A Novel Approach to The Training Foundation Models for Jet-Related Tasks Without Vector Quantization

Masahiro Morinaga

The University of Tokyo
(ICEPP ,Beyond AI)

Scaling Law for Transformer

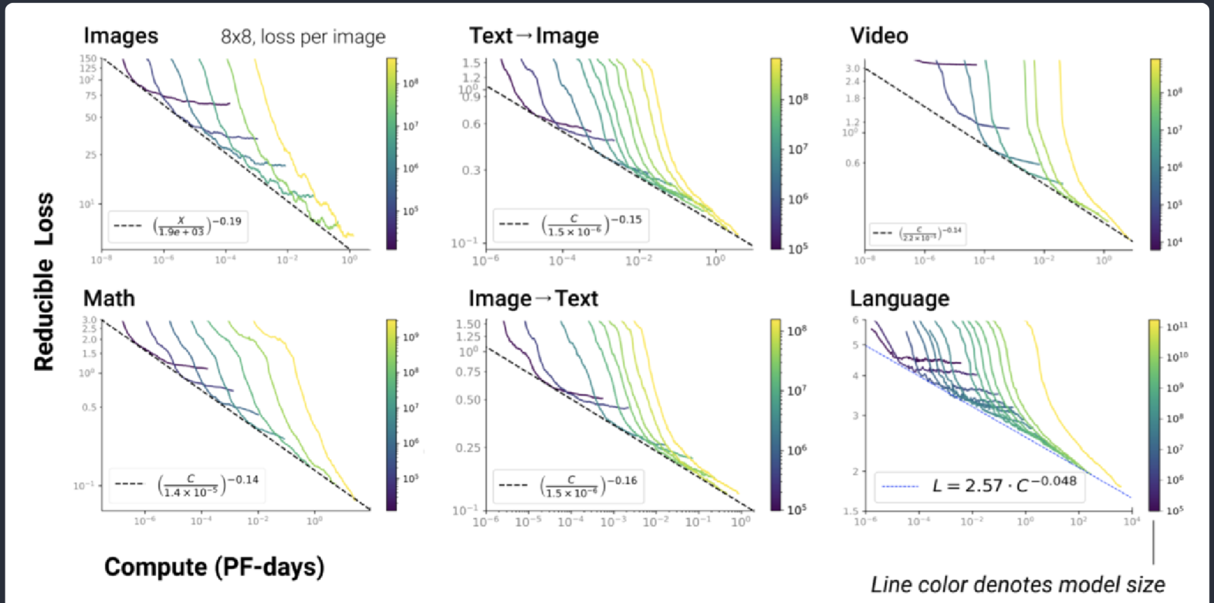
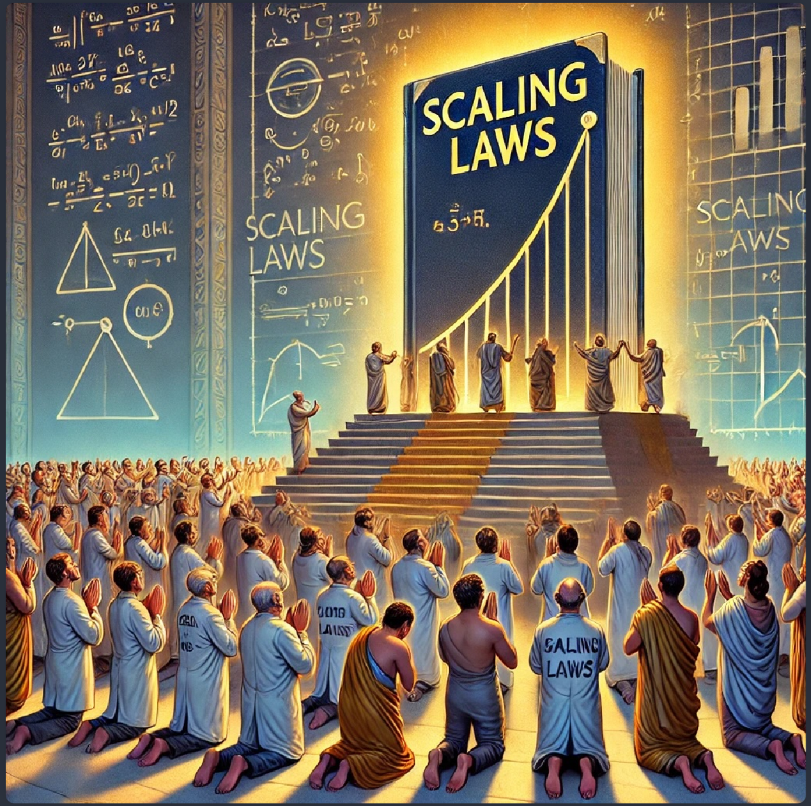


