Foundation Models and Representation Learning

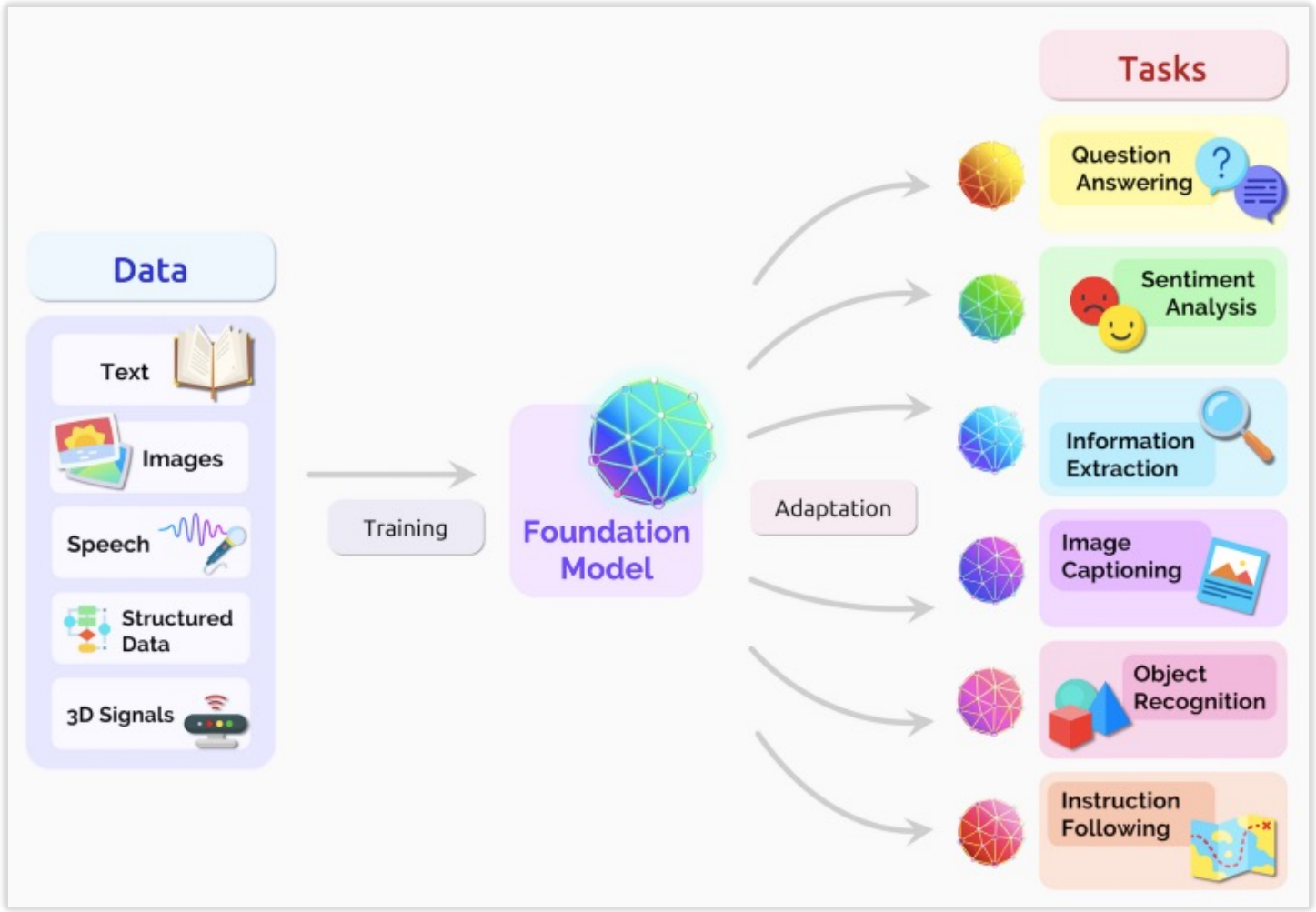
Michael Kagan, SLAC

IWR School on Machine Learning for Fundamental Physics
September 18, 2025

SLAC NATIONAL ACCELERATOR LABORATORY

Stanford University | U.S. DEPARTMENT OF ENERGY

What is a Foundation Model?

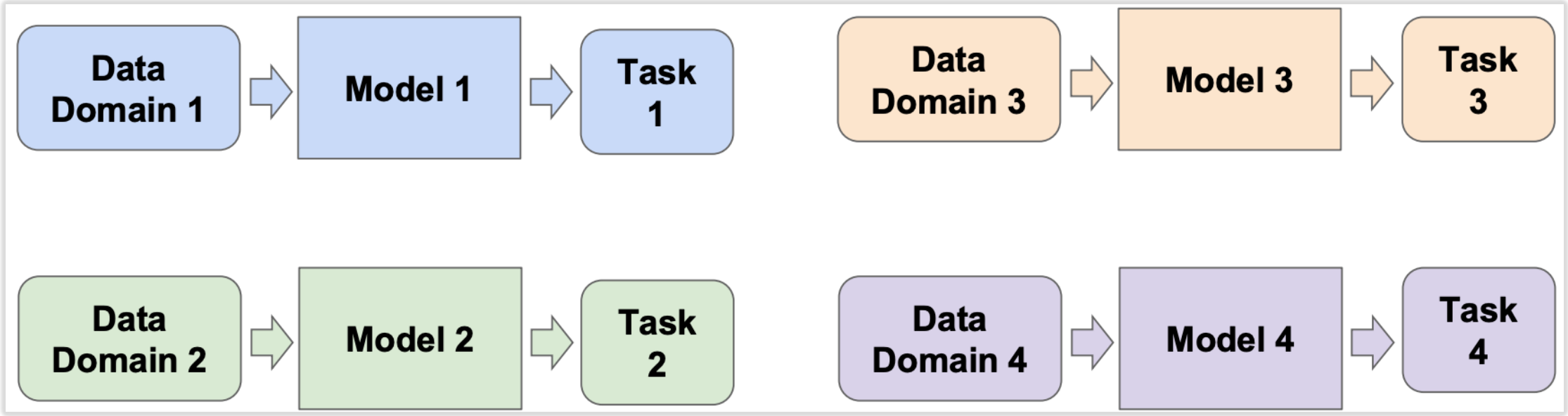


Prompt: `Several giant wooly mammoths approach treading through a snowy meadow […]` OpenAI Sora

Image credit: 2108.07258

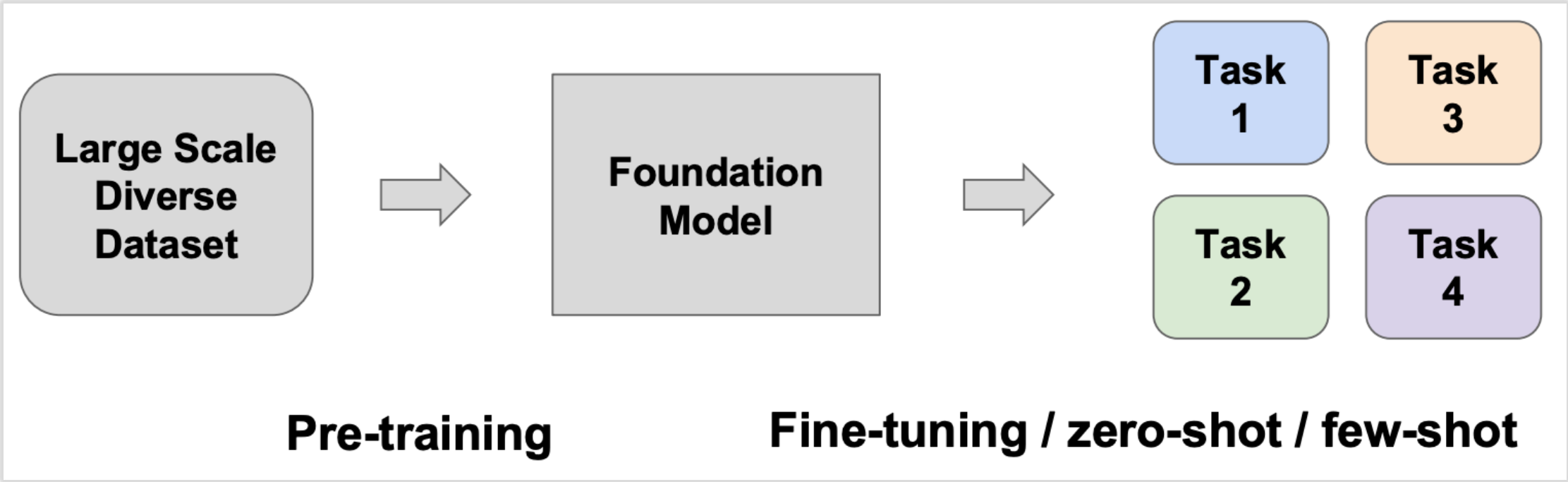
How we typically think about ML models

Train a specialized model to solve each task



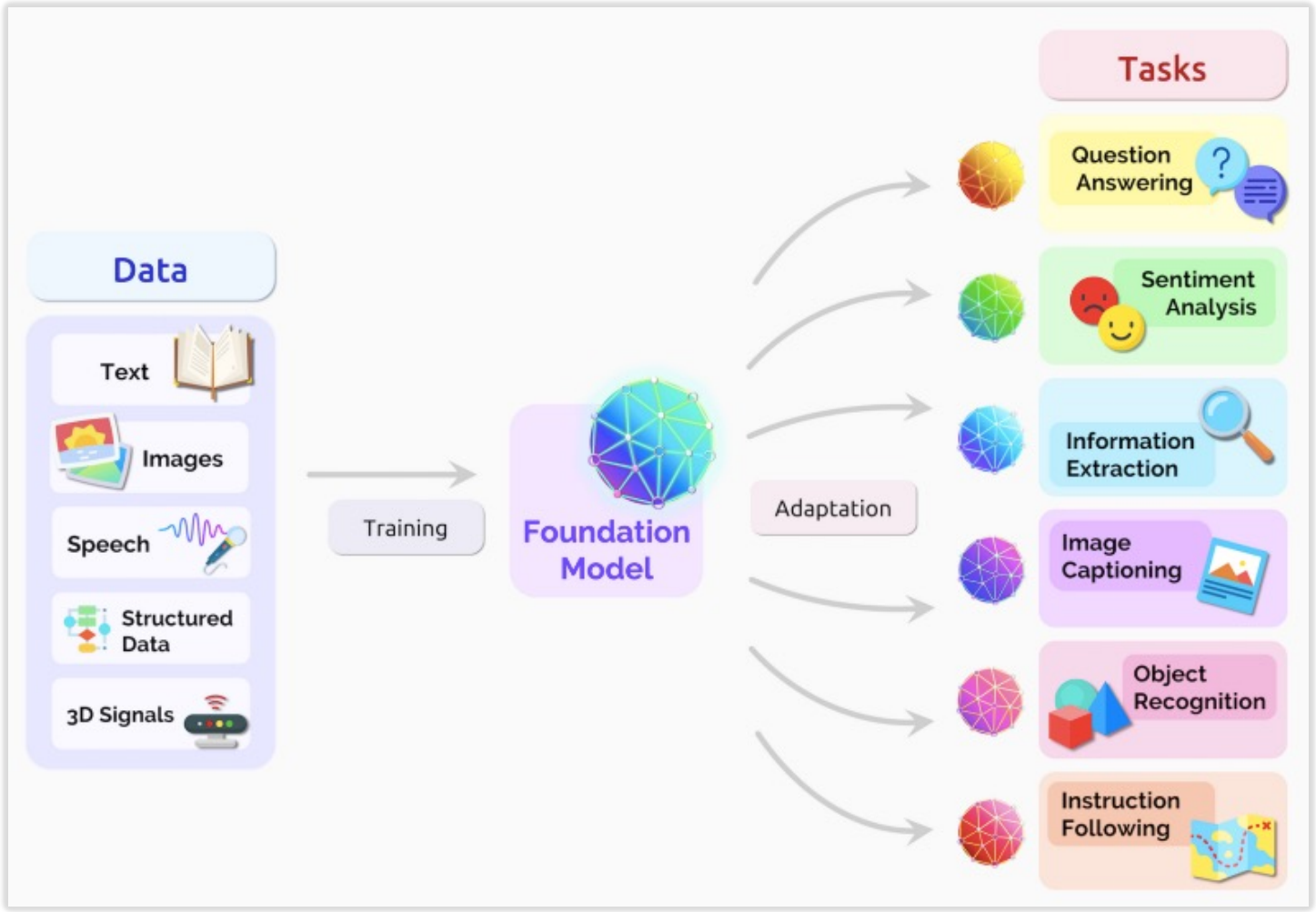
Foundation Models take a different approach

Pre-train one model that acts as the ***foundation*** for many different tasks



During pre-training foundation model must learn a good ***representation*** of the data that is useful / transferrable / tunable for many tasks.

What is a Foundation Model?



Always see with FMs:

- General /robust to many different tasks

Often see with FMs:

- Large # params
- Large amount of data
- ***Self-supervised*** pre-training

Image credit: 2108.07258

Same Model Solving Many Tasks

2005.14165

Language Models are Few-Shot Learners

Tom B. Brown* **Benjamin Mann*** **Nick Ryder*** **Melanie Subbiah***

Jared Kaplan† **Prafulla Dhariwal** **Arvind Neelakantan** **Pranav Shyam** **Girish Sastry**

Amanda Askell **Sandhini Agarwal** **Ariel Herbert-Voss** **Gretchen Krueger** **Tom Henighan**

Rewon Child **Aditya Ramesh** **Daniel M. Ziegler** **Jeffrey Wu** **Clemens Winter**

Christopher Hesse **Mark Chen** **Eric Sigler** **Mateusz Litwin** **Scott Gray**

Benjamin Chess **Jack Clark** **Christopher Berner**

Sam McCandlish **Alec Radford** **Ilya Sutskever** **Dario Amodei**

OpenAI

3 Results **10**
3.1 Language Modeling, Cloze, and Completion Tasks . . . 11
3.2 Closed Book Question Answering . . . 13
3.3 Translation . . . 14
3.4 Winograd-Style Tasks . . . 16
3.5 Common Sense Reasoning . . . 17
3.6 Reading Comprehension . . . 18
3.7 SuperGLUE . . . 18
3.8 NLI . . . 20
3.9 Synthetic and Qualitative Tasks . . . 21

2 Model architecture, data, training and evaluations **3**
2.1 Quantization . . . 4
2.2 Architecture . . . 4
2.3 Tokenizer . . . 5
2.4 Pretraining . . . 5
2.5 Post-Training for Reasoning and Tool Use . . . 6
2.5.1 Harmony Chat Format . . . 6
2.5.2 Variable Effort Reasoning Training . . . 7
2.5.3 Agentic Tool Use . . . 7
2.6 Evaluation . . . 7
2.6.1 Reasoning, Factuality and Tool Use . . . 8
2.6.2 Health Performance . . . 8
2.6.3 Multilingual Performance . . . 9
2.6.4 Full Evaluations . . . 10
3 Safety testing and mitigation approach **10**
4 Default Safety Performance: Observed Challeng
4.1 Disallowed Content . . .
4.2 Jailbreaks . . .
4.3 Instruction Hierarchy . . .
4.4 Hallucinated chains of thought . . .
4.5 Hallucinations . . .
4.6 Fairness and Bias . . .
5 Preparedness Framework
5.1 Adversarial Training . . .
5.1.1 External Safety expert feedback on adversarial training methodology . . . 17
5.2 Capability findings . . . 18
5.2.1 Biological and Chemical - Adversarially Fine-tuned . . . 18
5.2.1.1 Long-form Biological Risk Questions . . . 19
5.2.1.2 Multimodal Troubleshooting Virology . . . 20
5.2.1.3 ProtocolQA Open-Ended . . . 20
5.2.1.4 Tacit Knowledge and Troubleshooting . . . 21
5.2.1.5 TroubleshootingBench . . . 21
5.2.1.6 Evaluations and Red Teaming by External Safety Experts . . . 22
5.2.2 Cybersecurity - Adversarially fine-tuned . . . 22
5.2.2.1 Capture the Flag (CTF) Challenges . . . 23
5.2.2.2 Cyber range . . . 24
5.2.3 AI Self-Improvement . . . 26
5.2.3.1 SWE-bench Verified (N=477) . . . 26
5.2.3.2 OpenAI PRs . . . 27
5.2.3.3 PaperBench . . . 28

[gpt-oss model card]

