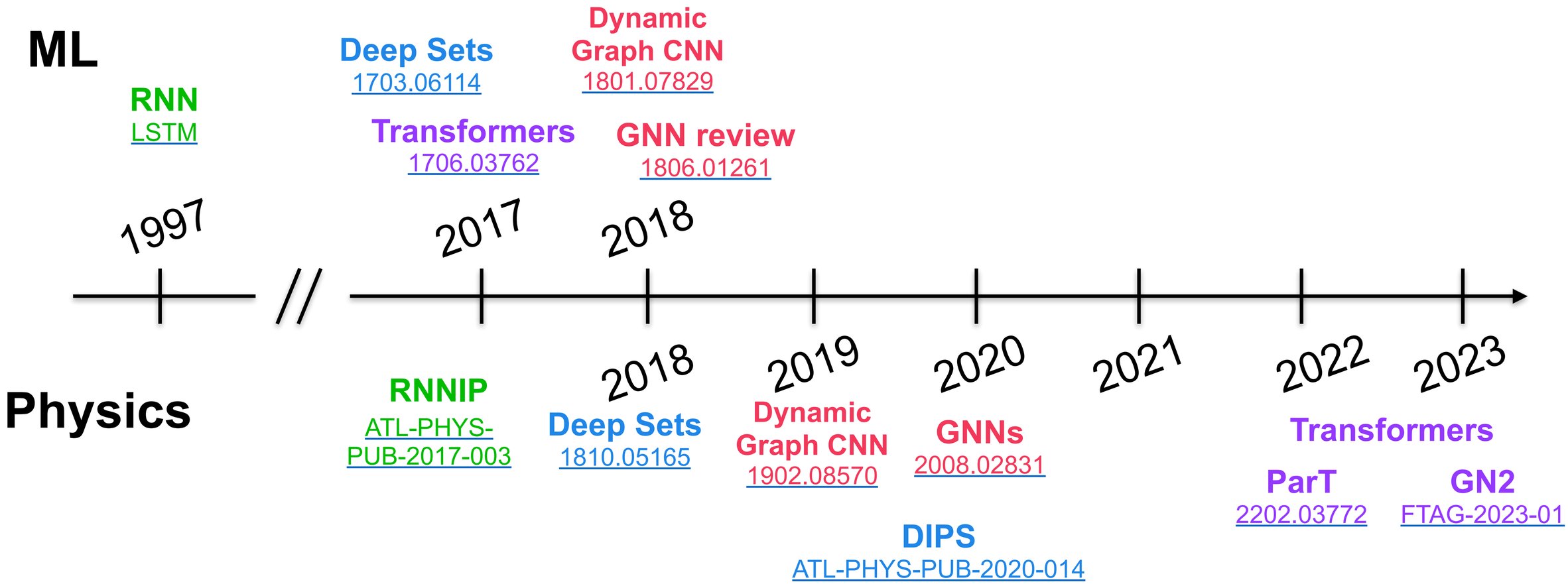


(Deployment on the grid through IaaS)

How much are we talking about



(Deployment on the grid through IaaS)


ML
RNN
LSTM
Deep Sets
1703.06114
Dynamic
Graph CNN
1801.07829
Transformers
1706.03762
GNN review
1806.01261
1997
2017
2018
2018
2019
2020
2021
2022
2023
Physics
RNNIP
ATL-PHYS-
PUB-2017-003
Deep Sets
1810.05165
Dynamic
Graph CNN
1902.08570
GNNs
2008.02831
DIPS
ATL-PHYS-PUB-2020-014
Transformers
ParT
2202.03772
GN2
FTAG-2023-01