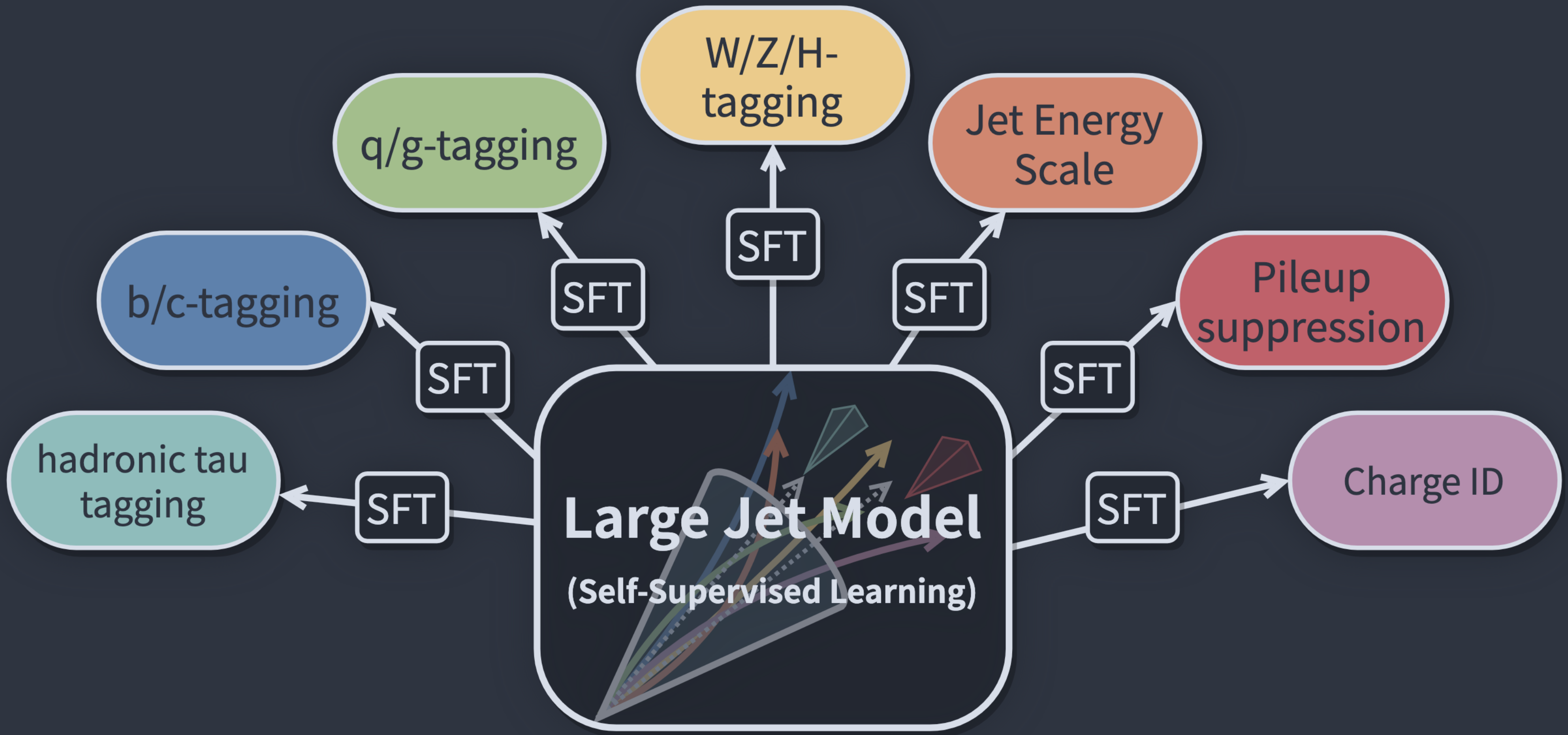
Figure 1 **Smooth scaling of *reducible* loss across domains—** We show power-law scaling laws for the *reducible* loss $L - L_\infty$ as a function of compute, where the irreducible loss L_∞ is a fitted domain-dependent constant. Under plausible assumptions concerning the infinite data and compute limits, the irreducible loss estimates the entropy of the underlying data distribution, while the reducible loss approximates the KL divergence between the data and model distributions. In the case of language we use results from [BMR+20], and only show the full loss L.