In-Context Learning

Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
1  Translate English to French:        ← task description
2  cheese =>                           ← prompt
```

One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

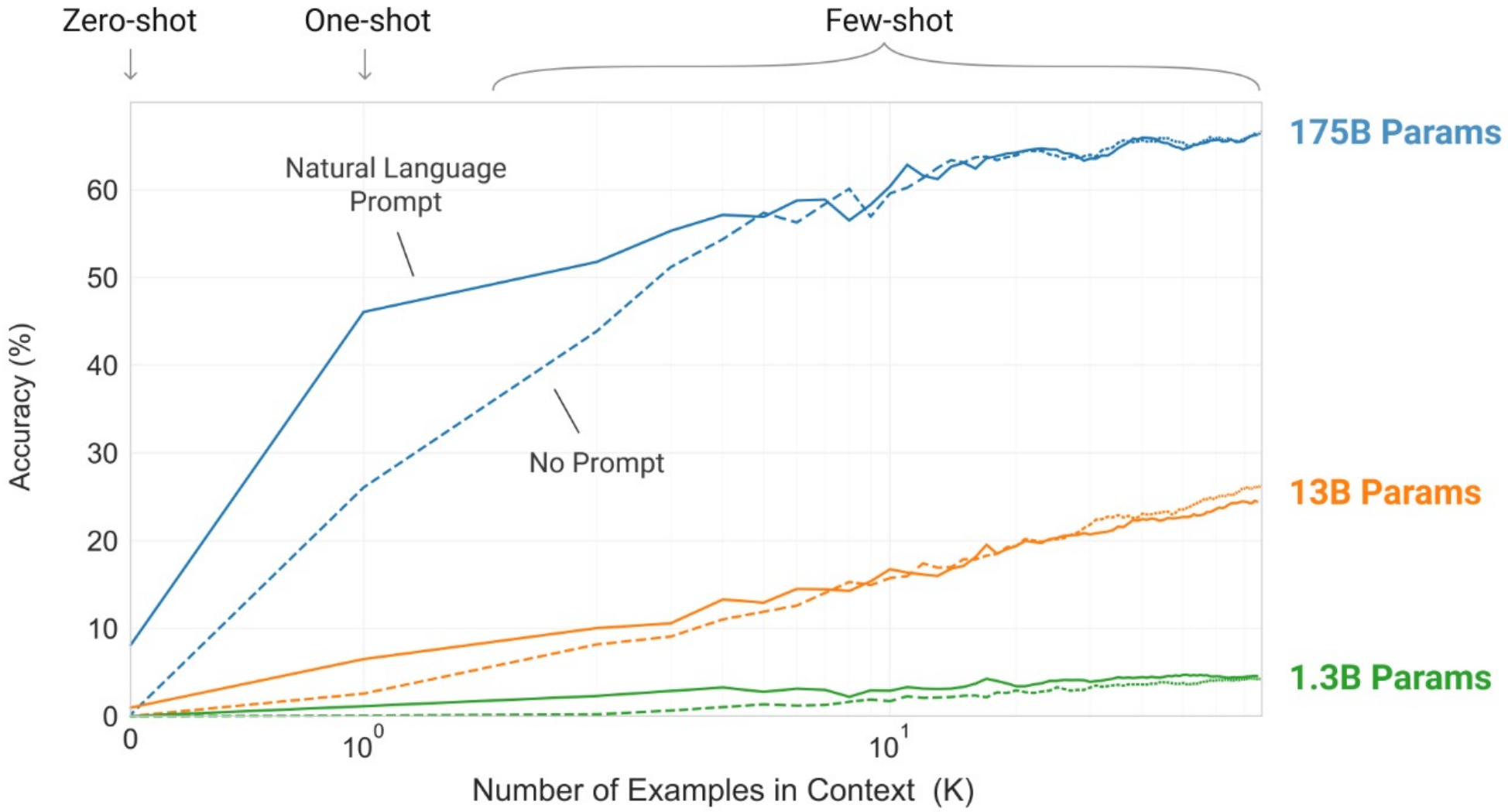
```
1  Translate English to French:        ← task description
2  sea otter => loutre de mer          ← example
3  cheese =>                           ← prompt
```

Few-shot

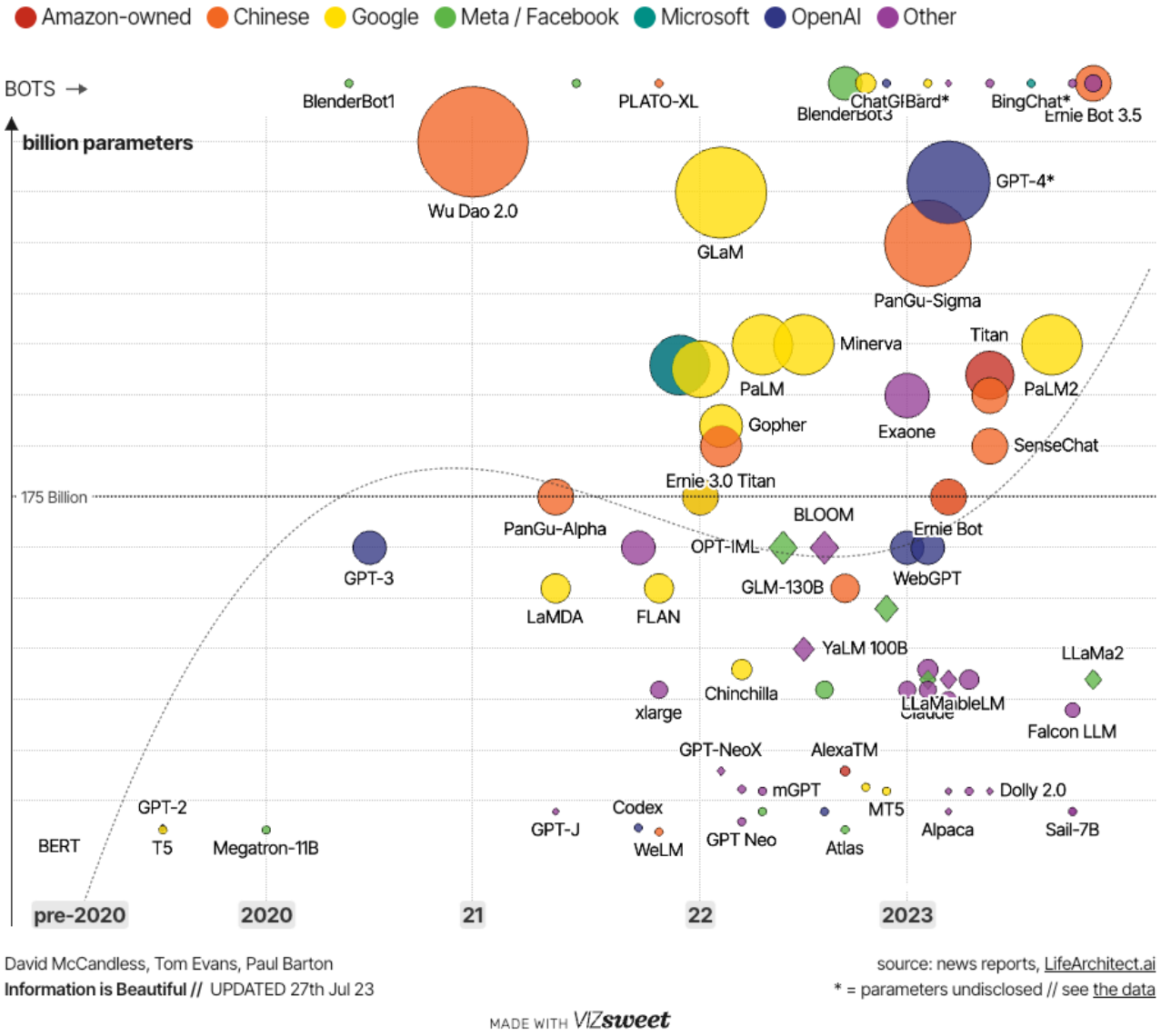
In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

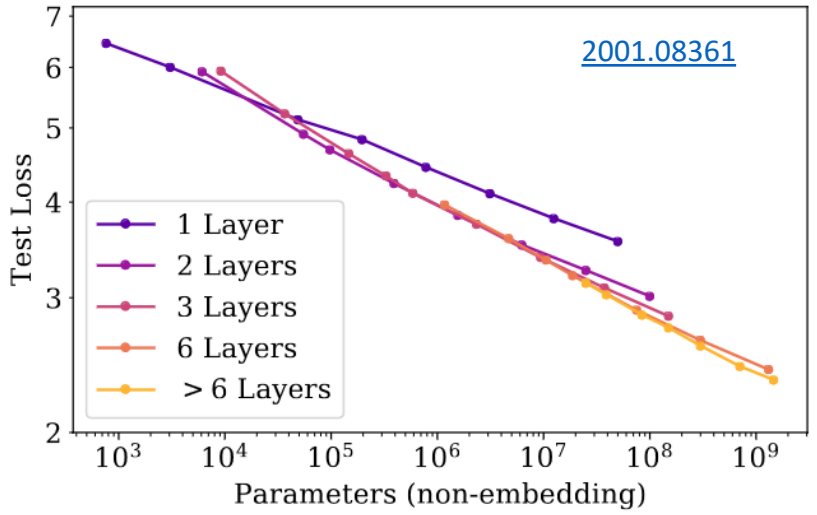
```
1  Translate English to French:        ← task description
2  sea otter => loutre de mer          ← examples
3  peppermint => menthe poivrée        ← examples
4  plush girafe => girafe peluche      ← examples
5  cheese =>                           ← prompt
```

In-Context Learning



The Power of Scale: Large Models, Data, Compute





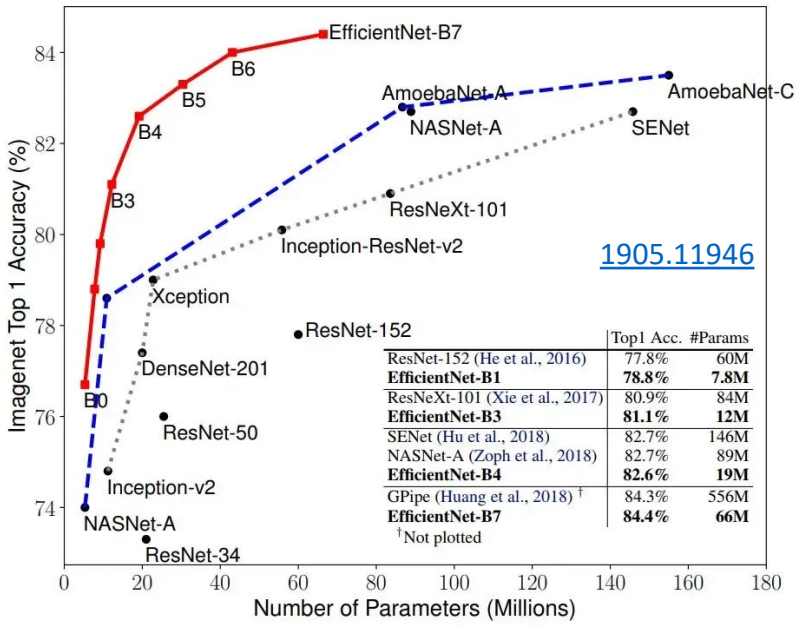
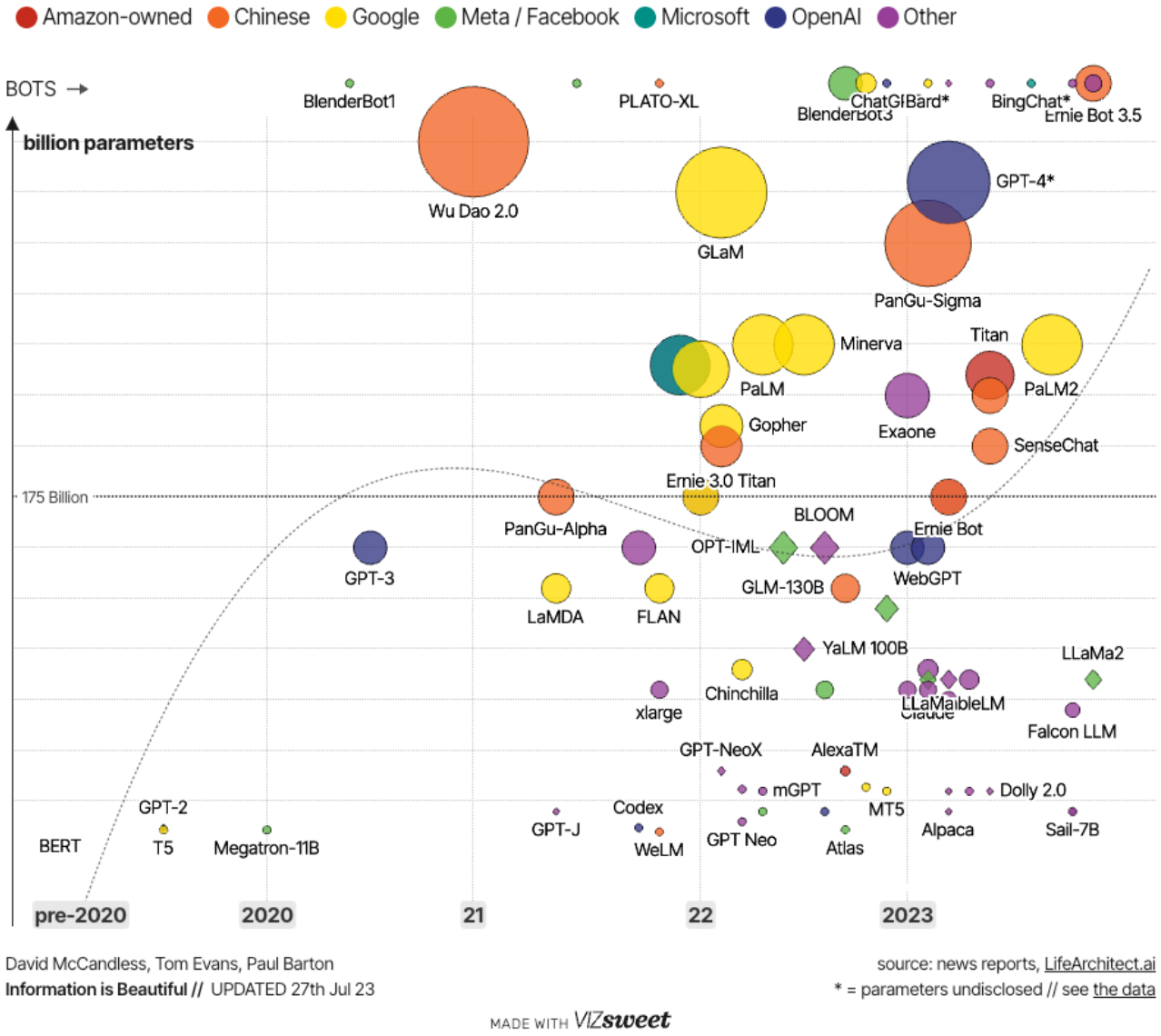


Image credit: D. McCandless, T. Evans, P. Barton

The Power of Scale: Large Models, Data, Compute



Problem with large-scale training

Need a lot of (labelled?) data

Is there a way to train neural networks without the need for huge manually labeled datasets?

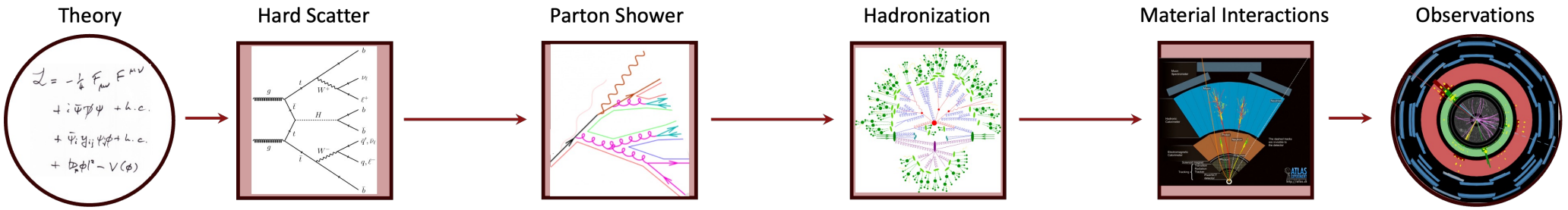
Self-supervised learning

Image credit: D. McCandless, T. Evans, P. Barton

What does this have to do with HEP?

AI and Fundamental Physics Have Similar Aims

Generate plausible, high-dim. data from high-level concepts



AI and Fundamental Physics Have Similar Aims

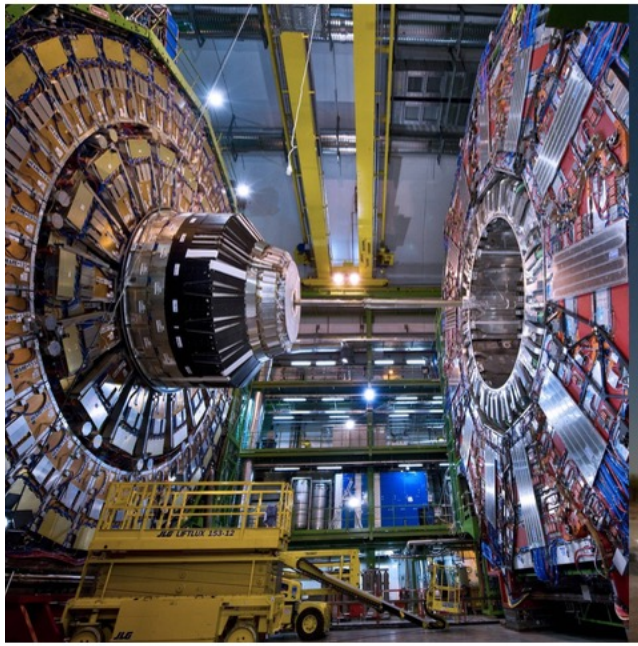
Generate plausible, high-dim data from high-level concepts

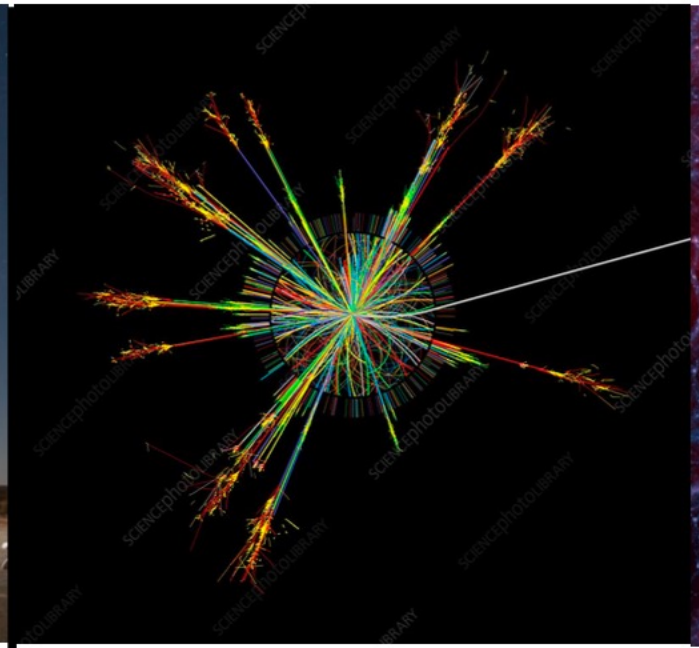
Prompt:
street style photo of a woman selling pho at a Vietnamese street market, sunset, shot on fujifilm

Extract high-level concepts from low-level, high-dim. data

Classification:
A photo of guacamole, a type of food

Massive and High-Quality Scientific Data





Exabytes of Experimental Data from Large-Scale Experiments
→ *much more than used for ChatGPT*

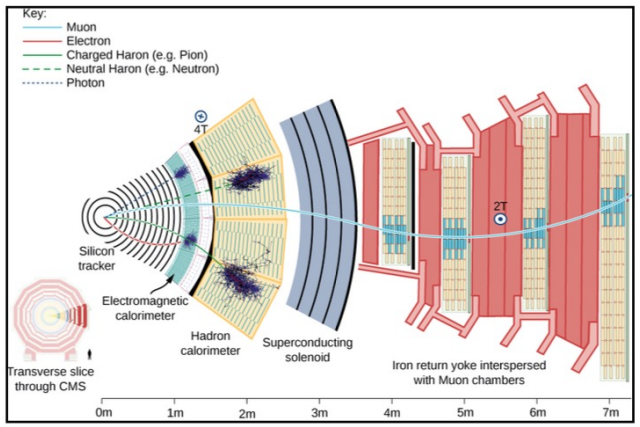
High Quality Simulators allow us to explore new hypothetical models of the universe
→ *perfect training data for AI*

Image credit: L. Heinrich

Sort of already have a HEP Foundation Model?

Reconstruction at a collider turns energy depositions in particle properties

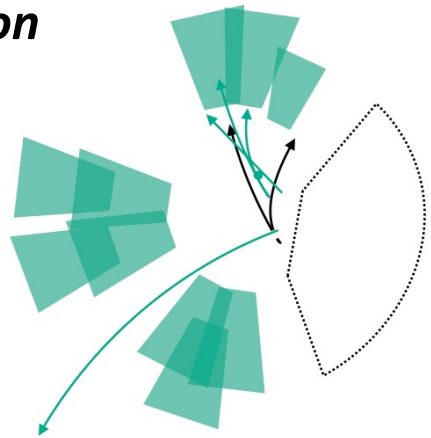
Reconstruction works, and can be tuned (a bit), for essentially all analyses!



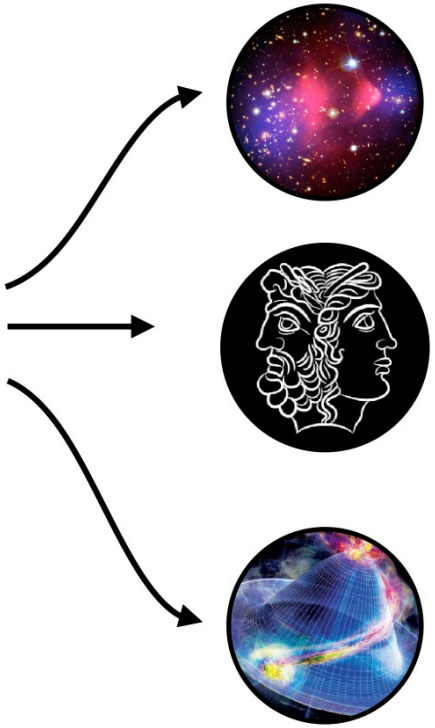
100 M- dimensional multi-modal data

Reconstruction

Multi-modal Pattern Recognition



O(1k-10k) high level patterns



Dark Matter

Matter & Antimatter

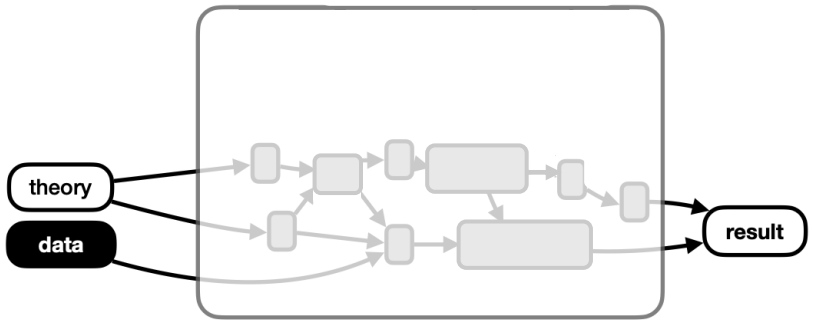
Origin of Mass

Image credit: L. Heinrich

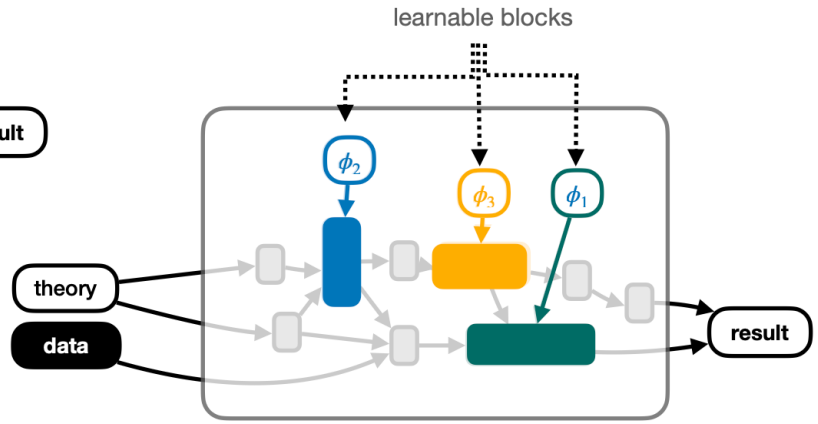
How Big Should the ML Components be?

ML is already been infused into this reconstruction pipeline

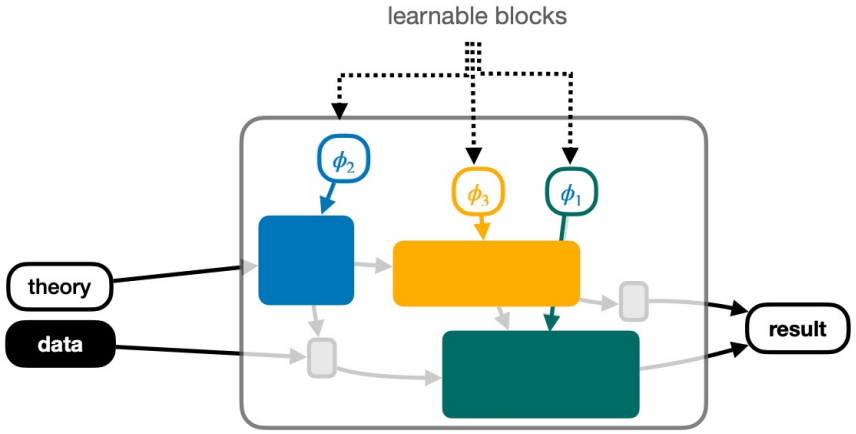
How big / how much ML should we be using?



Human designed algorithms



Mixing some ML with
Human designed algorithms

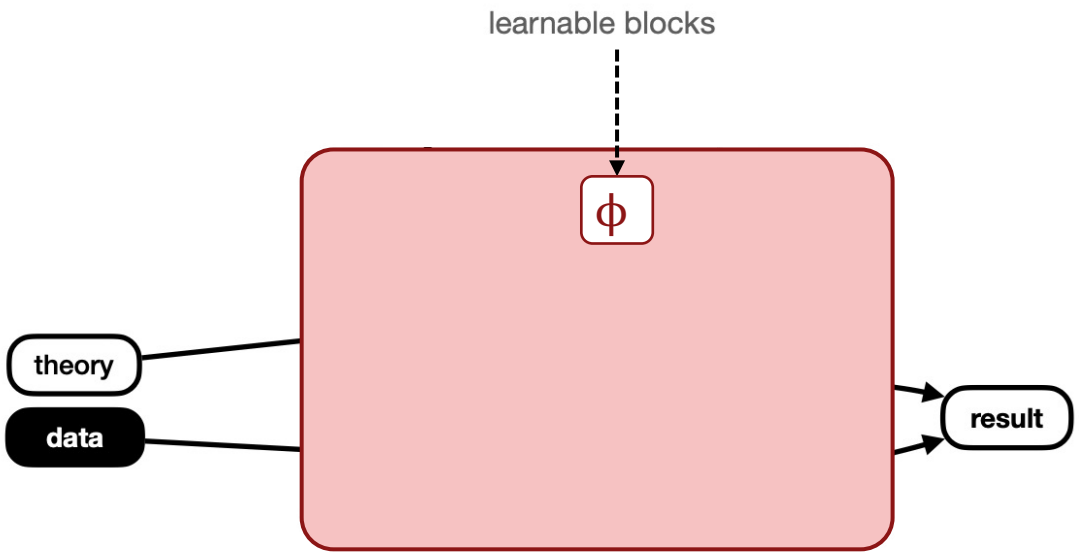


Bigger ML blocks, connected by
small pieces of human designed algos

Does HEP Foundation Model mean one BIG neural Network?

I don't believe this will work…

Or at least it won't work with any "reasonable" model size, data size, compute



Hits → Higgs: Yes? No?

Does HEP Foundation Model mean one BIG neural Network?

Some have tried…

success (if possible) likely requires a huge amount of data

Still enormous gap between using only hits

and using some physics knowledge

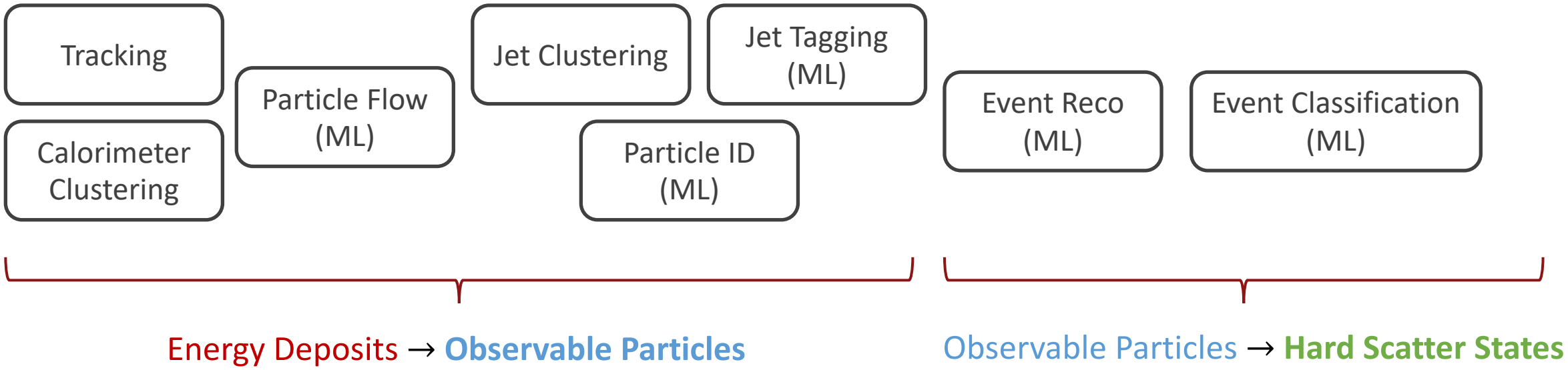
Model	Input	AUC @ n_{train} (k)		
		8	18	38
MLP[a] (*full detector*)	Reco objects	0.960	0.951	0.959
MLP[b] (*only b-jets*)		0.859	0.834	0.848
ParT[c] (*full detector*)	Reco objects	–	–	**0.972**
Higgsformer-small (*only inner tracker*)	Raw detector hits	0.704	0.757	0.779
Higgsformer-small (*augmented, only inner tracker*)		0.721	0.764	0.792

Transformers classifiers operating only on hits

2508.19190

The Structure of Particle Physics

"Inverting the Generative Model":
Choosing our intermediate representations based on known physics



There are raw data challenges in HEP, tracking, clustering, particle flow, neutrino physics, cosmology…

Deep learning on low level data is part of the story, but not the whole story

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 com
cons
perf
com
shor
that
othe
psyc
approach tends to complicate methods in ways that make them less suited to taking advantage of general methods leveraging computation. ...

Researchers seek to leverage their human knowledge of the domain, but the only thing that matters in the long run is the leveraging of compute

The Unreasonable Effectiveness of Mathematics in the Natural Sciences

Richard Courant Lecture in Mathematical Sciences delivered at New York University, May 11, 1959

EUGENE P. WIGNER
Princeton University

"and it is probable that there is some secret here which remains to be discovered." (*C. S. Peirce*)

Foundation Model For Science?

Defining Foundation Models for Computational Science: A Call for Clarity and Rigor

Youngsoo Choi, Siu Wun Cheung, Youngkyu Kim, Ping-Hsuan Tsai, Alejandro N. Diaz, Ivan Zanardi, Seung Whan Chung, Dylan Matthew Copeland, Coleman Kendrick, William Anderson, Traian Iliescu, Matthias Heinkenschloss

SPECIALIZED FOUNDATION MODELS STRUGGLE TO BEAT SUPERVISED BASELINES

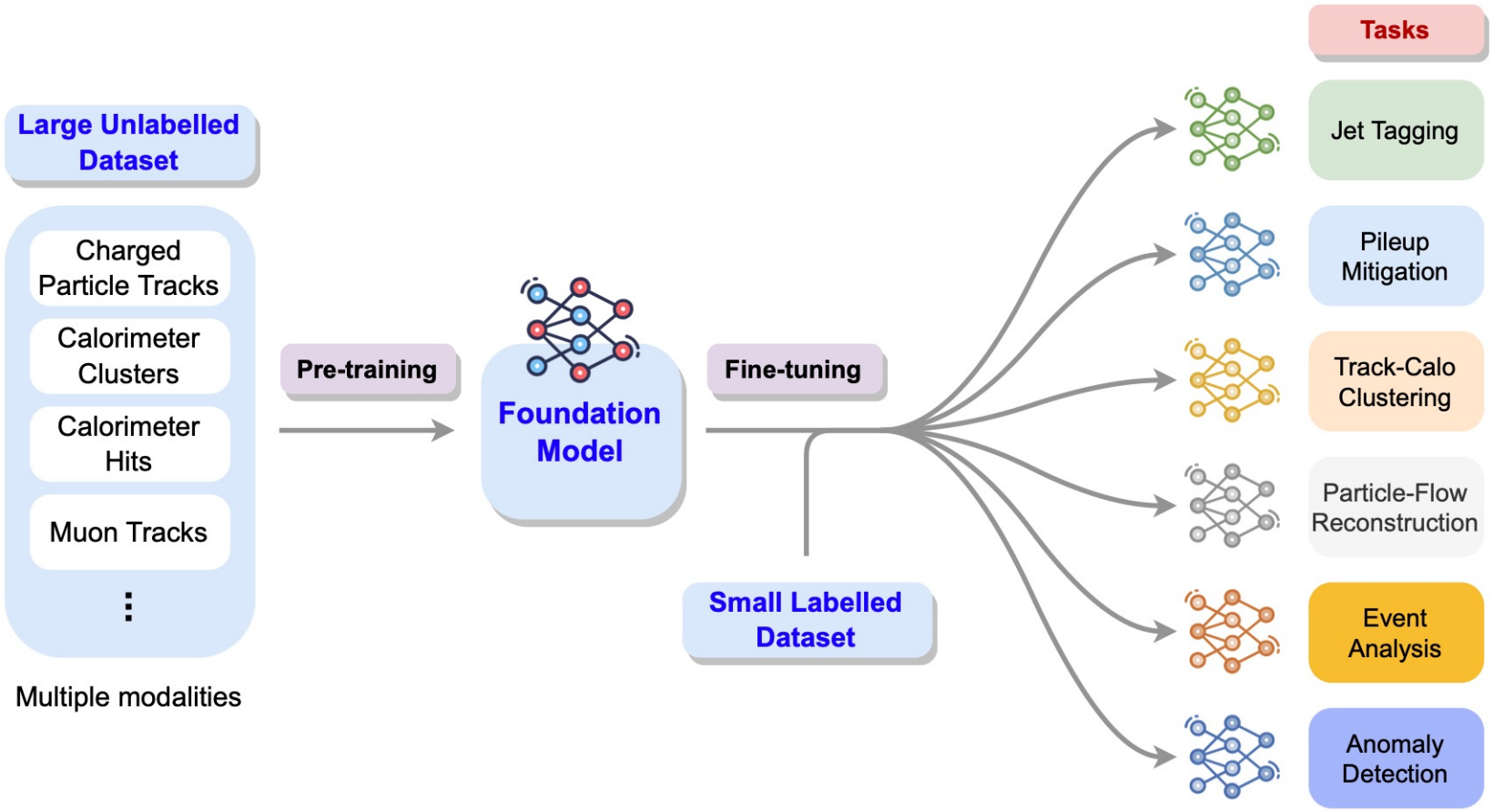
Zongzhe Xu,* Ritvik Gupta,* Wenduo Cheng, Alexander Shen, Junhong Shen
Carnegie Mellon University
{zongzhex,ritvikgu,wenduoc,ajshen,junhongs}@andrew.cmu.edu
* denotes equal contribution; order decided by coin flip

Ameet Talwalkar
Carnegie Mellon University & Datadog, Inc.
talwalkar@cmu.edu

Mikhail Khodak
Princeton University
mkhodak@cs.princeton.edu

Foundation Model For Fundamental Physics

Klein, …, **MK**, 2401.13537



Different data analysis goals often share similar pattern recognition needs

Can we build tools that can generalize across tasks & even experiments?

Why Use this Strategy in Fundamental Physics?

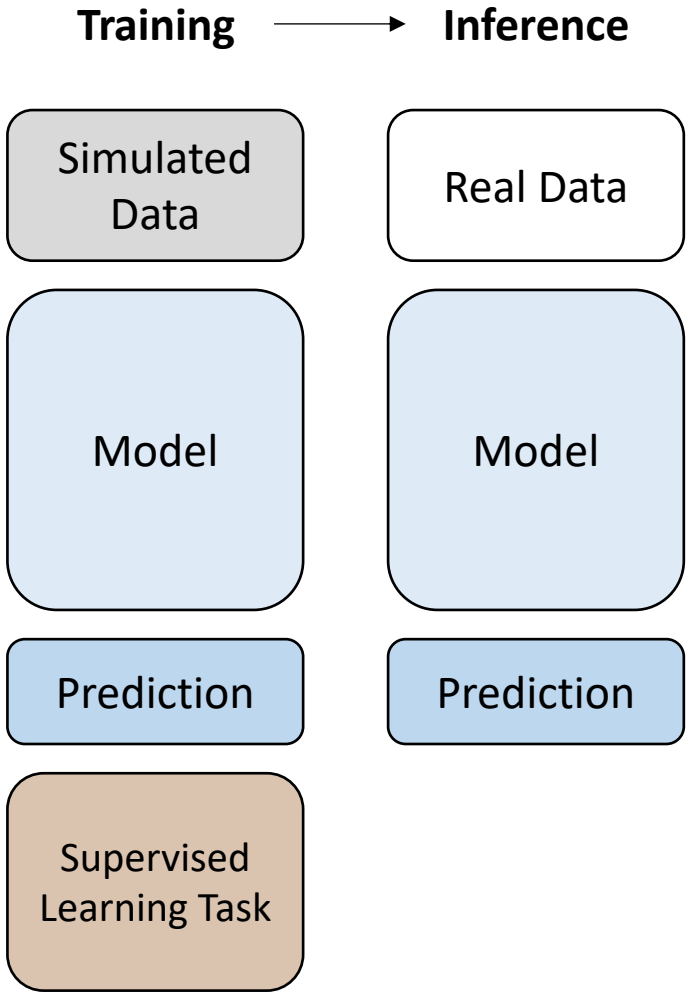
Reusable – One backbone used for several tasks.
Fine tune for different data analyses, experiments, detector designs

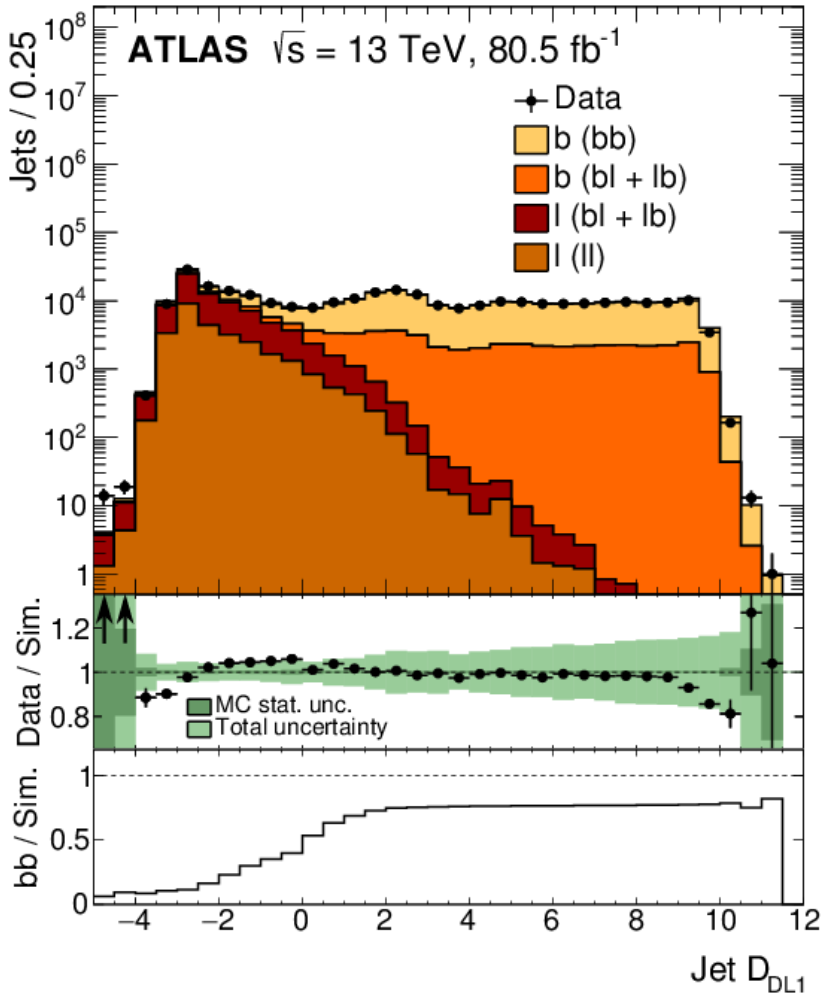
Train on huge real data – Leverage experimental data

Uncertainty reduction – Reduce dependence on simulation-based training

Leverage multi-modal methods– Combine data from different detectors to address more complex tasks

The Challenge of Systematic Uncertainties



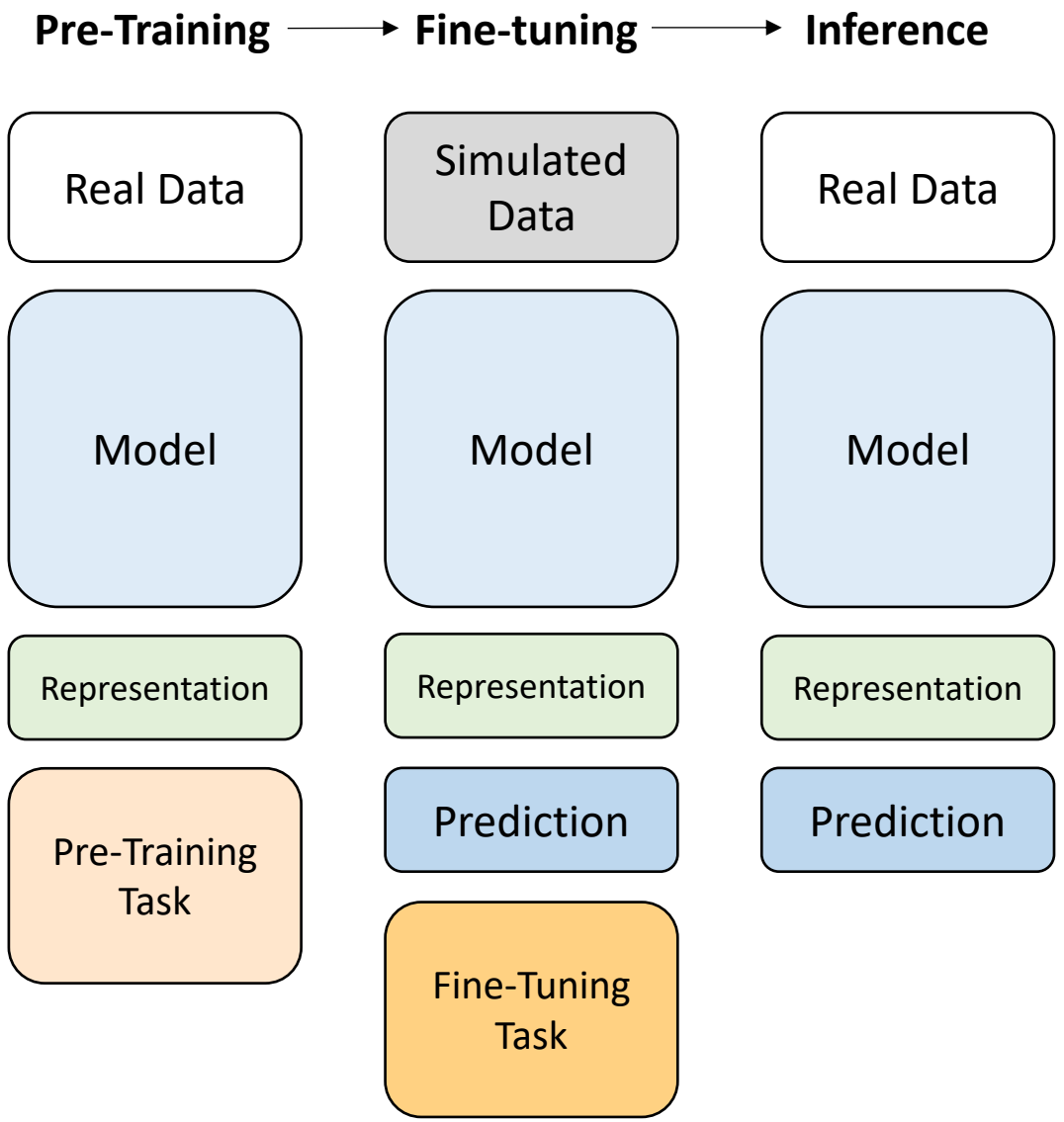


Uncertainty from distribution shift between sim. & data

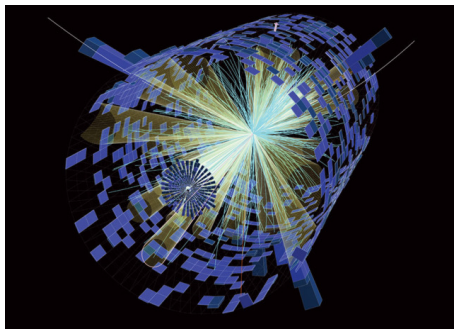
Can Pre-training Strategies Help?

Learn representations with reduced sensitivity to simulator variations?

Pre-training on real data?



Towards Optimizable and More Automated Data Analysis



Object-level (Reconstruction)

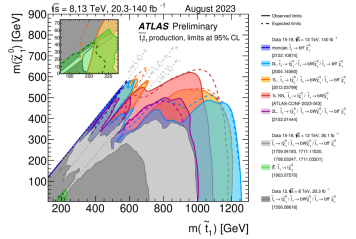
Foundation Model
Structured
Pre-trained
Fine-tunable

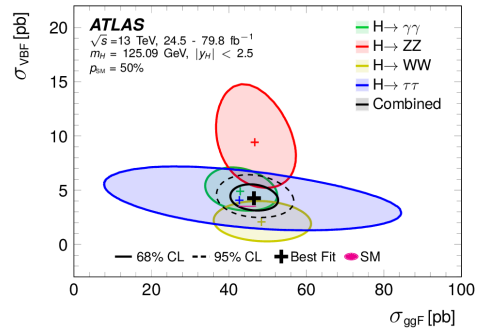
Population-level

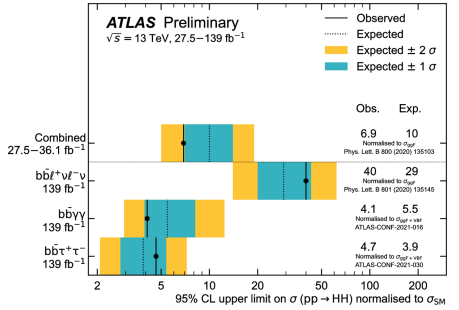
Event Selection

Signal Discrimination

Statistical analysis







Pre-trained, generalized tools → Fully optimizable analysis

End-to-End Fine-Tuning for each data analysis

More optimal, rapid, and broad searches & measurements

What this approach can mean

Never have to retrain my own neural networks from scratch

- Existing pre-trained models would already be near optimal, no matter the task

Practical large scale Deep Learning even in **very few example regime**

If the information is embedded in a space where it becomes linearly accessible, **very simple analysis tools are enough** for downstream analysis

Could it even mean?

Single pre-trained model which can **operate on any input data type**

- I no longer need to worry about what network to use on some data

Deep understanding of the data, informed by cross-modal information

- A downstream task could be specified with just a few examples

Representation Learning

"I stand at the window and see a house, trees, sky. Theoretically I might say there were 327 brightnesses and nuances of colour. Do I have "327"? No. I have sky, house, and trees."

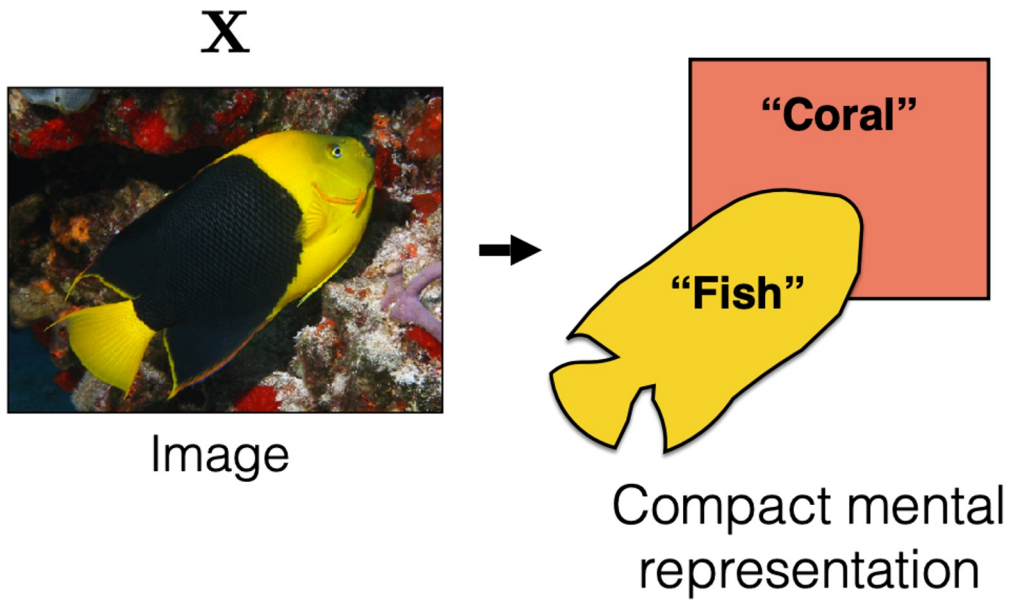
— Max Wertheimer, 1923

“AI must [...] learn to identify and disentangle the underlying explanatory factors hidden in the observed milieu of low-level sensory data”

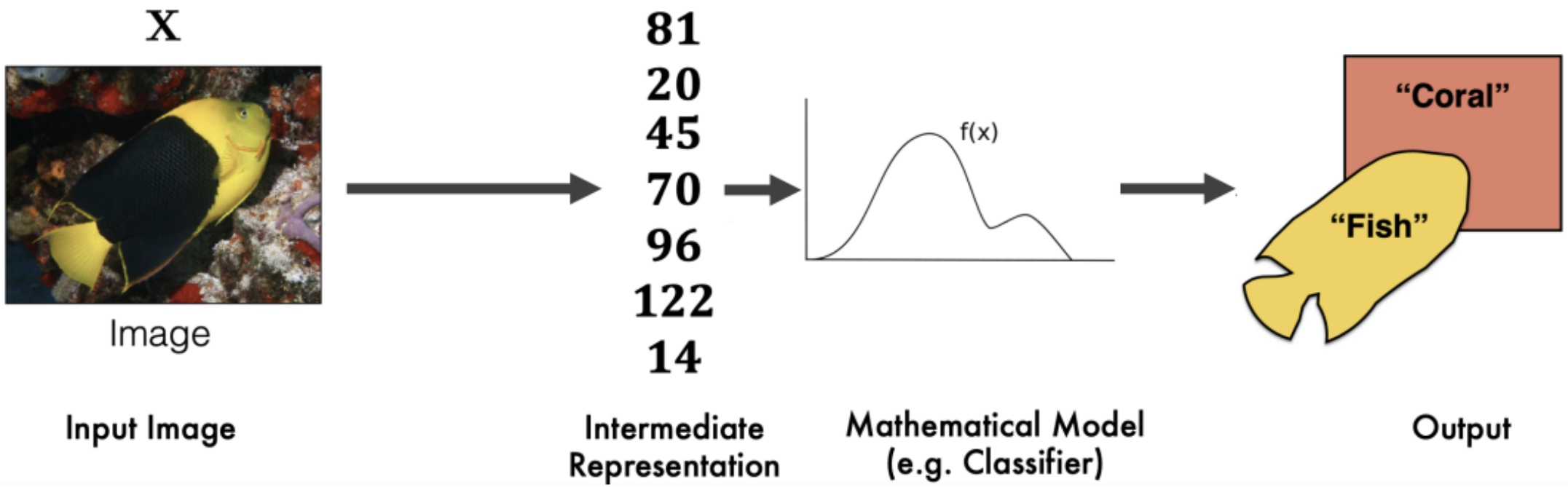
“representation learning [means] learning representations of the data that make it easier to extract useful information when building classifiers or other predictors”

Y. Bengio, A. Courville, P. Vincent,
Representation learning: A Review and New Perspectives, 2012

Representations



Representations



What Makes a Good Representation?

Make subsequent problem solving easy

Compact

- Contains only the essential information, removing redundant details.

Predictive

- Being able to take actions that achieve desirable future outcomes.

Disentangled

- Each dimension represents a distinct attribute.

[For more discussion, See e.g. "Representation Learning", Bengio, Courville, Vincent 2013]

Interpretable

- It would be good if we human can understand it!

Transferability

- Ideally, one rep. to make all later problems (that human cares about) easy!

What is a Representation?

Representation z is an abstraction of data x mapped by an encoder function f

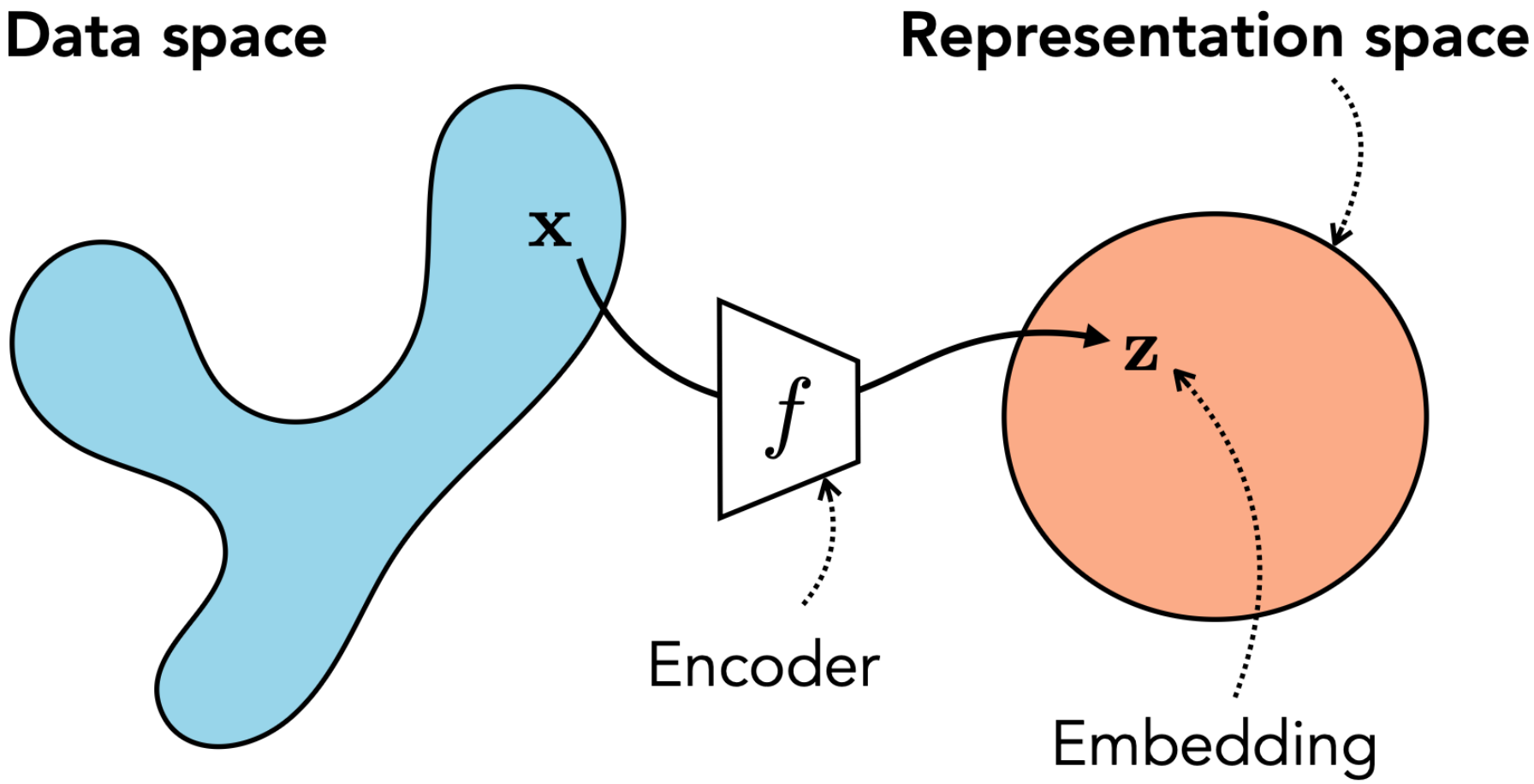
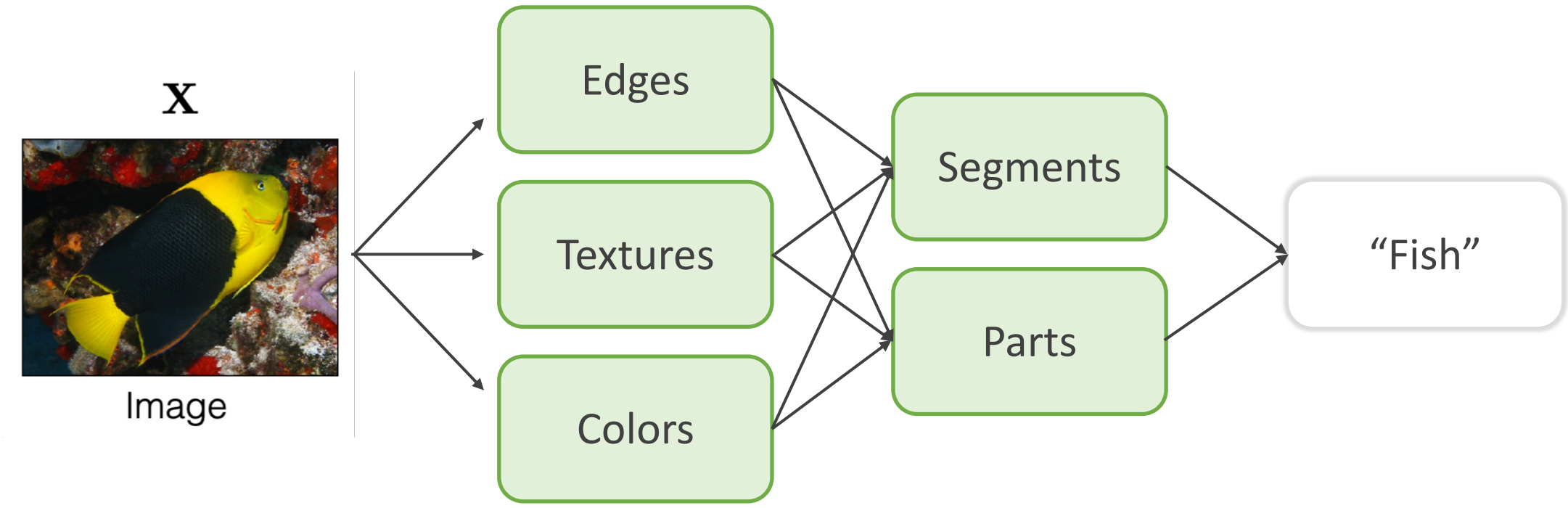
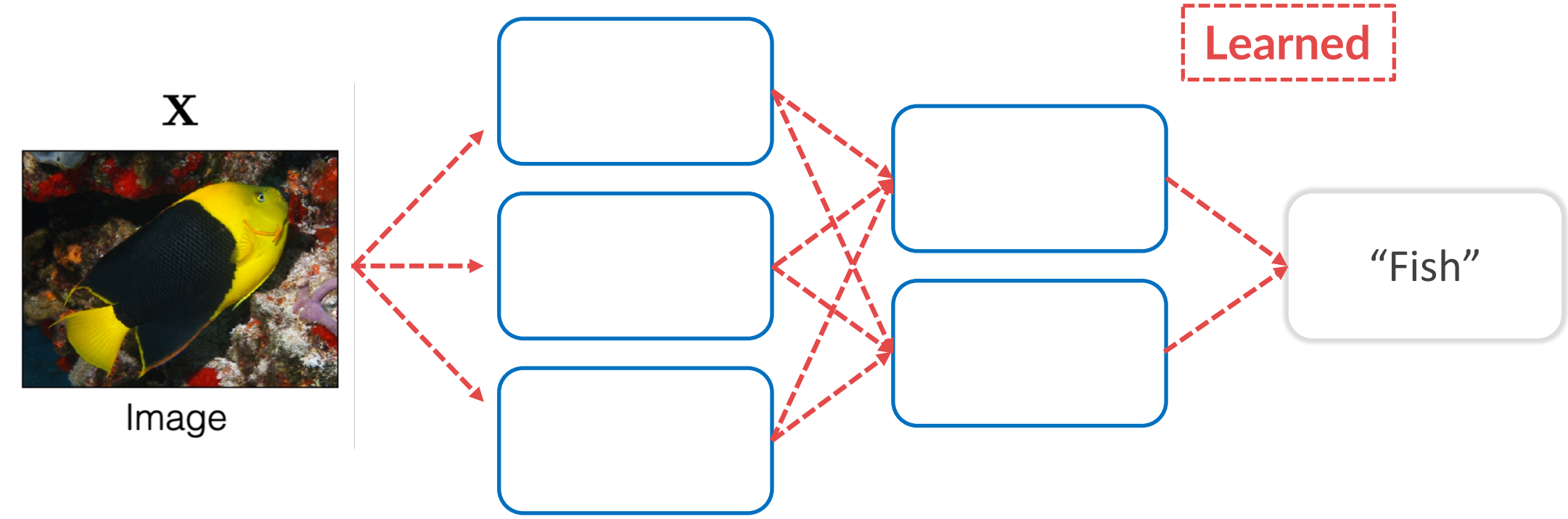


Image Credit: Foundations of Computer Vision

Classic Object Detection



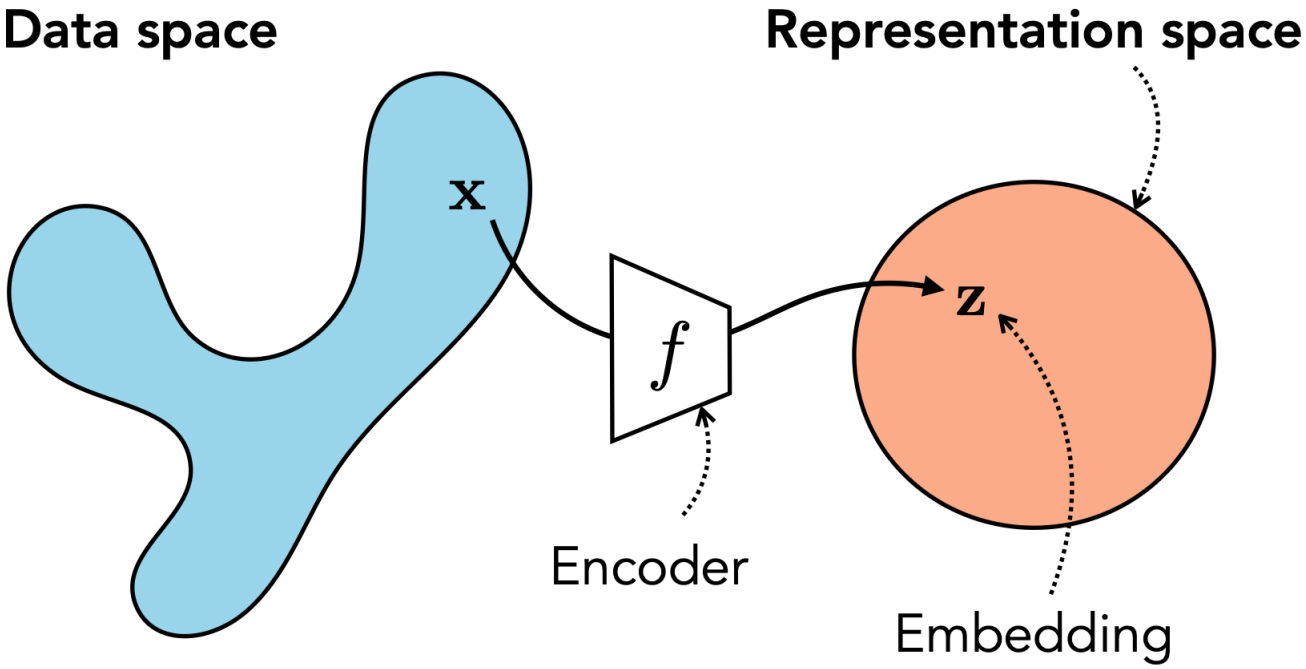
Deep Learning



What is Representation Learning?

Typically find **encoder** through optimization of an **objective** or loss $L[\cdot]$

Representation learning is about designing encoders and objectives

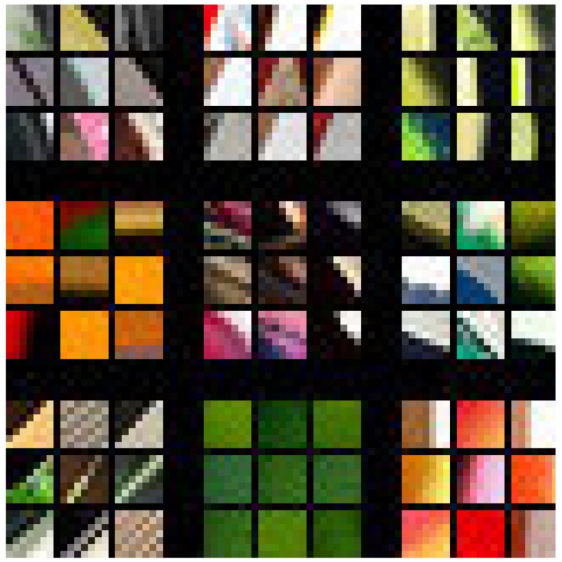


$$\min_f L[f(x)]$$

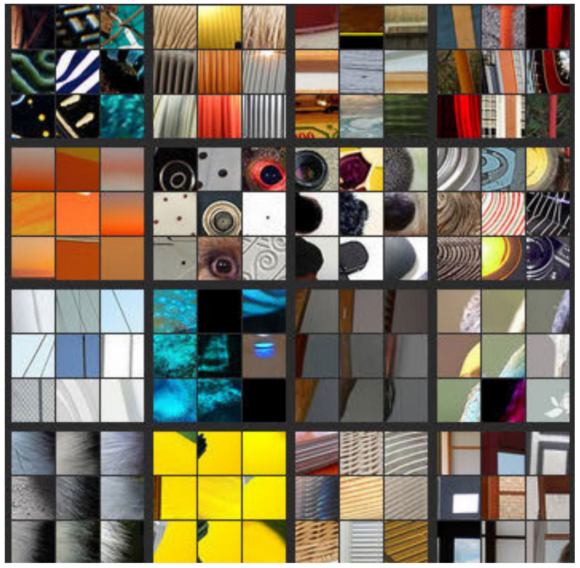
Image Credit: Foundations of Computer Vision

What Do Deep Nets Internally Learn?

Image patches that activate several CNN neurons most strongly

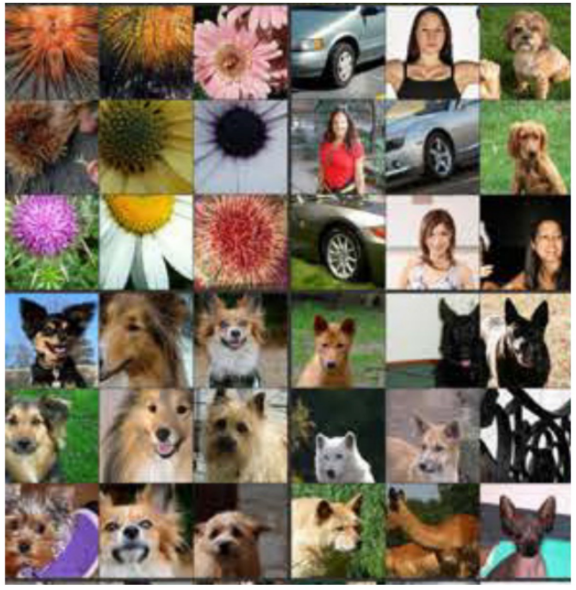


Layer 1



Layer 2

Layer 3

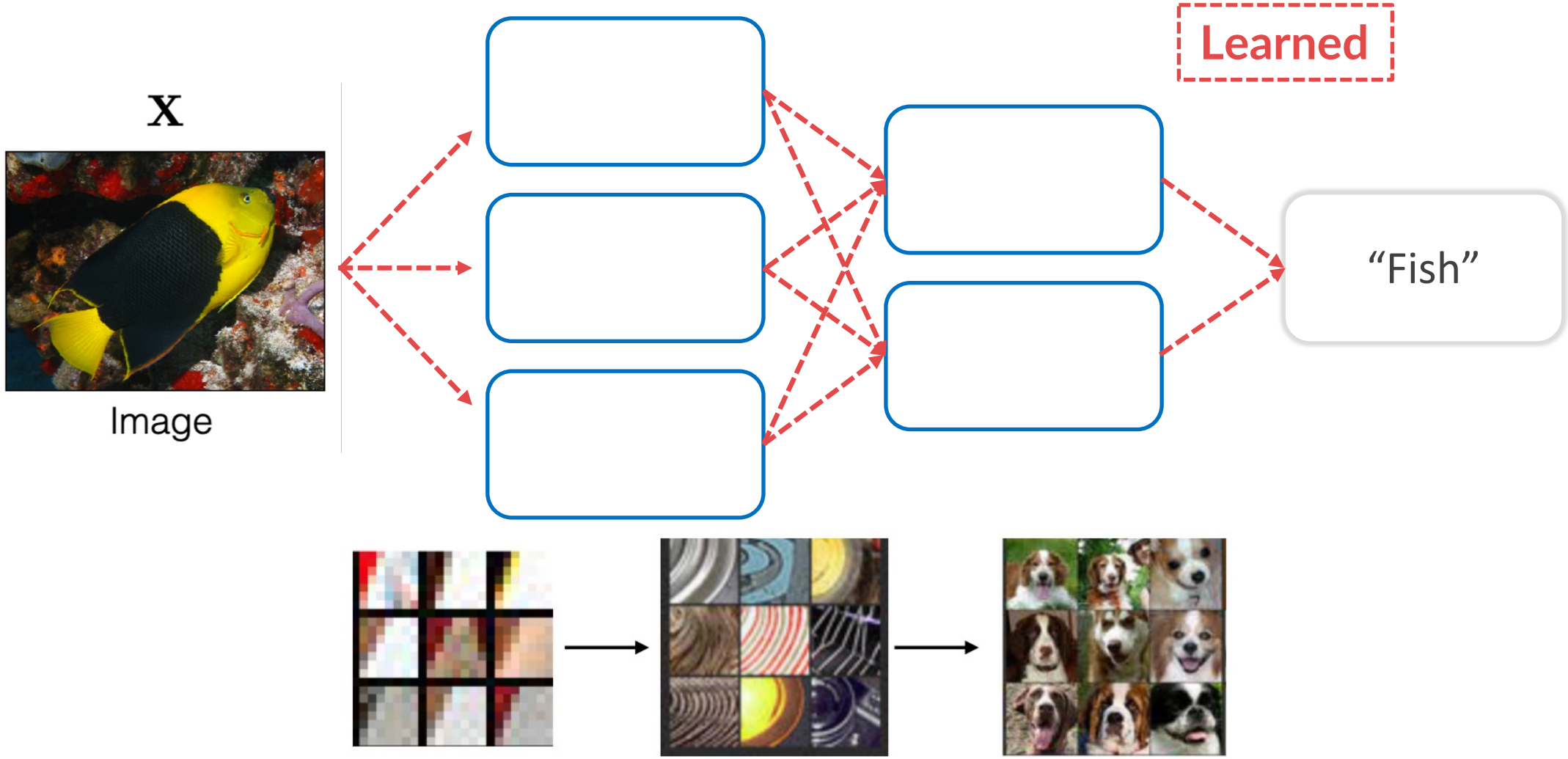


Layer 5

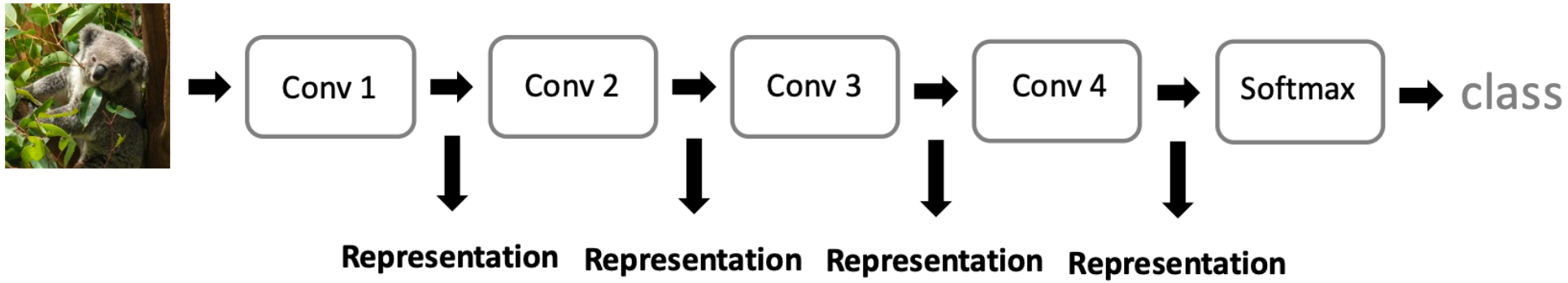
1311.2901

Deep Learning

Learned Intermediate Representations represent those in traditional pipeline



Representations from Deep Neural Networks



Hard to interpret

Build multiple levels of representations

Reduce domain knowledge and feature engineering

Deep representations are transferrable

Learned Representations are Transferrable

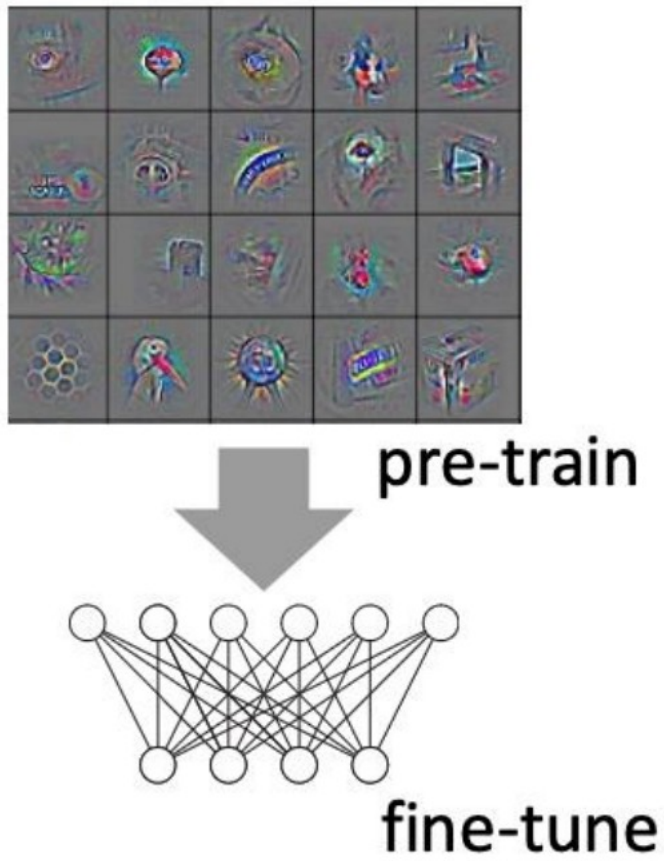
One of the most important discoveries of the deep learning revolution

Transfer learning:

- Pre-train on large-scale data
- Fine-tune on small-scale data

- Enabled deep learning for small datasets
- Revolutionized computer vision

- Taking a supervised pre-trained ResNet and fine-tuning it for new tasks /dataset totally changed how people could used models



DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition, 1310.1531
Visualizing and Understanding Convolutional Networks, 1311.2901
CNN Features off-the-shelf: an Astounding Baseline for Recognition, 1403.6382

Slide credit: Isola, Freeman, Li, Intro to Computer Vision

How do you learn a representation?

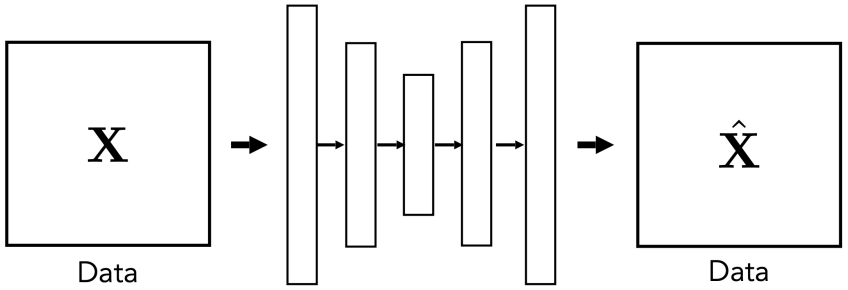
Broadly two approaches, ***Compression*** and ***Prediction***

Learning Method	Learning Principle	Short Summary
Autoencoding	Compression	Remove redundant information
Contrastive	Compression	Achieve invariance to viewing transformations
Clustering	Compression	Quantize continuous data into discrete categories
Future prediction	Prediction	Predict the future
Imputation	Prediction	Predict missing data
Pretext tasks	Prediction	Predict abstract properties of your data

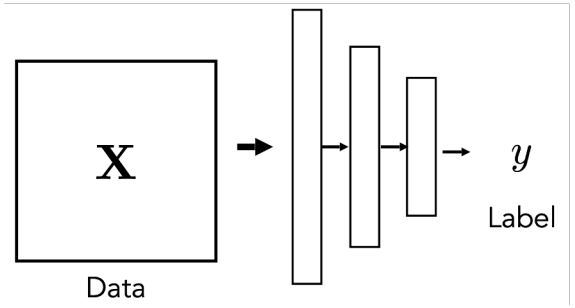
Torralba, Isola, Freeman, Foundations of Computer Vision

How do you learn a representation?

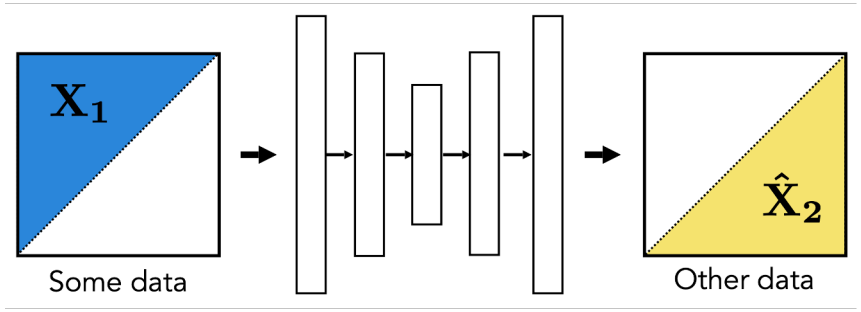
Data Compression



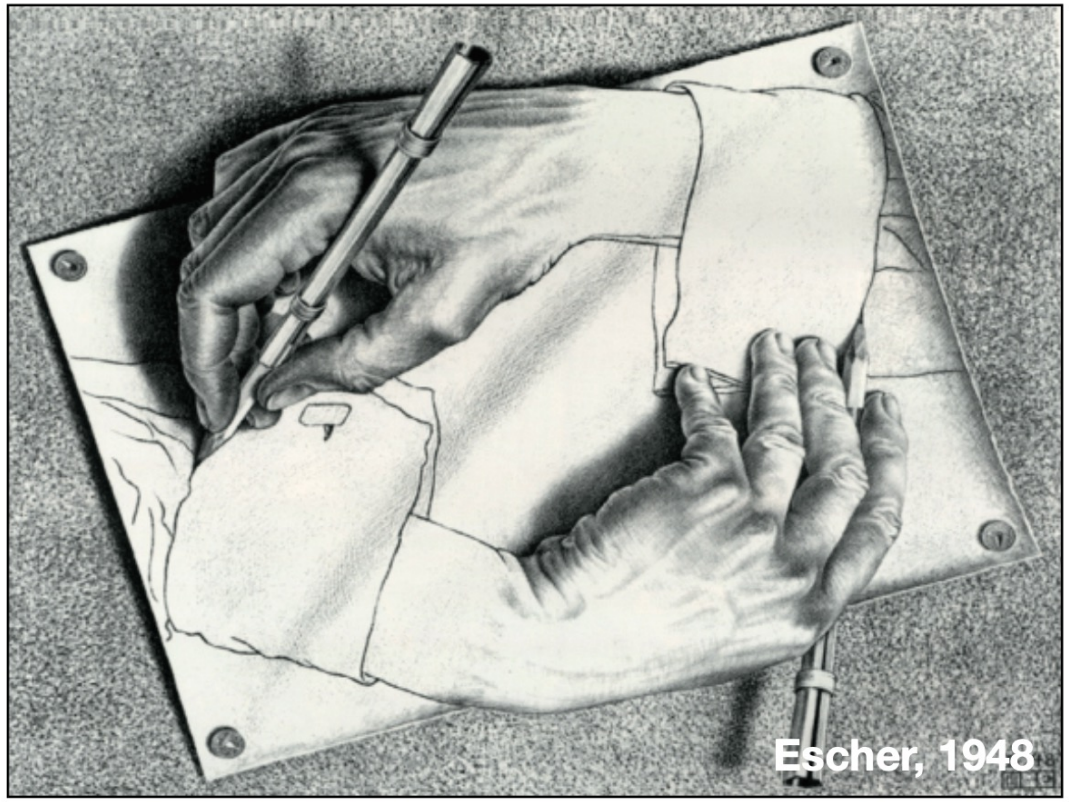
Label Prediction



Data Prediction, i.e. Self-Supervision



Self-Supervised Learning



Common trick:

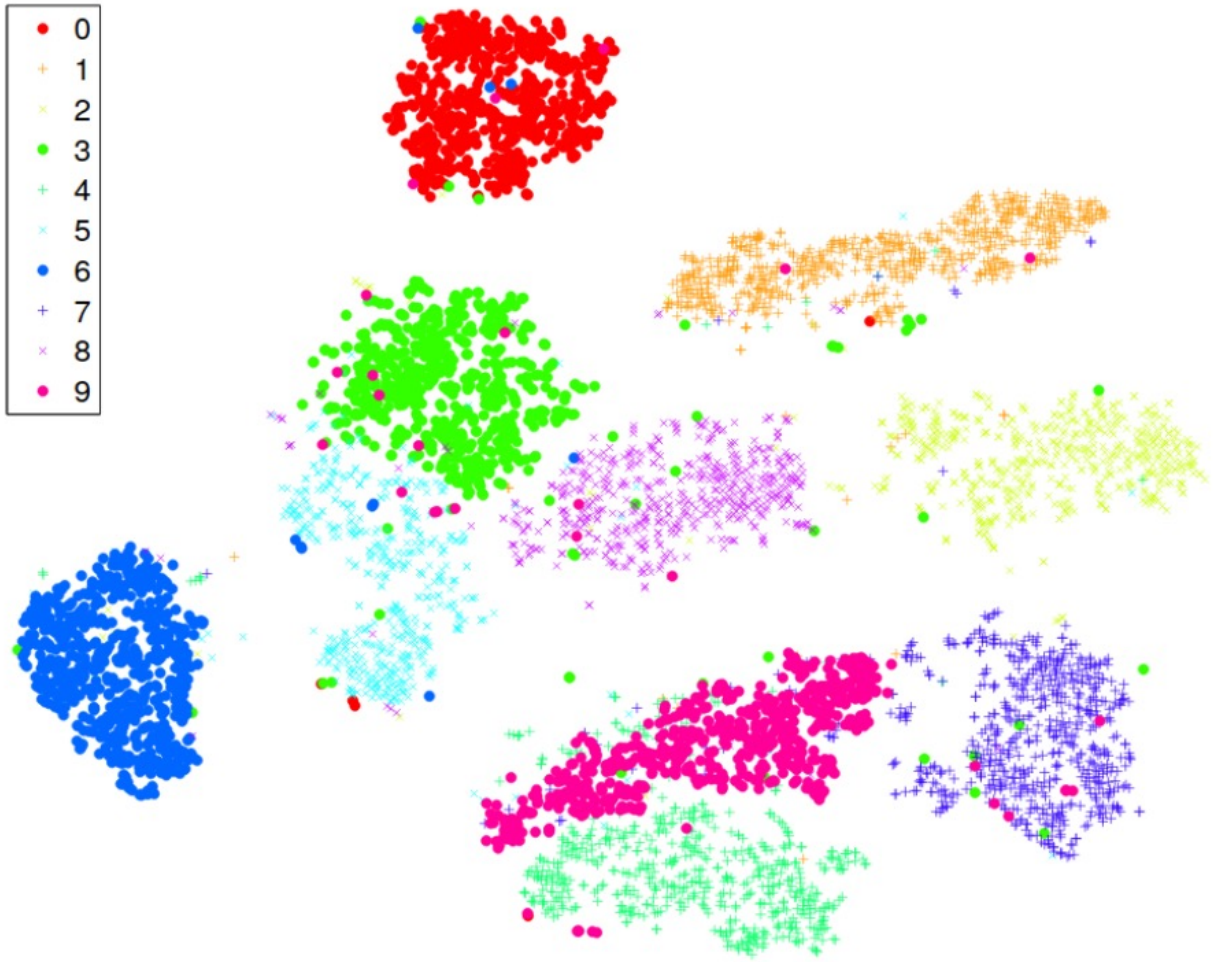
Convert “unsupervised” problem into “supervised” empirical risk minimization

Do so by cooking up “labels” (prediction targets) from the raw data itself — called **pretext task**

How to evaluate the quality of a representation?

Dimensionality Reduction & Visualization

tSNE visualization
of MNIST dataset



How to evaluate the quality of a representation?

Dimensionality Reduction & Visualization

Figure 1: **Visualization of the first PCA components.** We compute a PCA between the patches of the images from the same column (a, b, c and d) and show their first 3 components. Each component is matched to a different color channel. Same parts are matched between related images despite changes of pose, style or even objects. Background is removed by thresholding the first PCA component.

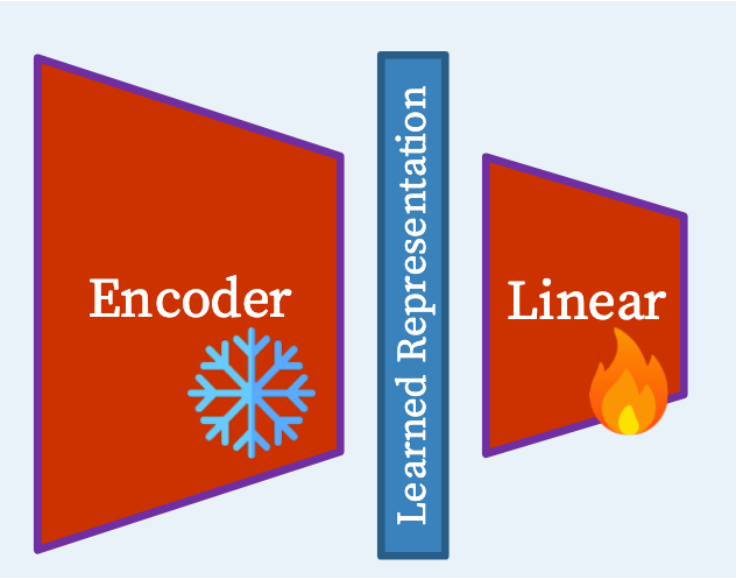
2304.07193

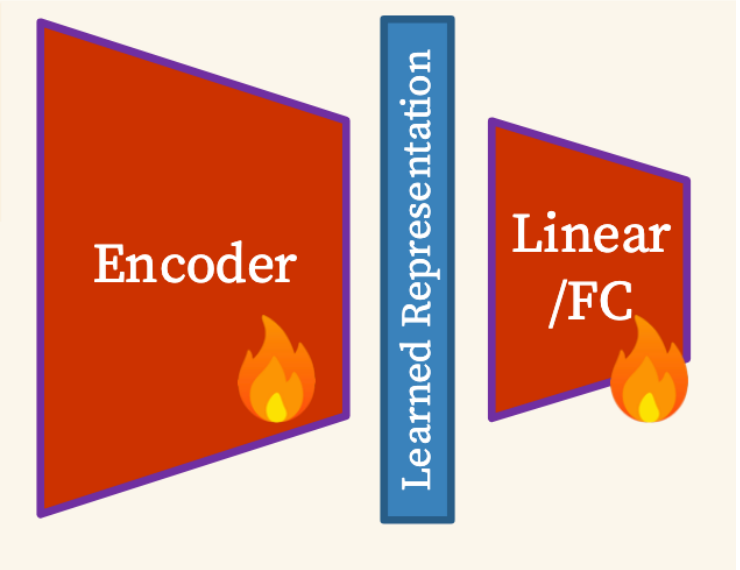
How to evaluate the quality of a representation?

Check downstream task performance of features

Train small “linear probe” on top of representation

Fine-tune network for some downstream task

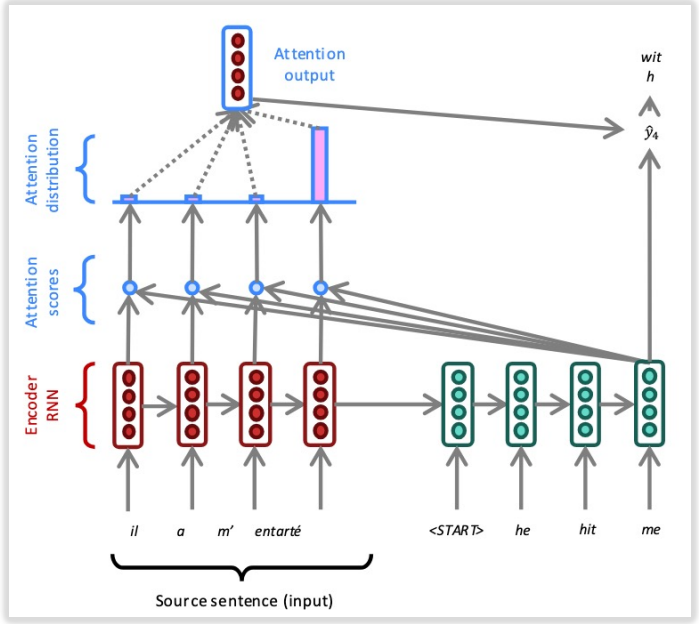


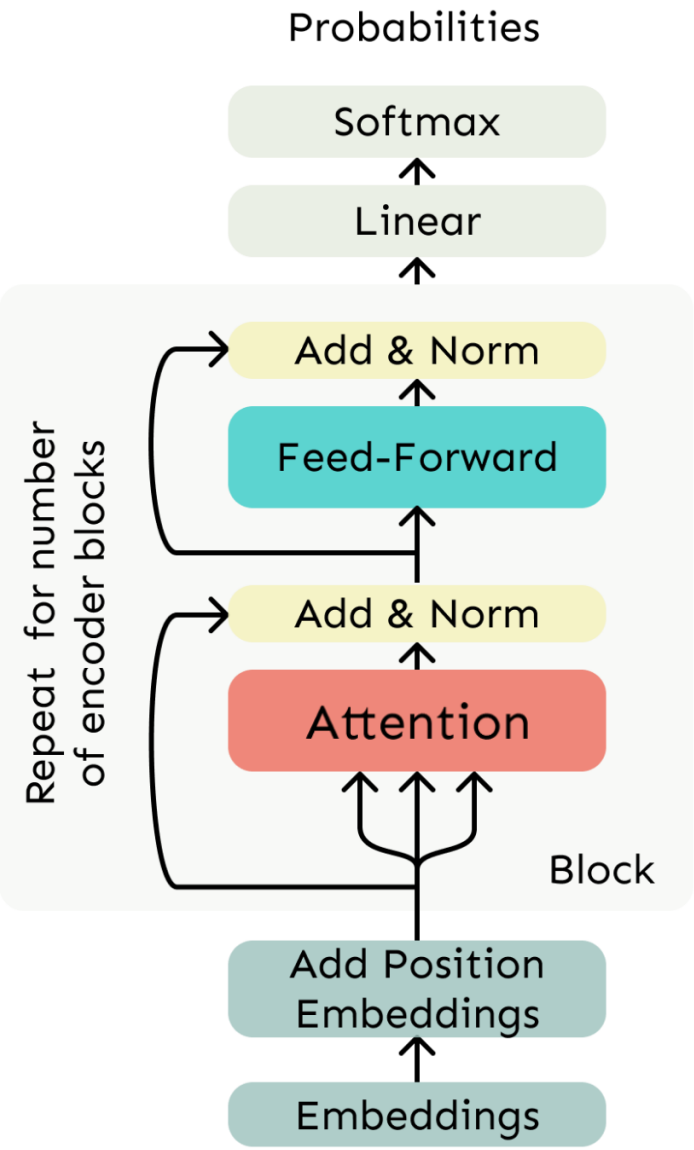


Transformers

Transformers

Starting from the idea of using **attention** to enable recurrent neural networks to look across inputs while processing a sequence





Question: why do we need RNN at all?

[Attention is all you need]

Scaled Dot-Product Attention

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V$$

where

$$Q \in \mathbb{R}^{n \times d}$$
$$K \in \mathbb{R}^{m \times d}$$
$$V \in \mathbb{R}^{m \times d_v}$$

Query

$$Q = \begin{pmatrix} \vec{q}_1 \\ \vdots \\ \vec{q}_n \end{pmatrix}_{n \times d}$$

Key

$$K = \begin{pmatrix} \vec{k}_1 \\ \vdots \\ \vec{k}_m \end{pmatrix}_{m \times d}$$

Value

$$V = \begin{pmatrix} \vec{v}_1 \\ \vdots \\ \vec{v}_m \end{pmatrix}_{m \times d_v}$$