Scaling Law : Current AI's principle

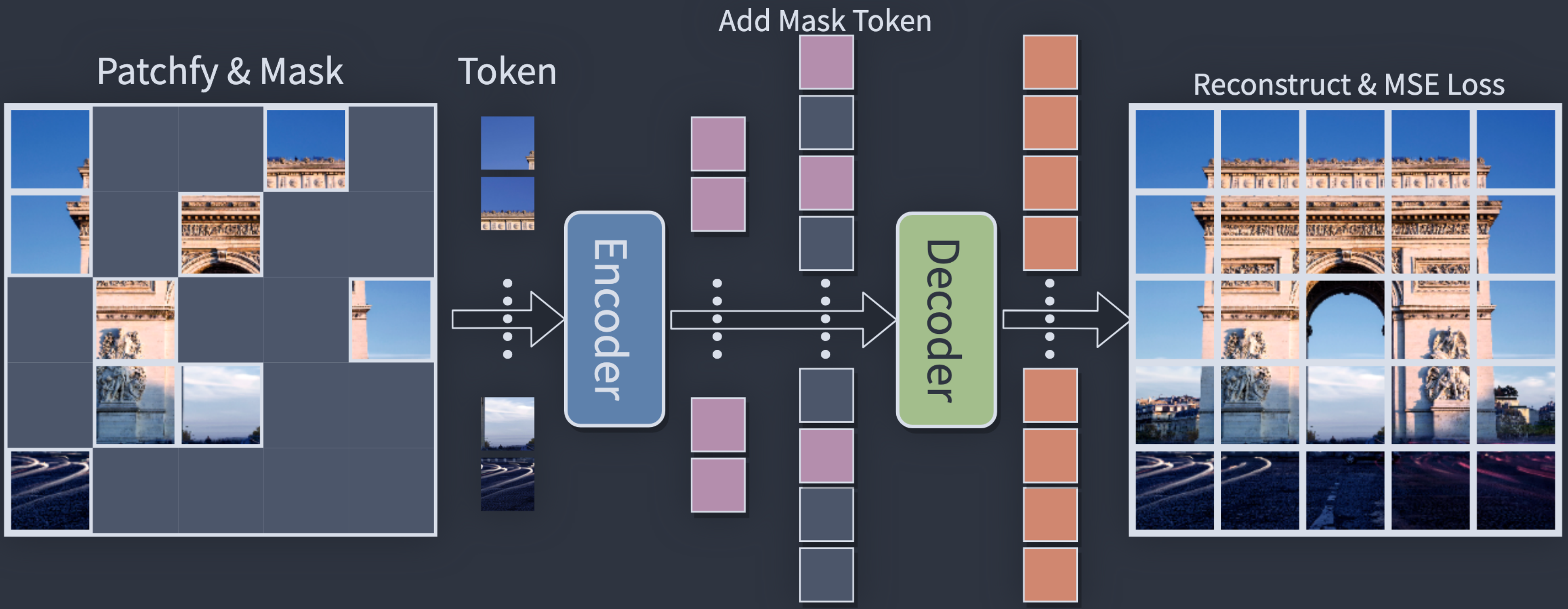
- The NN loss will continue to decrease as the number of model parameters, training data, and training time
- Not only languege, but also other fields show this, so then particle pehysics?
- Larger model has better performance in jets physics?

(Realistic) Foundation Model for Jet Related Task



- At the LHC, there are numerous tasks related to jets, but each uses its own models and data
- Larger models require more data, training itself become difficult.
- Pre-training that can generalize to various tasks is important.
- Then, how can we train such model? ⟶ ***Main topic of this talk!***