Project input **Query** onto **Key** to compute weights for corresponding **Value**

Return the weighted value

Scaled Dot-Product Attention

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V$$

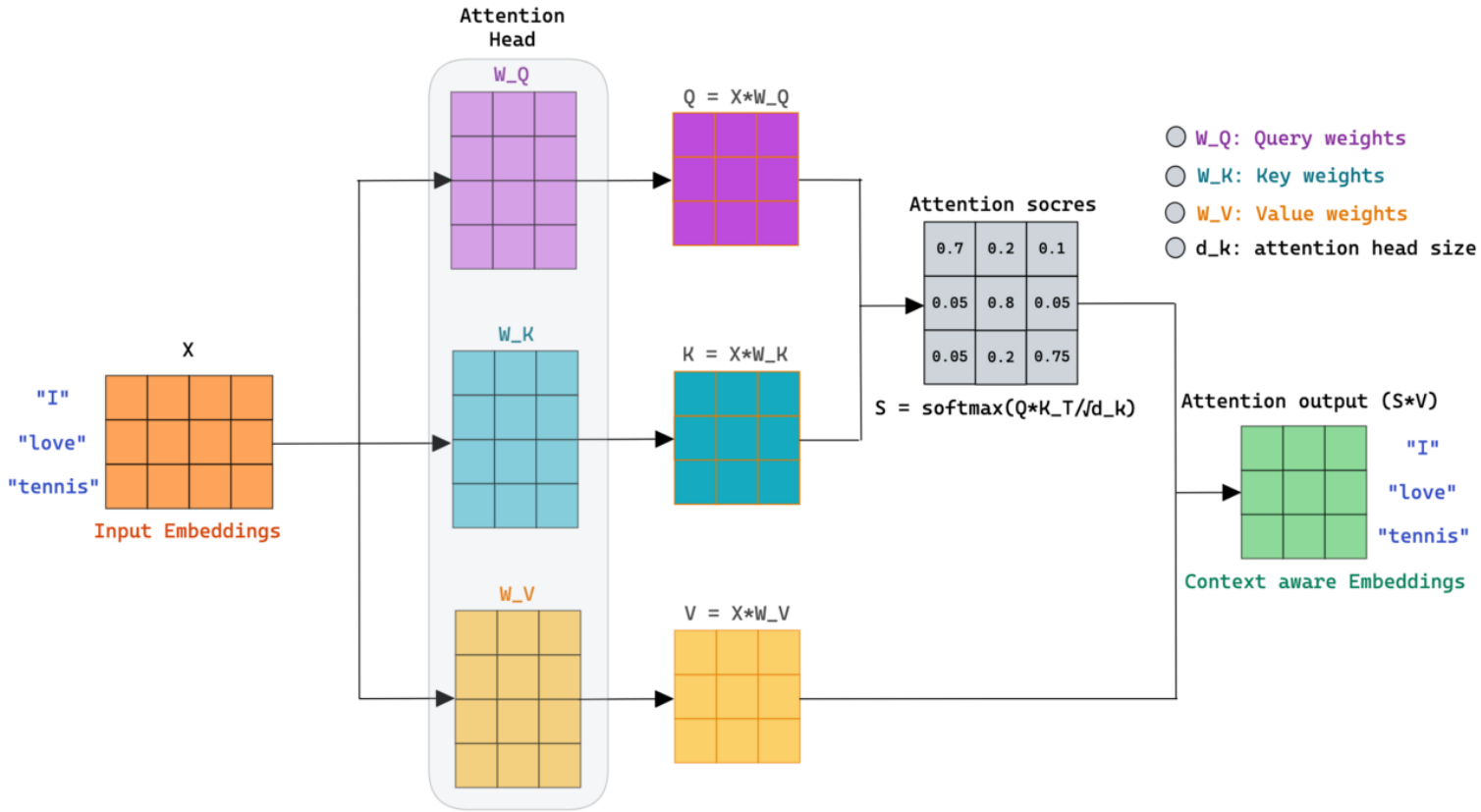
where $Q \in \mathbb{R}^{n \times d}$, $K \in \mathbb{R}^{m \times d}$, $V \in \mathbb{R}^{m \times d_v}$

Self-Attention:
Use input X to define Q,K,V

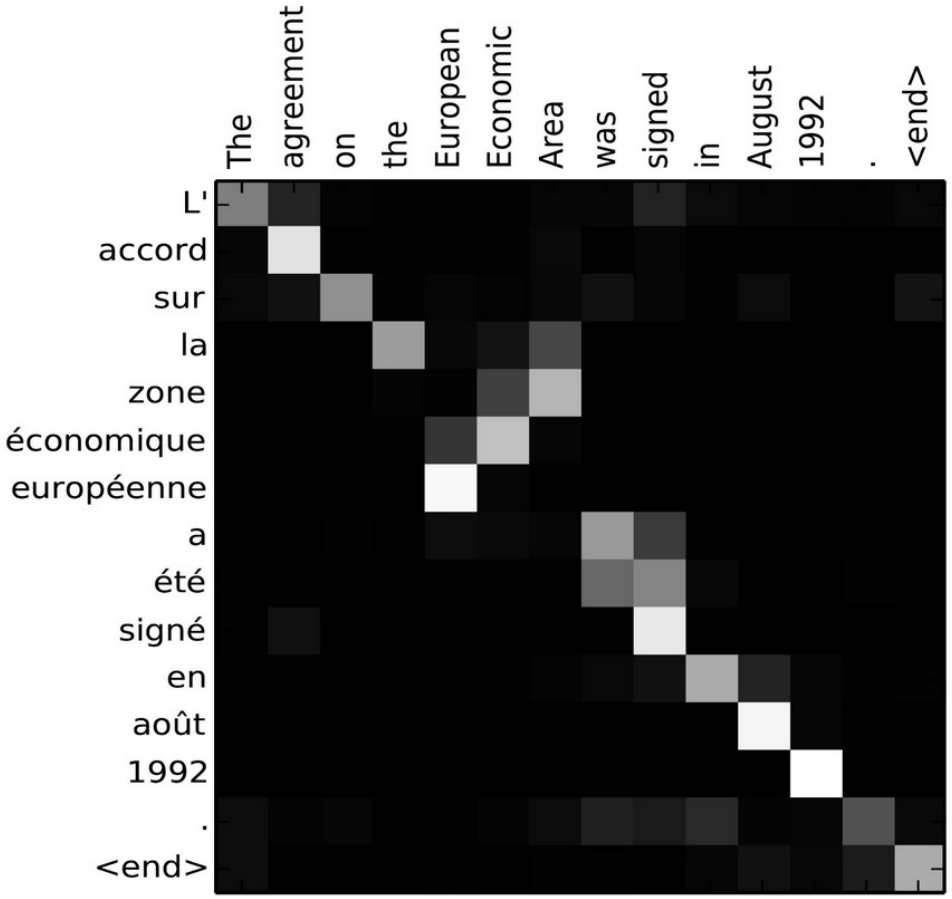
$$Q = XW_Q$$

$$K = XW_K$$

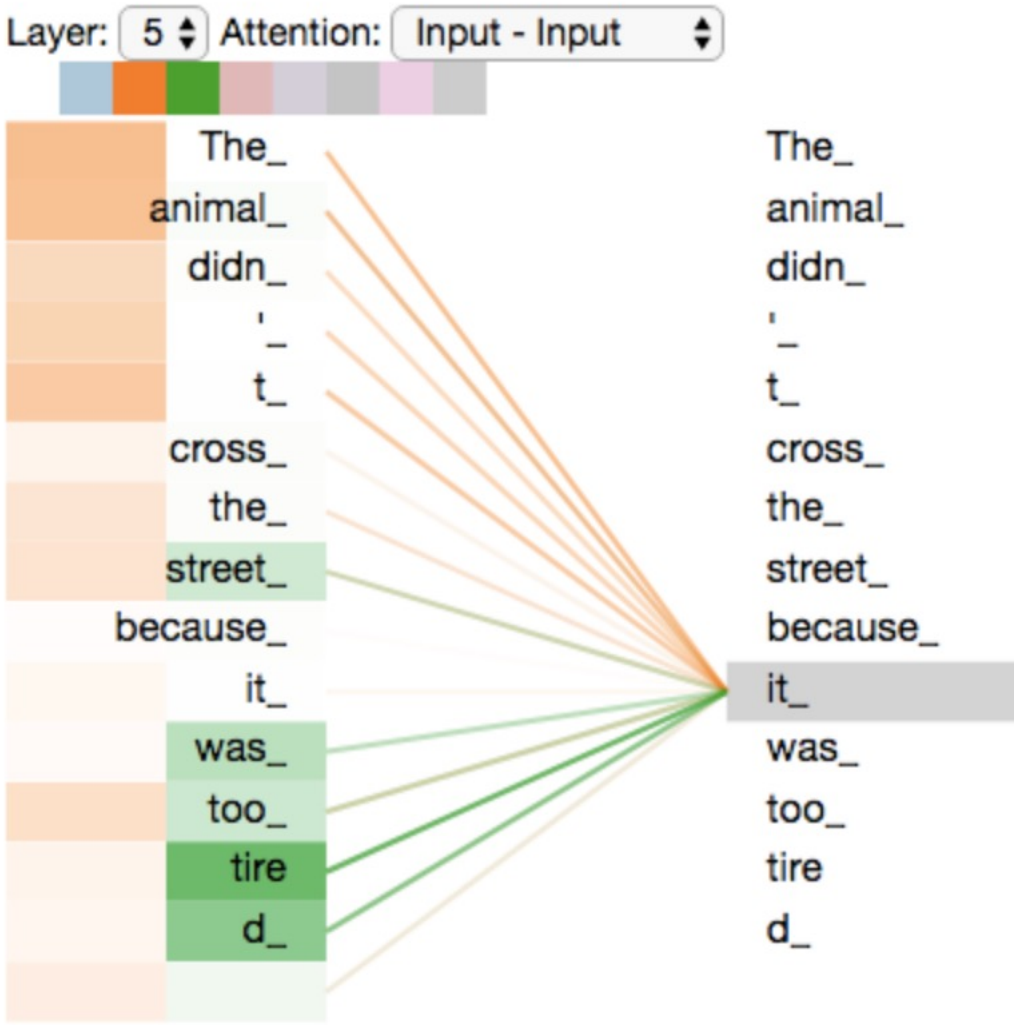
$$V = XW_V$$



Attention Visualization



1409.0473



BertViz

Transformer Encoders vs Decoders

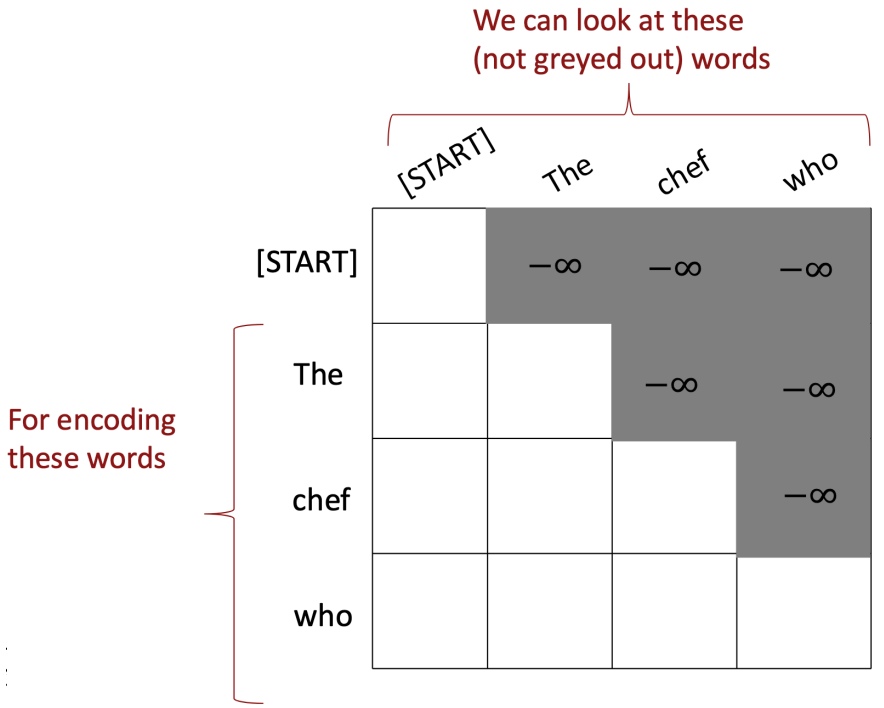
Transformer **decoder** uses *causal masking*

- When processing an input, can only look at previous inputs
- Can only "look into the past"

Transformer **encoder** doesn't use masking

- "Bi-directional" context, i.e. can look at all the inputs at every step
- Can "look into the future"

Causal Masking



Data Order

Self-attention is permutation invariant

- Great if your data is a set … Not so great for sequences

If we want / need order information must build it in ourselves,
we need to encode the order of the sentence in our keys, queries, and values.

Represent each sequence index as a vector

$$p_i \in \mathbb{R}^d, \text{ for i} \in \{1,2, \dots , n\} \text{ are position vectors}$$

- Easy to incorporate this info into our self-attention block: add p_i to our inputs!

- Let xi is the embedding of the word at index i. The positioned embedding is:

$$\tilde{x}_i = x_i + p_i$$

Typically just do this at the first input layer

Absolute Positional Encoding

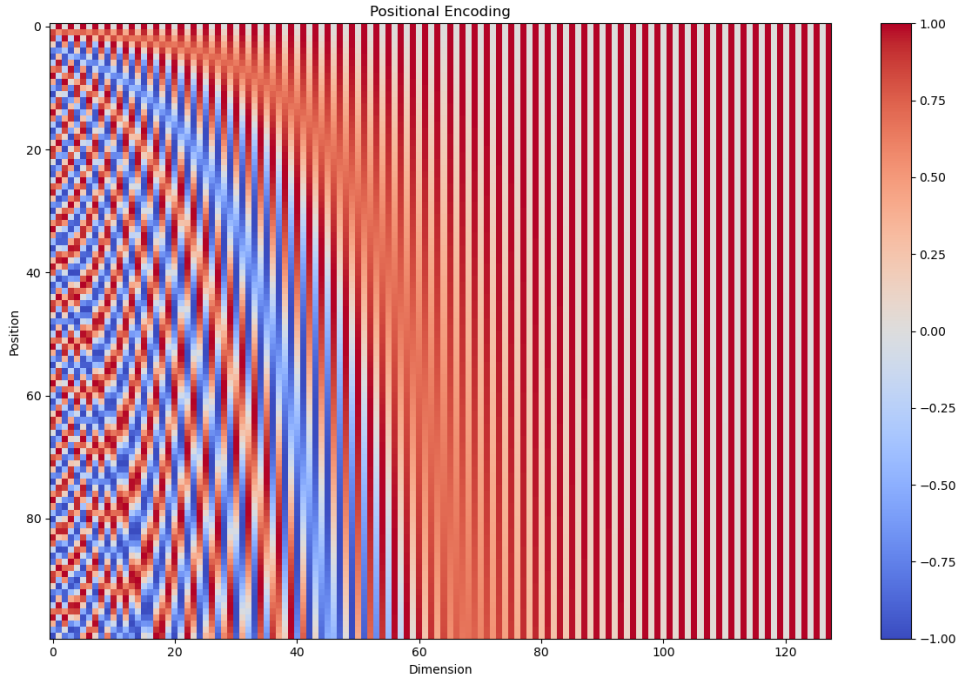
Pros:

- Periodicity indicates that maybe "absolute position" isn't as important
- Maybe can extrapolate to longer sequences as periods restart!

Cons:

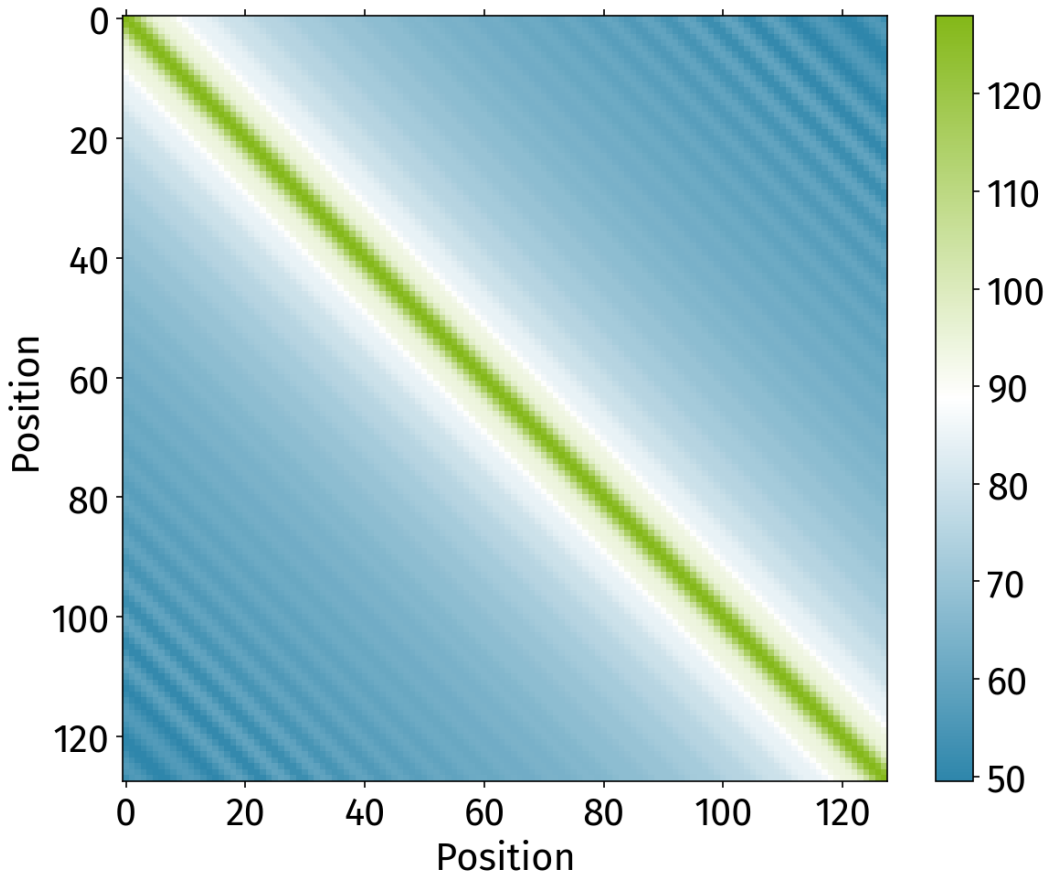
- Not learnable
- Extrapolation doesn't work well!

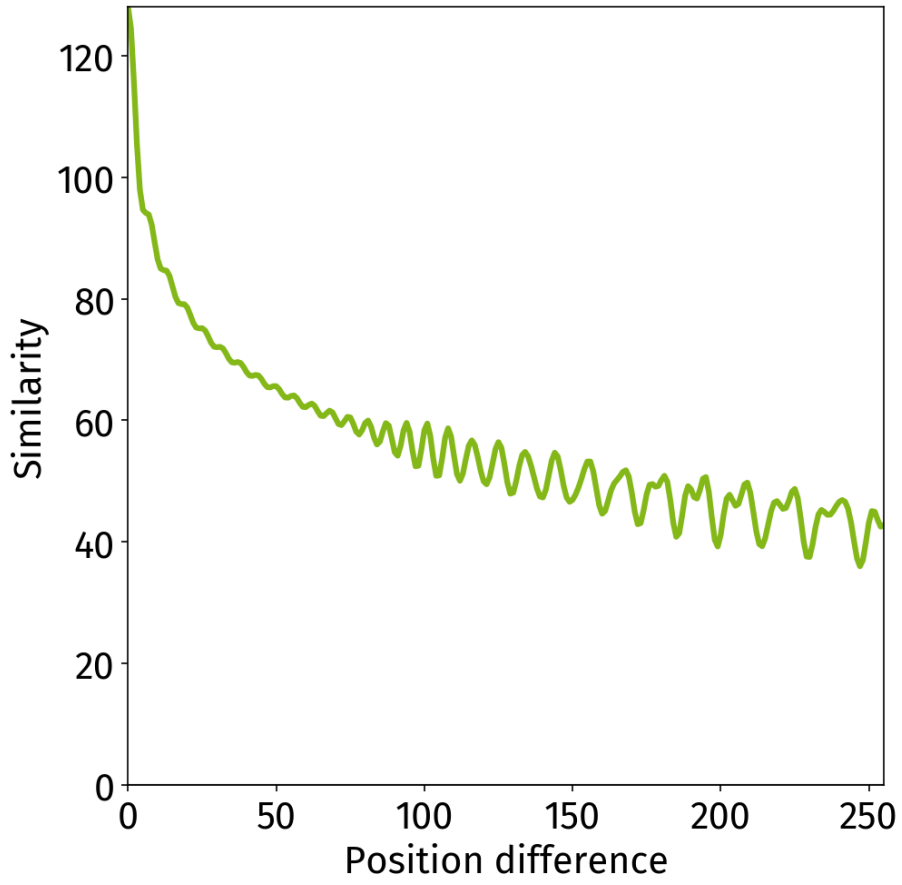
$$PE_{(pos,i)} = \begin{cases} \sin\left(\frac{pos}{10000^{i/d_{model}}}\right) & \text{if } i \text{ is even} \\ \cos\left(\frac{pos}{10000^{(i-1)/d_{model}}}\right) & \text{if } i \text{ is odd} \end{cases}$$



Absolute Positional Encoding

Dot product between positional encoding vectors $PE_a \cdot PE_b$





Learnable Positional Encoding

Learned absolute position representations:

Let all p_i be learnable parameters!

Learn a matrix $P \in \mathbb{R}^{d \times n}$ and let each p_i be a column of that matrix!

Pros:

- Flexibility: each position gets to be learned to fit the data

Cons:

- Definitely can't extrapolate to indices outside $1, \ldots, n$.

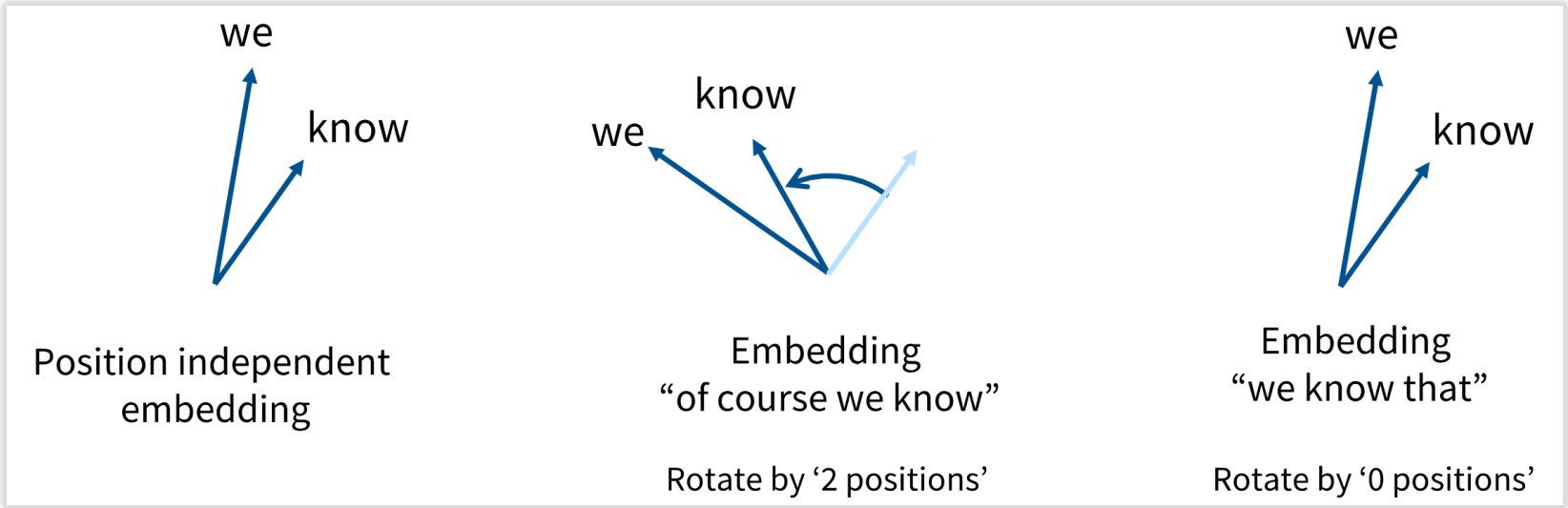
Many systems use this!

Relative Positional Encoding

Want attention to only depend on relative position $(i - j)$

Relative positional embedding should be function $f(x, i)$ such that

$$f(x, i) \cdot f(y, j) = g(x, y, i - j)$$



2104.09864

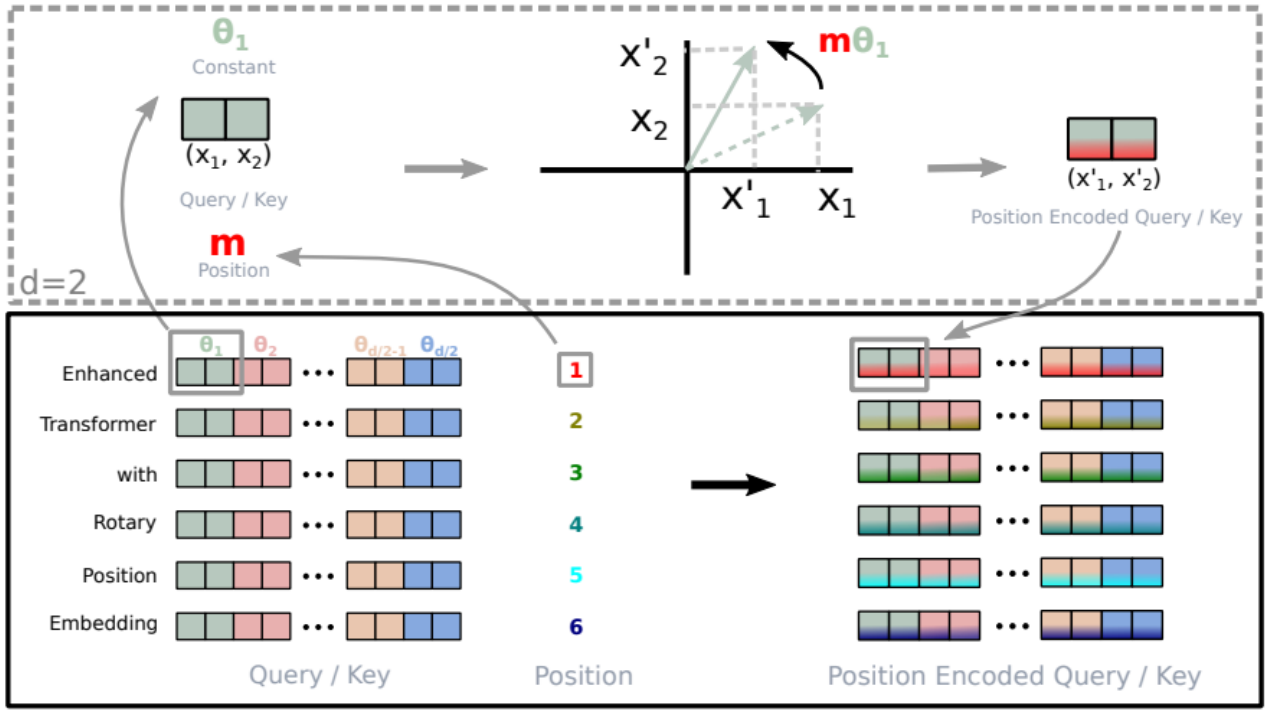
RoPE: Rotary Position Embedding

Instead of adding absolute position, multiple by “rotation” matrix

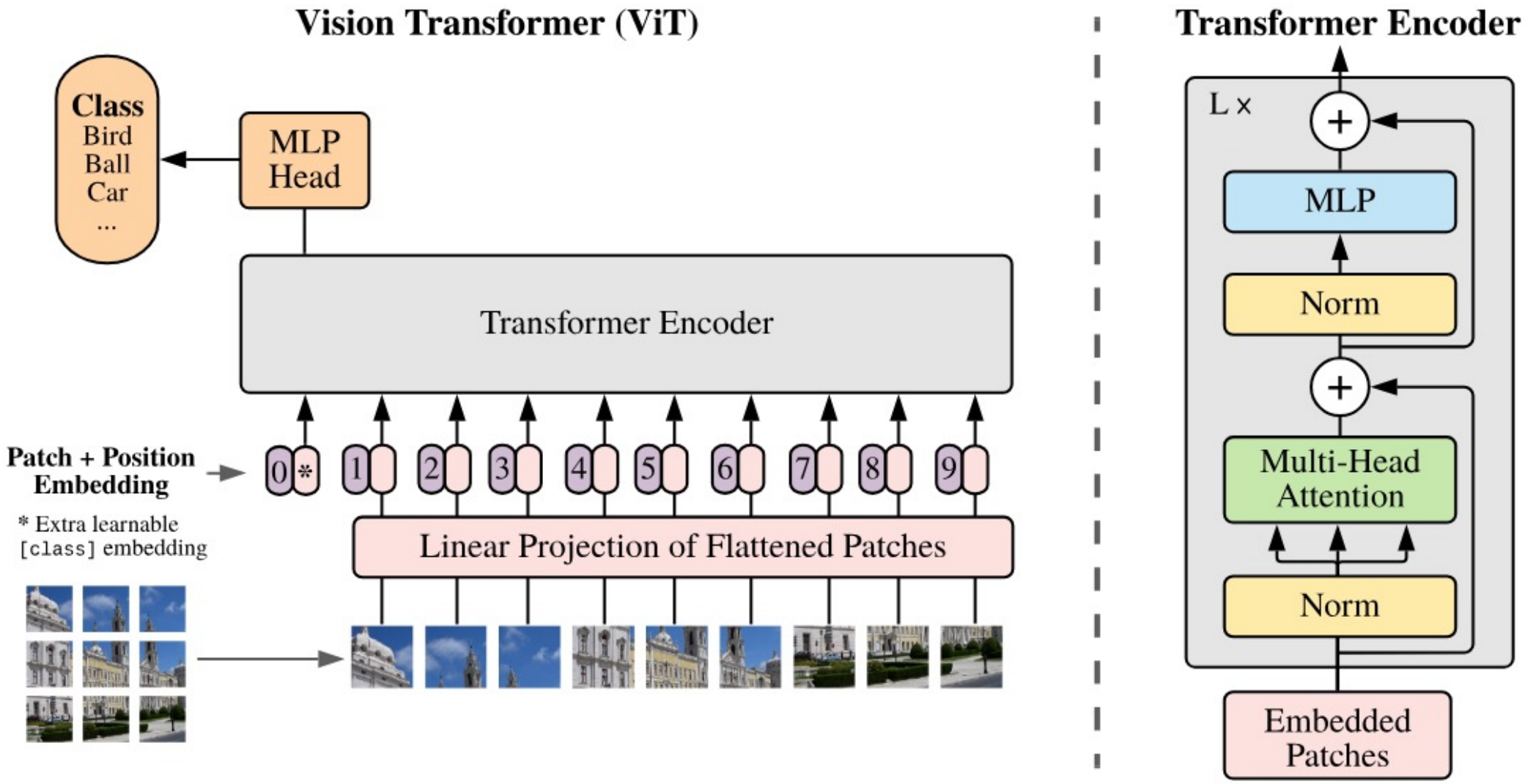
$$f_{\{q,k\}}(\boldsymbol{x}_m, m) = \boldsymbol{R}^d_{\Theta,m}\boldsymbol{W}_{\{q,k\}}\boldsymbol{x}_m$$

$$\boldsymbol{R}^d_{\Theta,m} = \begin{pmatrix} \cos m\theta_1 & -\sin m\theta_1 & 0 & 0 & \cdots & 0 & 0 \\ \sin m\theta_1 & \cos m\theta_1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cos m\theta_2 & -\sin m\theta_2 & \cdots & 0 & 0 \\ 0 & 0 & \sin m\theta_2 & \cos m\theta_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \cos m\theta_{d/2} & -\sin m\theta_{d/2} \\ 0 & 0 & 0 & 0 & \cdots & \sin m\theta_{d/2} & \cos m\theta_{d/2} \end{pmatrix}$$

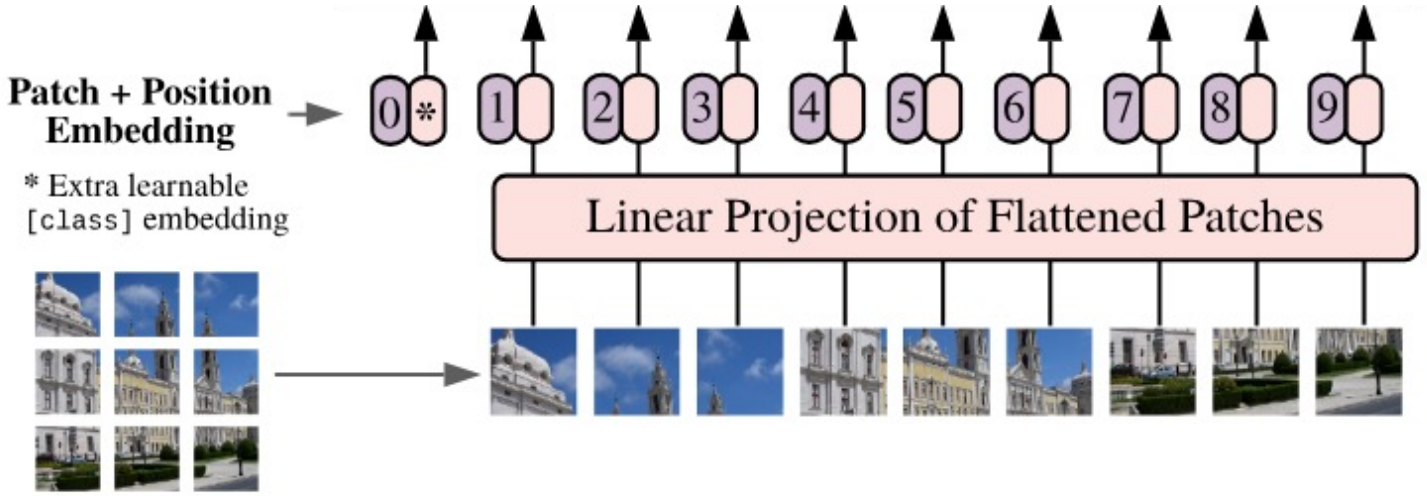
$$\boldsymbol{q}_m^\intercal \boldsymbol{k}_n = (\boldsymbol{R}^d_{\Theta,m}\boldsymbol{W}_q\boldsymbol{x}_m)^\intercal(\boldsymbol{R}^d_{\Theta,n}\boldsymbol{W}_k\boldsymbol{x}_n) = \boldsymbol{x}^\intercal\boldsymbol{W}_q R^d_{\Theta,n-m}\boldsymbol{W}_k\boldsymbol{x}_n$$



What about other Modalities? Vision Transformer



Vision Transformer – Patches as Tokens



Vision transformer processes an image like a sequence

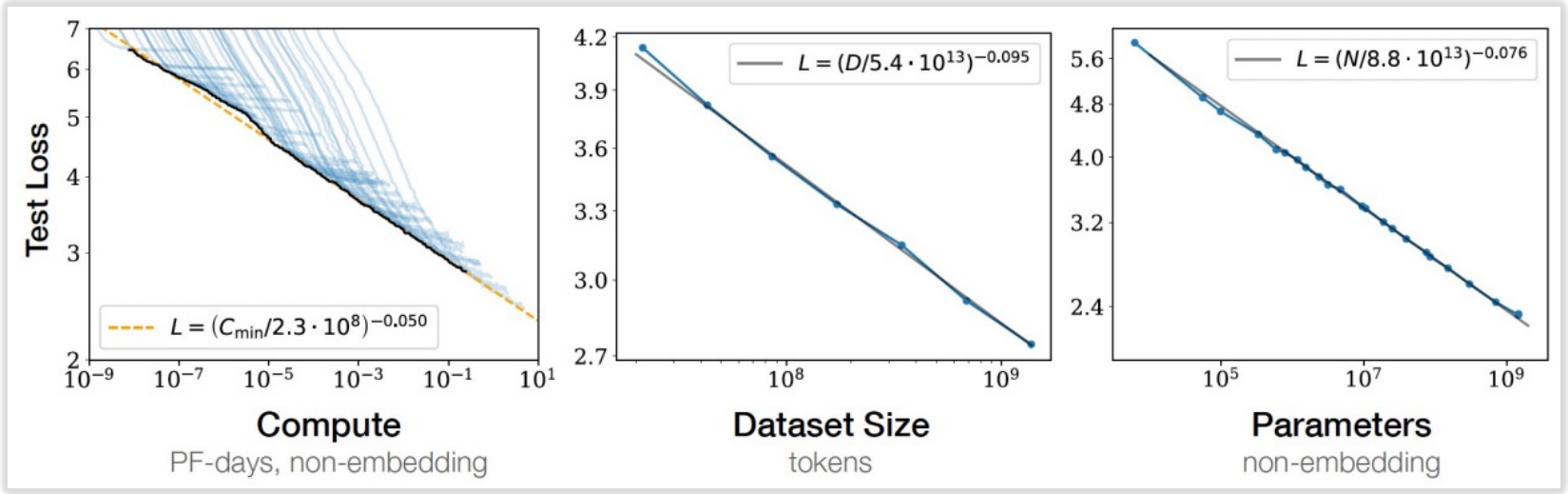
Tokens formed from taking patches of the image

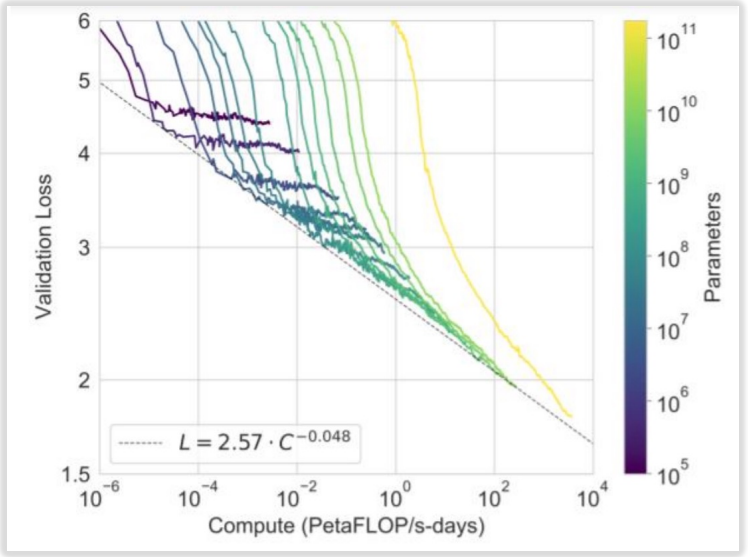
patch → flatten → linear transformation

These are not “discretized” tokens, linear projections can be anything

Scaling Laws

2001.08361
2102.06701
+ many more

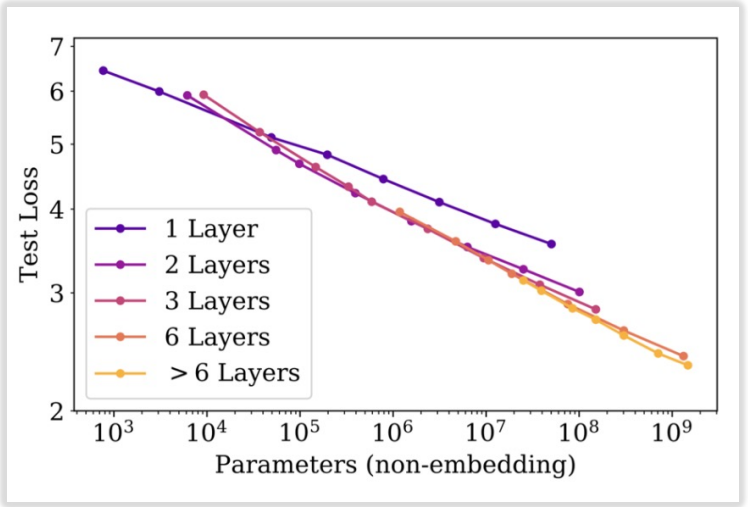




Scaling up models leads to reliable gains in loss reduction

Scaling laws can help identify model size – data tradeoffs

Predictable scaling helps us make intelligent decisions about architectures etc.



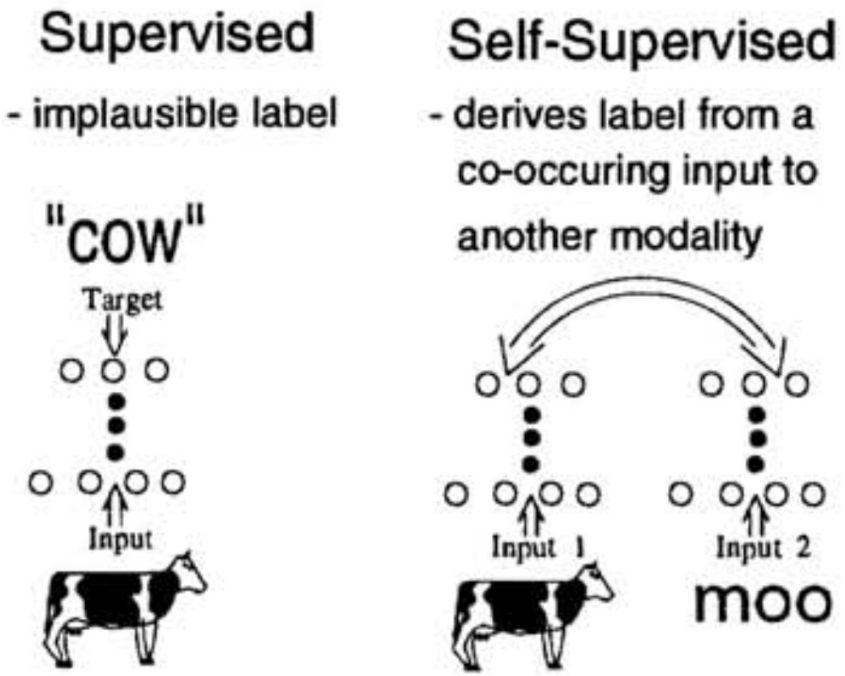
Self-Supervised Learning

Self-Supervised Learning

Making the World Differentiable: On Using Self-Supervised Fully Recurrent Neural Networks for Dynamic Reinforcement Learning and Planning in Non-Stationary Environments

Jürgen Schmidhuber*
Institut für Informatik
Technische Universität München
Arcisstr. 21, 8000 München 2, Germany
schmidhu@tumult.informatik.tu-muenchen.de

[Schmidhuber, 1990]



[Virginia Da Sa, *Learning Classificaion with Unlabeled Data*, NeurIPS 1993]

Self-Supervision

Self-supervision:
Train a model to answer procedurally generated questions about the data.

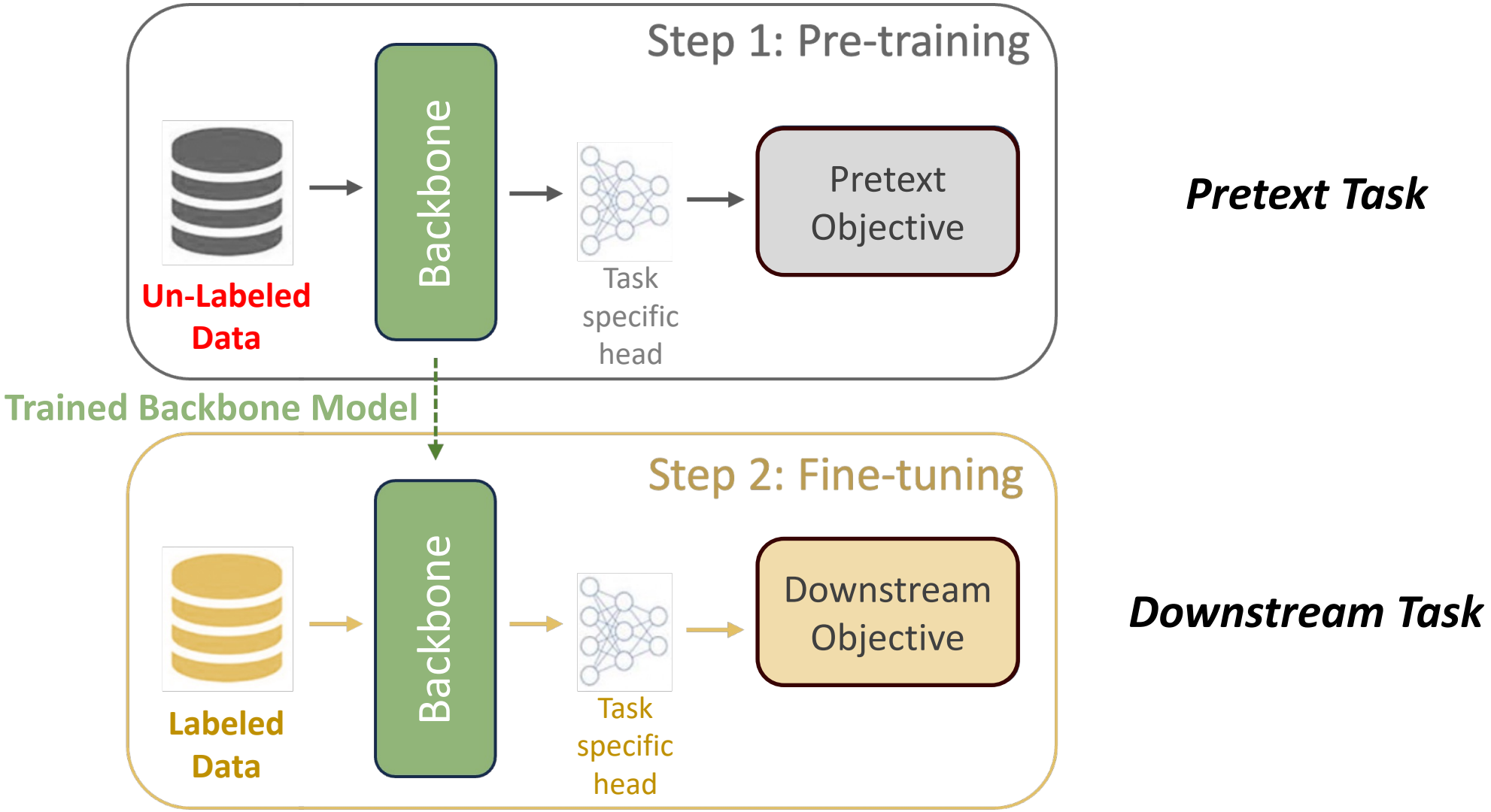
Good:

- Can procedurally generate potentially infinite amounts of annotation.
- We can borrow tricks from supervised learning without labels.
- Focus on only the information that you need (e.g., not pixels).
- Answering these questions requires more fundamental understanding of data.

Not so good:

- designing good questions also requires some fundamental understanding of the data (e.g., structure).

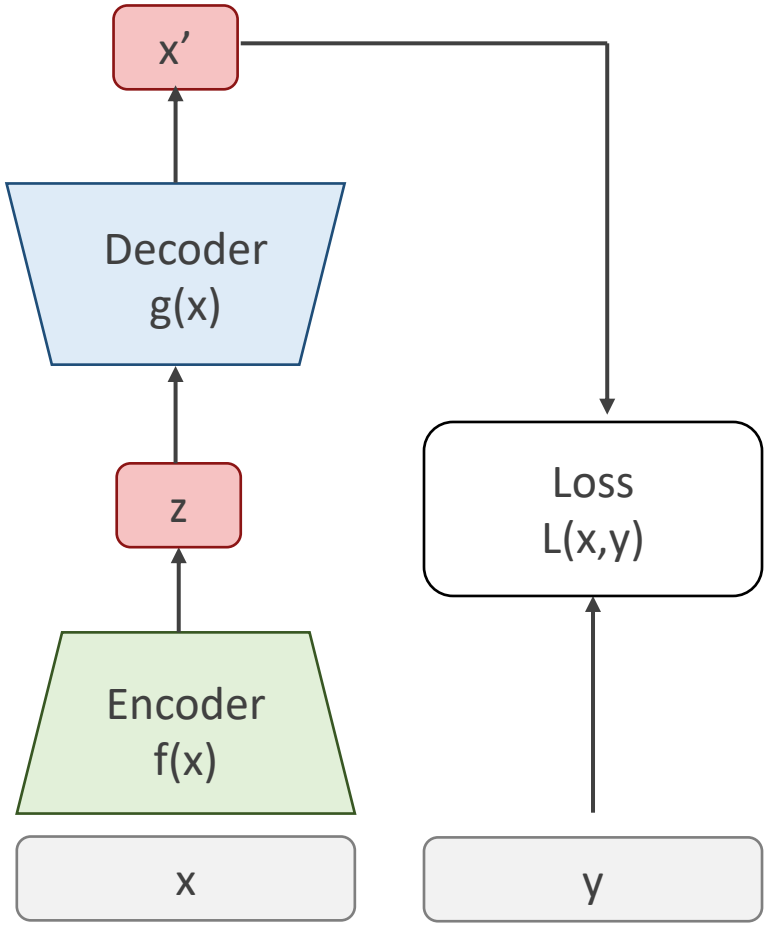
Self-Supervised Learning and Fine-Tuning

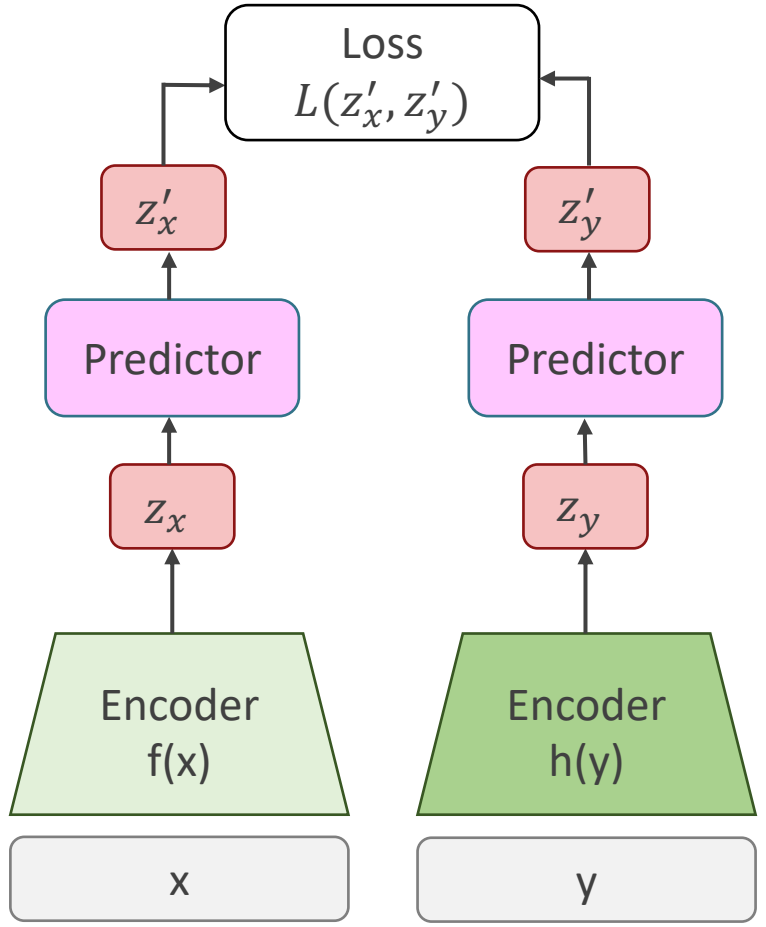


Pretext Task

Downstream Task

Classes of Self-Supervision





"Generative" like approaches

Compare at "pixel" / "token" level

"Predictive" like approaches

Compare at representation level

Generative vs Predictive (& Contrastive) vs Supervised Approaches

Generative Approaches

- Aim to generate missing or corrupted data in ***original input / token space***
- *Criticism*: may require learning very fine and often irrelevant details or noise

Predictive Approaches

- Make and compare predictions ***in (learned) representation space***
- *Criticism*: Only can learn encoder, can't generate data
- *Criticism*: Often requires choosing good augmentations and/or lots of negative examples

Supervised Approaches

- Learning representation required to solve a labelled supervised learning task
- *Criticism*: Requires lots of labelled data, hard to scale
- *Criticism*: If supervised objective not general enough, learn representations that are not generally useful and are hard to adapt / fine-tung to new tasks

Towards SSL & Foundation Models in HEP – Sep. 2025

Contrastive

- JetCLR - symmetry augmentations
- R3SL - re-simulation
- RINO – clustering augmentations with DINO distillation

Generative

- Mask particle type prediction
- Mask Particle Modeling [1, 2]
- Next Particle Token Prediction