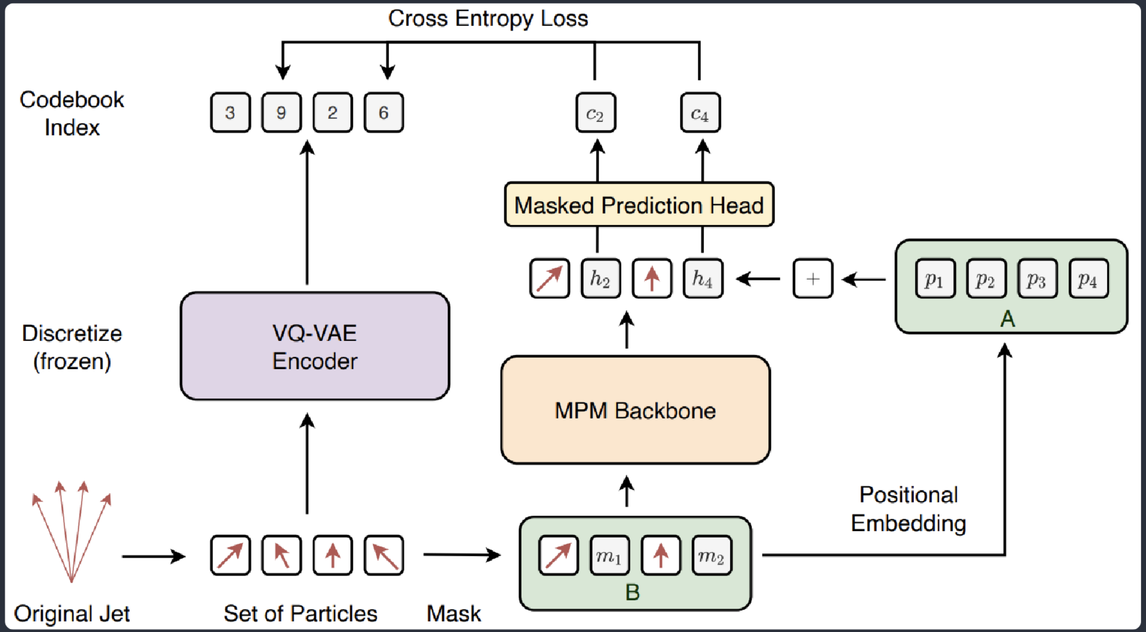
Self-Supervised Learning: MAE



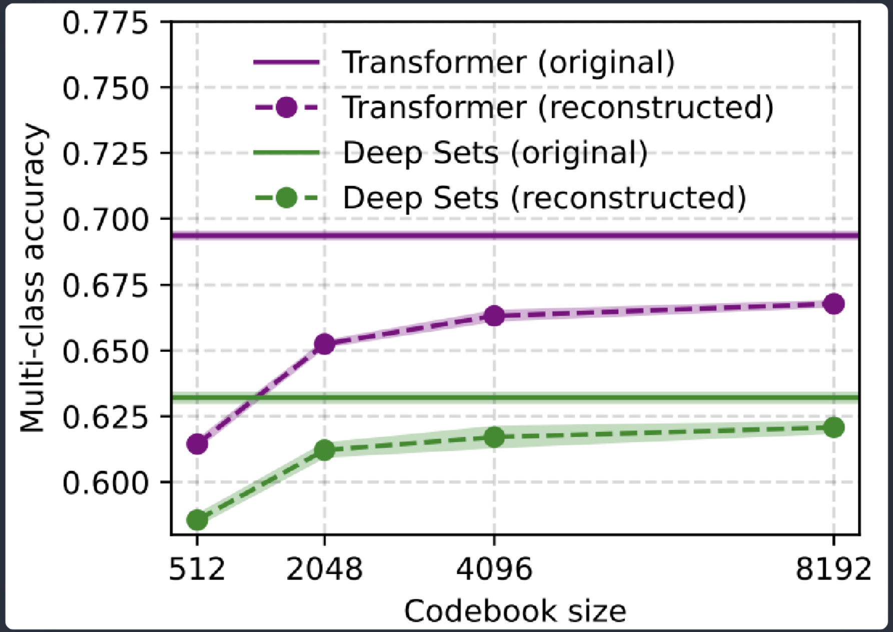
Masked Autoencoder: Self-Supervised Learing for Computer Vision

- Learn how to encode tokens from subset of patches of image
- Reconstruc using MSE loss directly from latent tokens
- **Good**: self-supervised training, reduce computatinal cost at encoder
- **Bad** : Feature collaspe happened