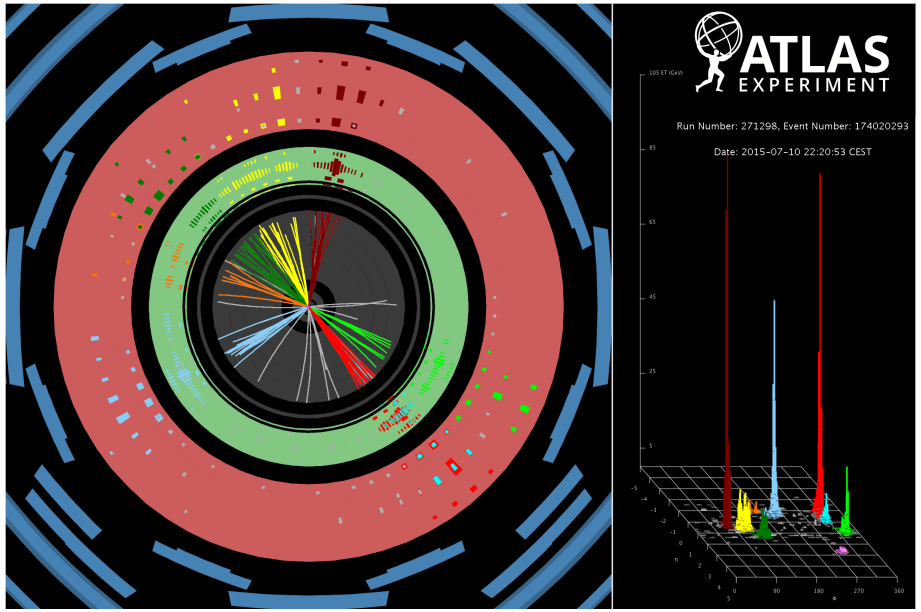
Joint Embedding Predictive Architectures (JEPA)

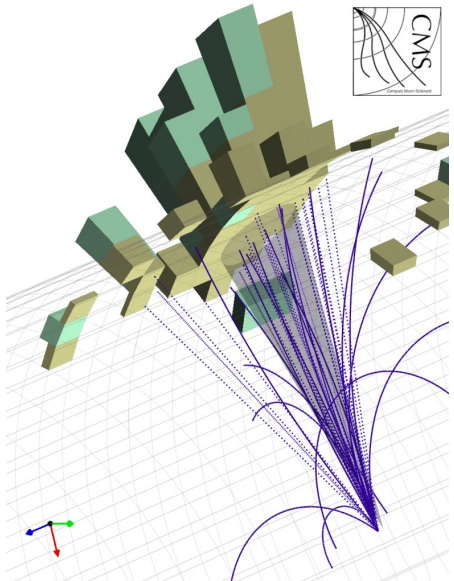
- J-JEPA – cluster particles into subjets
- P-JEPA – use random masks
- HEP-JEPA – blocks of particles

Supervised

- Supervised classification and generation
- Large-scale fine-grained classification

A note about data: Jets





Jet = Unordered set of particles

Each particles has a list of features:

Particle = {momentum, direction, position, … }

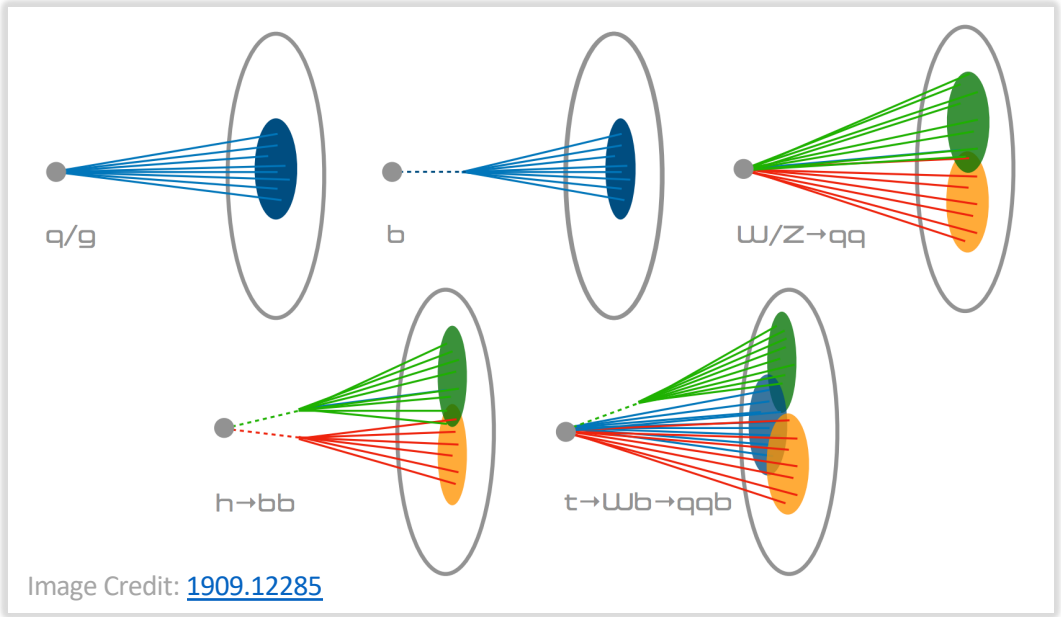
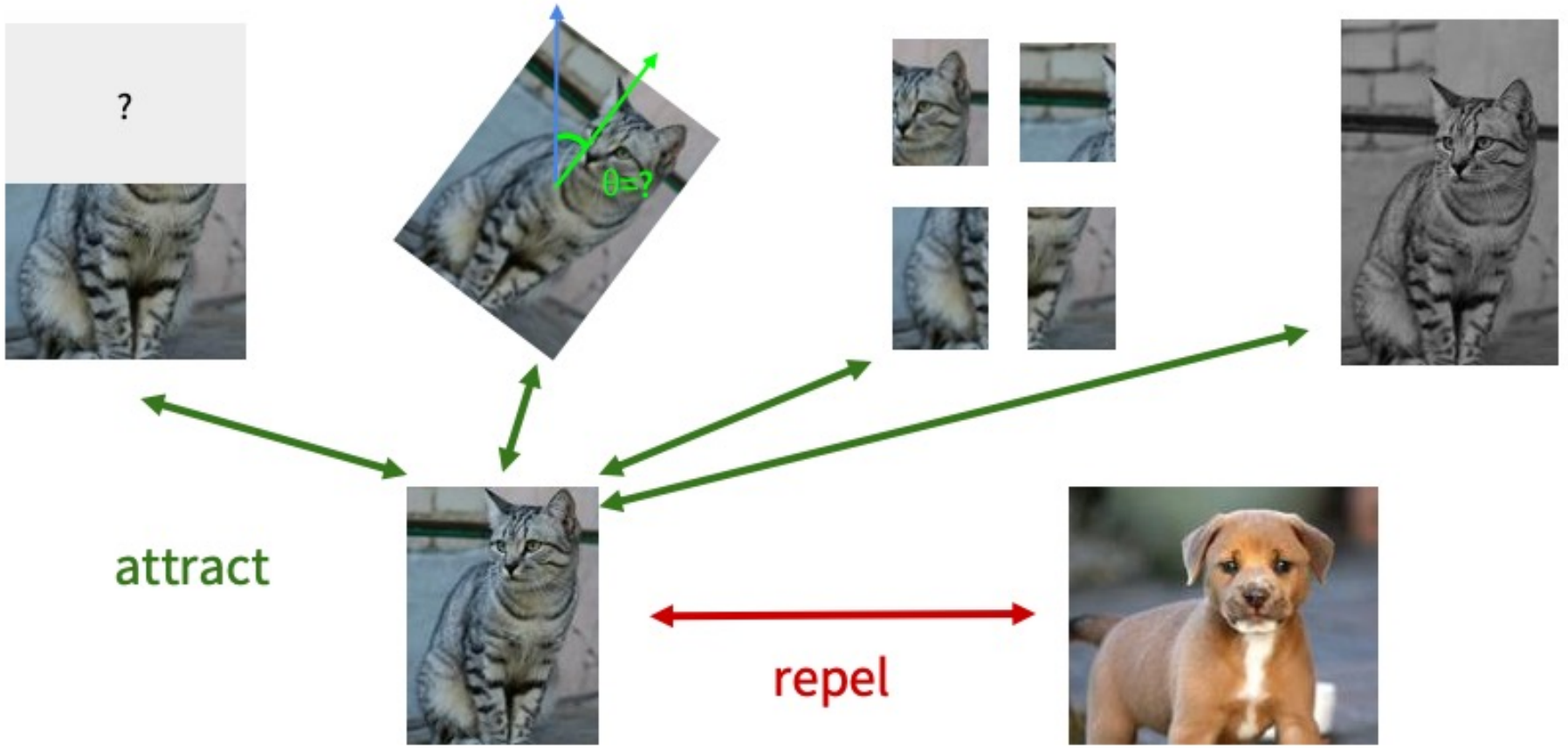


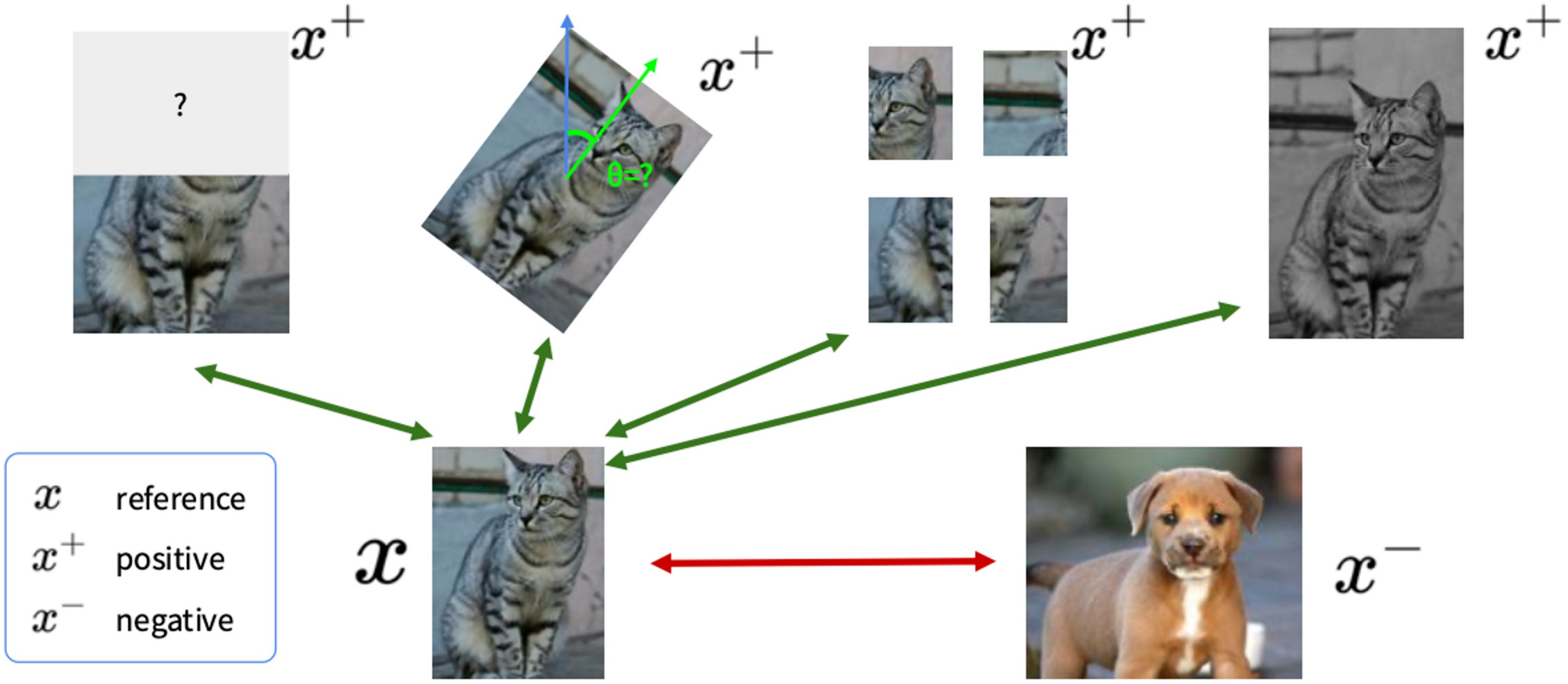
Image Credit: 1909.12285

Contrastive Learning

Contrastive Representation Learning



Contrastive Representation Learning



Strategy: Given input & set of examples,
determine *which examples are different* and *which are similar* (***an augmentation***)

A formulation of contrastive learning

Encoder model $f(\cdot)$ embeds and give representation of data, we want:

$$Score\big(f(x), f(x^+)\big) \gg Score\big(f(x), f(x^-)\big)$$

x =reference sample, x^+ = positive sample, x^- = negative sample

Want to learn an encoder function $f(\cdot)$ that gives

High score for positive pairs (x, x^+)

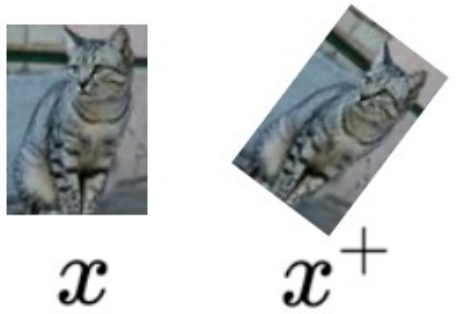
Low score for negative pairs (x, x^-)

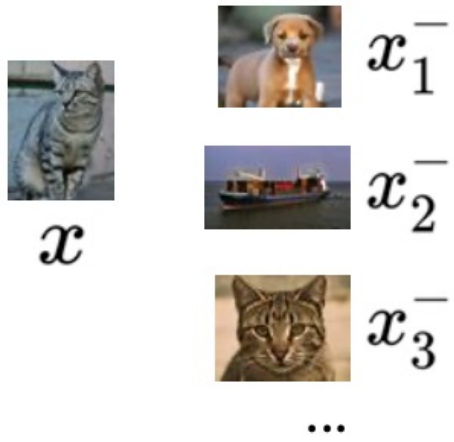
A formulation of contrastive learning

A loss function with **1 positive sample** and **N-1 negative samples**

$$L = -\mathbb{E}_X \left[\log \frac{\exp\left(s\left(f(x), f(x^+)\right)\right)}{\exp\left(s\left(f(x), f(x^+)\right)\right) + \sum_{j=1}^{N-1} \exp\left(s\left(f(x), f\left(x_j^-\right)\right)\right)} \right]$$

Score for positive pair **Score for N-1 negative pairs**





A formulation of contrastive learning

A loss function with **1 positive sample** and **N-1 negative samples**

$$L = -\mathbb{E}_X \left[\log \frac{\exp\left(s\left(f(x), f(x^+)\right)\right)}{\exp\left(s\left(f(x), f(x^+)\right)\right) + \sum_{j=1}^{N-1} \exp\left(s\left(f(x), f\left(x_j^-\right)\right)\right)} \right]$$

Cross-entropy loss for N-way softmax classifier:

Learn to find positive sample from N samples

Also known as InfoNCE loss [1807.03748]

lower bound on mutual information: $\mathrm{I}\left(f(x), f(x^+)\right) - LogN \geq -L$

[1905.06922]

What's going on here? Invariance perspective

Want embedding of two views (x, x^+) to be as similar are possible

So embeddings need to be insensitive to effect of the augmentations

- i.e. identify that it is the same underlying object

Pushes model to learn embedding that are invariant to augmentations

Choosing good augmentation becomes critical!

- Want to be invariant to irrelevant details
- Want to be sensitive to important *semantic* information

SimCLR: A simple framework for contrastive learning

Score function is cosine similarity

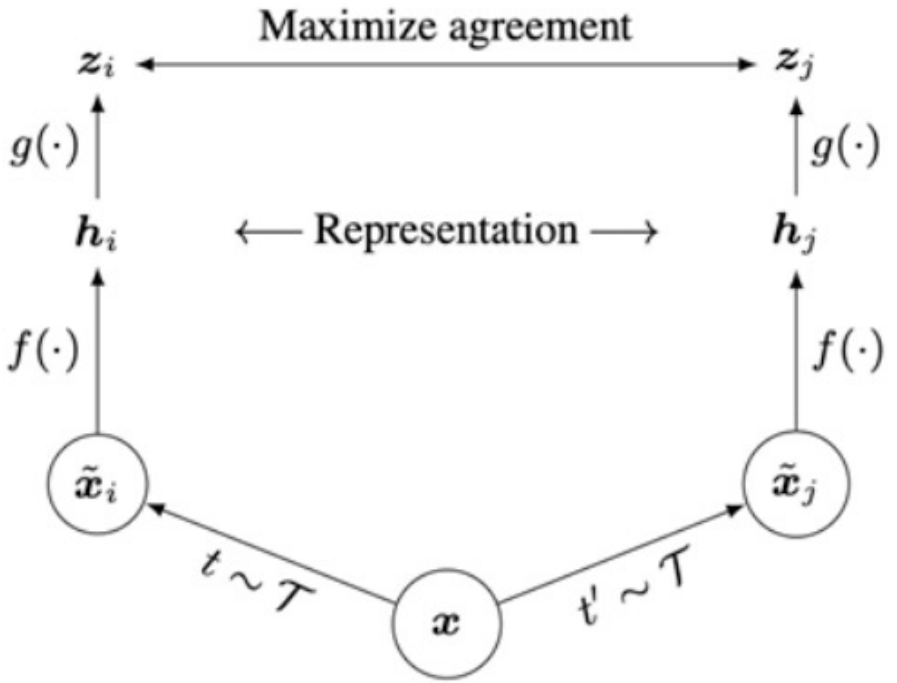
$$s(u, v) = \frac{u^T v}{\|u\|\|v\|}$$

Encoder $f(\cdot)$ computes representation h

Projection network $g(\cdot)$

- Project feature to space where contrastive learning is applied
- Separates making representation from loss

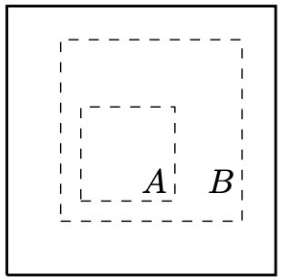
Generate positive samples with augmentation: crop, blur, color distort, …



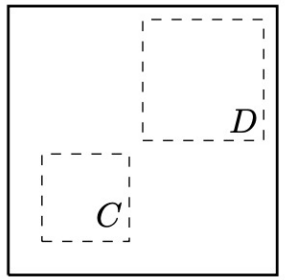
[2002.05709]

SimCLR Augmentations

Original



(a) Global and local views.



(b) Adjacent views.

Crops can be overlapping or not.

Crop and resize

Gaussian blur

SimCLR uses many different augmentations (more than methods prev. models)

Started trend of using augmentation to drive SSL

Color distort. (drop)

Color distort. (jitter)

SimCLR: A simple framework for contrastive learning

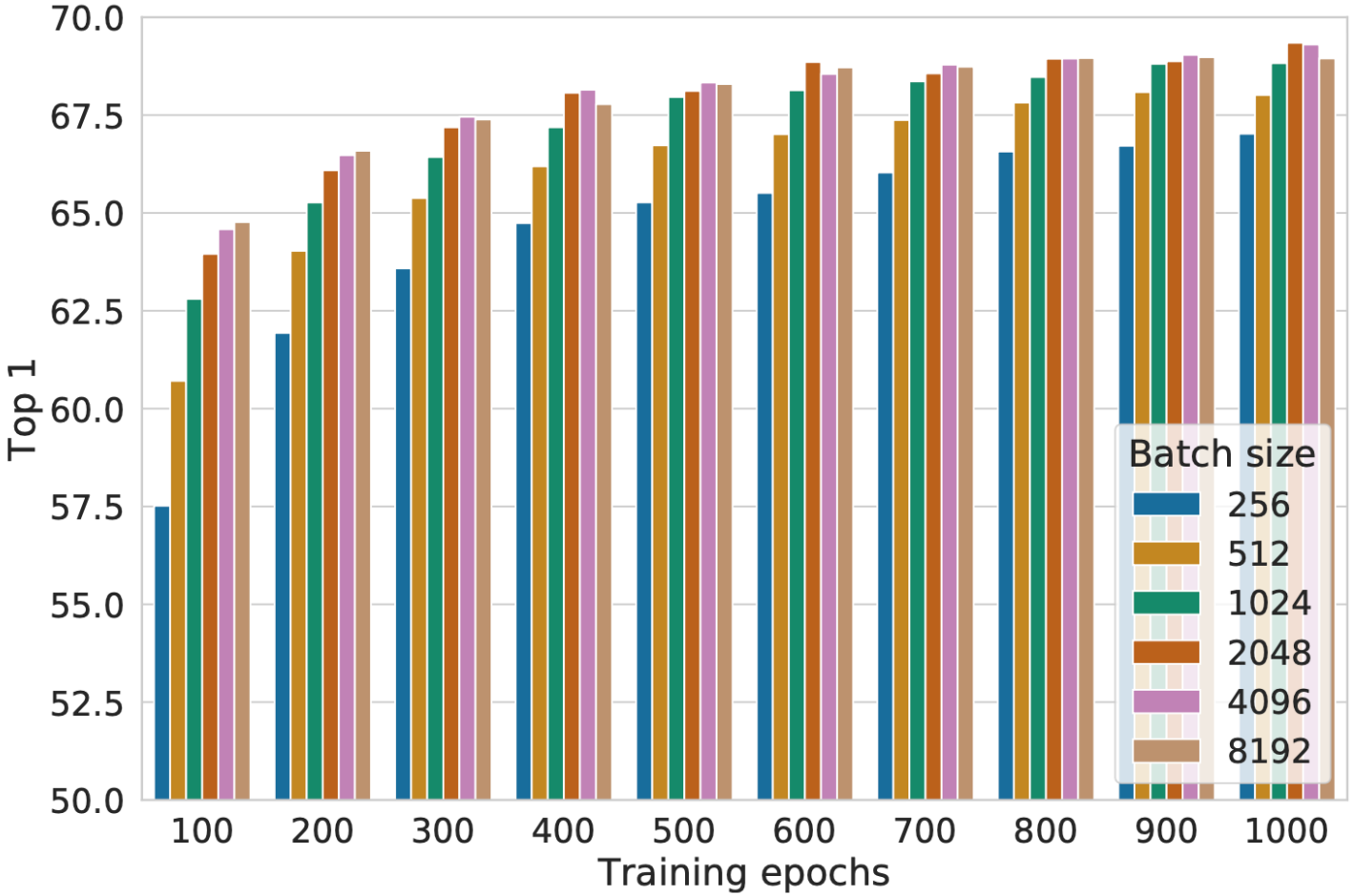


Figure 9. Linear evaluation models (ResNet-50) trained with different batch size and epochs. Each bar is a single run from scratch.[10]

Linear evaluation on image classification

SimCLR works best with large batch size and long training times

[2002.05709]

JetCLR

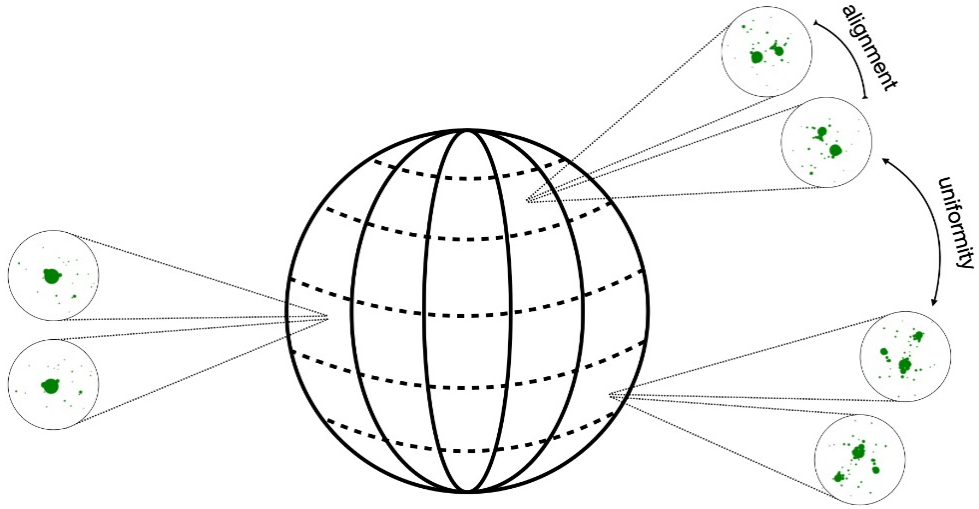
SimCLR was among the first SSL methods applied to HEP data, specifically for jets

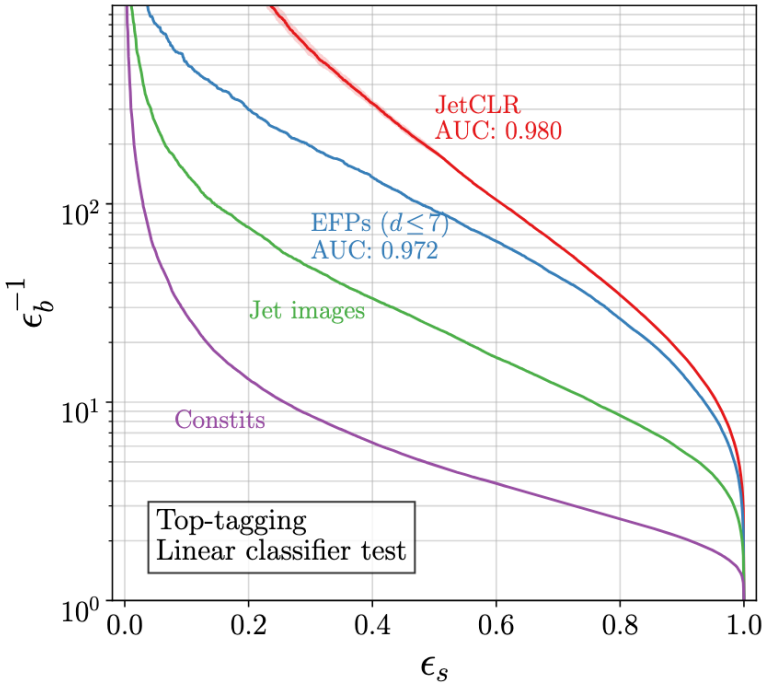
Physics inspired augmentations

- Translations
- Rotations
- Soft-splitting
- Collinear splitting

Tested with linear classifier head on top of frozen backbone

Is there a general way to generate physics-driven augmentations?



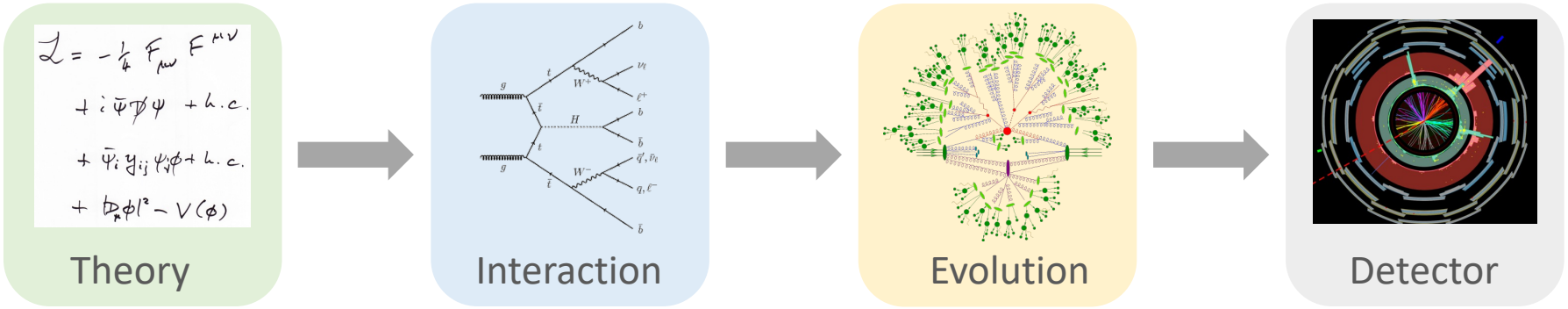


2108.04253

Re-Simulation Based Self-Supervised Learning

Simulators hold domain knowledge

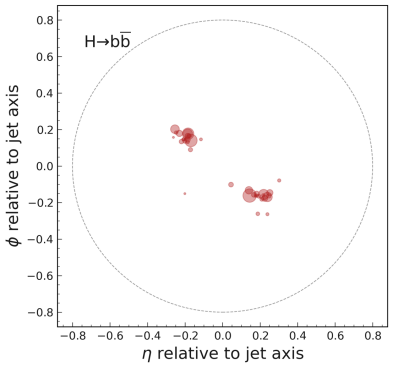
→ generate plausible outcomes of experiment

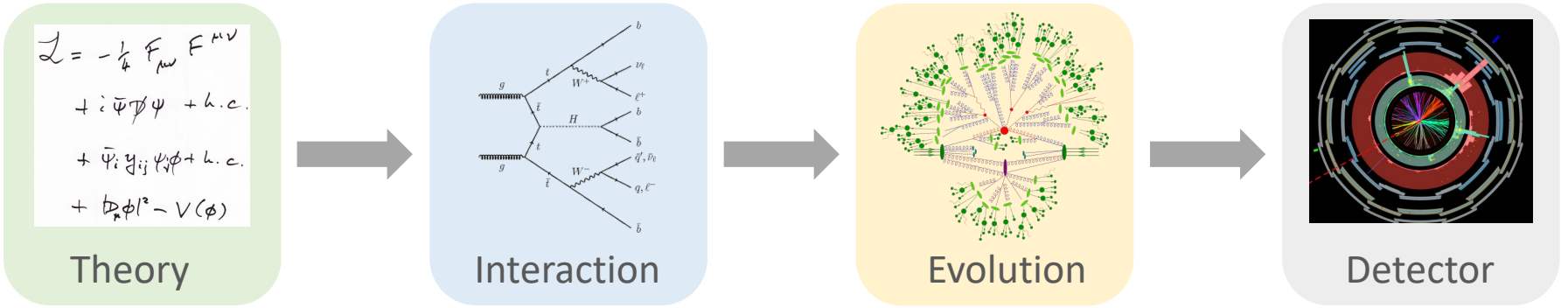


HEP high-fidelity multi-step stochastic simulator

2403.07066

Re-Simulation Based Self-Supervised Learning

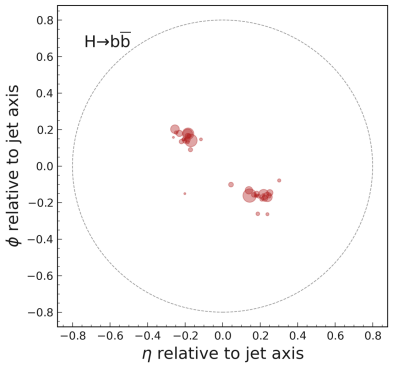




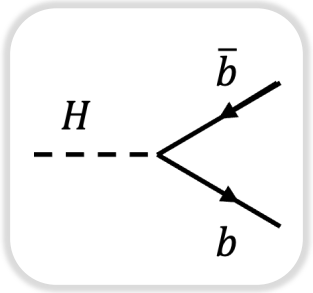
HEP high-fidelity multi-step stochastic simulator

2403.07066

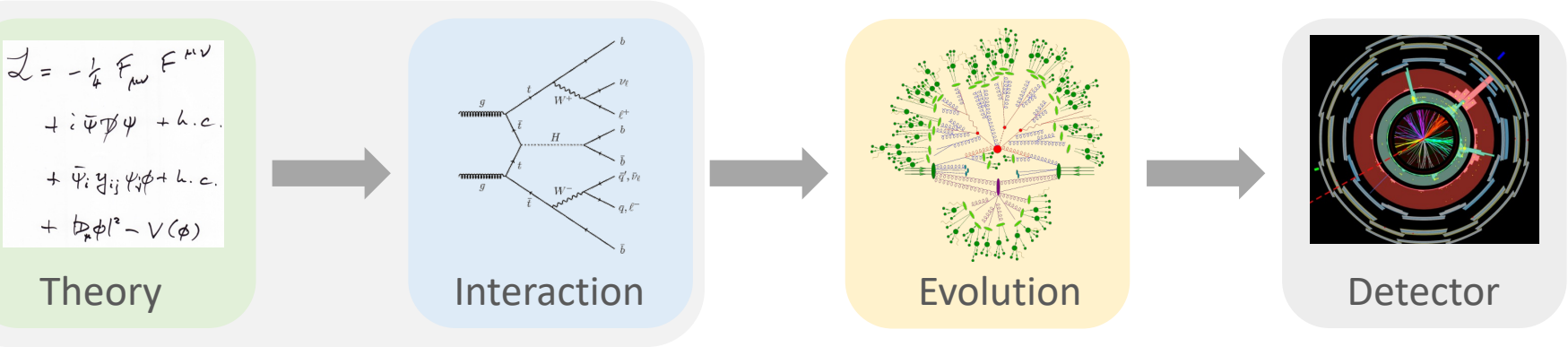
Re-Simulation Based Self-Supervised Learning



Fix



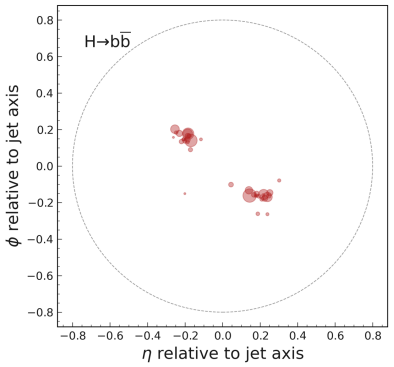
Fix



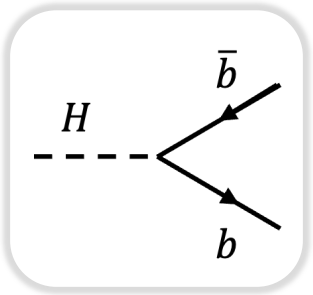
HEP high-fidelity multi-step stochastic simulator

2403.07066

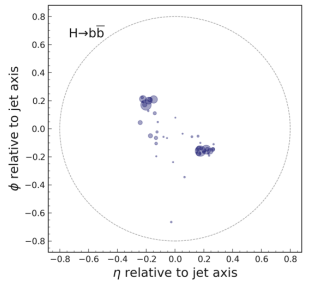
Re-Simulation Based Self-Supervised Learning



Fix

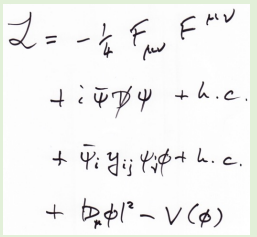


Same Sim.

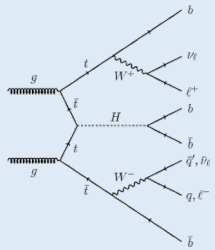


In-Domain

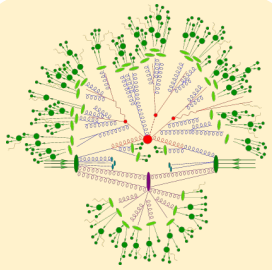
Fix



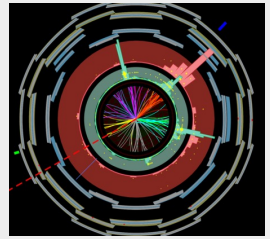
Theory



Interaction



Evolution



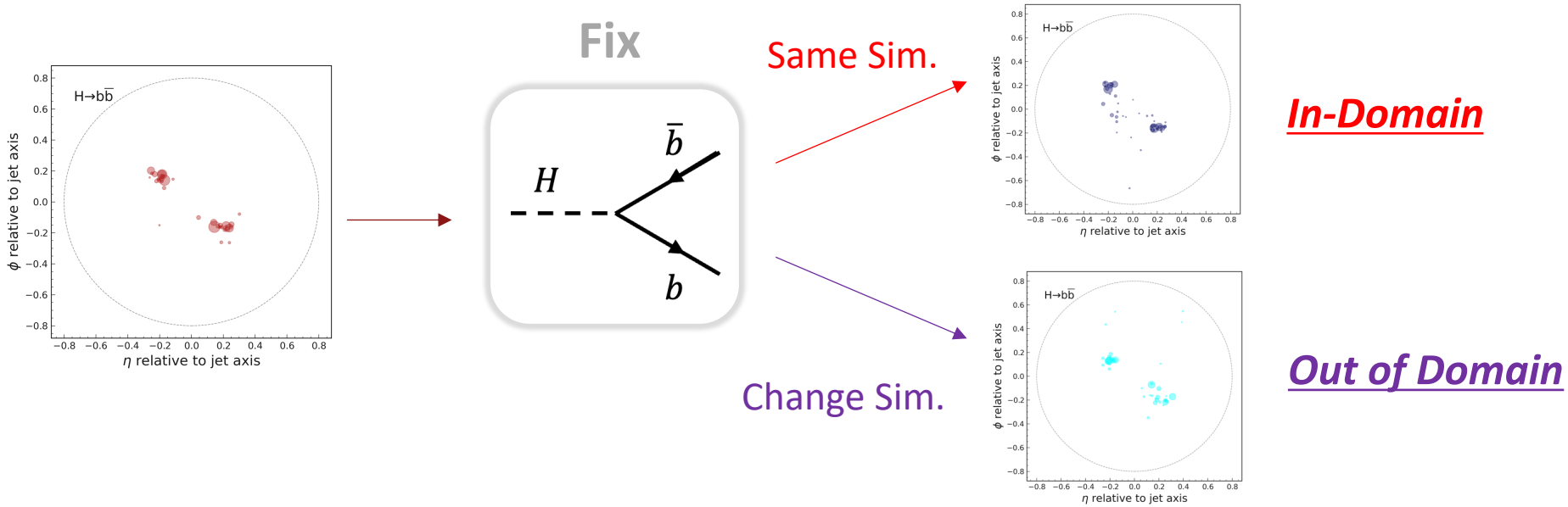
Detector

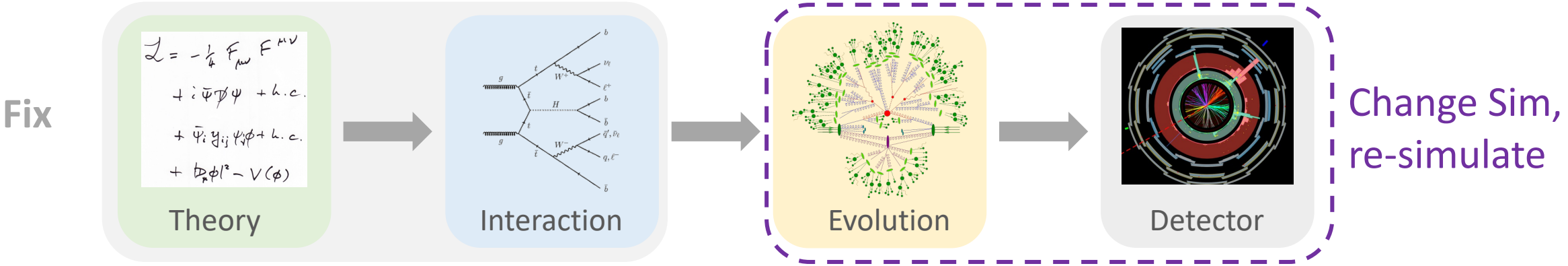
Re-simulate

HEP high-fidelity multi-step stochastic simulator

2403.07066

Re-Simulation Based Self-Supervised Learning

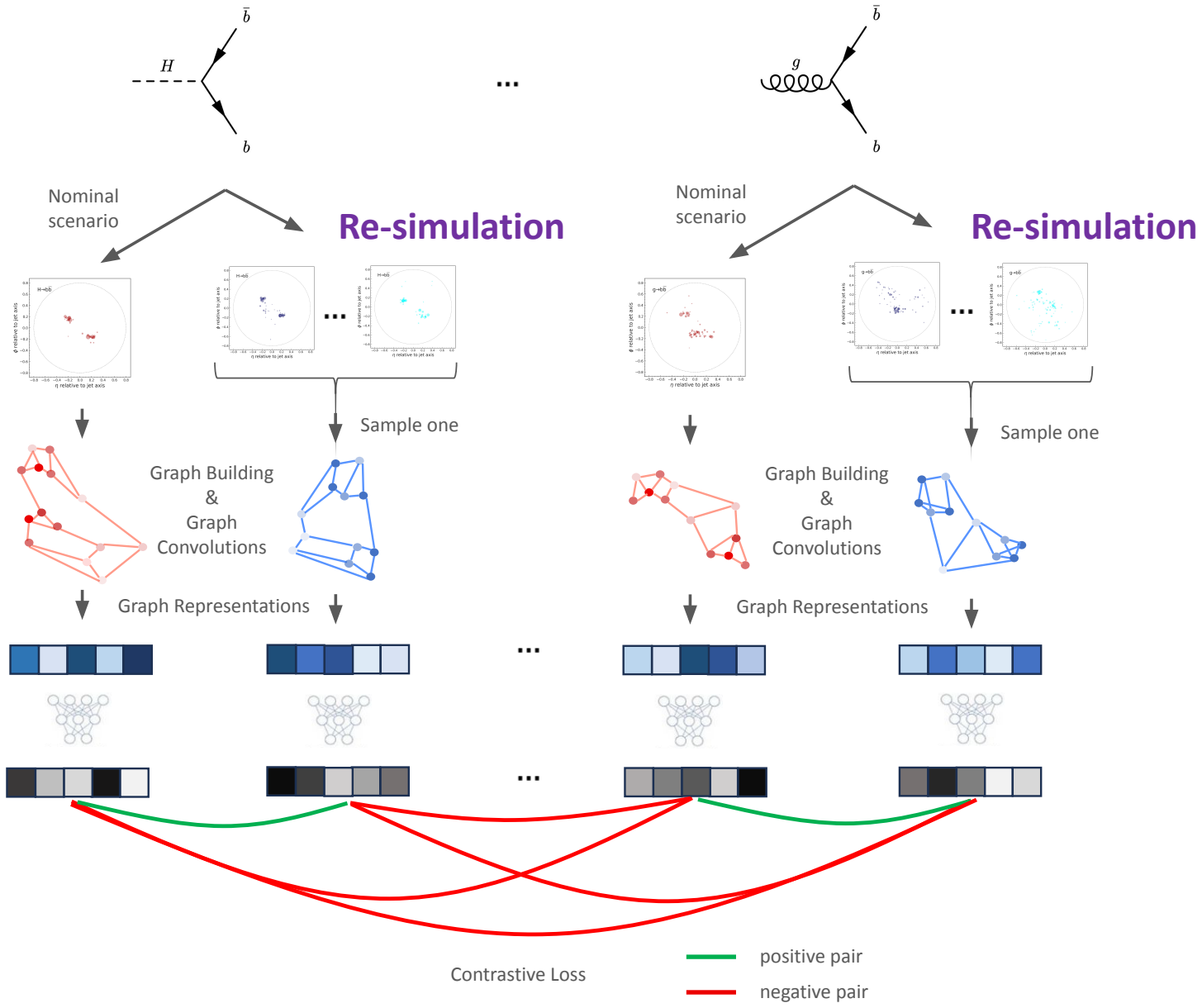




HEP high-fidelity multi-step stochastic simulator

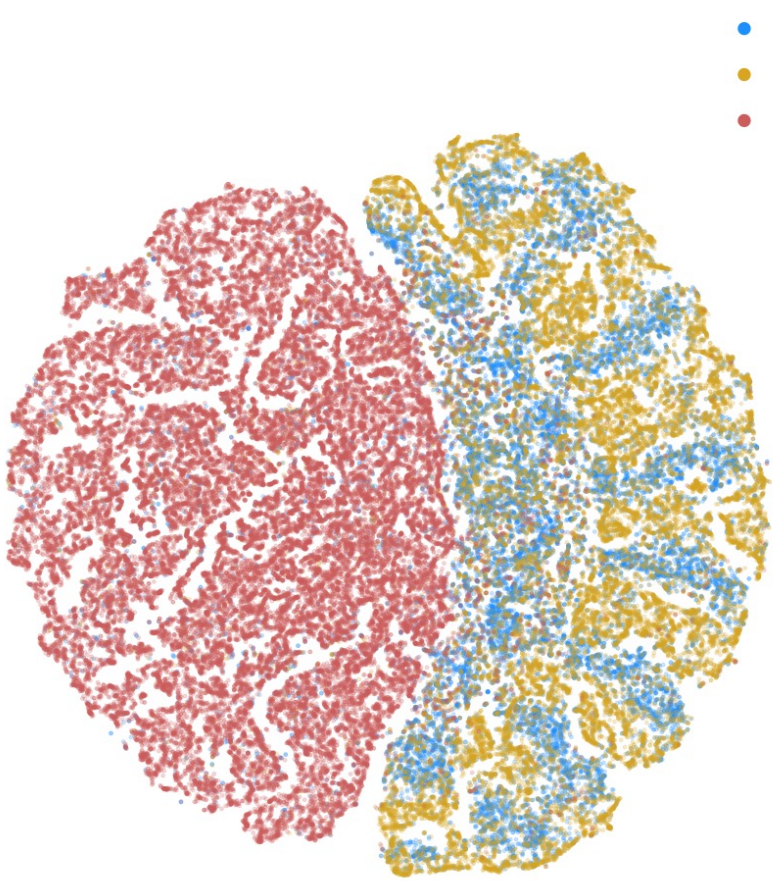
2403.07066

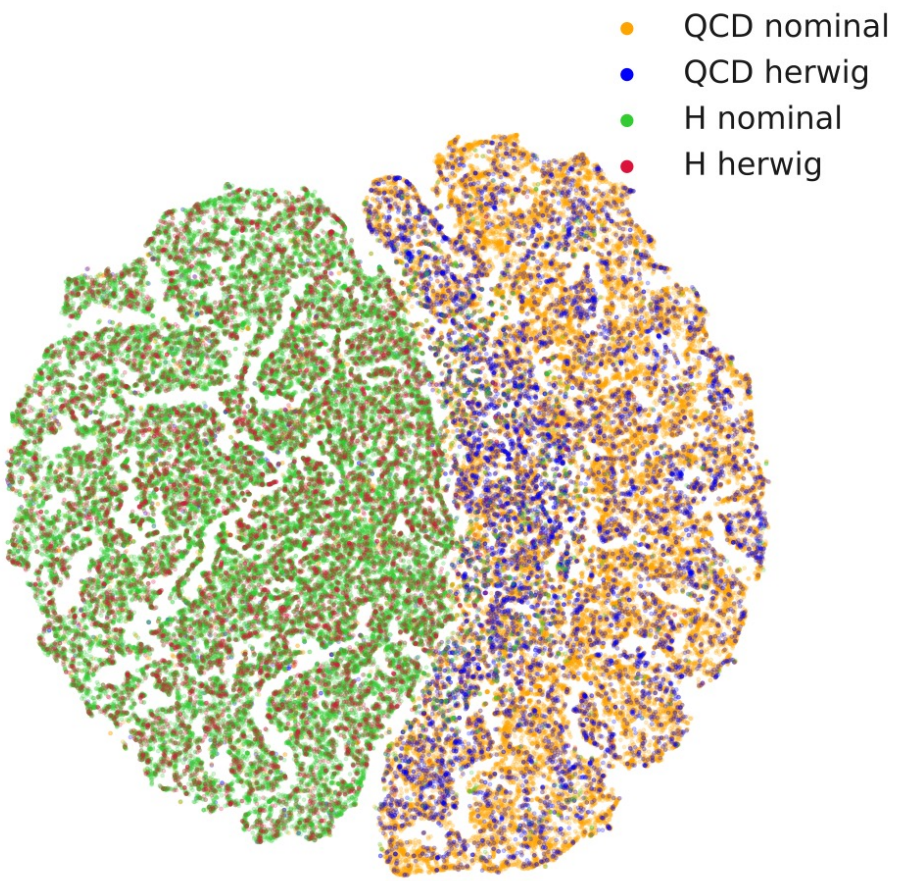
Re-Simulation Based Contrastive Learning



2403.07066

tSNE Visualization of Pre-Trained Representations

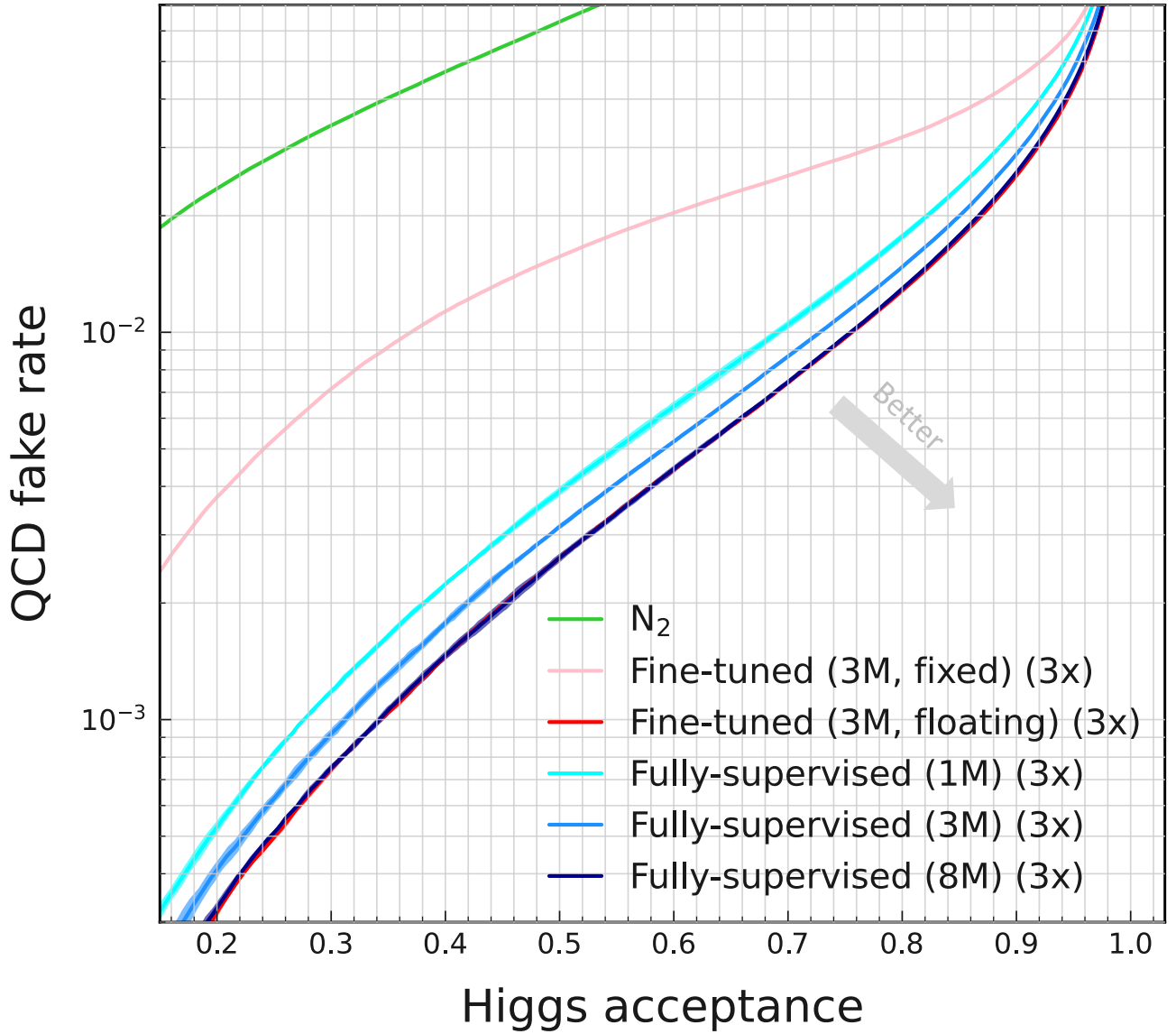




Class separation & alignment of representations across domains

2403.07066

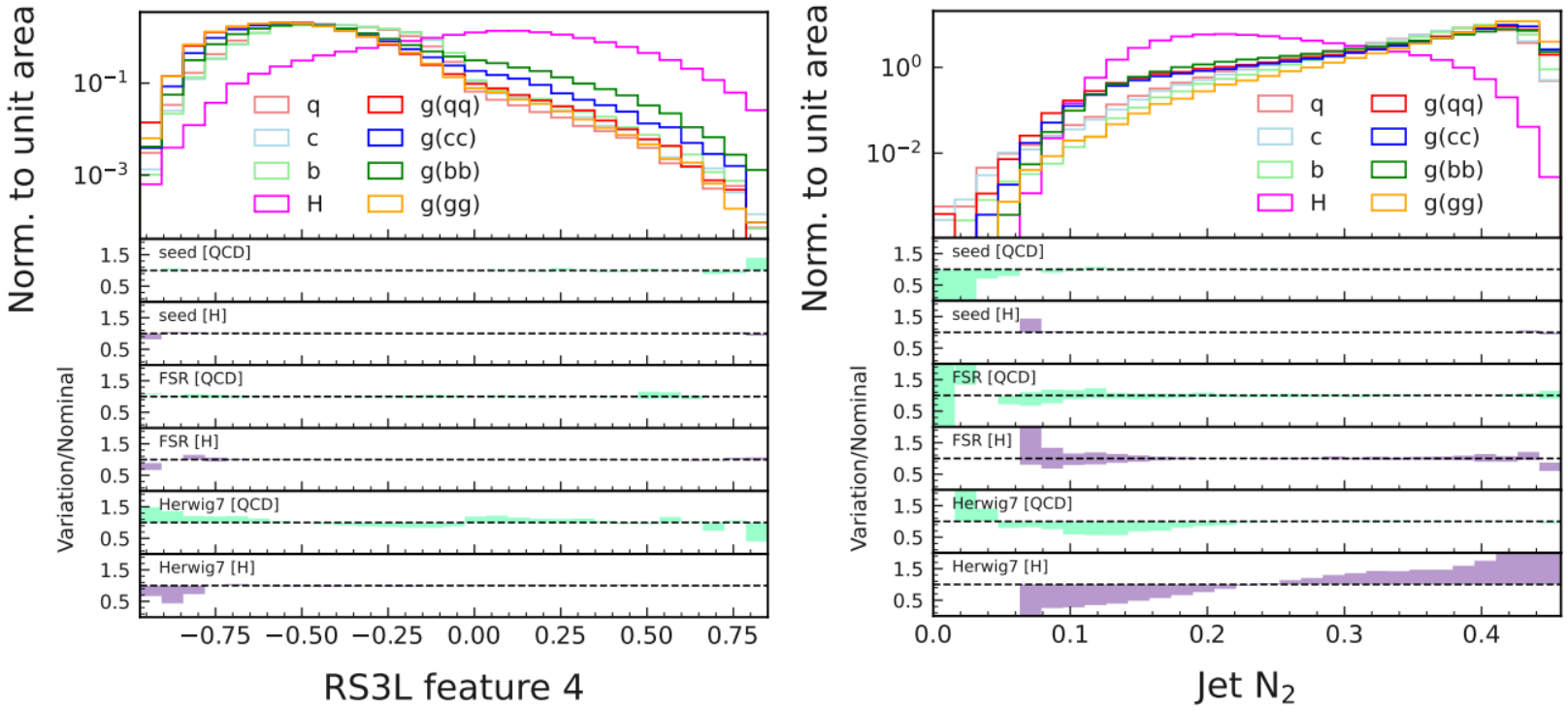
Fine-Tuned Performance



2403.07066

New work building on this idea in: 2503.11632

Sensitivity to Simulator Variations



Some reduction of sensitivity to simulator variations for some learned features

Far from robust… still a lot of work needed on systematic uncertainty robustness

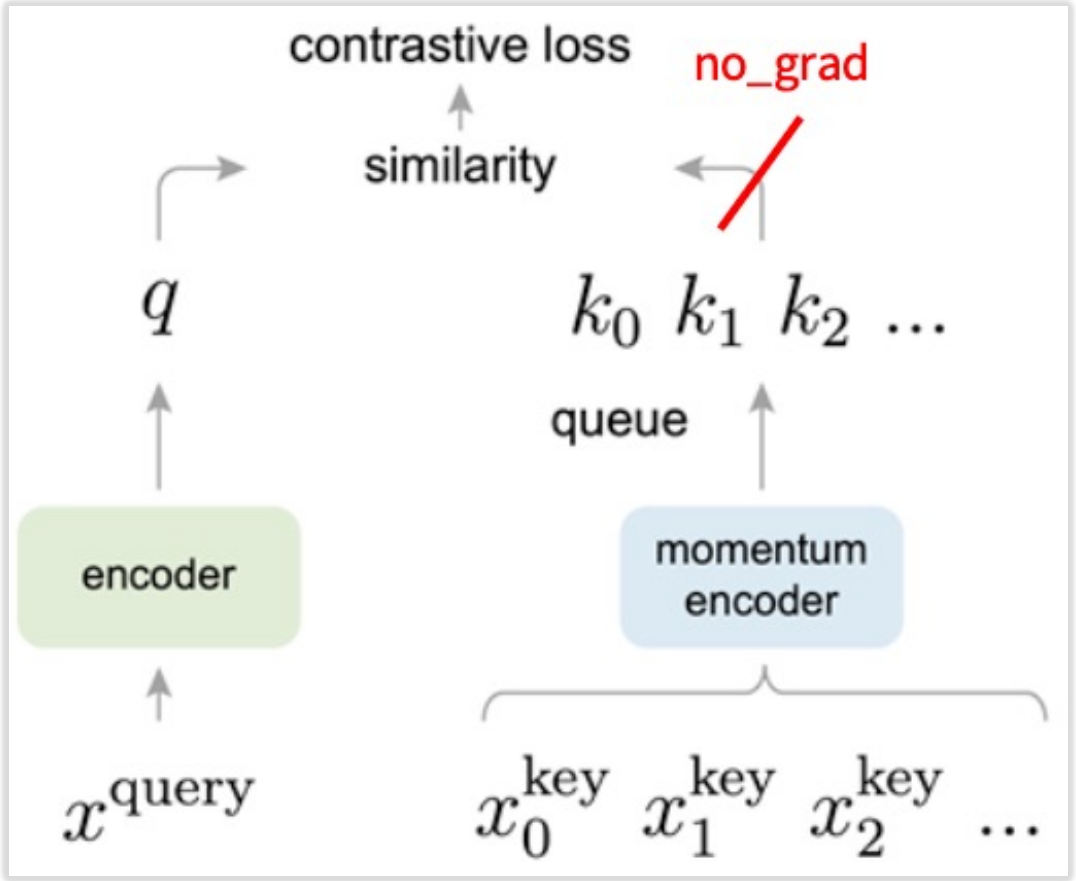
Momentum Contrastive Learning (MoCo)

Difference to SimCLR

- Running queue of keys (negative samples)
- Only update encoder through query
- Decouple mini-batch size with num. keys
  → **Can support large number of samples**
- Key encoder is slowly progressing:
  momentum update rule

$$\theta_k \leftarrow m\theta_k + (1-m)\theta_q$$

…enter EMA (exponential moving average)



1911.05722

What's the deal with the EMA?

Why would comparing a model with EMA of itself drive learning?

$$\mathcal{L}_q = -\log \frac{\exp(q \cdot k_+ / \tau)}{\sum_{i=0}^{K} \exp(q \cdot k_i / \tau)}$$

Contrastive loss like mapping encoded query q onto dictionary of encoded keys k_i, and want to learn to find positive key k_+

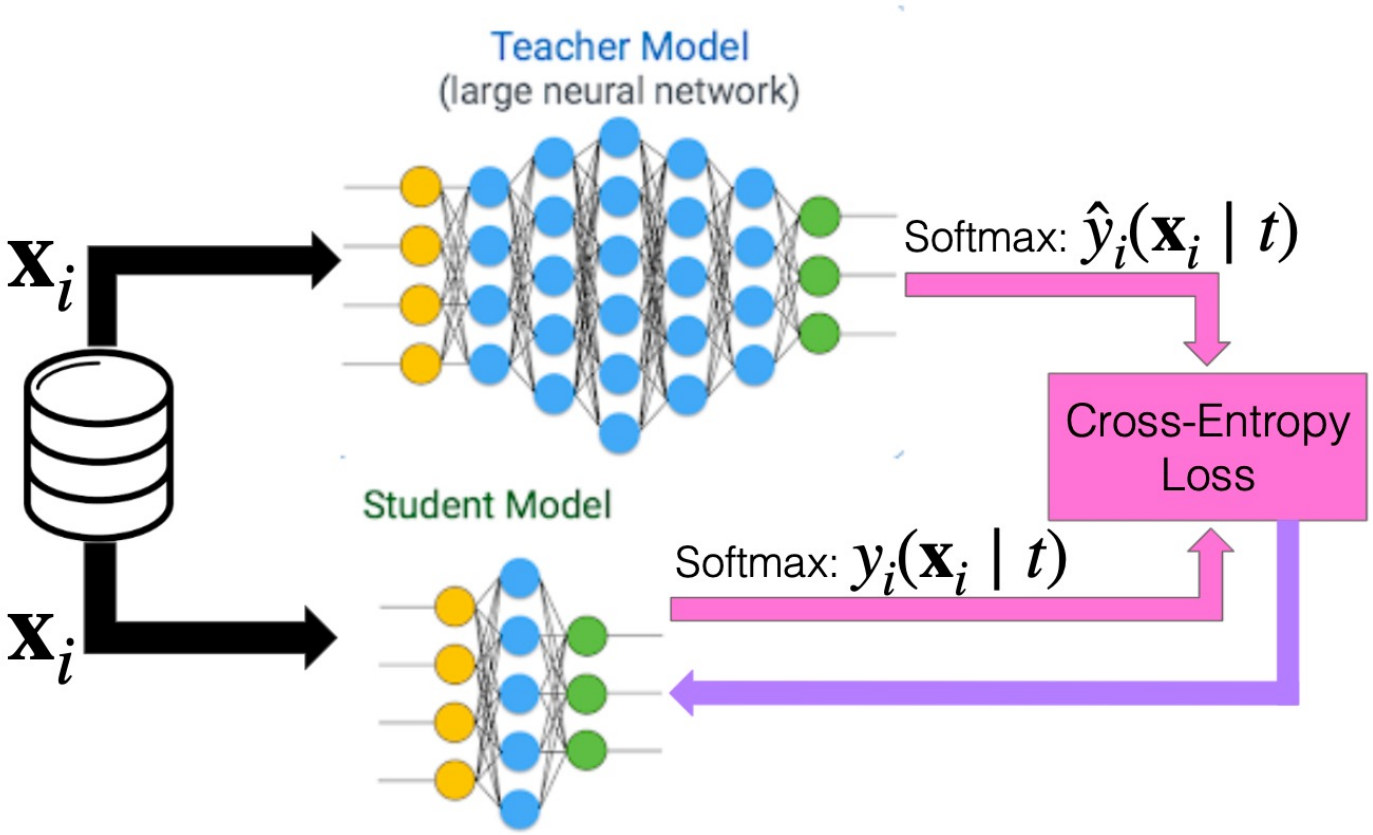
Claim: if the key encoder is changing too rapidly, ensuring similarity between encoded query and key becomes challenged

→ Want a slowly evolving key encoder

I still struggle to understand why this works… But it seems to

1911.05722

Knowledge Distillation



Knowledge Distillation is a process of transferring knowledge from a NN teacher to a NN student of equal or smaller size.

Teacher softmax out considered a soft classifier target for the student model with Cross-Entropy loss:

$$L = -\sum_i \hat{y}(x_i|t) \log y(x_i|t)$$

Where t is temperature parameter

Dino: Self-Distillation with No Labels

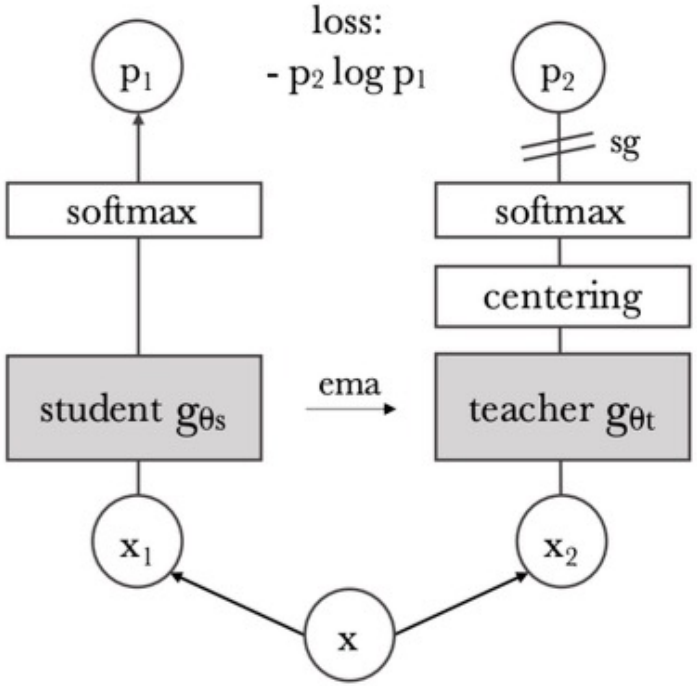
Train encoder (student) by comparing predictions to EMA of encoder (teacher)

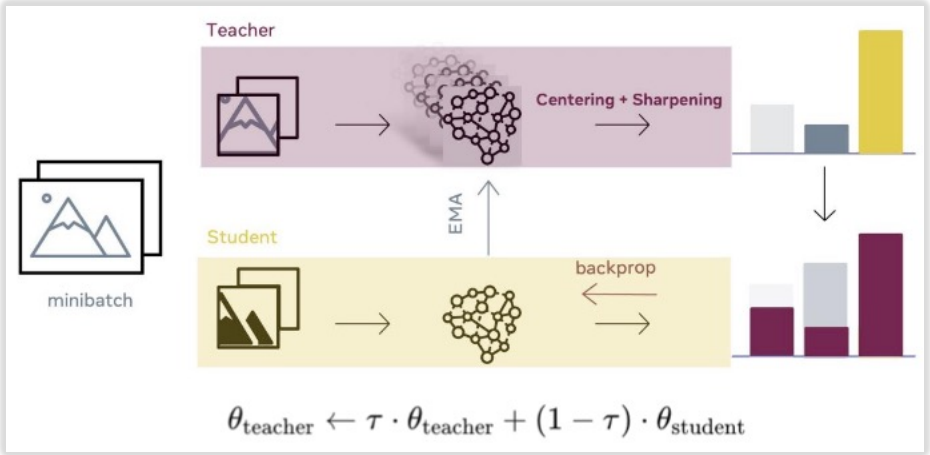
- Different augmentation in student and teacher
- Global and patch views

Compare softmax predictions

- No contrastive loss
- Compare augmentations against each other like SimCLR

Notably, no negative samples needed!





Dino: Self-Distillation with No Labels

This works strikingly well!

Unsupervised object segmentation

Emerging Properties in Self-Supervised Vision Transformers

Mathilde Caron[1,2] Hugo Touvron[1,3] Ishan Misra[1] Hervé Jegou[1]
Julien Mairal[2] Piotr Bojanowski[1] Armand Joulin[1]

[1] Facebook AI Research [2] Inria* [3] Sorbonne University

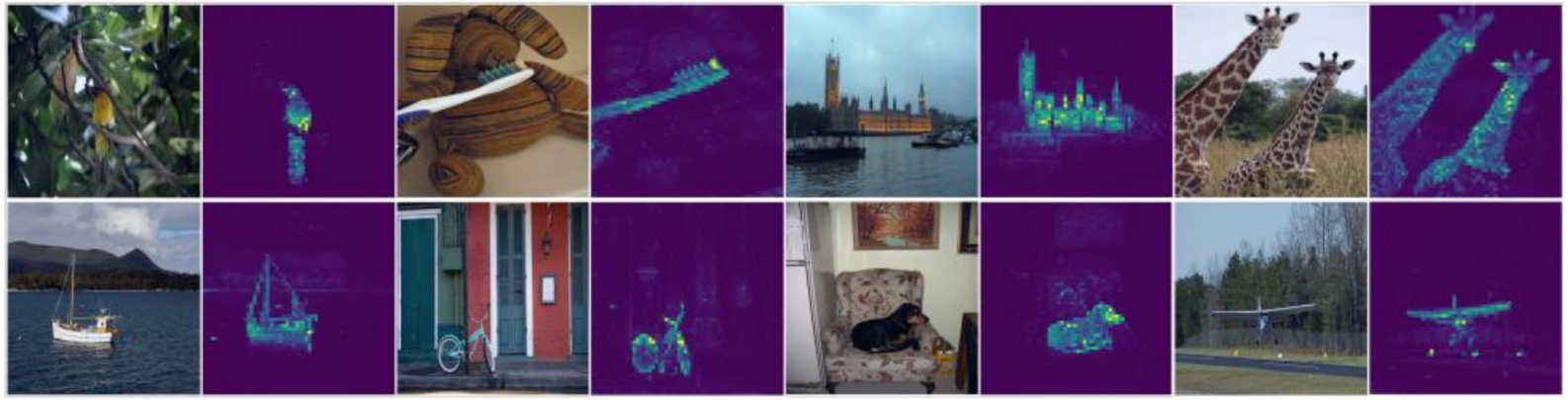


Figure 1: **Self-attention from a Vision Transformer with 8×8 patches trained with no supervision.** We look at the self-attention of the `[CLS]` token on the heads of the last layer. This token is not attached to any label nor supervision. These maps show that the model automatically learns class-specific features leading to unsupervised object segmentations.

*In this paper, we question if self-supervised learning provides new properties to Vision Transformer (ViT) [16] that stand out compared to convolutional networks (convnets). Beyond the fact that adapting self-supervised methods to this architecture works particularly well, we make the following observations: first, self-supervised ViT features contain explicit information about the semantic segmentation of an image, which does not emerge as clearly with supervised ViTs, nor with convnets. Second, these features are also excellent k-NN classifiers, reaching 78.3% top-1 on ImageNet with a small ViT. Our study also underlines the importance of momentum encoder [26], multi-crop training [9], and the use of small patches with ViTs. We implement our findings into a simple self-supervised method, called DINO, which we interpret as a form of self-**di**stillation with **no** labels. We show the synergy between DINO and ViTs by achieving 80.1% top-1 on ImageNet in linear evaluation with ViT-Base.*

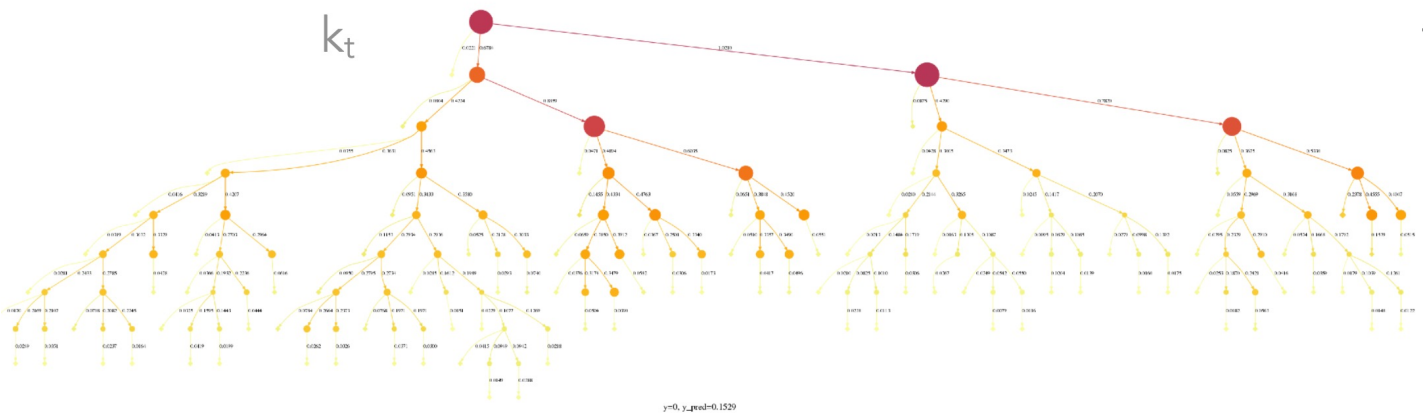
RINO: Renormalization group Invariance with NO labels

SSL training with DINO framework, but using k_t-clustering history as the augmentation method for jets

k_t-clustering steps interpreted as probing jet at different resolution scales

- Augmentations from n-prong subjets from k_t-clustering

By training model to be approximately invariant under examination at different steps of k_t-clustering → model learns invariance to resolution scale



2509.07486

Model size	Strategy	JETCLASS Accuracy	JETNET Accuracy
nano	Supervised	0.601 ± 0.060	$\mathbf{0.910 \pm 0.001}$
nano	RINO-Linear	$\mathbf{0.796 \pm 0.003}$	0.858 ± 0.001
nano	RINO-MLP	0.710 ± 0.052	0.859 ± 0.002
lite	Supervised	0.551 ± 0.038	$\mathbf{0.910 \pm 0.001}$
lite	RINO-Linear	$\mathbf{0.776 \pm 0.005}$	0.862 ± 0.003
lite	RINO-MLP	0.699 ± 0.025	0.867 ± 0.002
mini	Supervised	0.595 ± 0.049	$\mathbf{0.910 \pm 0.001}$
mini	RINO-Linear	0.772 ± 0.006	0.872 ± 0.003
mini	RINO-MLP	$\mathbf{0.803 \pm 0.019}$	0.871 ± 0.003
base	Supervised	0.629 ± 0.062	$\mathbf{0.910 \pm 0.001}$
base	RINO-Linear	$\mathbf{0.766 \pm 0.010}$	0.877 ± 0.002
base	RINO-MLP	0.752 ± 0.008	0.879 ± 0.003

Masked Modeling

Masked Language Modeling

Strategy: Mask some elements of input sequence, then predict missing tokens

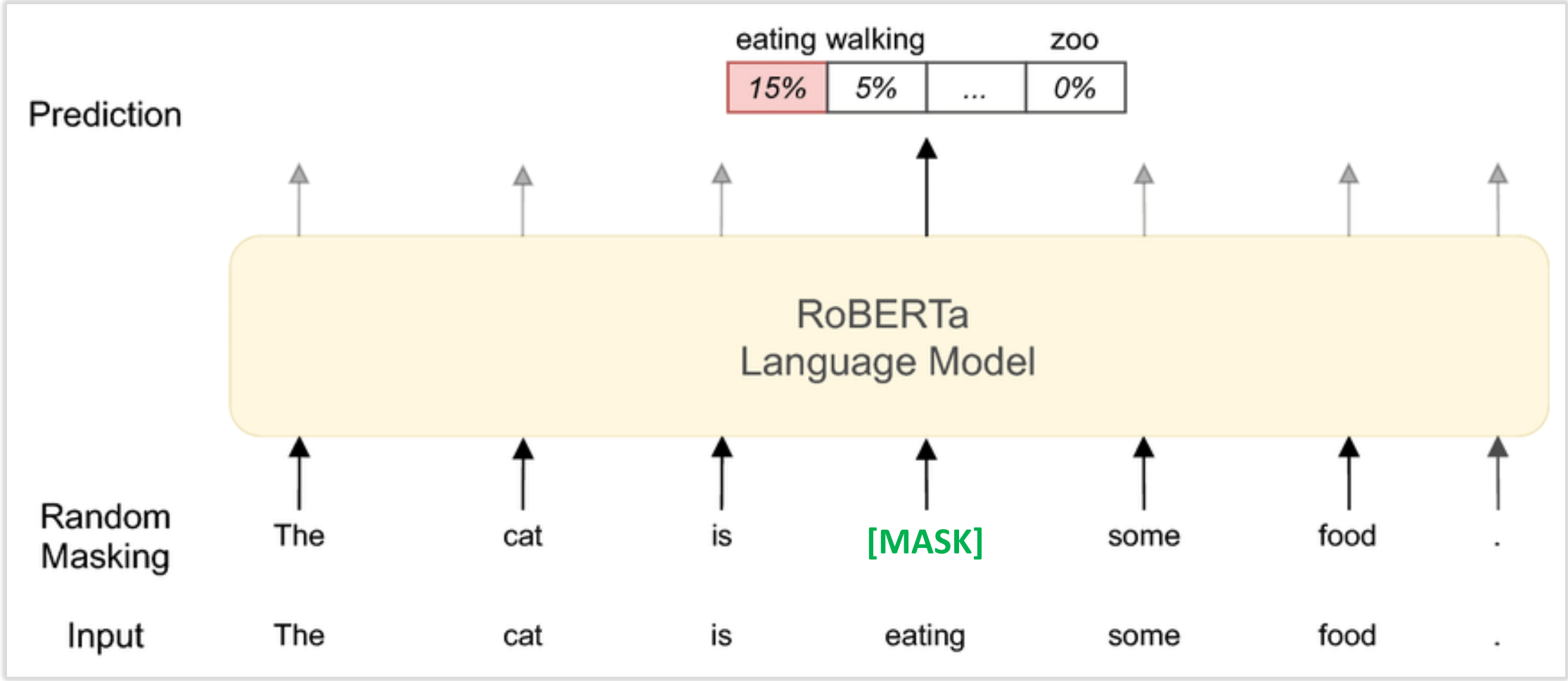


Image credit: 2104.01642

Masked Language Modeling (MLM)

What is the model learning?

Let $x = input$, and $x_M = inputs\ with\ some\ items\ masked$

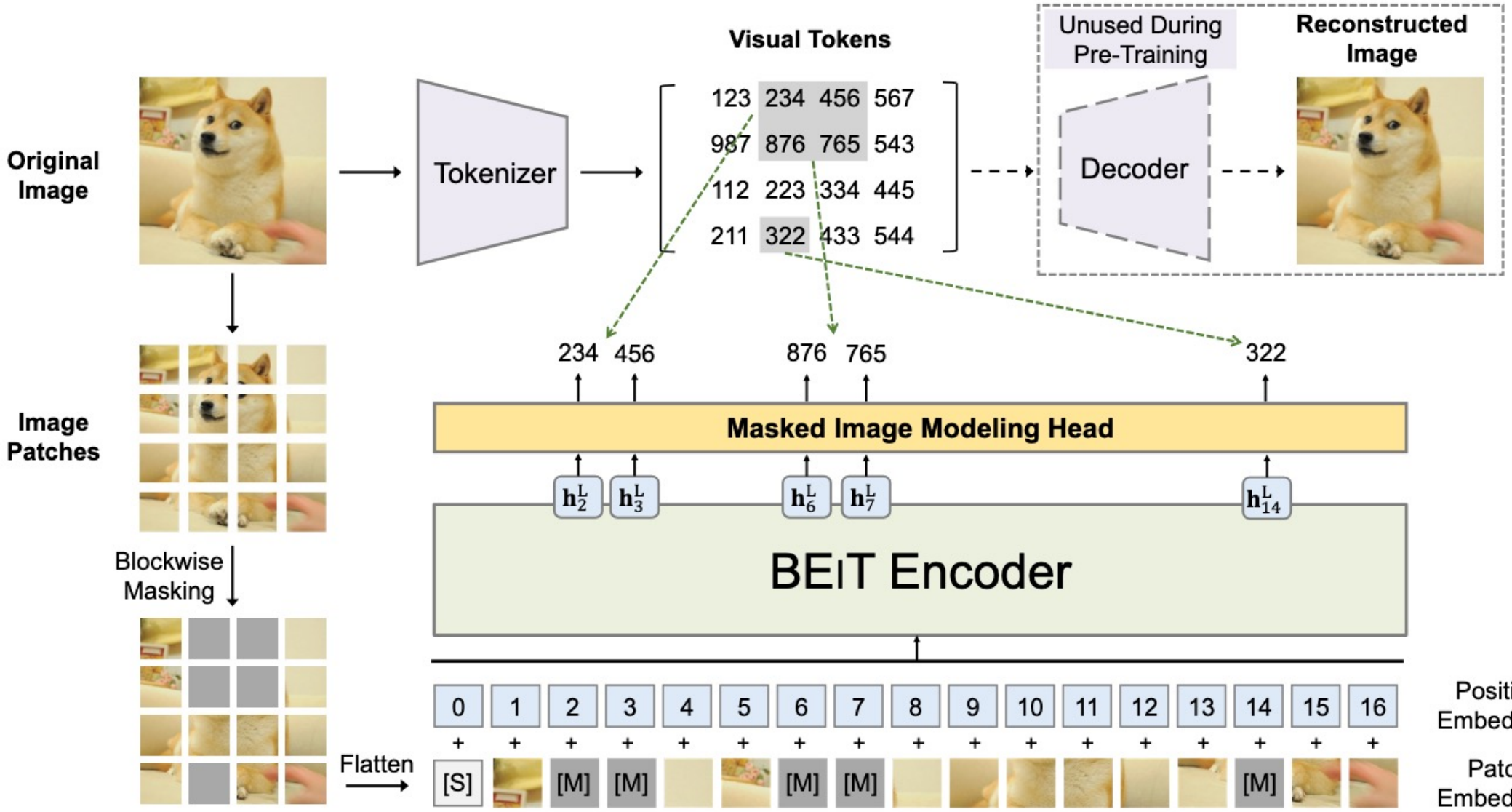
$$p(x \mid x_M)$$

Training target: masked token probability given unmasked tokens

When applied, all tokens unmasked
Why is the representation useful for unmasked tokens at inference time?

Encourages learning a contextualized meaning, not a meaning in isolation.
Once learned, contextualized meaning useful for all tokens.

Masked Image Modeling

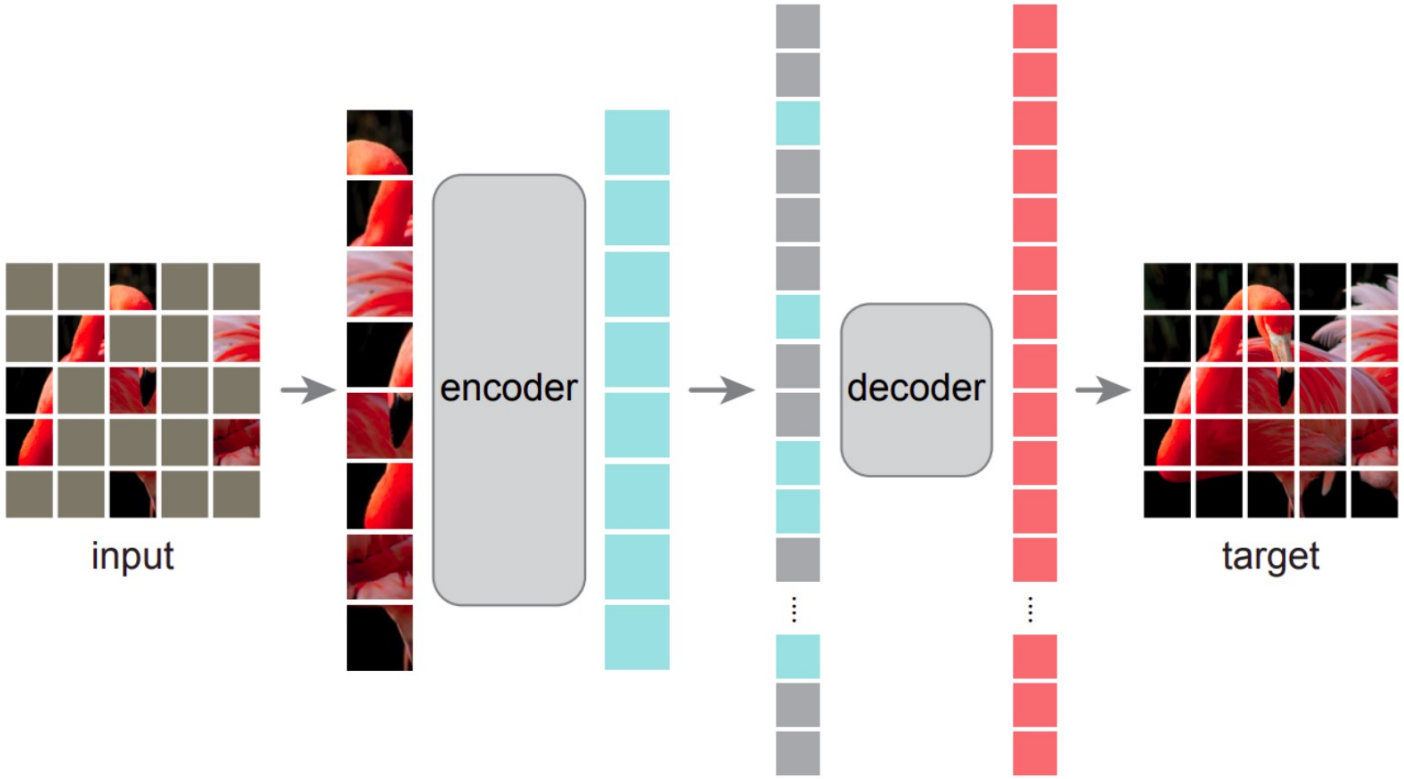


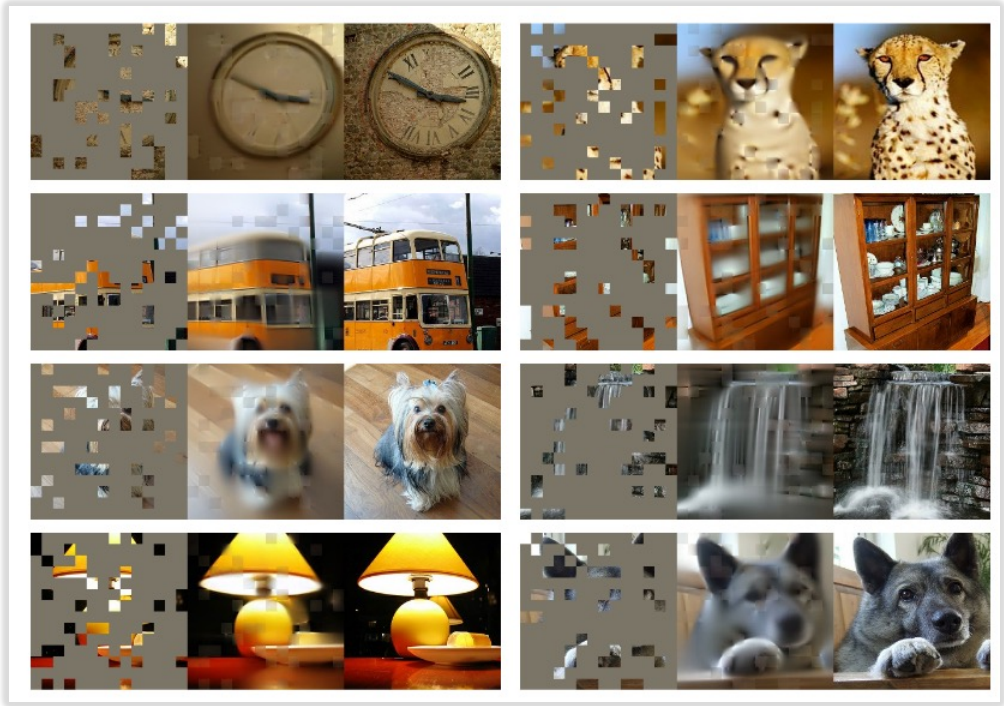
2106.08254
2208.06366

Masked AutoEncoder (MAE)

MAE only encodes unmasked elements. Decoder also see masked tokens

Significantly more scalable that MIM, since only encode small fraction of inputs

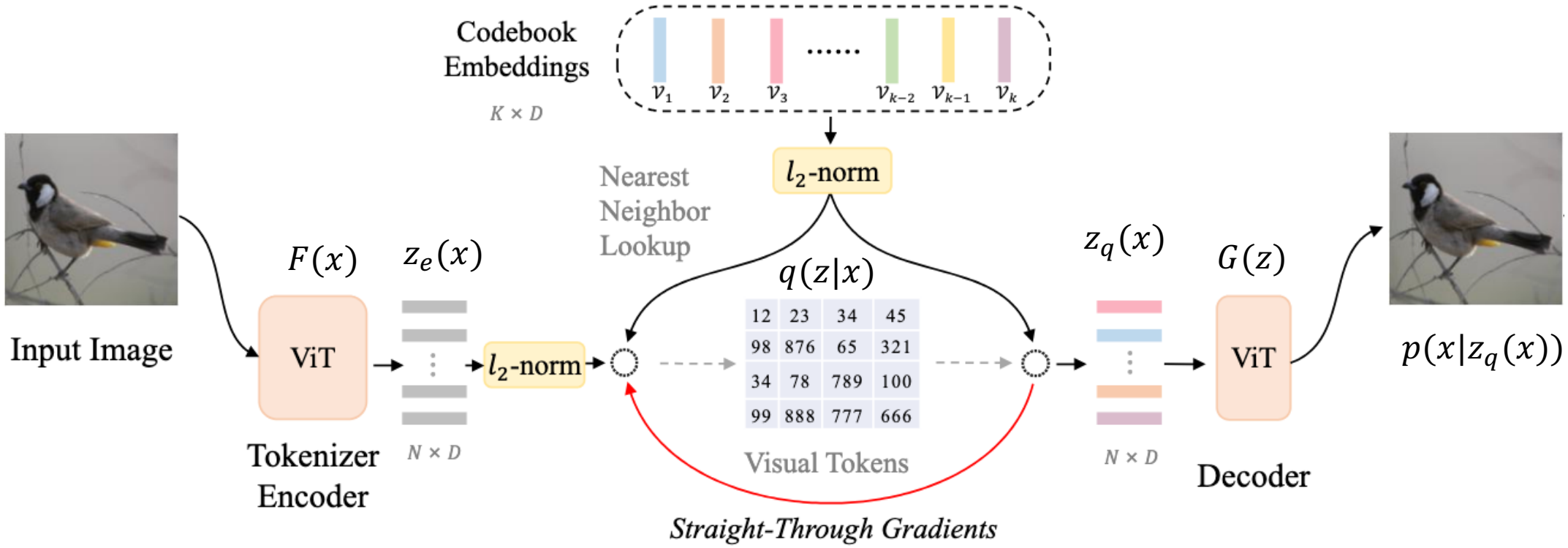




2111.06377

Aside: Tokenizing Continuous Data

Discretizing Tokens with Vector Quantized (VQ)-VAE



VAE but (a) only discrete set of latent vectors allowed, (b) deterministic

VQ-VAE

Latent vector z_q determined by closest codebook vector e_j to encoder-vector z_e

$$q(z=k|x) = \begin{cases} 1 & \text{for } k = \text{argmin}_j \|z_e(x) - e_j\|_2, \\ 0 & \text{otherwise} \end{cases}$$

Loss is a combination of usual MSE reconstruction loss, updates to codebook, and updates to encoder

$$z_q(x) = e_k, \quad \text{where} \quad k = \text{argmin}_j \|z_e(x) - e_j\|_2$$

Discretization is non-differentiable

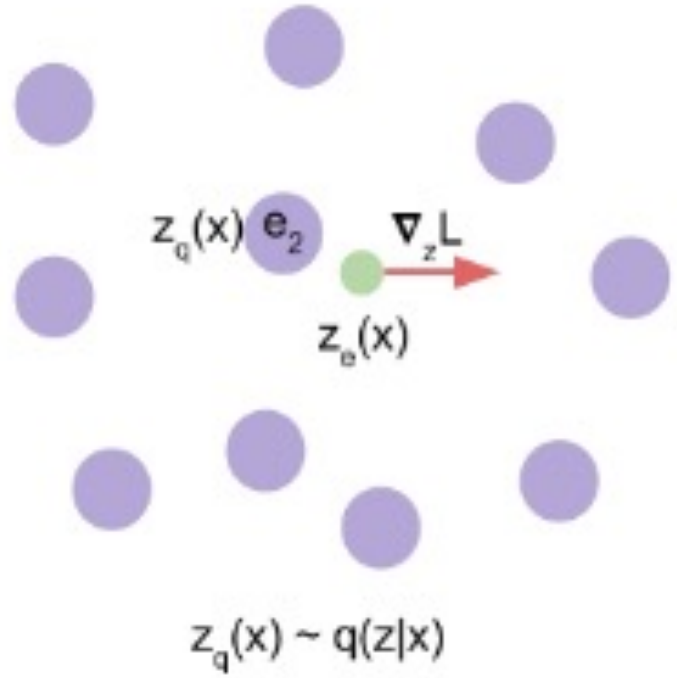
- Straight-through estimator on reconstruction to get encoder grads
- Additional losses to learn codebook

$$L = \log p(x|z_q(x)) + \|\text{sg}[z_e(x)] - e\|_2^2 + \beta\|z_e(x) - \text{sg}[e]\|_2^2,$$

"stop-gradient"

Training with a Straight-Through Estimator

$$\frac{\partial L}{\partial F} = \frac{\partial L}{\partial x'} \frac{\partial x'}{\partial z_q} \frac{\partial z_q}{\partial z_e} \frac{\partial z_e}{\partial F}$$

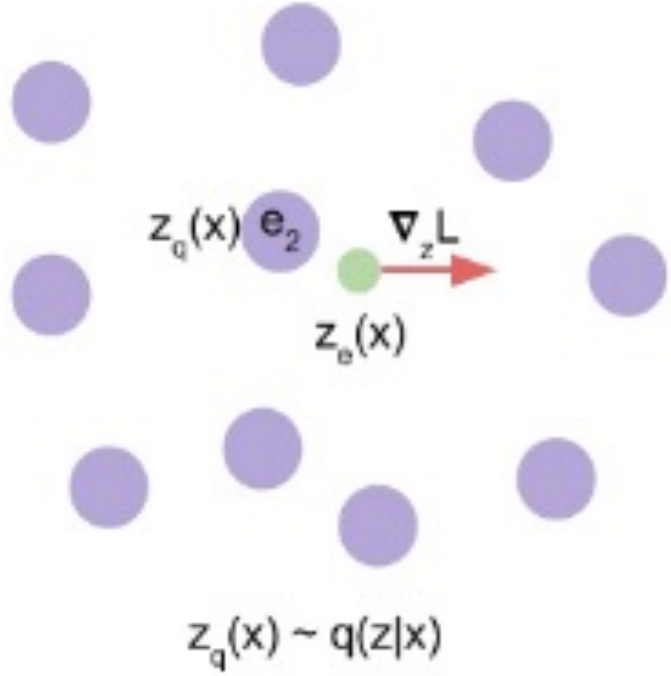


Training with a Straight-Through Estimator

$$\frac{\partial L}{\partial F} \approx \frac{\partial L}{\partial x'} \frac{\partial x'}{\partial z_q} \frac{\partial z_q}{\partial z_e} \frac{\partial z_e}{\partial F} = \frac{\widehat{\partial L}}{\partial F}$$

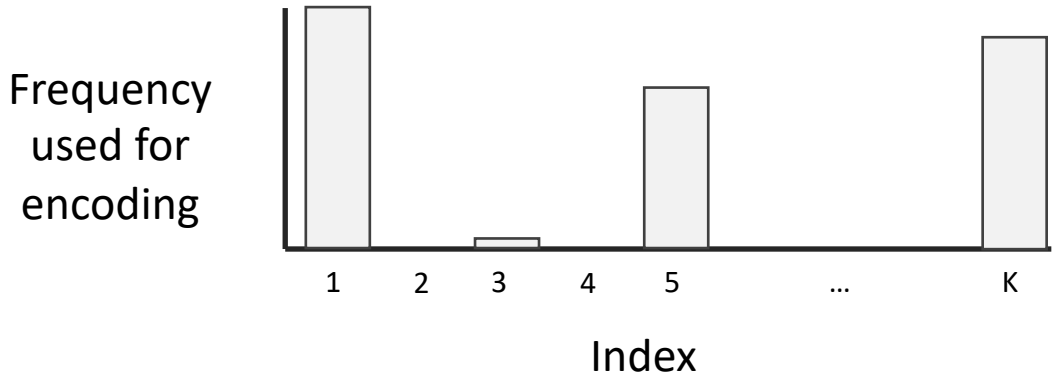
$\frac{\partial z_q}{\partial z_e}$ → 1

Straight-Through



Ignore gradient of discretization step

Codebook Collapse



During training, when examining the frequency of selecting codebooks elements

Often observe “codebook collapse”: only few of the codes are selected and used

Require careful initialization of codebook vectors, and re-initializing un-used codebook vectors during training time

- Not going through all the tricks now… see 2305.08842 … Bear in mind when training!

VQ-VAE Seems Painful… Why do we want it?

A. For vision-language models… language space is tokenized. If we want to “talk” to images, we probably need to tokenize them