Masked Particle Modeling / Next Token Prediction



Schematic view of MPM

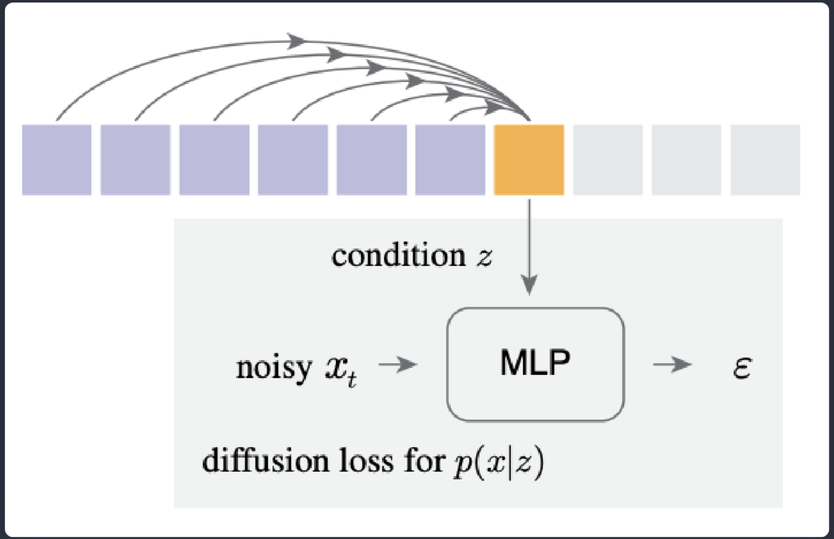


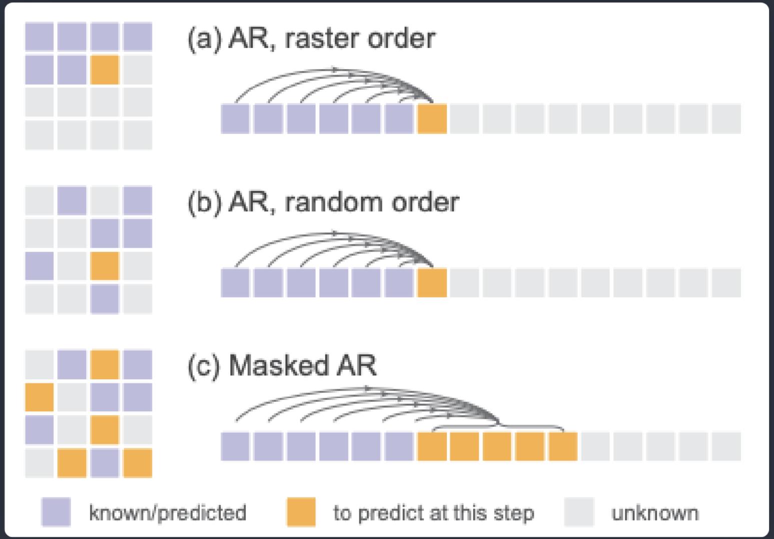
Results of OmniJet-α

Masked Particle Modeling, OmniJet-α : BEiT, Decoder

- Masking and replacing jet constitunes, predict masked tokens as discrete index from VQVAE
- Clear improvements at downstream tasks if fewer training samples
- Unlike BEiT or PointBERT, not clear at full scale dataset...
- A lot of SSLs with VQ are proposed so far...

Predict Tokens Using Diffusion Process

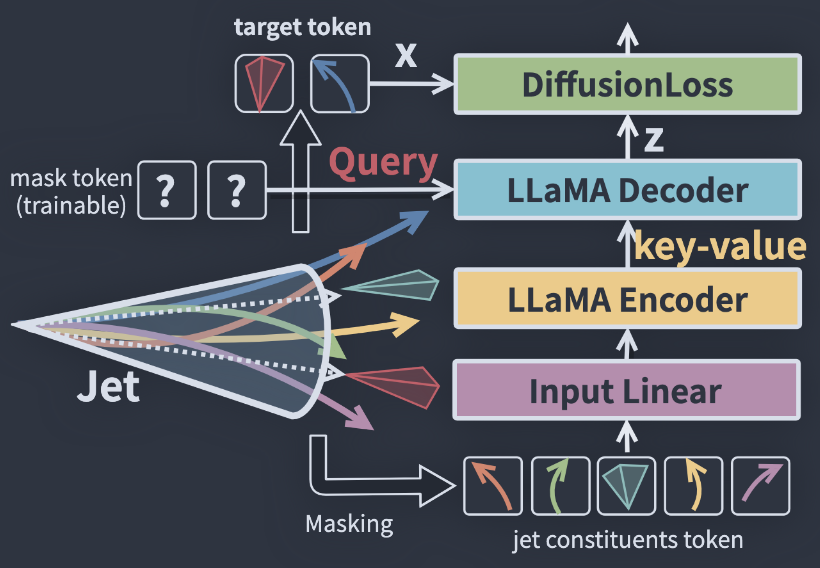




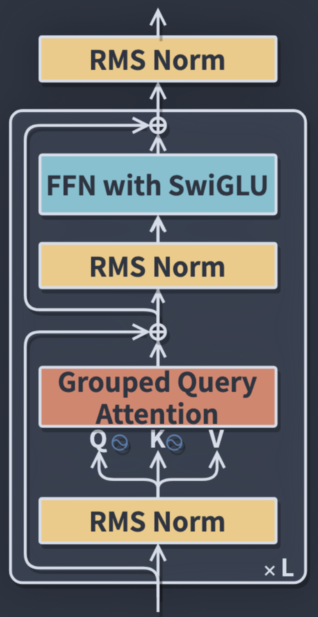
MAR: Autoregressive Image Generation without Vector Quantization

- Predict not tokens but *noise added from duffision process*, conditioned by decoder outputs
- Predict noise distribution for each token $\longrightarrow$ could avoid feature collaspe $\longrightarrow$ No need to train VQVAE
- Original method is for autoregressive model, but it can be suitable for our interest.
  - We could use this for encoder model, but can use it for generative model

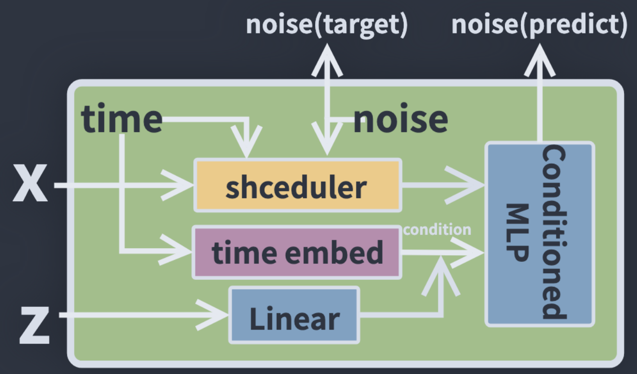
Pretraining Model for This Study



SSL Model



LLaMA



DiffusionLoss Layer

- **SSL Model** : Using LLaMA transformer as encoder/decoder, encoder is for downstream tasks.
  - LLaMA : Grouped Query Attention, FFN with SwiGLU, RoPE
  - CrossMAE : Efficient MAE prediction
  - DiffusionLoss : To avoid feature collaspes and vector quantization
- Hyperparameters : Just two parameters to be sweaped
  - *Mask ratio* : how many tokens would be masked?
  - *Pred ratio* : how many tokens would be predicted?

Training Setups

Model Architechture

- LLaMA type Transformer: **987K** parameters
  - `depth=4, dim=128`
  - `#of Group=2, head dim=32, n_heads=2`
  - `FFN expansion=4, dropout=0`
- library : `jax/flax`
- Full BF16 for params/state

Data and Input Variables

- JetClass dataset for SSL and SFT
- Total 21 variables which are almost same as ParT's paper
  - Token Four vector : $p_\mathrm{T}, \eta, \phi, E, \log p_\mathrm{T}^\mathrm{token}, \log E$
  - Ralative to jet axis : $\Delta\phi, \Delta\eta, \Delta R$
  - Ralative to jet momentum : $\log p_\mathrm{T}^\mathrm{token}/p_\mathrm{T}^\mathrm{jet}, \log E^\mathrm{token}/E^\mathrm{jet}$
  - Track : $q/p_\mathrm{T}, d_0, z_0, \sigma(d_0), \sigma(z_0)$
  - PID : 5 bits for `el, mu, photon, charged had, neutral had`
- Apply standardization

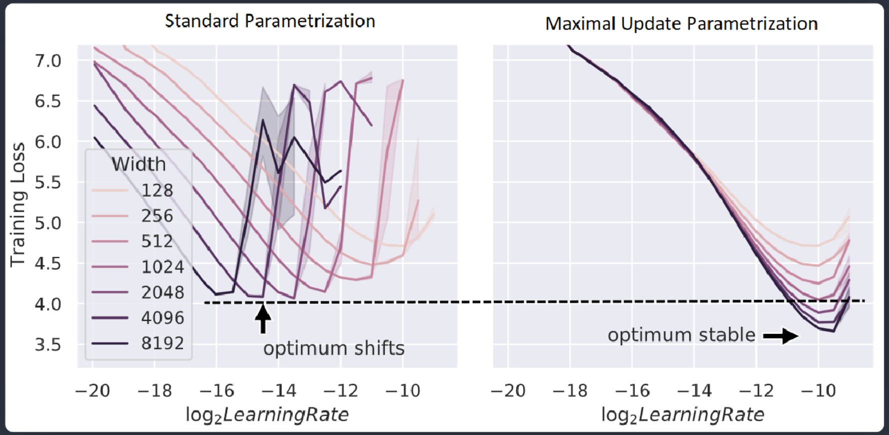
Pretraining

- Batch size : `2048`
- Mask ratio : sweep `{12.5, 25, 50, 75}%`
- LR: `1.0`, 1000 steps for lr warmup, then constant
- Optimizer : Schedule free AdamW, $\beta_1, \beta_2 = (0.9, 0.95)$
- weight decay : `1.0e-2`