B. Even outside language, transformers operate very well sequences of discretized tokens… is it absolutely needed? Is it a secret inductive bias? Not totally clear

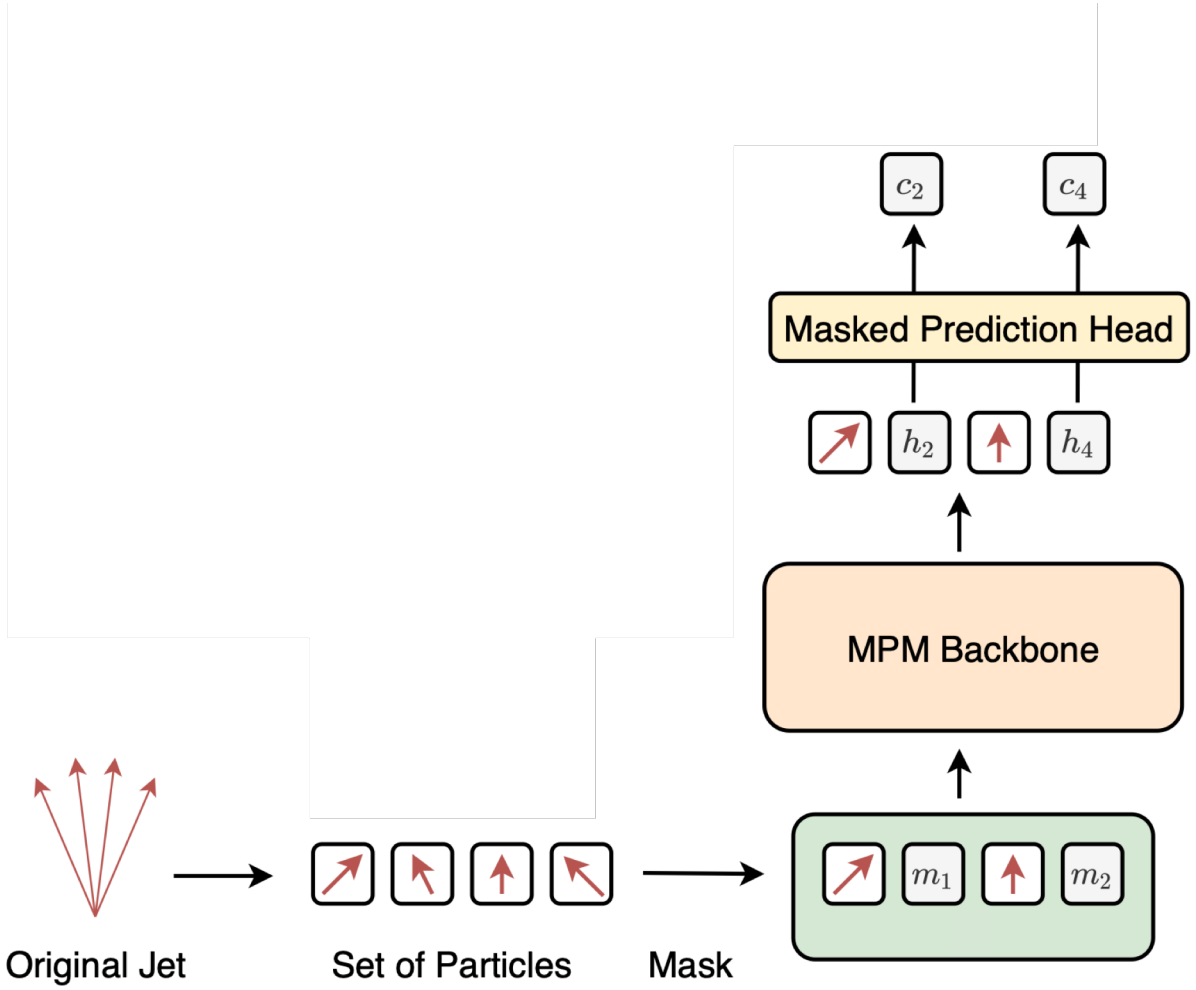
C. Practical: Easier to predict discrete tokens / categorical posteriors

End Aside

Masked Particle Modeling

Mask individual particles and predict their properties

- 40M parameter backbone model with 100M jets for pre-training



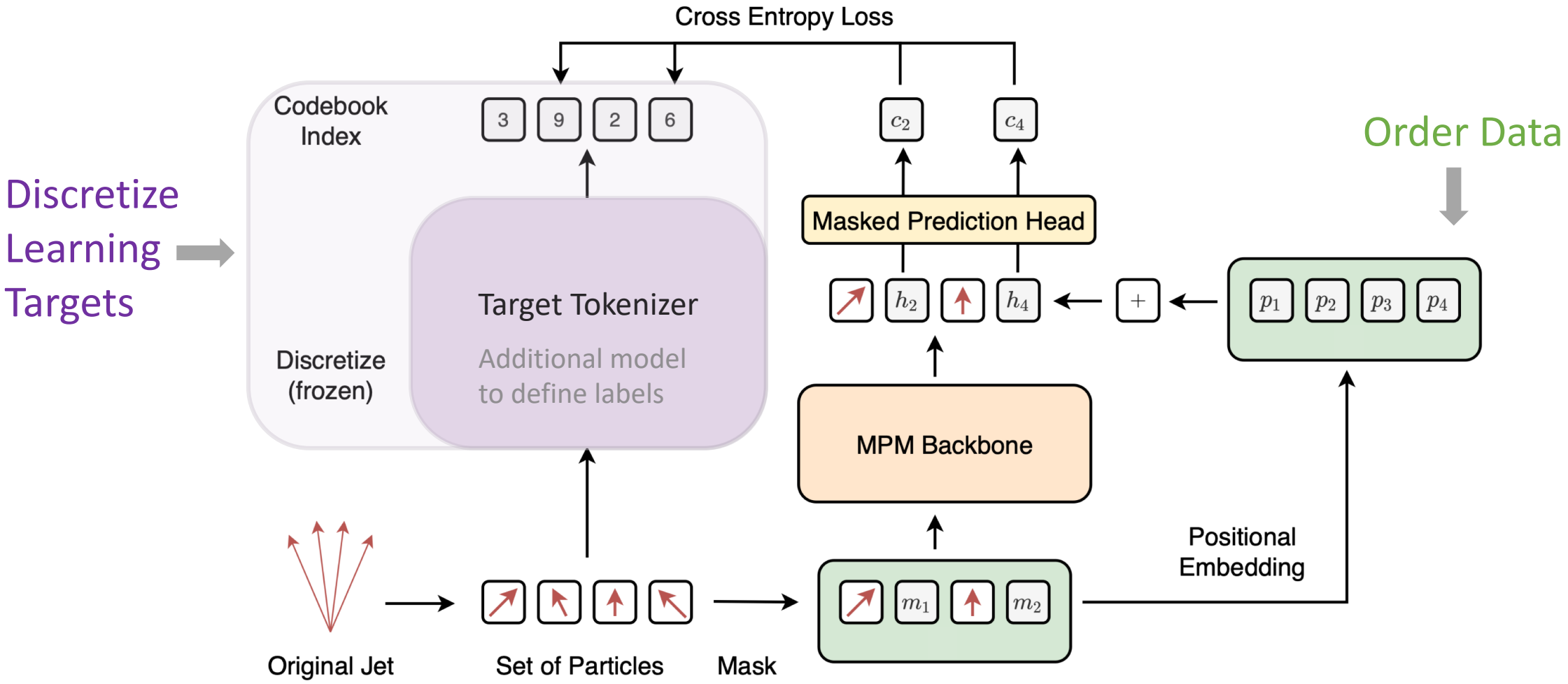
Weird thing we might want:
Permutation invariant backbone

Why?
Particles are an unordered set

Masked Particle Modeling

Mask individual particles and predict their properties

- 40M parameter backbone model with 100M jets for pre-training

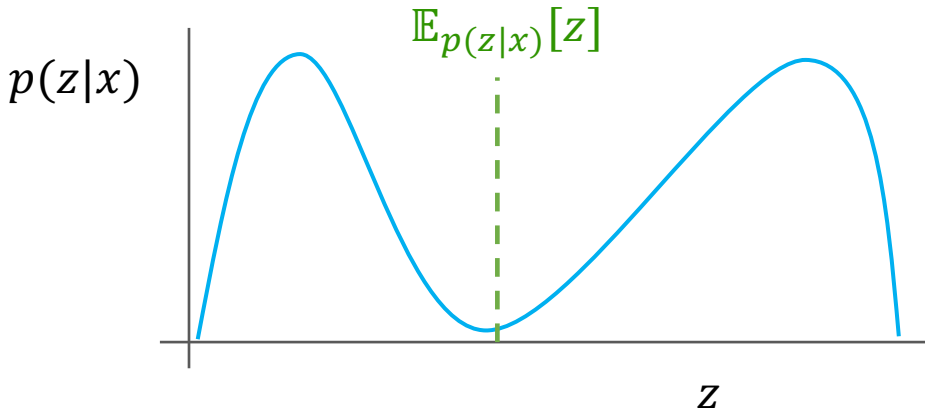


Challenges

1) Particle Features (momentum, position, …) are not discrete:

Regress continuous values?

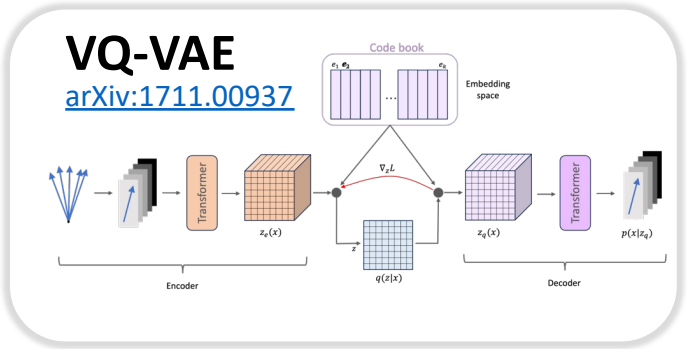
In multi-modal distributions regression may not provide a useful prediction



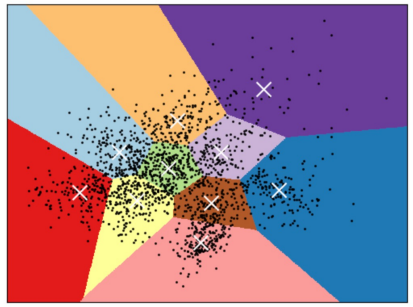
Challenges

1) Particle Features (momentum, position, …) are not discrete:

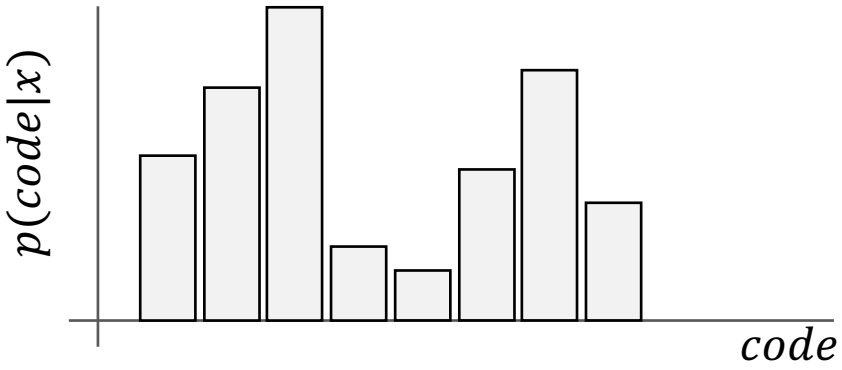
Tokenize Training Targets



K-Means Clustering



Discrete Density Estimation of Target



With discretized training targets, predict a categorical distribution over codes

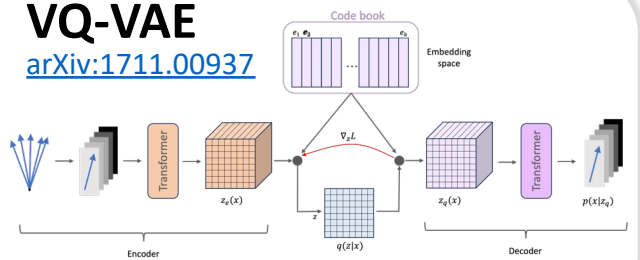
This is a full (discretized) posterior over outputs, not an average (like regression)

Challenges

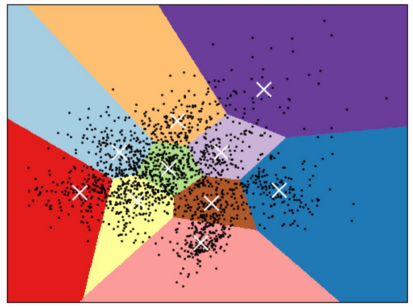
1) Particle Features (momentum, position, ...) are not discrete:

Tokenize Training Targets

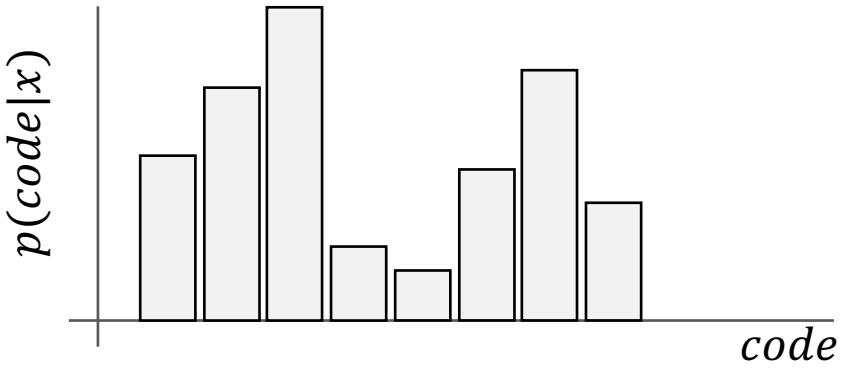
VQ-VAE
arXiv:1711.00937



K-Means Clustering



Discrete Density Estimation of Target



What about tokenizing inputs to the model?

Loss of input resolution seemed to hurt performance

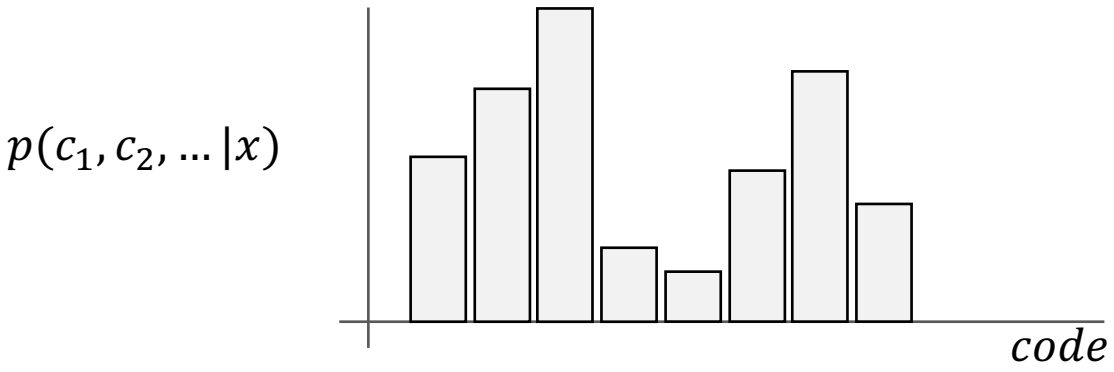
Ordering	Inputs	Loss	Accuracy
no ordering	continuous	VQ-VAE classification	54.1%
order head	**continuous**	**VQ-VAE classification**	**56.8%**
order backbone	continuous	VQ-VAE classification	53.4%
order head	quantized	VQ-VAE classification	51.1%
order head	quantized	K-means classification	49.3%
order head	continuous	K-means classification	56.2%
order head	continuous	regression	48.9%
order backbone	continuous	regression	46.3%

Challenges

2) Particles are not an ordered sequence, does order matter?

Without ordering, all masked elements have the same predicted target distribution

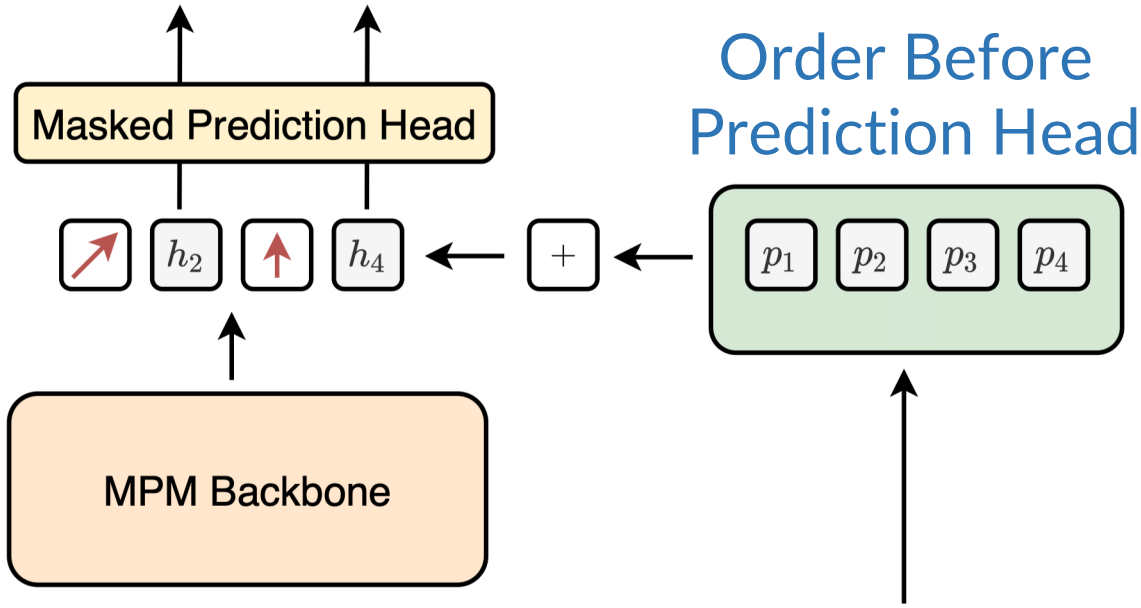
Every masked token looks the same to the model



Challenges

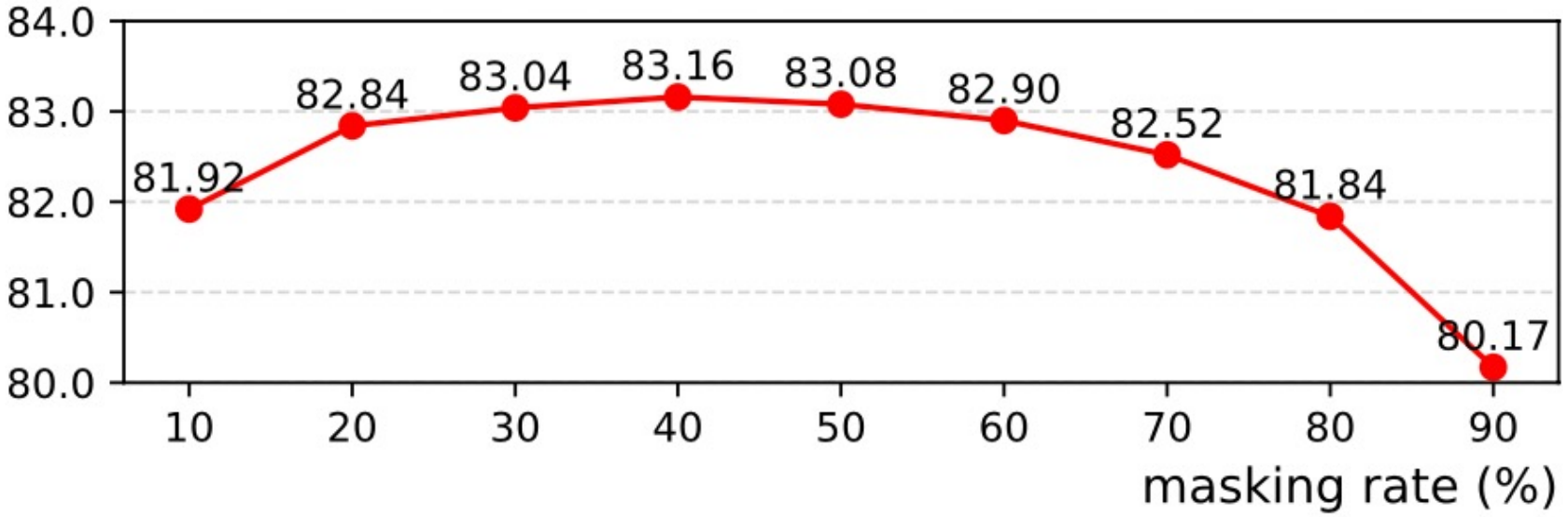
2) Particles are not an ordered sequence, does order matter?

Ordering only prediction head maintains backbone permutation symmetry



How much to mask?

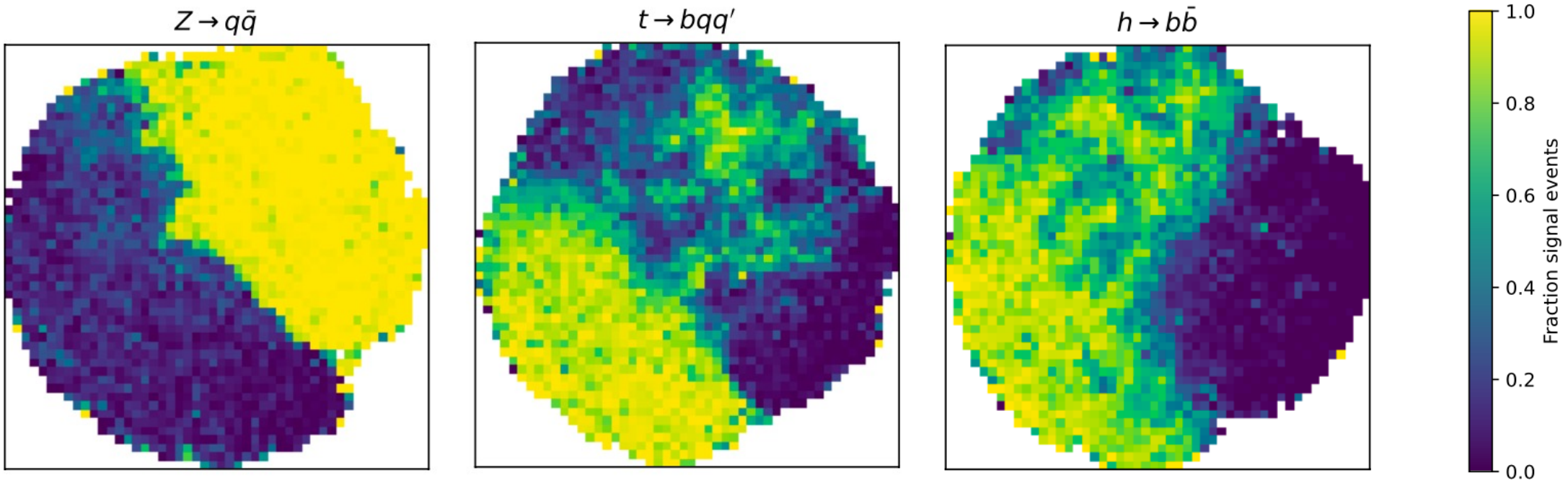
Classification Accuracy [%]

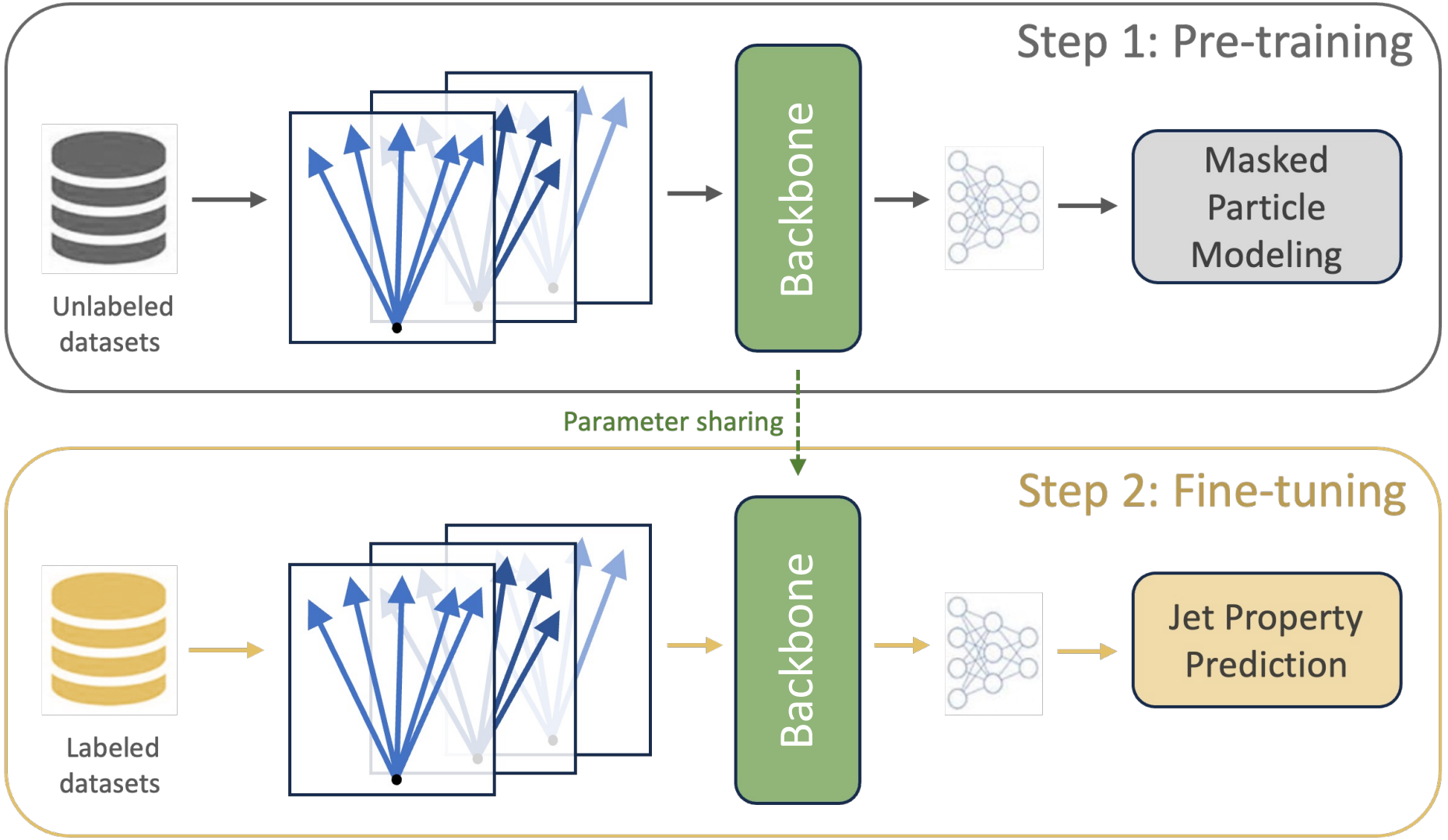


Quite a lot…

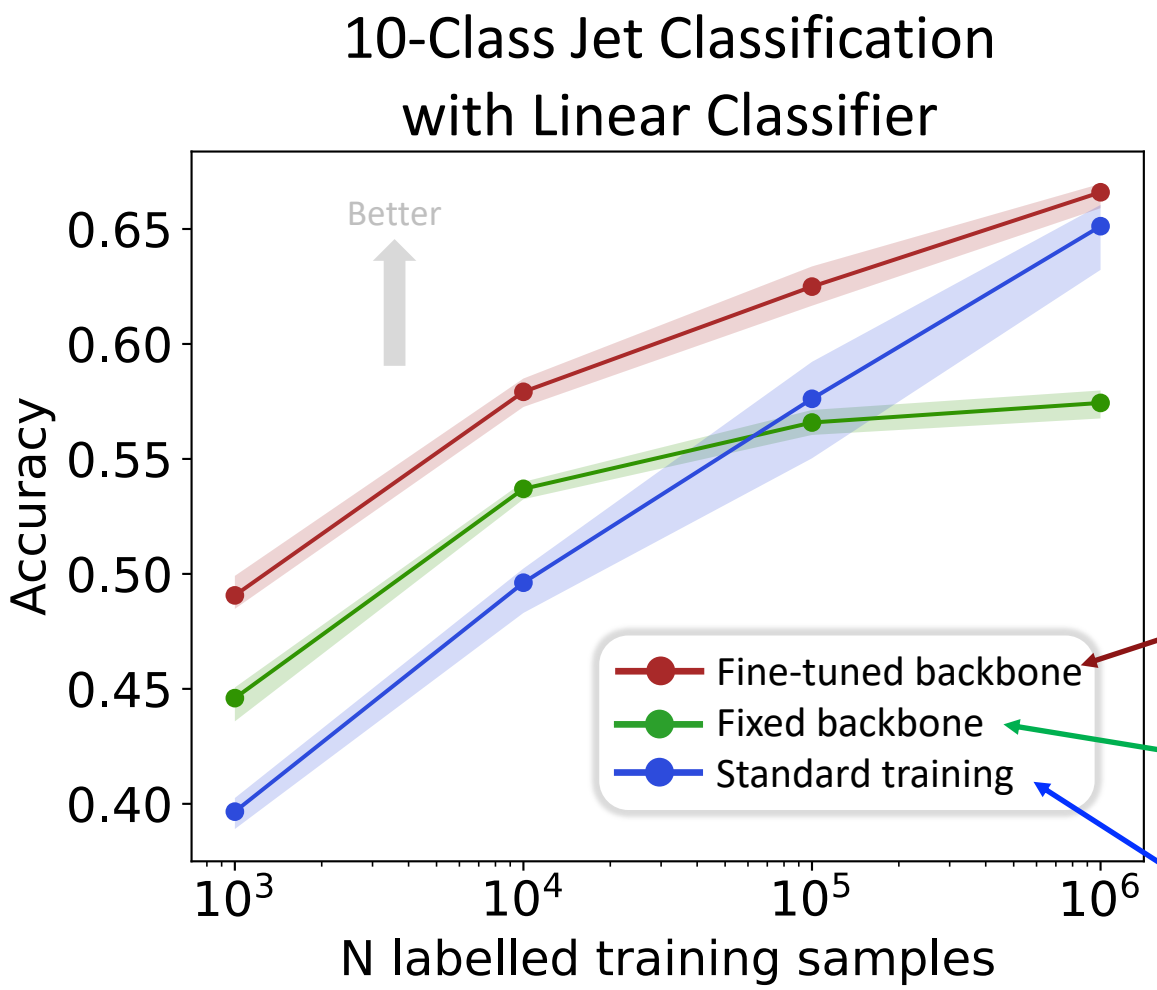
Consistent with MLM and MIM approaches (up to 70 or 80% masked)

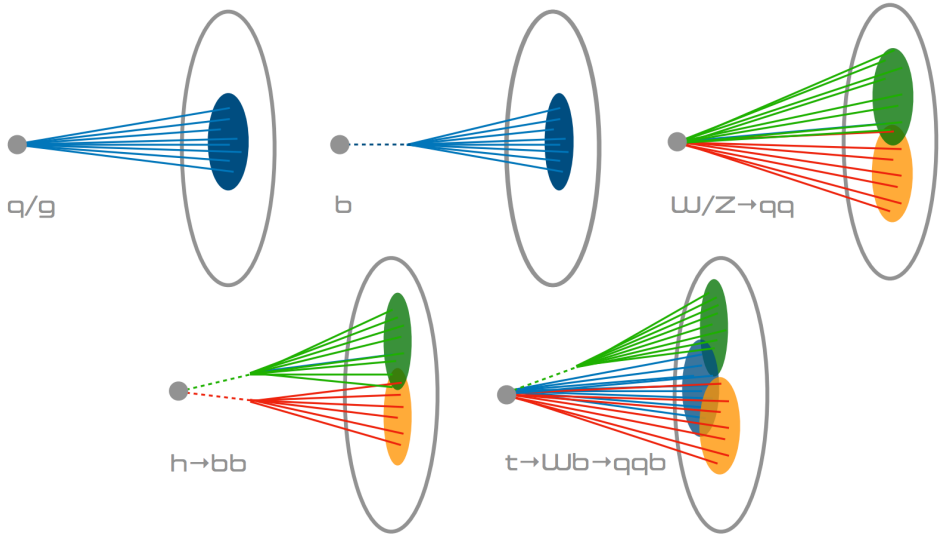
2D tSNE Projections of Representations




Step 1: Pre-training
Unlabeled datasets
Backbone
Masked Particle Modeling
Parameter sharing
Step 2: Fine-tuning
Labeled datasets
Backbone
Jet Property Prediction

Pre-training Gives Better Performance and Can Use Less Data on Downstream Tasks





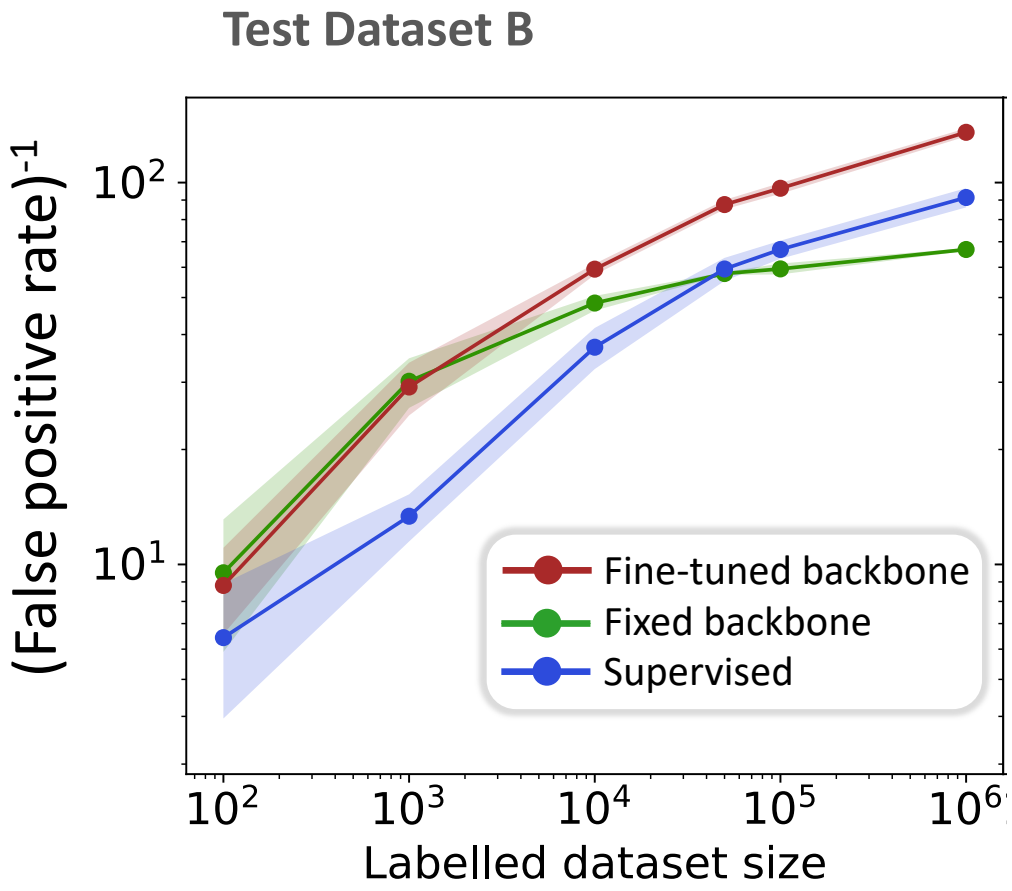
Fine-tune backbone along with small task specific classifier

Fix backbone after pre-training, train small task specific classifier

Standard supervised training on data for the specific task

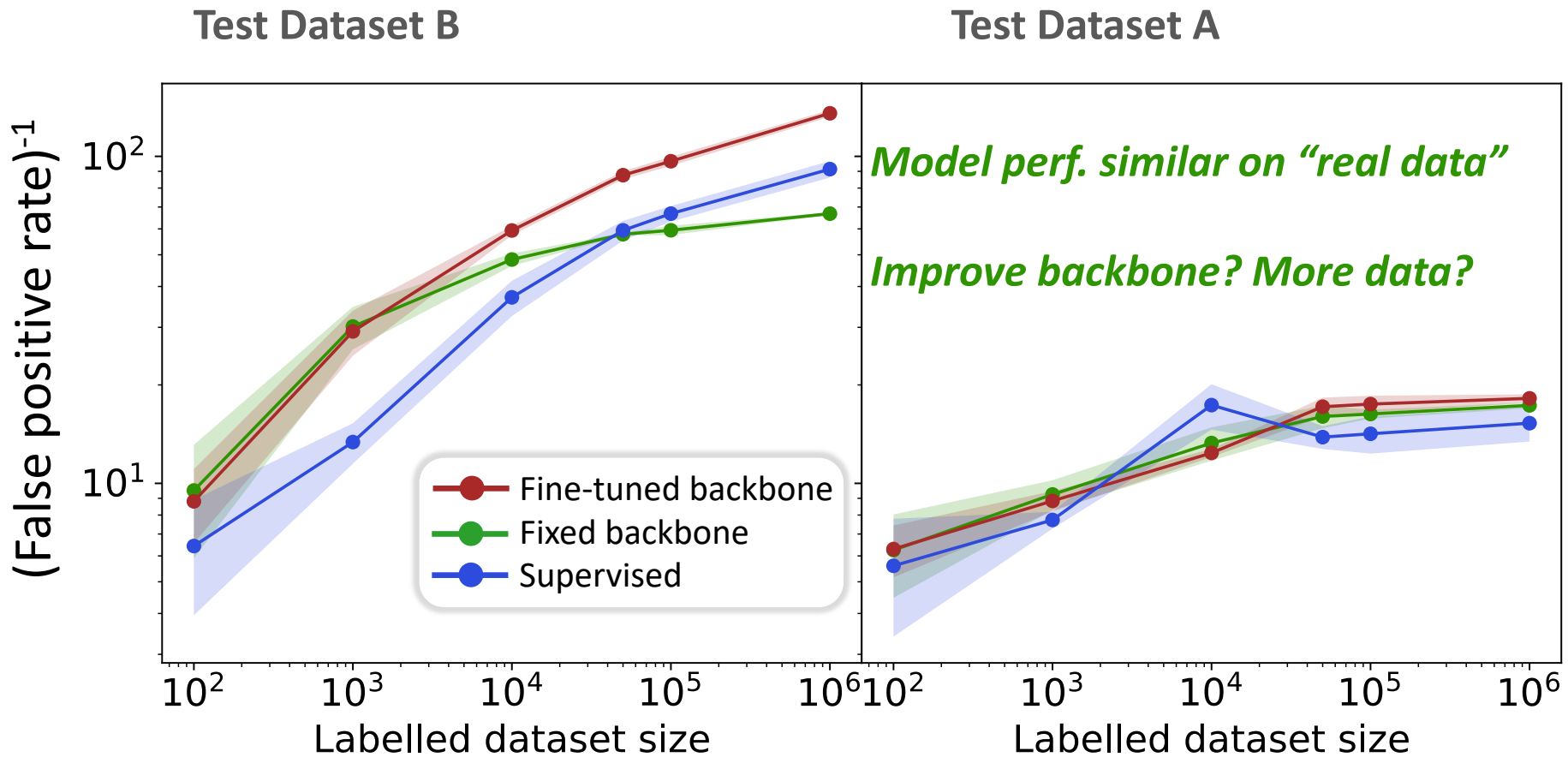
Dataset Transfer... Towards Domain Adaptation

1. Pre-train on “unlabeled” dataset A → treat like “real data”
2. Fine-tune on labeled new dataset B → treat like simulations



Dataset Transfer... Towards Domain Adaptation

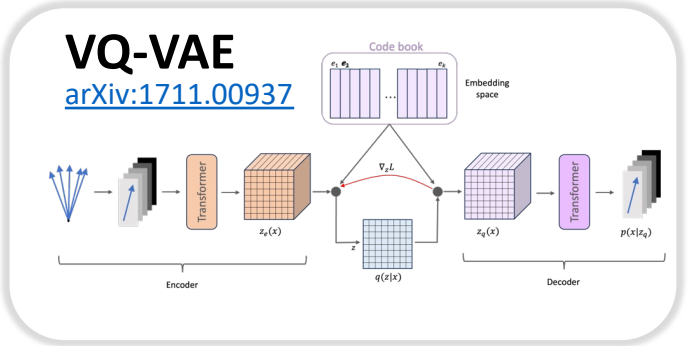
1. Pre-train on “unlabeled” dataset A → treat like “real data”
2. Fine-tune on labeled new dataset B → treat like simulations
3. Examine performance on “real” dataset A

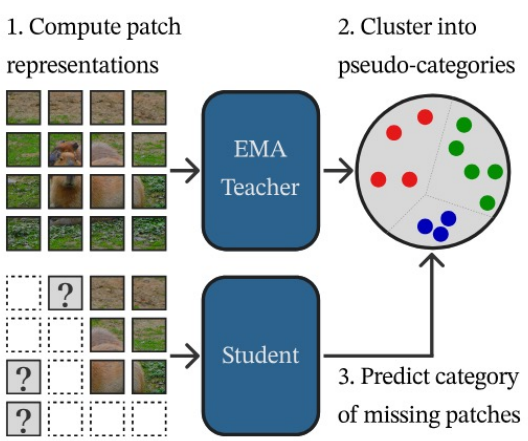


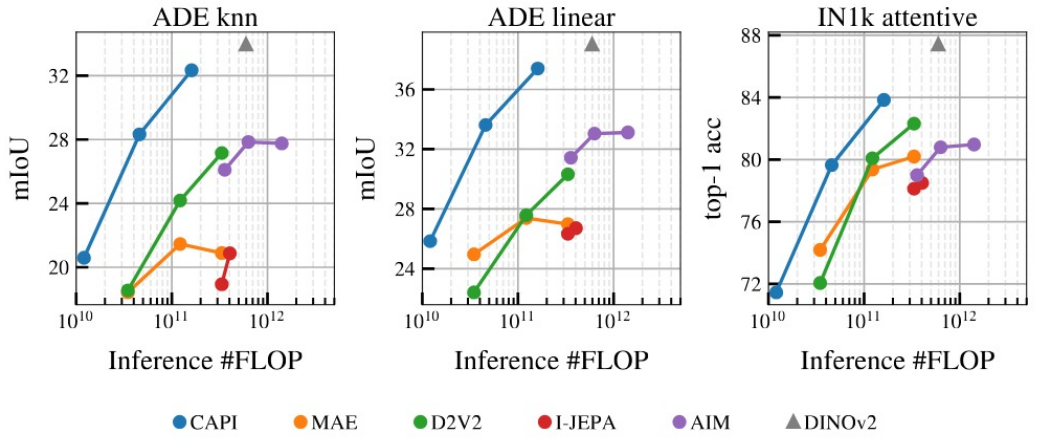
Is Tokenization Needed? [2409.12589](2409.12589)

Tokenizing enables **"binned" density estimation**

- K-Means clustering *(easier to train than VQ-VAE)*







Related idea in masked image modeling [2502.08769](2502.08769)

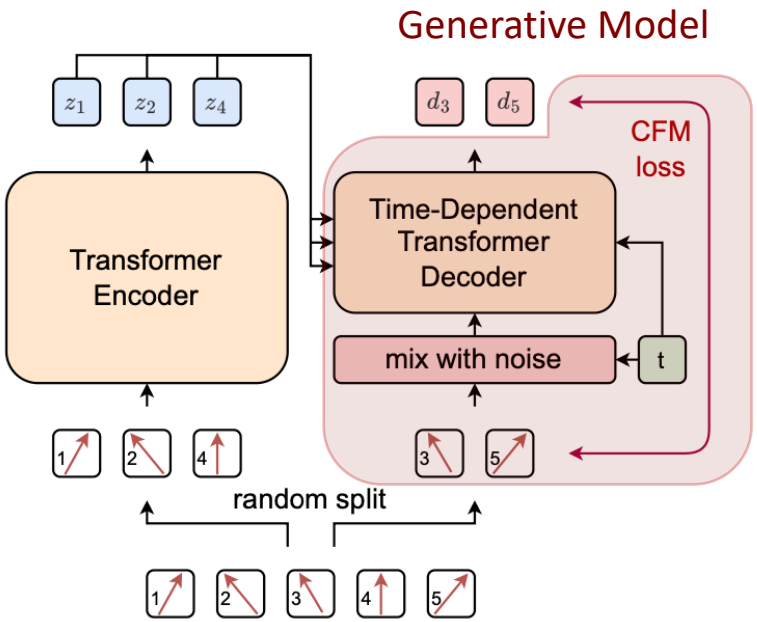
Is Tokenization Needed?

[2409.12589](2409.12589)

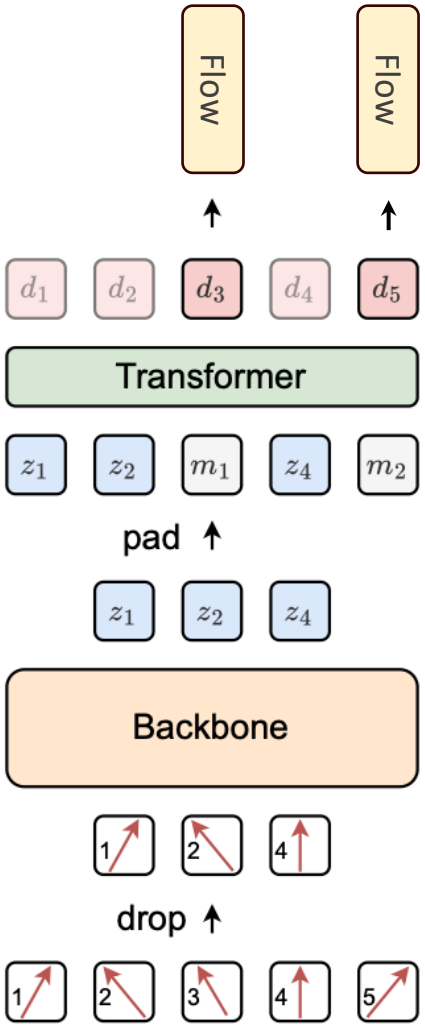
Tokenizing enables **"binned" density estimation**

- K-Means clustering *(easier to train than VQ-VAE)*

Can do **continuous density estimation** using **generative models** conditioned on unmasked data



Also updated model to MAE architecture



Is Tokenization Needed?

2409.12589

Tokenizing enables **"binned" density estimation**

- K-Means clustering *(easier to train than VQ-VAE)*

Can do **continuous density estimation** using **generative models** conditioned on unmasked data

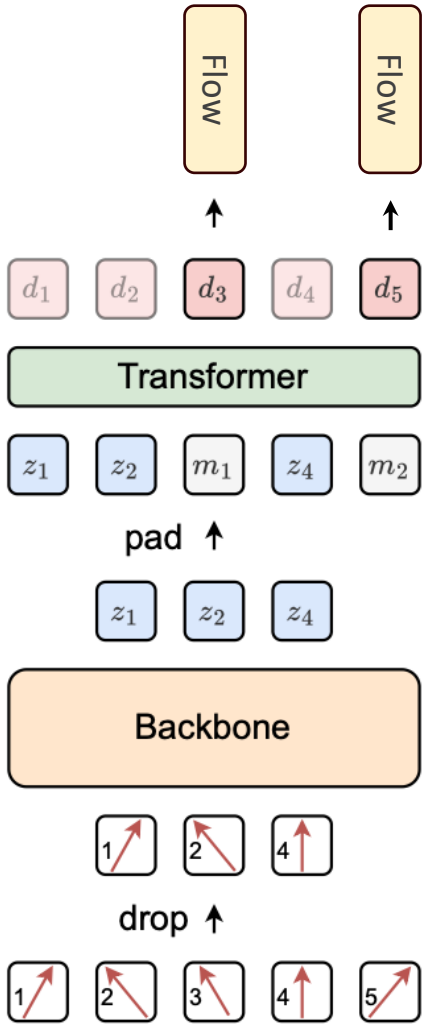
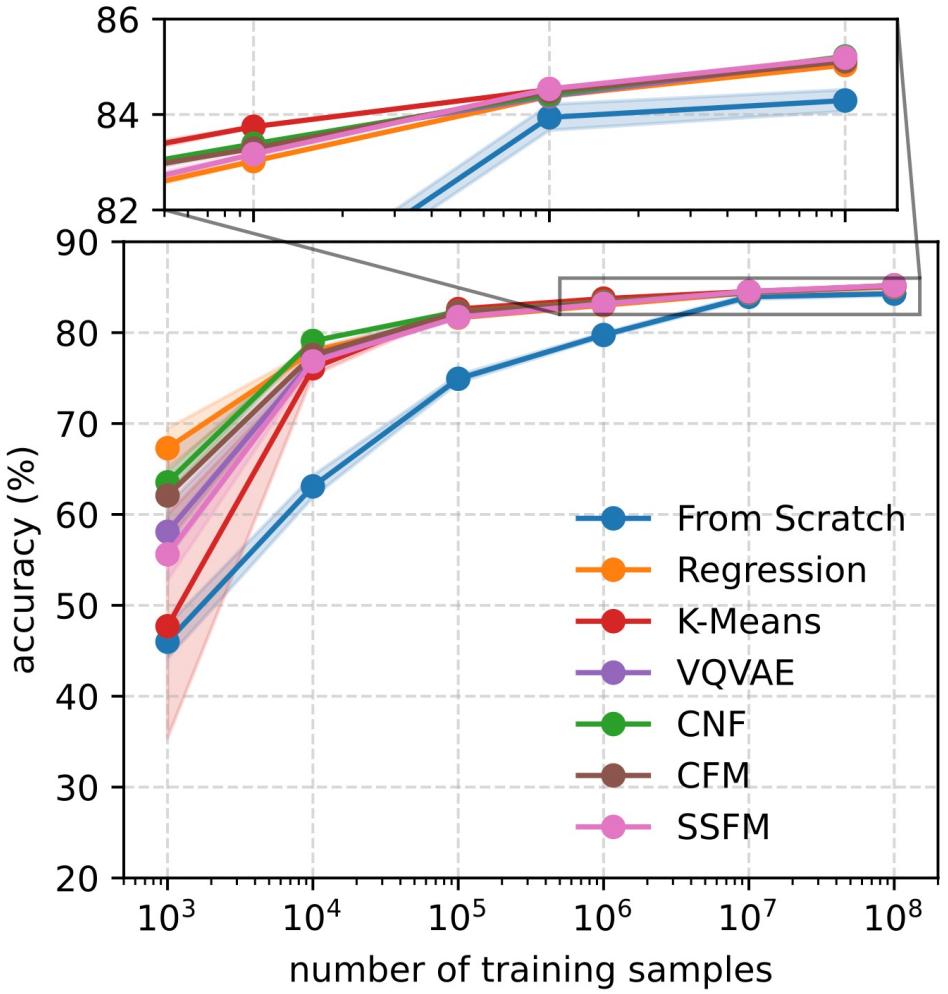


Table 1: The effects of the model redesign on the accuracy of a classifier head trained using the encoder outputs. All models except the final iteration were trained using 200k training steps, a mask rate of 30%, and a 2-layer decoder.

	k-means	
MPMv1 using (p_T, η, ϕ) [1]	56.2	
+ updated transformer layers	62.2	↑ 6.0
+ impact parameter features	70.2	↑ 8.0
+ constituent ID feature and ID reconstruction task	74.0	↑ 3.8
+ transformer as decoder (MAE)	81.4	↑ 7.4
+ registers	83.0	↑ 1.6
+ longer train (1M steps) + deeper decoder + 40% mask rate	84.0	↑ 1.0

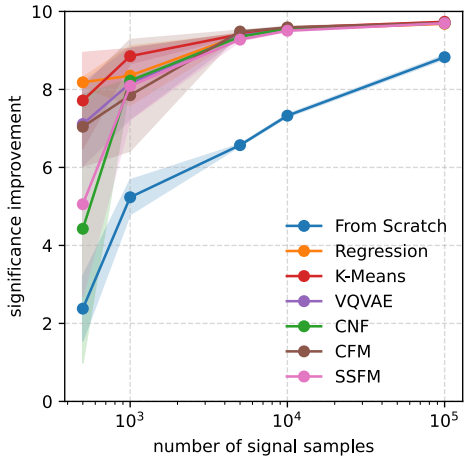
Many Different Masked Learning Strategies Work!

10-Class Jet Classification
with Transformer Head

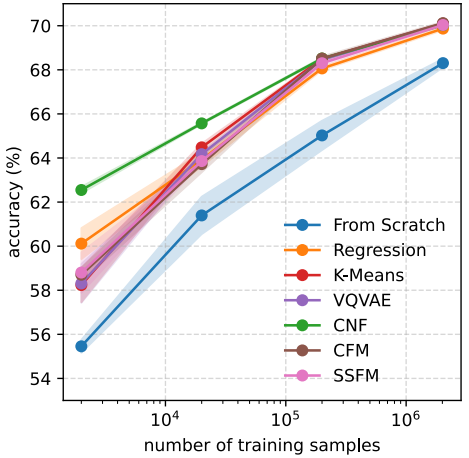


Successful Transfer & Improved Performance on Many Downstream Tasks

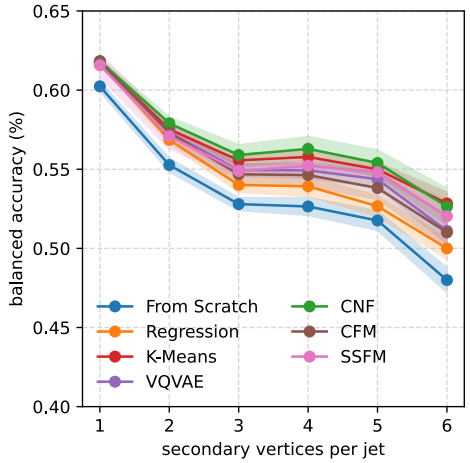
Weakly Supervised Classification



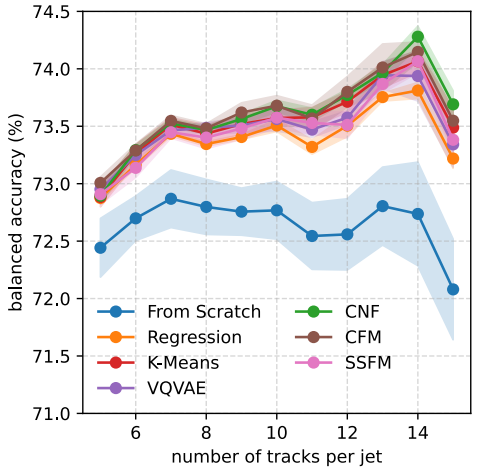
Out-of Distribution Classification on different data set (b-tagging)



Segmentation (vertex finding)

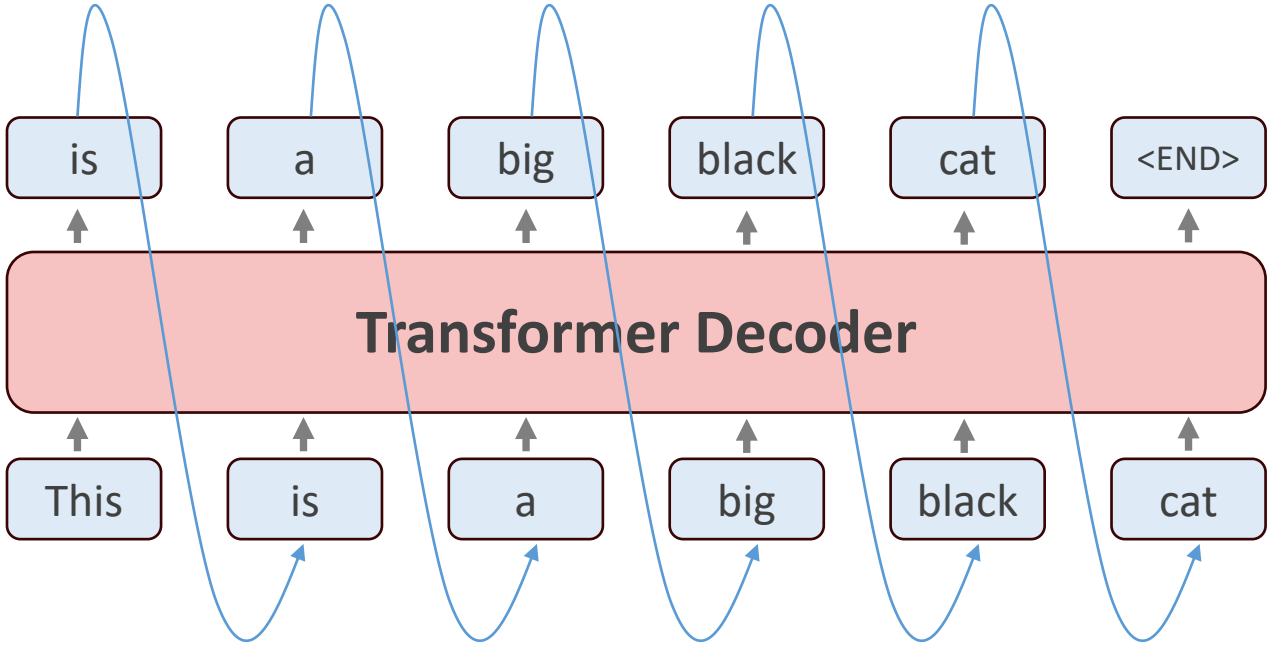


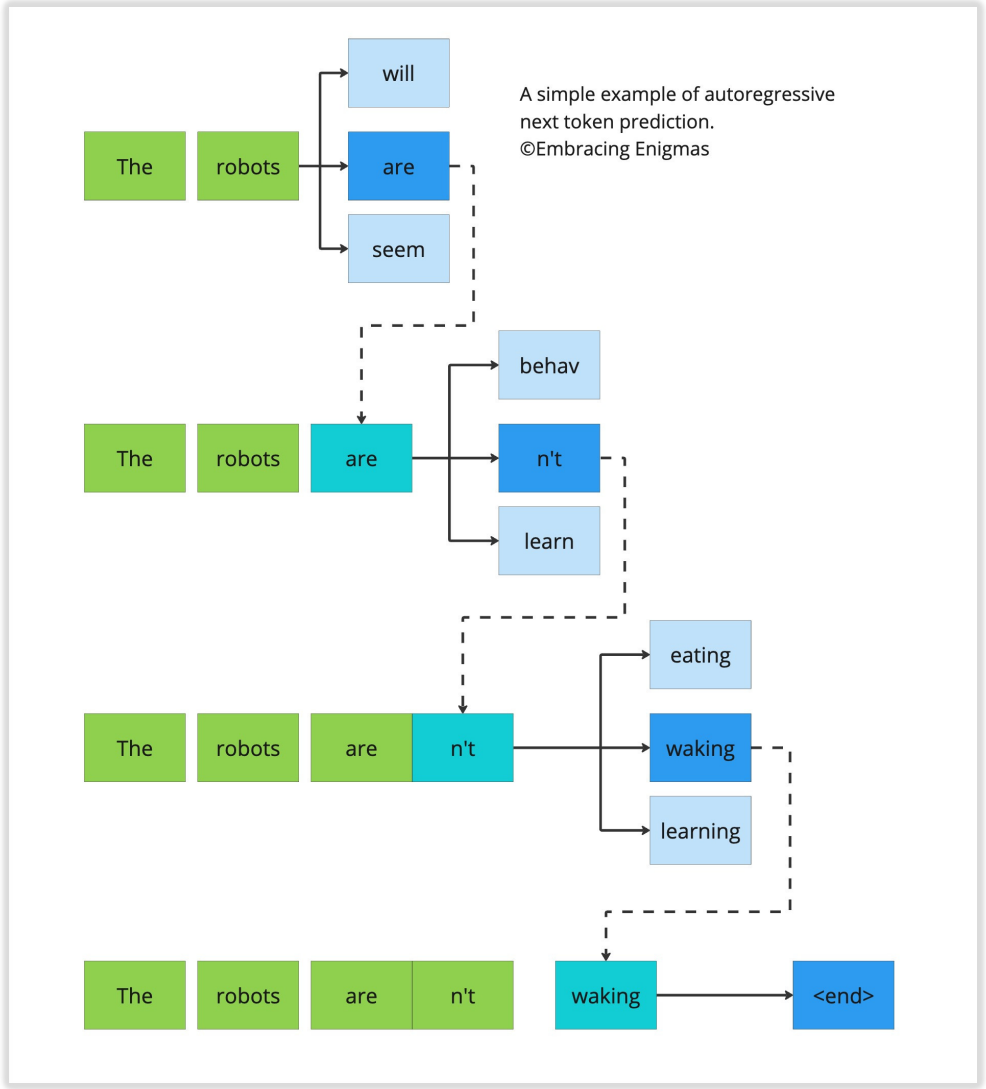
Element-wise (track) classification



Next token prediction

Next token prediction, i.e. GPT style training





Auto-regressively predict next token in sequence

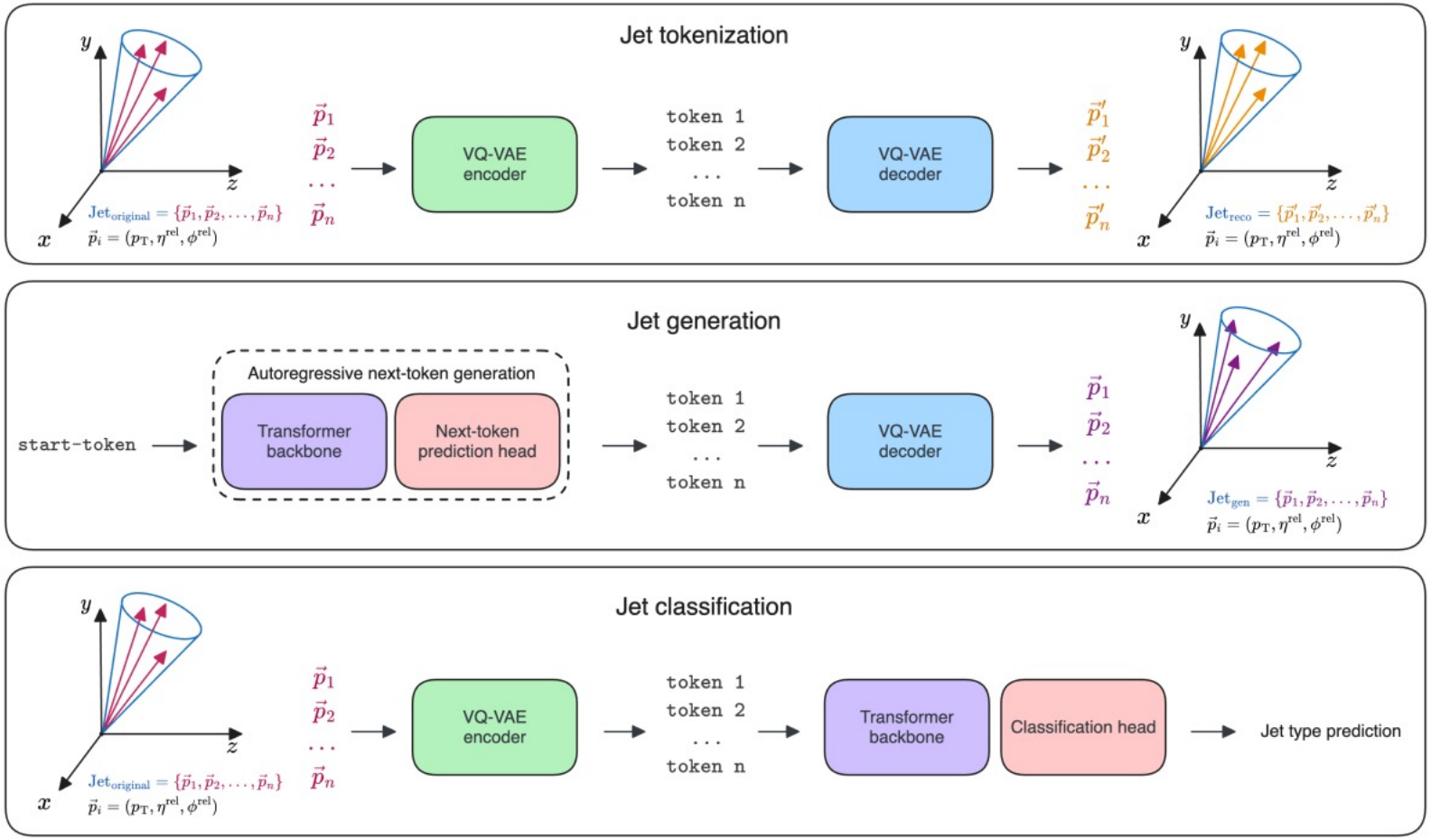
$$p(x_t|x_{1:t-1})$$

Train with log-likelihood:

$$\mathrm{L} = \sum_i \sum_t \log p(x_t^{(i)}|x_{1:t-1}^{(i)})$$

Next Particle Prediction: Omnijet-α

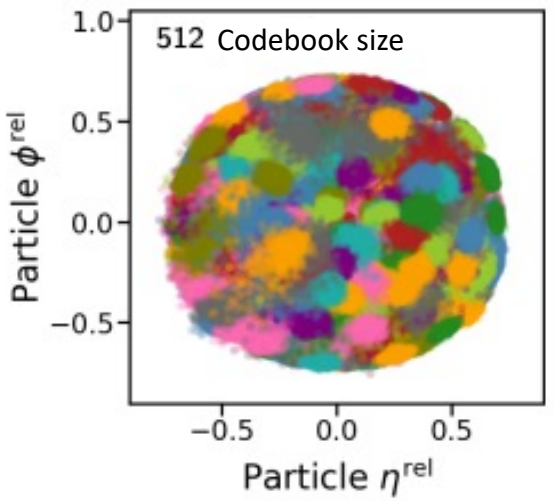
Next token prediction can also work for particles, if we
(a) tokenize particles, (b) give them an order, in this case by p_T

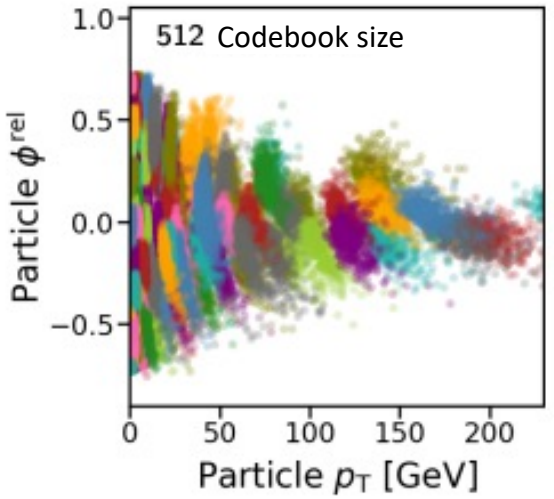


2403.05618

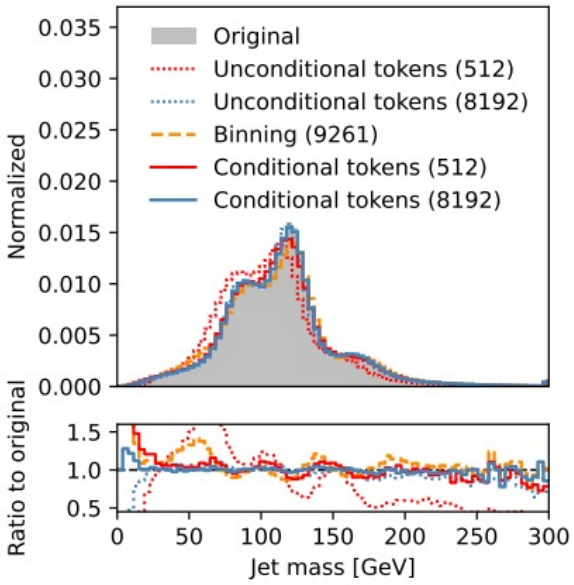
Tokenization Performance

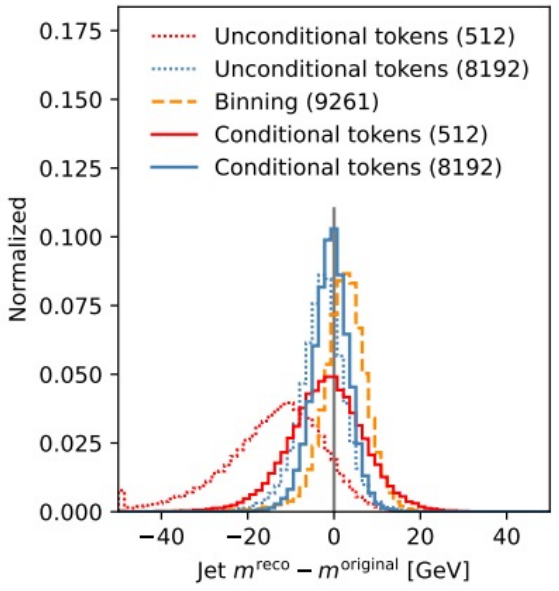
Token Visualization

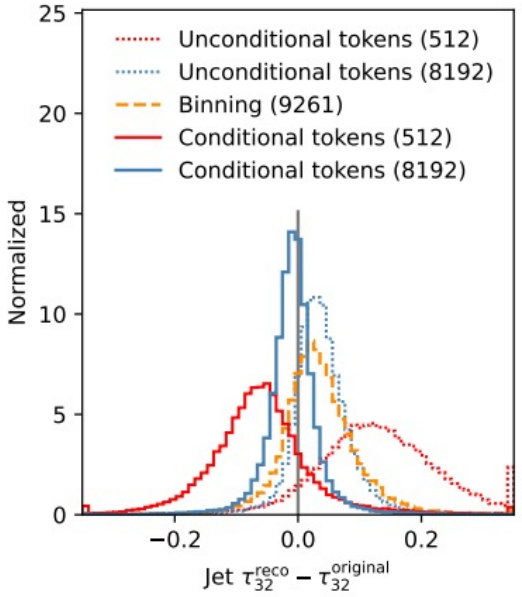




Jet properties from decoded particle tokens

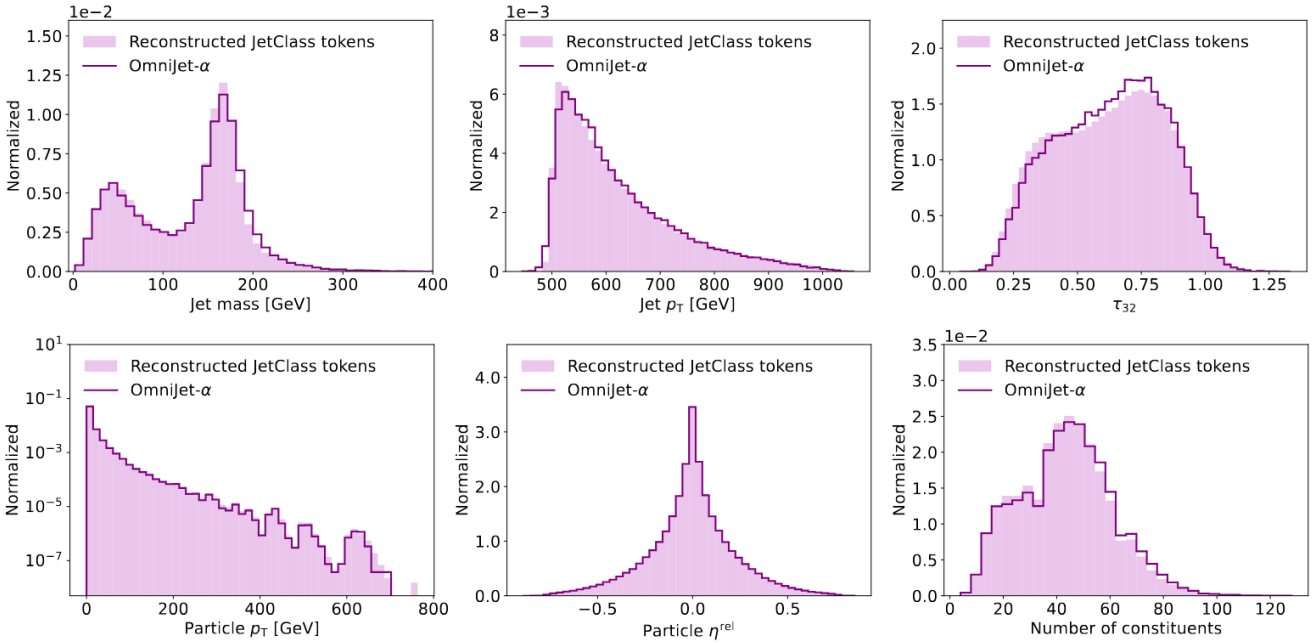




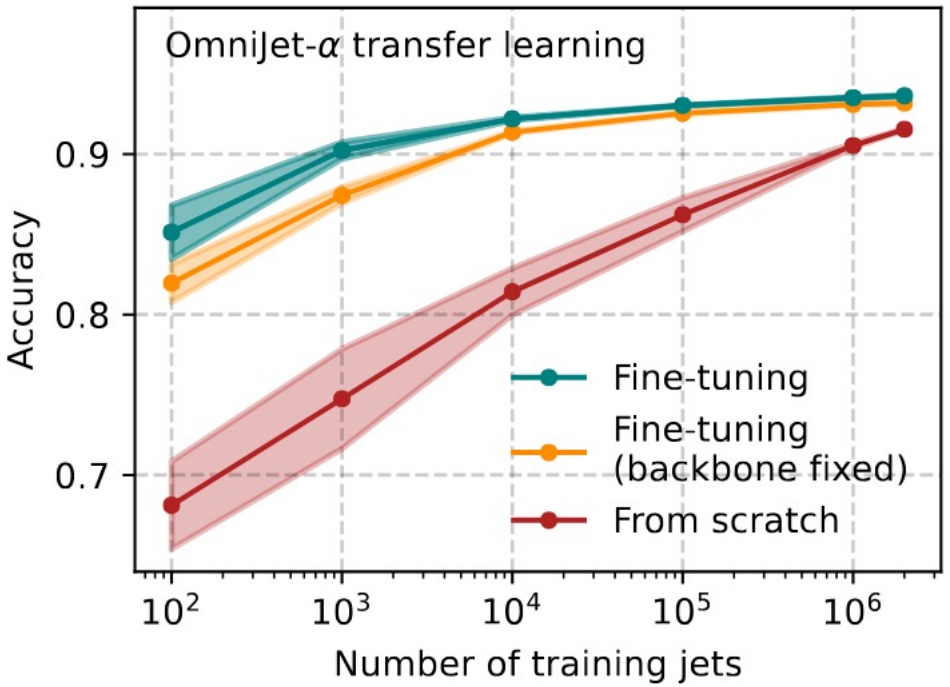


Omnijet-α Performance

Generative Modeling