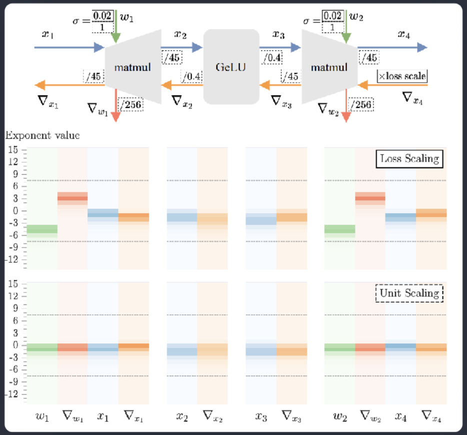
Downstream task

- Batch size : `2048`
- LR: `1.0`, 1000 steps for lr warmup, then constant
- Optimizer : Schedule free AdamW, $\beta_1, \beta_2 = (0.9, 0.95)$
- weight decay : `1.0e-4`

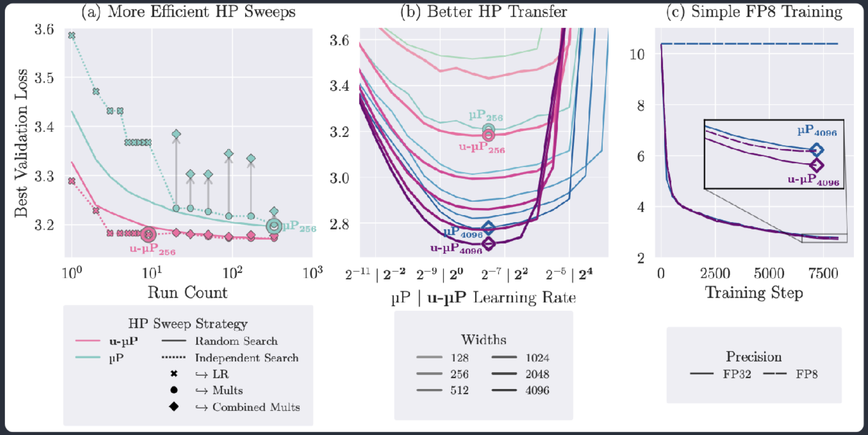
μTransfer : muP and u-muP



muP



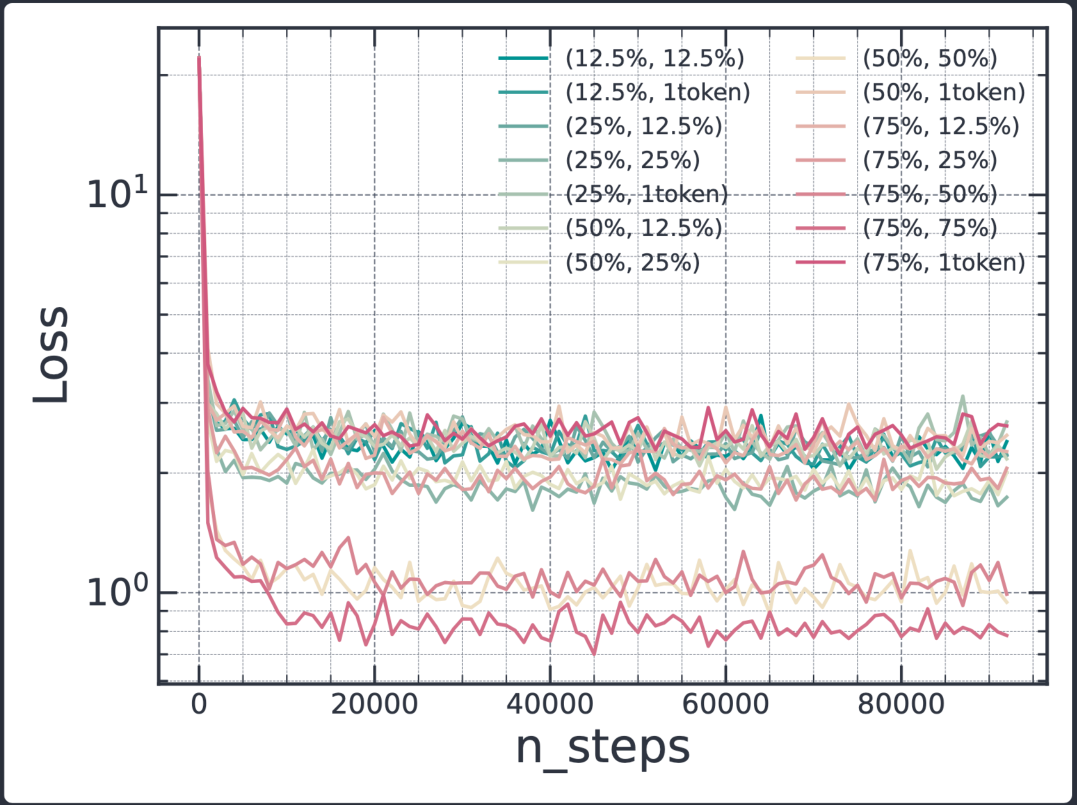
unit-scaling



u-muP

- Maximal Update Parametarization(muP) : Control the activation so that the optimal LR does not depend on width
  - Adjust initial weight and introduce weight-wise LR
- Unit Scaling : Controlling the deviation of network activity to perform low-precision learning
  - Adjust the initial value and the values for forward and backward propagation of gradient updates using fanin/fanout so that the activity falls within the valid range for FP8.