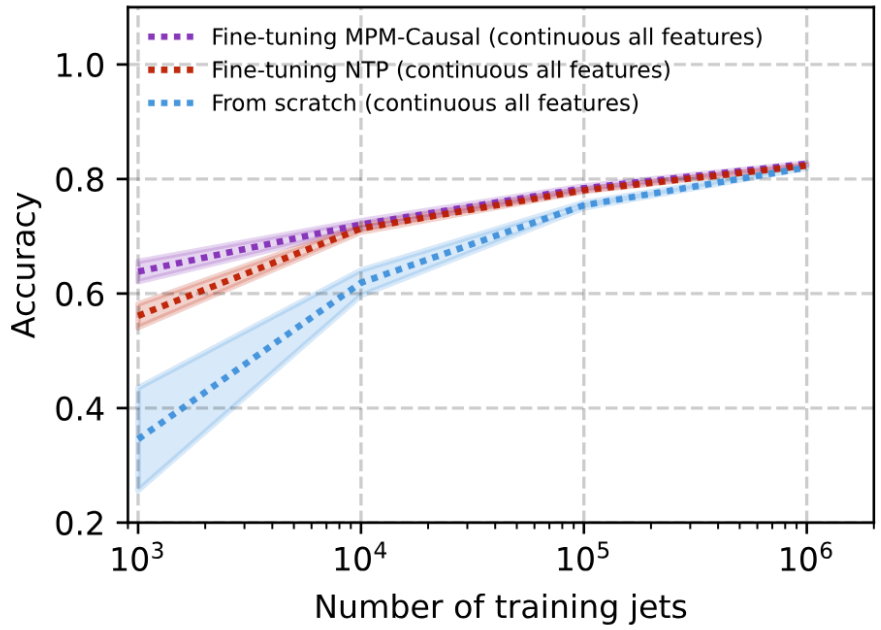
Fine-tuning for top vs QCD classification



Next vs. Masked Token Prediction

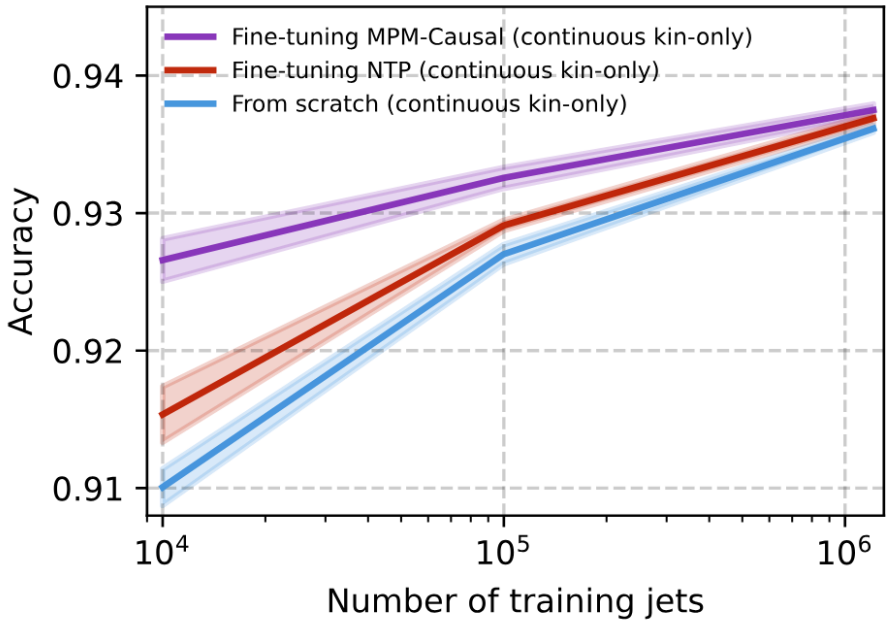
Comparison within the same dataset (JetClass)

Pre-training: jet generation (all 10 jet types)
Fine-tuning: jet classification (all 10 jet types)



Comparison when switching dataset

Pre-training: JetClass generation (all 10 jet types)
Fine-tuning: top tagging dataset [2] classification



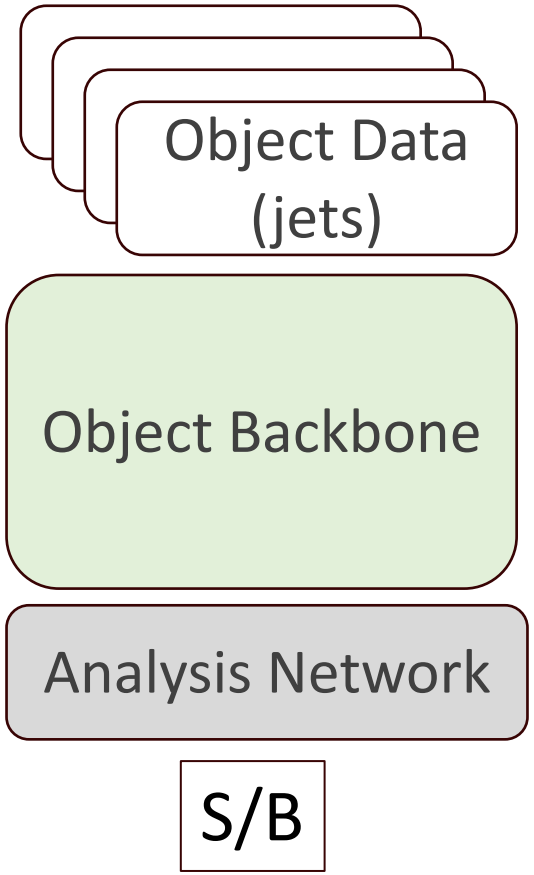
- MPM-Causal = modified version of Masked Particle Modeling (MPM) [1, 2]
- **MPM-Causal** leads to **more expressive backbone** compared to **next token prediction**

Slide credit: J. Birk

Does pre-training + fine-tuning help practically?

Is it only for SSL?

Does Pre-training help in Measurements?



Explore impact of pre-training and then fine-tuning a jet classifier as part of HH→4b analysis

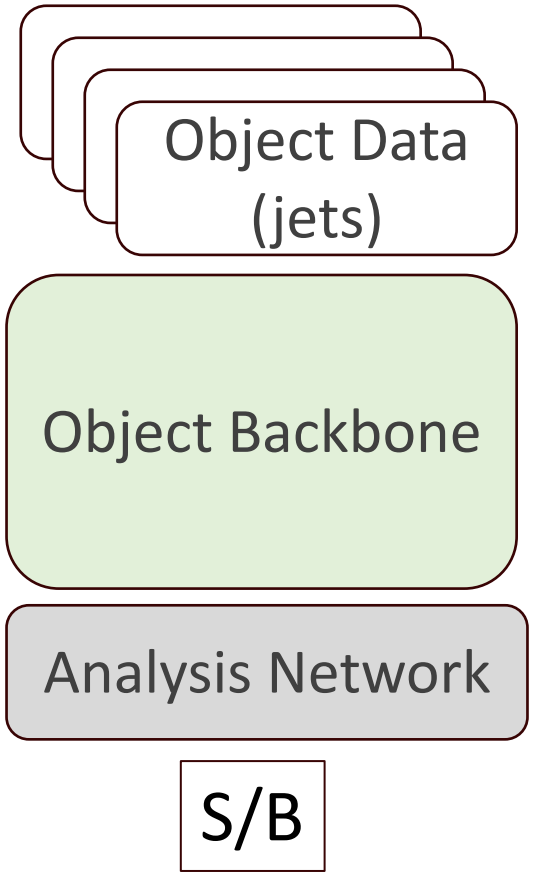
Pre-train “object backbone” = jet classifier

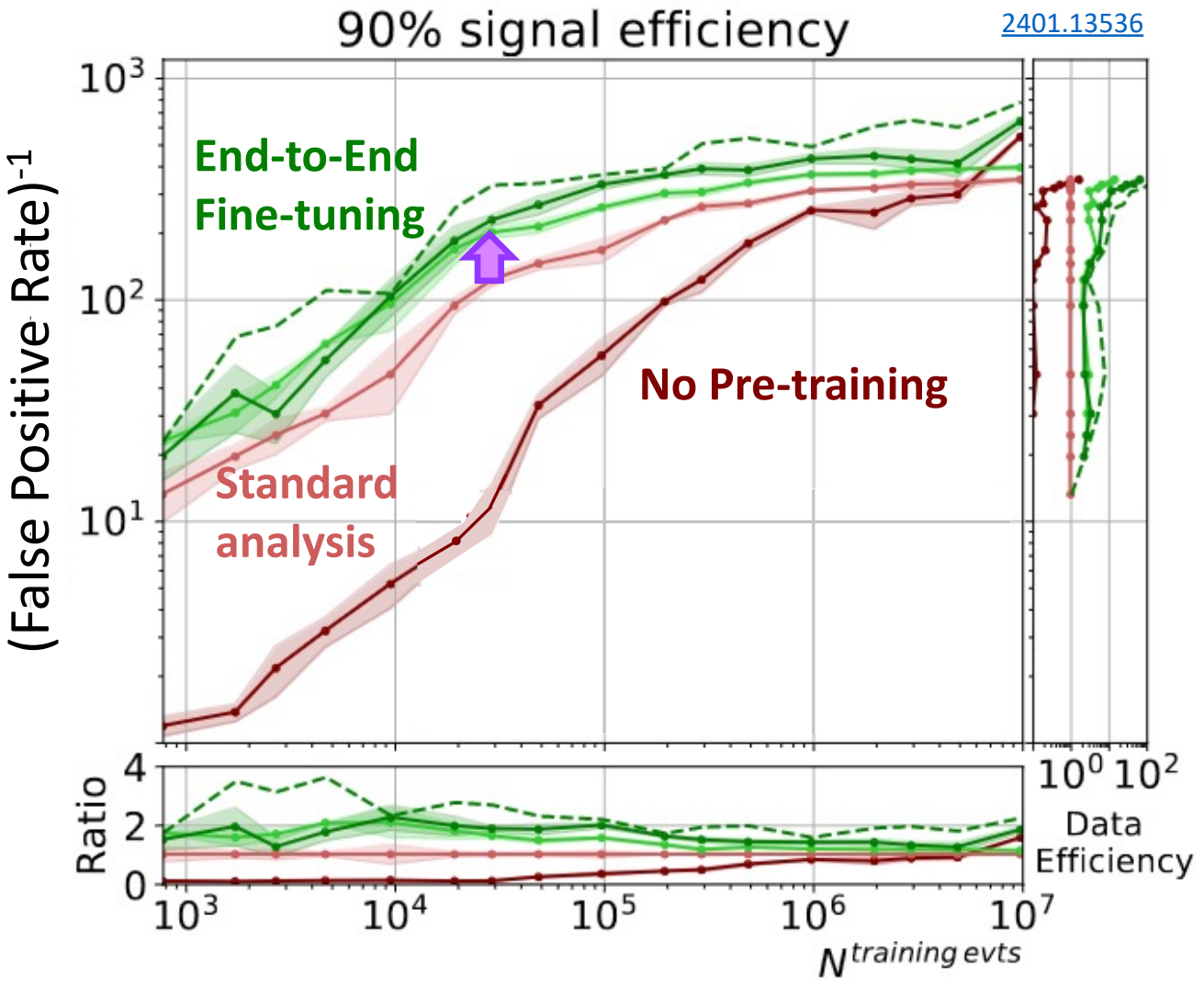
Want to train event classifier on objects

Options:

1. Fix object backbone, train analysis classifier (standard approach)
2. Fine-tune object backbone while also training analysis classifier
3. Retrain everything from scratch

Does Pre-training help in Measurements?





Up to:

2-4X better Signal / Background

10X more data efficient

Double Higgs HH→4b vs Background Events

So where are we with Foundation Models in HEP?

My take: We've seen an exploration of several self-supervised methods and the development of proto-foundation models in HEP

- Models mainly applied to particle in jets → need to move to events
- Not always clear what are the downstream tasks (esp. for jet models)
- Have not solved the systematic uncertainties / domain shift problem
  - Pre-training on data and fune-tuning on simulation with labels has not yet led to a more robust model
- Models fairly small so far, not yet benefitting from scaling
- As a community, learned a great deal about SSL

Multi-Modal Learning

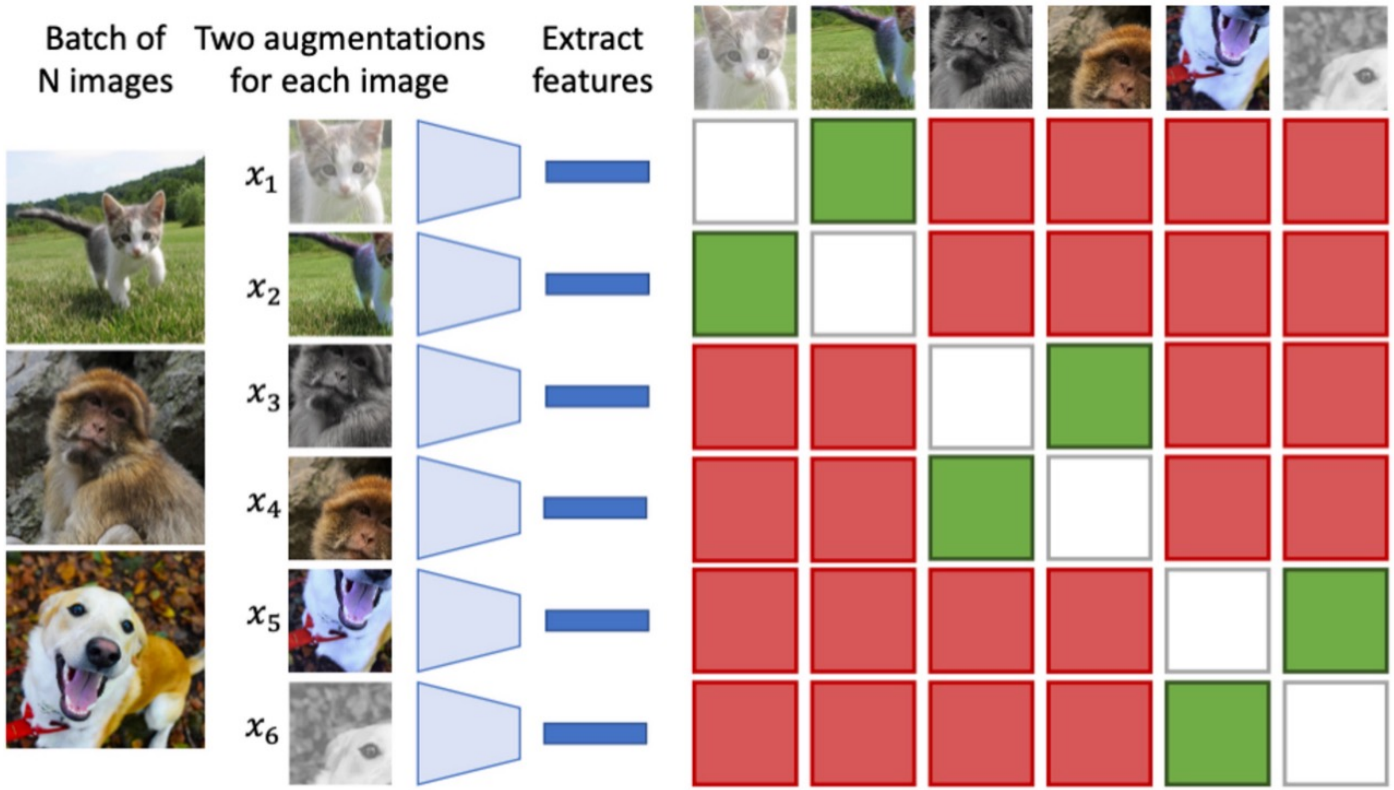
SimCLR → Multi-Modal Contrastive Learning

Remember SimCLR:

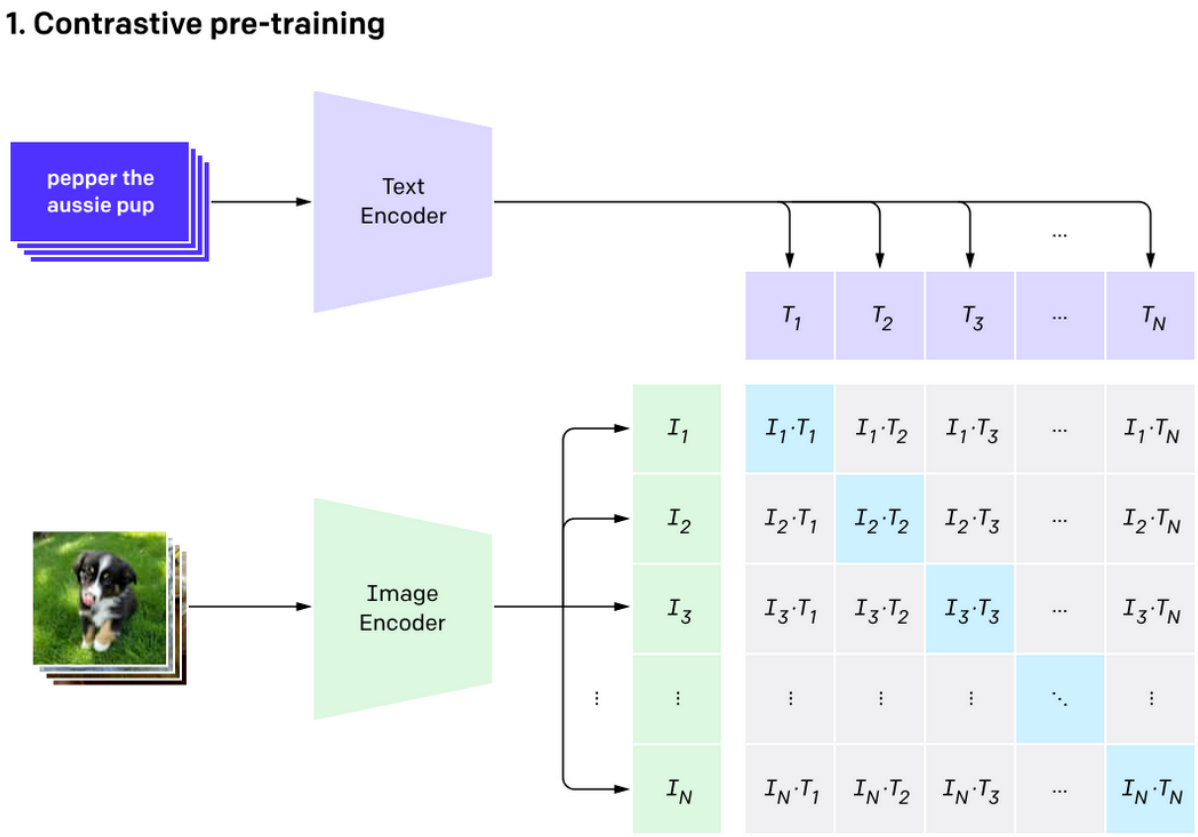
Compare representations of different augmentations of inputs using contrastive loss

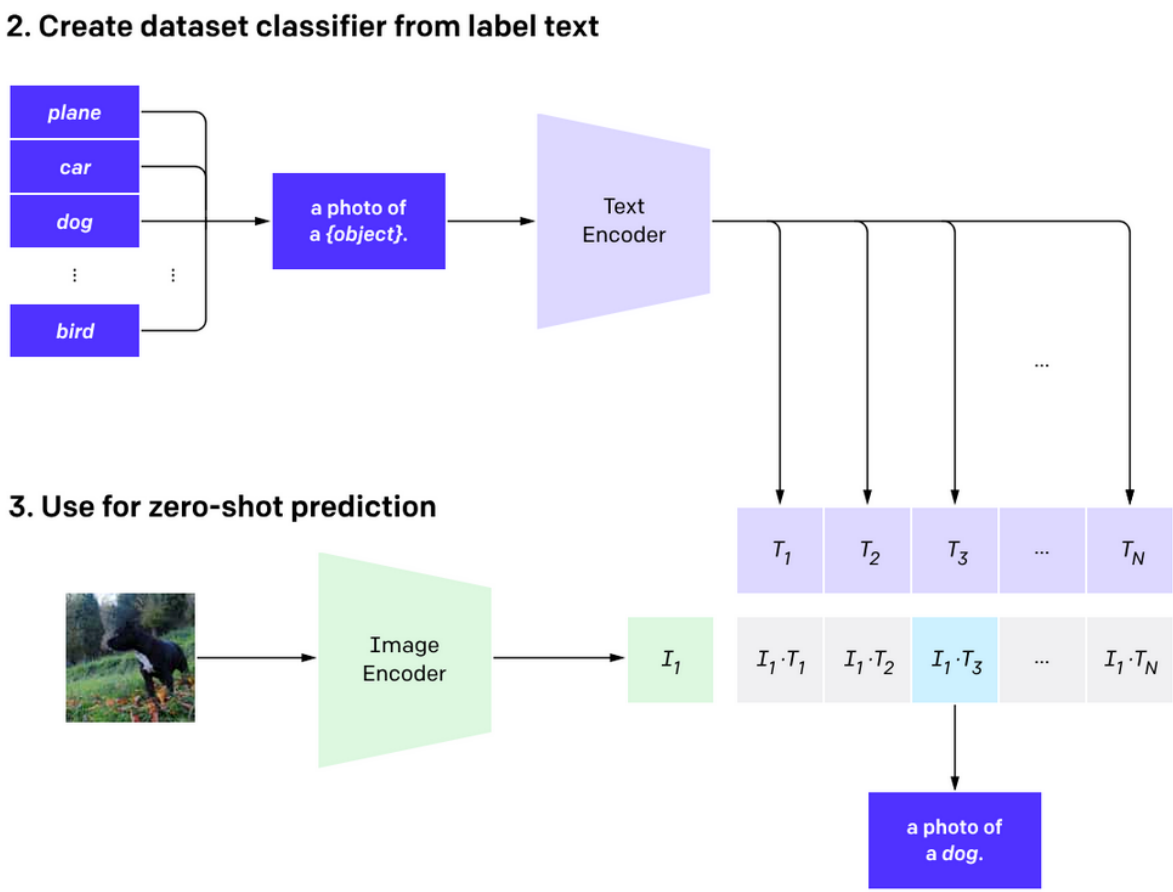
Bring same image together, push different images apart

Can we generalize beyond a single modality of data?



CLIP: Contrastive Language Image Pretraining

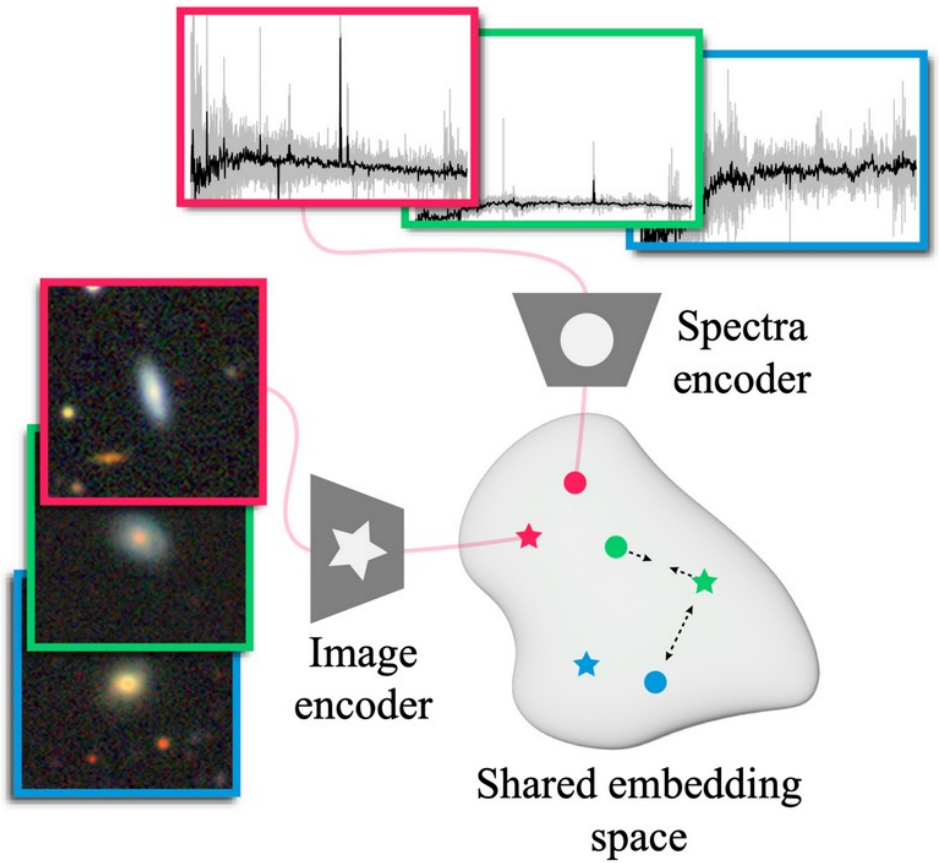




$$L_{\mathcal{I},\mathcal{M}} = -\log \frac{\exp(\mathbf{q}_i^\intercal \mathbf{k}_i/\tau)}{\exp(\mathbf{q}_i^\intercal \mathbf{k}_i/\tau) + \sum_{j \neq i} \exp(\mathbf{q}_i^\intercal \mathbf{k}_j/\tau)}$$

2103.00020

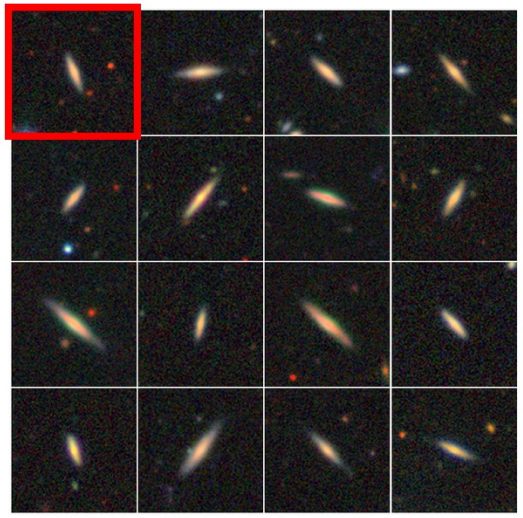
AstroCLIP



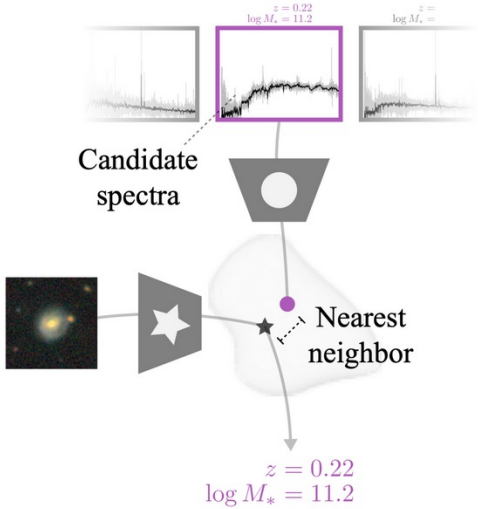
$$L_{\mathcal{I},\mathcal{M}} = -\log \frac{\exp(\mathbf{q}_i^\intercal \mathbf{k}_i/\tau)}{\exp(\mathbf{q}_i^\intercal \mathbf{k}_i/\tau) + \sum_{j\neq i} \exp(\mathbf{q}_i^\intercal \mathbf{k}_j/\tau)}$$

Spectra and multi-band **images** as two different views for the same underlying object.

DESI Legacy Surveys *(g,r,z)* images, and DESI EDR galaxy spectra.



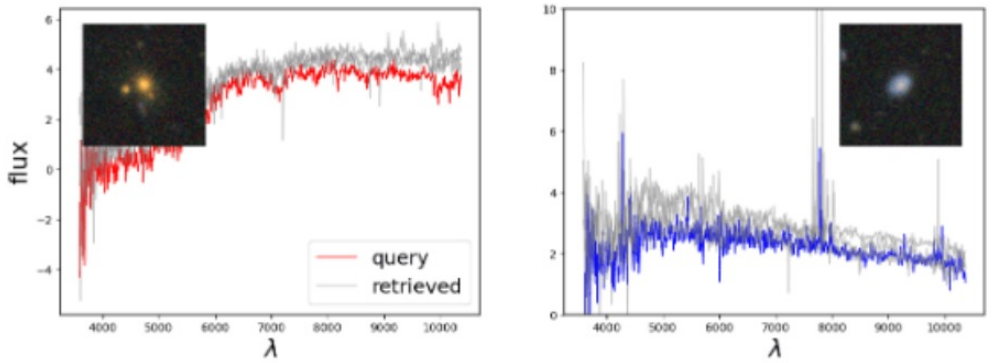
Cosine similarity search



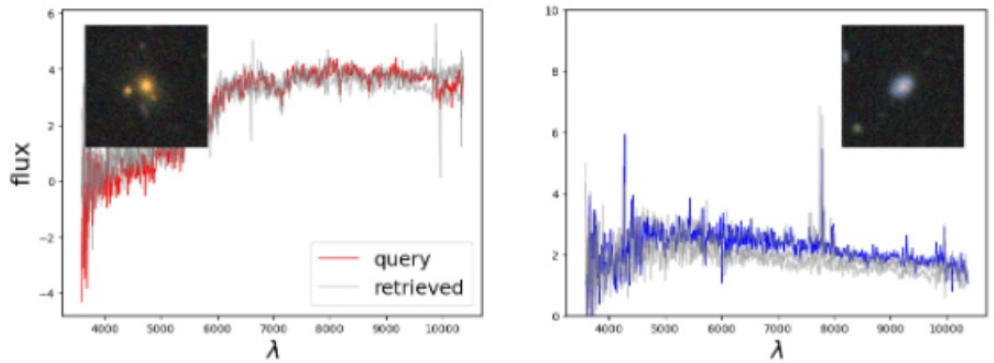
Zero-shot prediction

2310.03024

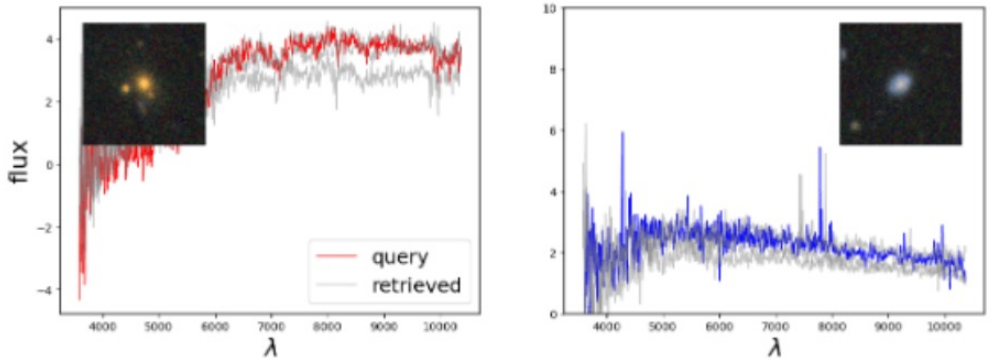
AstroCLIP: Cross-Modal Similarity Search



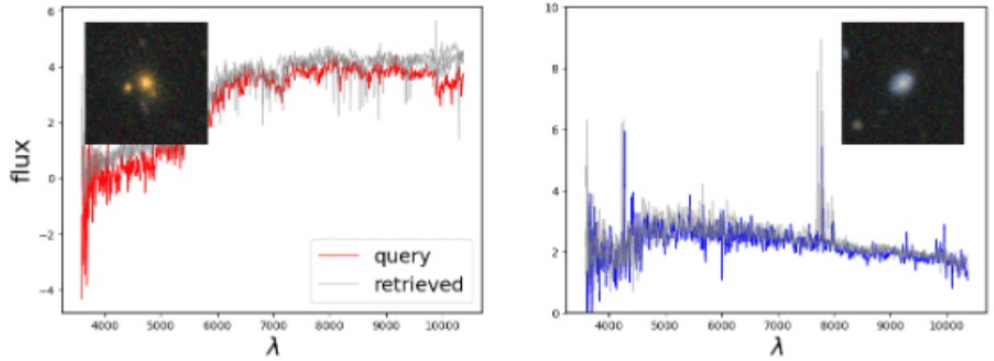
(f) $S_C(\mathbf{z}_q^{\text{im}}, \mathbf{z}^{\text{im}})$



(g) $S_C(\mathbf{z}_q^{\text{sp}}, \mathbf{z}^{\text{sp}})$



(h) $S_C(\mathbf{z}_q^{\text{sp}}, \mathbf{z}^{\text{im}})$

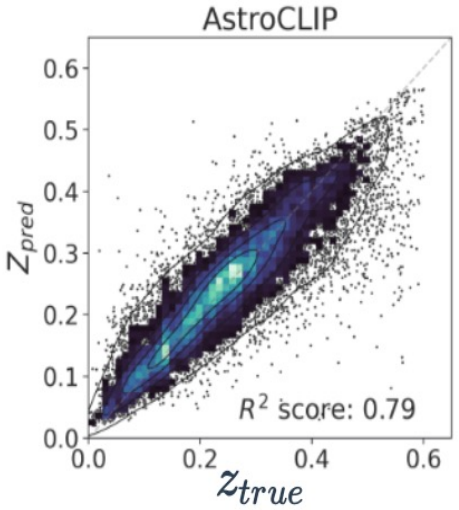


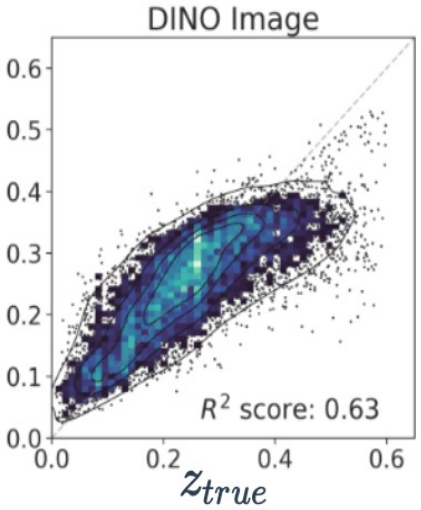
(i) $S_C(\mathbf{z}_q^{\text{im}}, \mathbf{z}^{\text{sp}})$

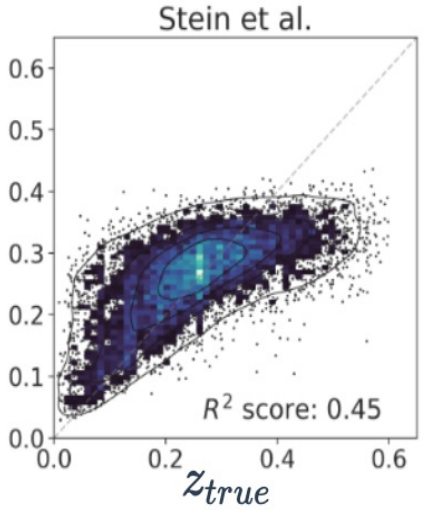
$$S_C(\mathbf{z}_i^{sp}, \mathbf{z}_j^{im}) = (\mathbf{z}_i^{sp} \cdot \mathbf{z}_j^{im}) / \| \mathbf{z}_i^{sp} \| \| \mathbf{z}_j^{im} \|$$

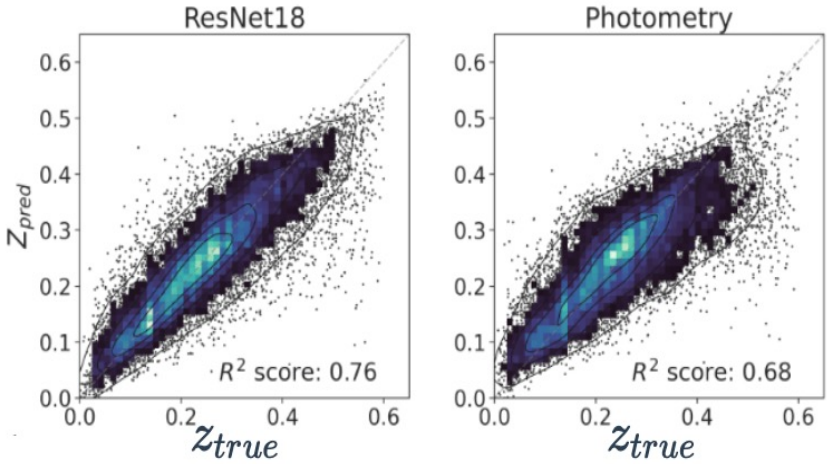
2310.03024

AstroCLIP: Redshift Estimation from Images







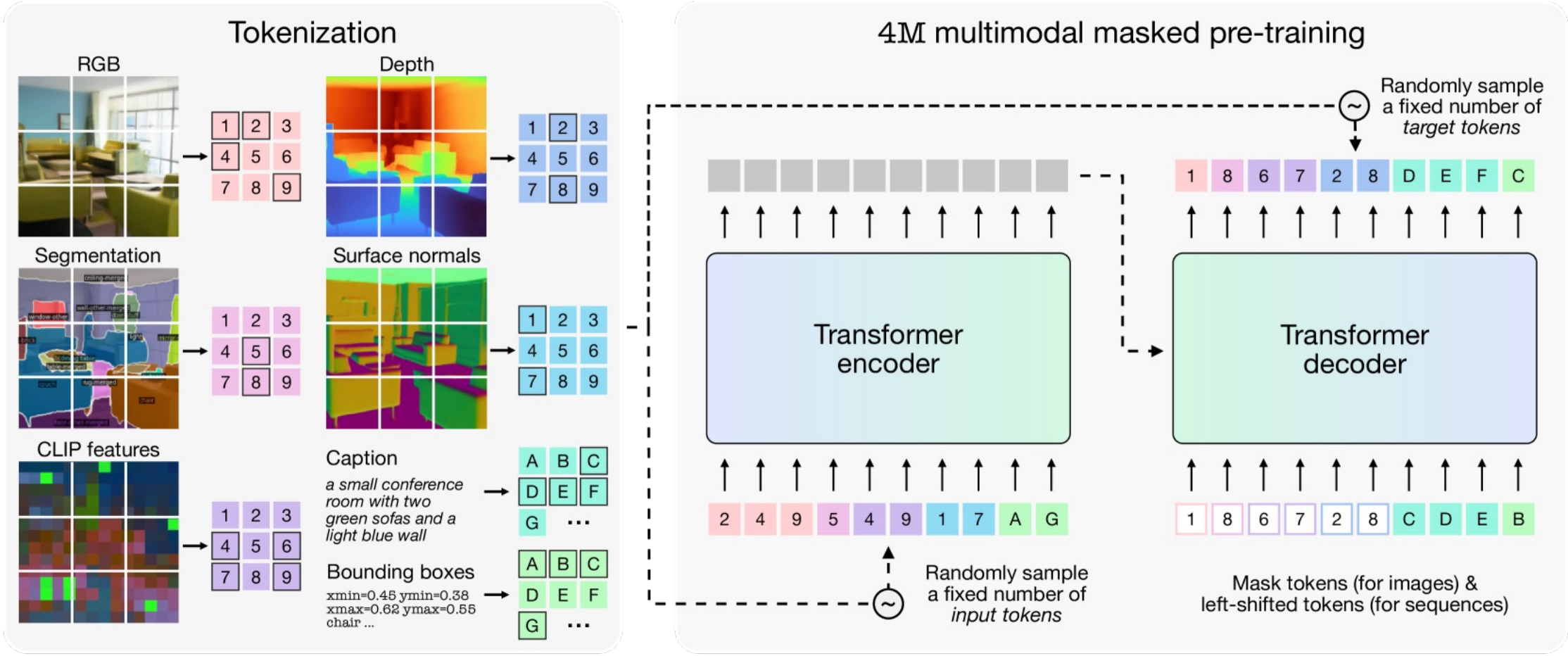


Supervised baseline

- Zero-shot prediction
  - k-NN regression

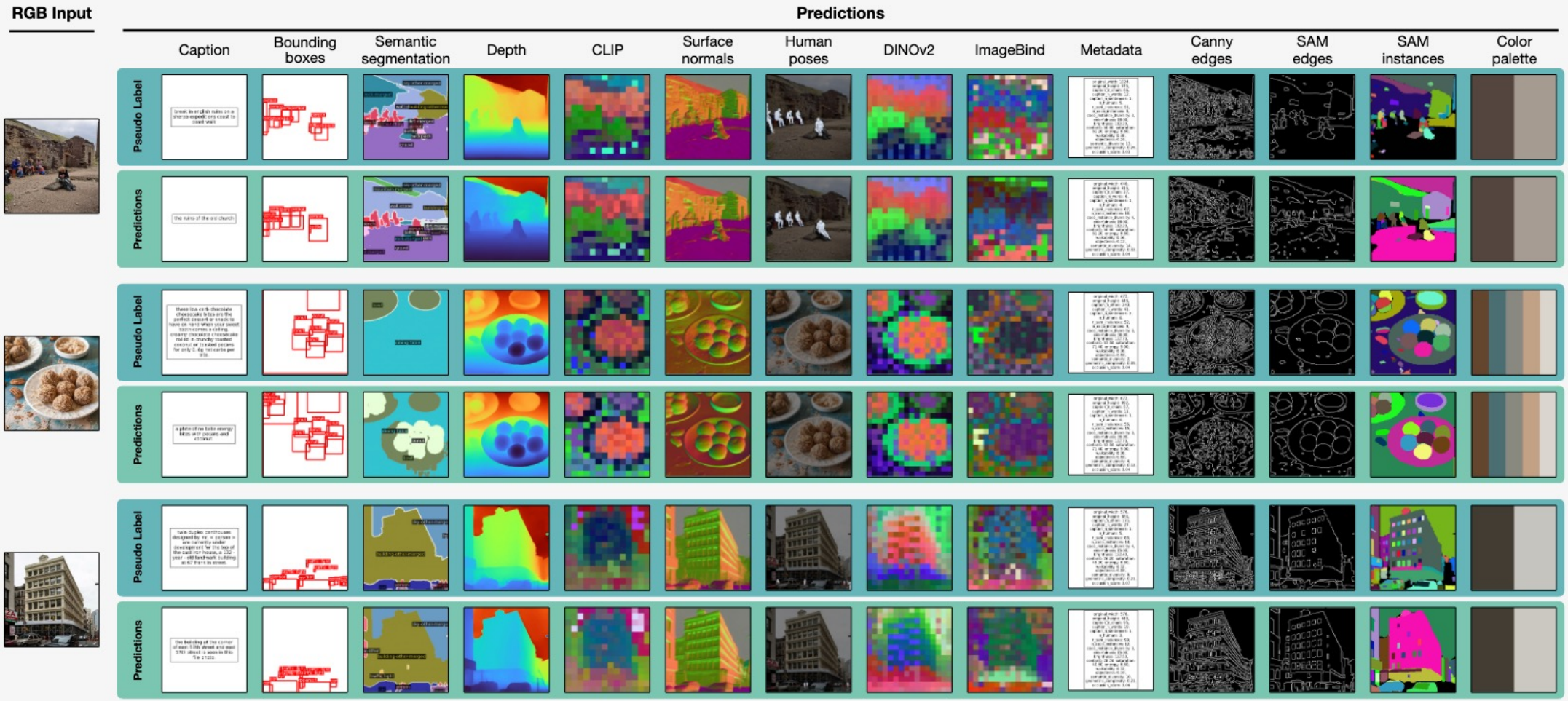
2310.03024

4M: Massively Multi-Modal Masked Modeling



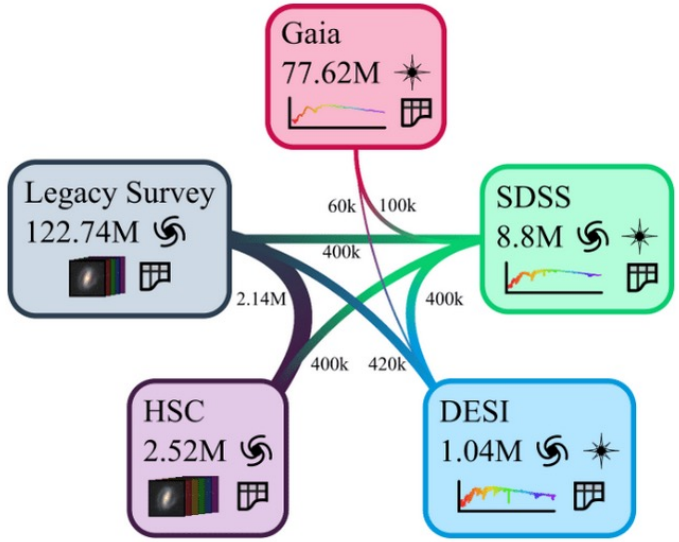
2312.06647
2406.09406

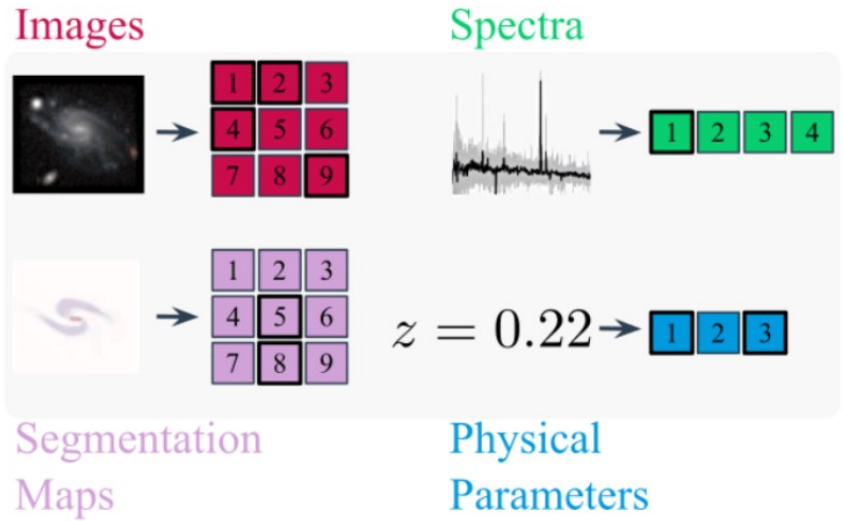
4M: Massively Multi-Modal Masked Modeling

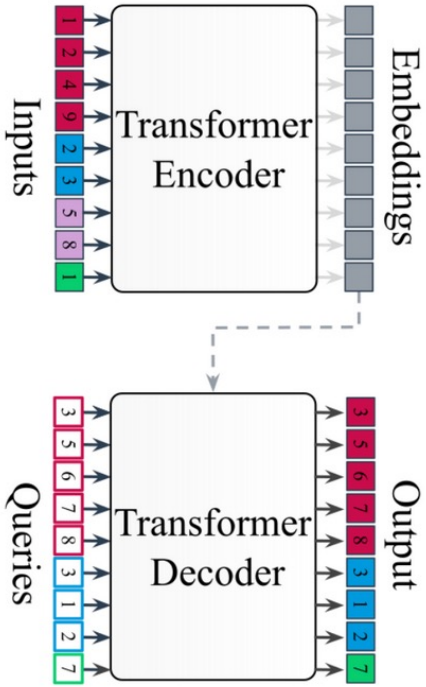


2312.06647
2406.09406

Any-to-Any Modeling with 4M-style Generative Masked Modeling







Training is done by **pairing observations of the same objects from different instruments**.

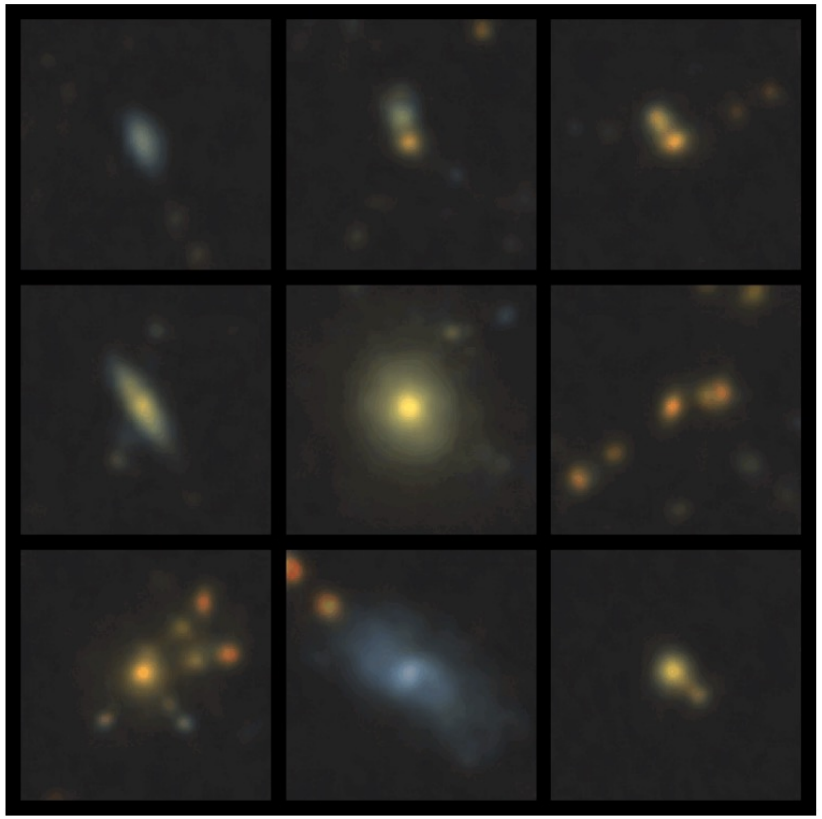
Each input token is tagged with a ***modality embedding*** that specifies provenance metadata.

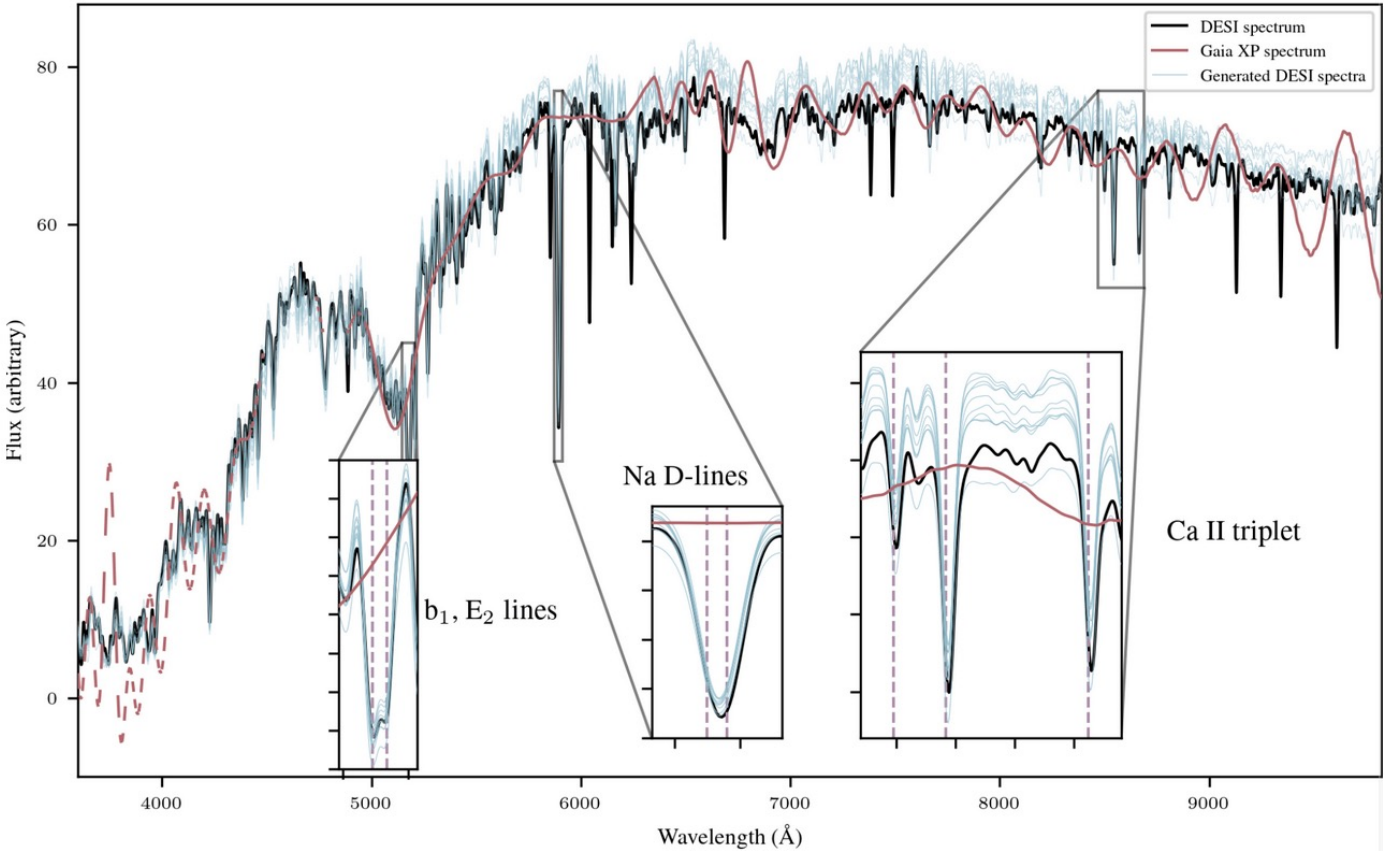
Model is trained by cross-modal generative masked modeling (i.e. 4M-style)

- Learns the ***joint and all conditional distributions of provided modalities***: $\forall m, n \quad p(x_m|x_n)$

Slide credit: F. Lanusse

Capabilities



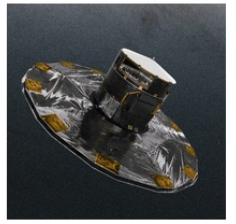


Survey translation

$p(\boldsymbol{x}_{HSC}|\boldsymbol{x}_{DES})$

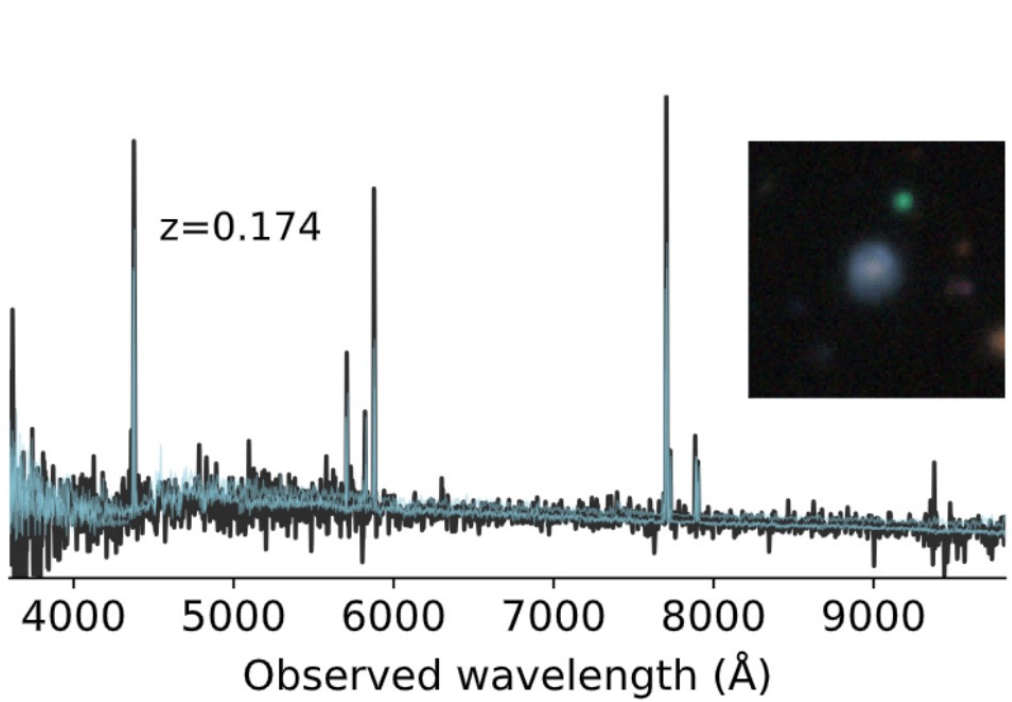
Spectrum super-resolution

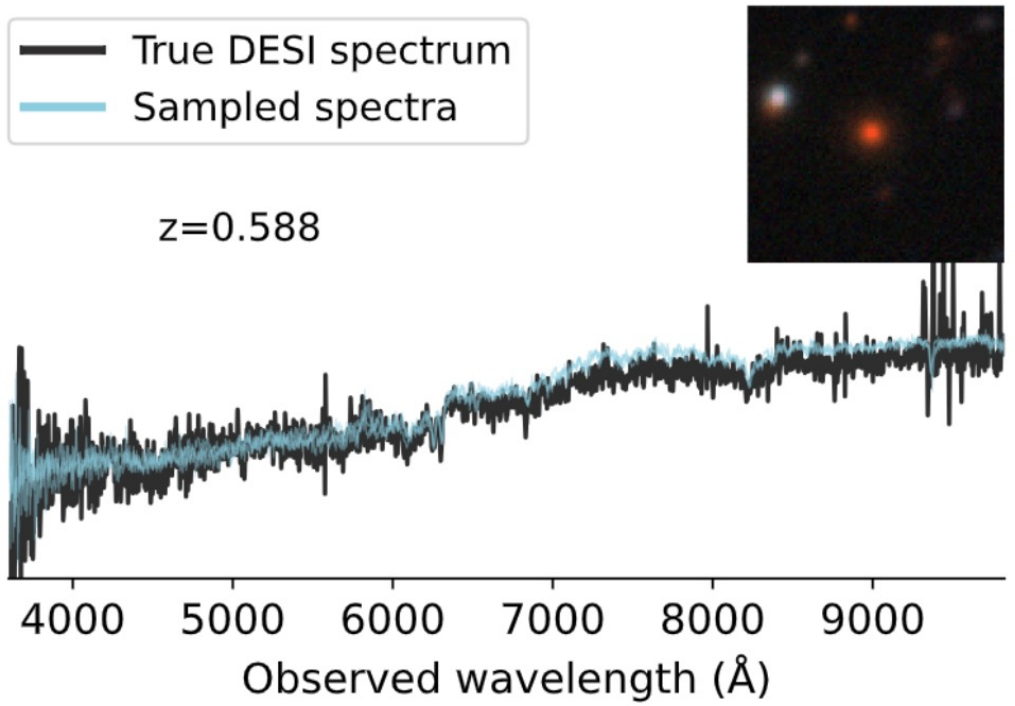
$p(\boldsymbol{x}_{DESI}|\boldsymbol{x}_{GAIA})$

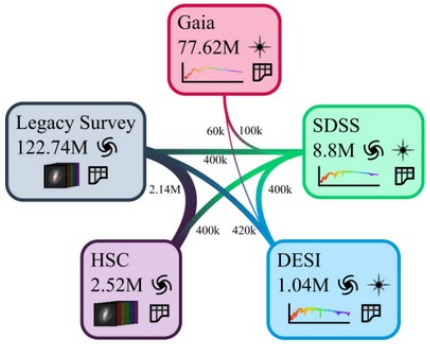


Slide credit: F. Lanusse

Capabilities







$$p(\boldsymbol{x}_{DESI}|\boldsymbol{x}_{HSC})$$

- Direct association between DESI and HSC was excluded during pretraining
  => This task is out of distribution!

Slide credit: F. Lanusse

What about Language Models, do they have a role to play?

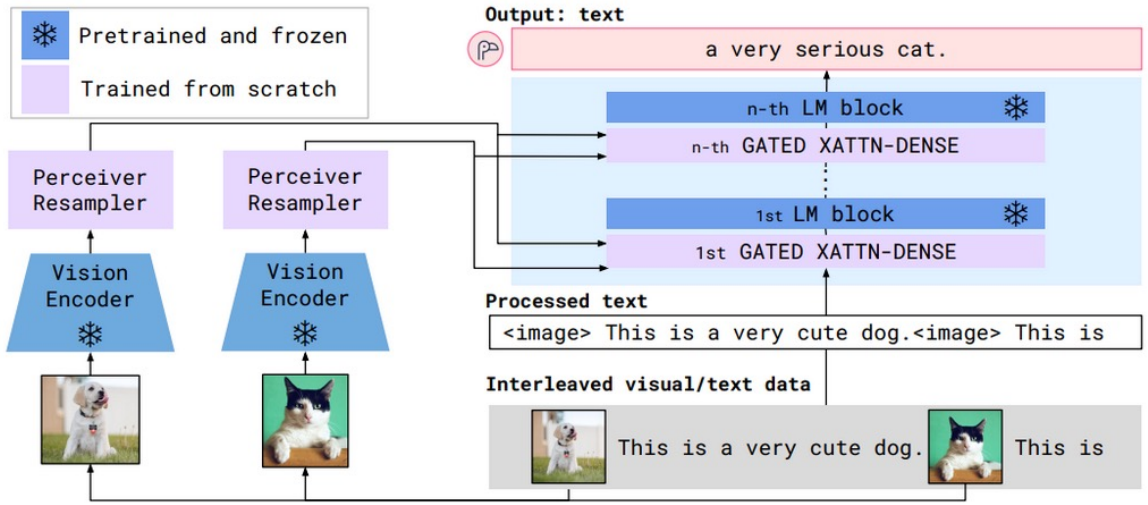
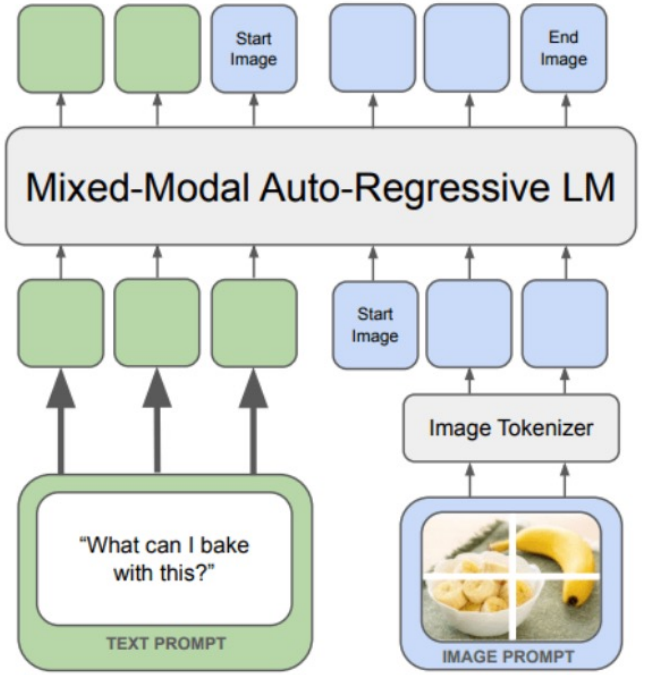


Figure 3: **Flamingo architecture overview.** Flamingo is a family of visual language models (VLMs) that take as input visual data interleaved with text and produce free-form text as output.

Flamingo: a Visual Language Model for Few-Shot Learning (2204.14198)



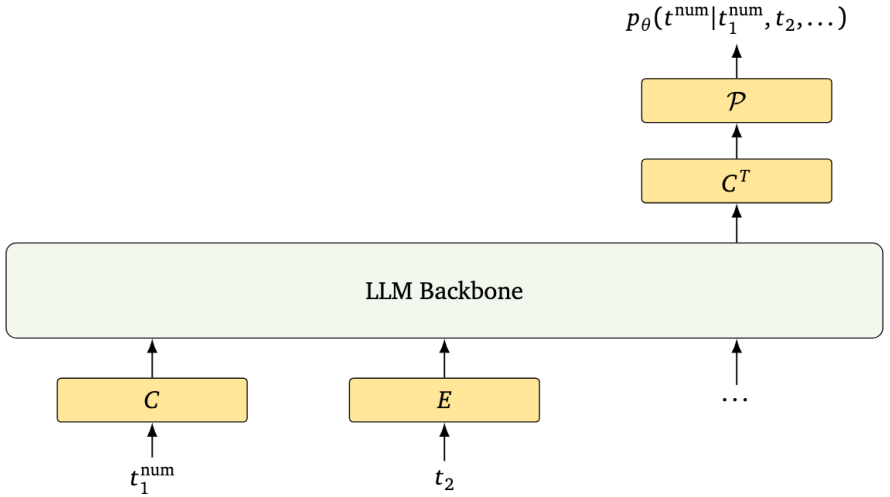
Chameleon: Mixed-Modal Early-Fusion Foundation Models (2405.09818)

Language-Vision models have been a big success…

Can language models interact with HEP data in some way?

First steps of integrating science data with LLMs

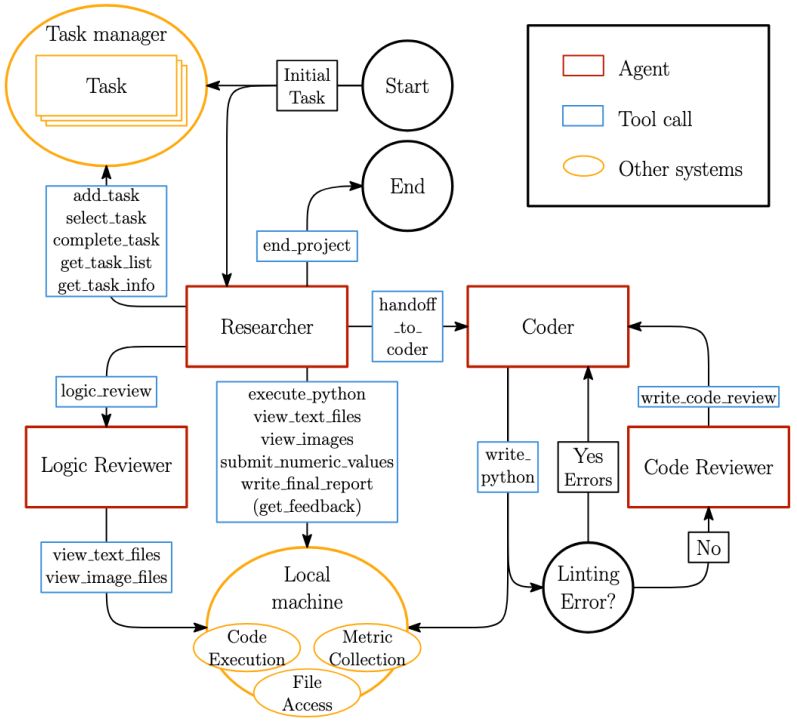
Using LLM backbone with language and numerical data for SKA physics




```
<|im_start|>system
<|im_end|>
<|im_start|>user
t_1^BT ... t_47^BT<|im_end|>
<|im_start|>assistant
<|ska-param|><|im_end|>
```

2506.14757

Using LLM agents to do LHC data analysis



2509.08535

Summary

Foundation models have emerged as a modern paradigm in AI, especially in language and vision modeling

Built on large scale representation learning with self-supervision

- Learn the useful and transferrable features from the data
- Choose a good pre-text task that will enable broad feature learning

Can adapt / fine-tune foundation models many downstream tasks

Building expertise in HEP for foundation models, adapting them to our data and exploring multi-modal capabilities

Will these efforts usher in a new way to do HEP science?