- unit scaling muP(u-muP) : Combine muP and unit-scaling
  - Adjust so that the activation values fall within the valid range in FP8 and are not dependent on the width and depth of the model
  - Control the variance of all activation values to 1 using different scaling for forward and backward propagation

SSL for This Study : Effect of Mask/Pred Ratio

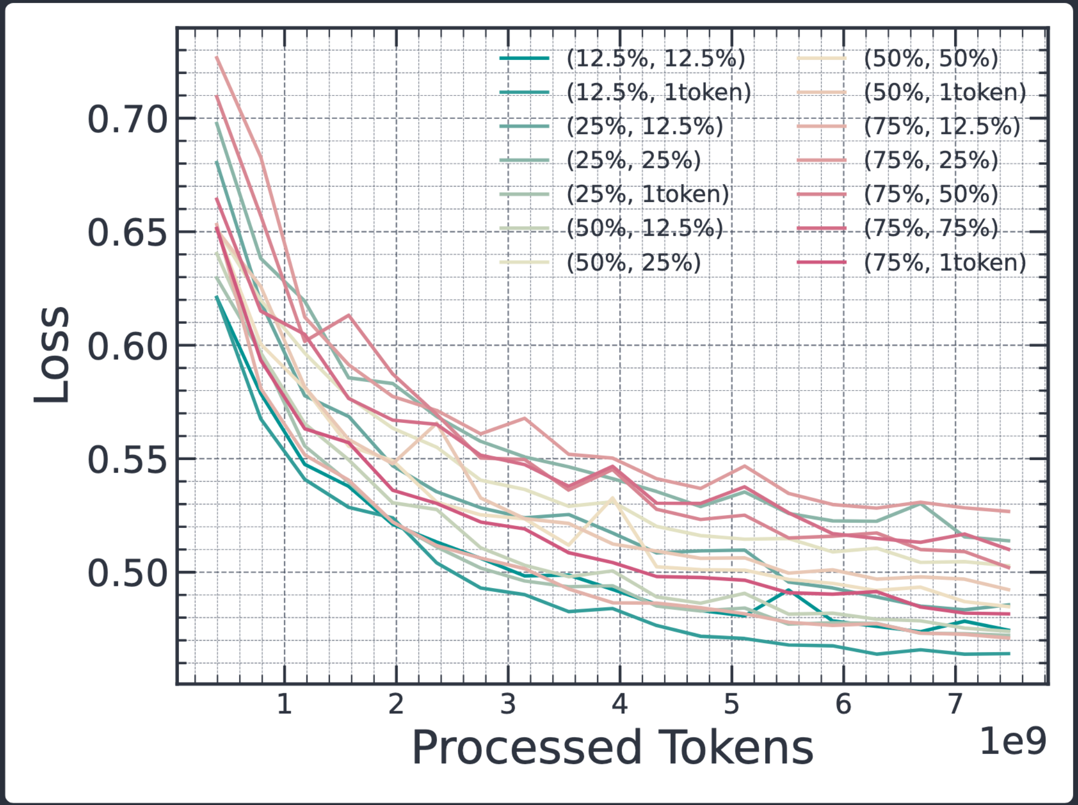


MSE Loss(train sample)

Effect of mask/pred ratio

- *Mask ratio* : (12.5%, 25%, 50%, 75%)
- *Pred ratio* : (1token, 12.5%, 25%, 50%, 75%)
- **MSE Loss** : $(\mathrm{noise}^{\mathrm{target}} - \mathrm{noise}^{\mathrm{predict}})^2$
- Higher mask ratio tend to have better MSE loss
- But how it effects on downstream task?
- 2 epochs for pretrainin

SSL for This Study : Effect of Mask/Pred Ratio



Loss for 10 class(test sample)

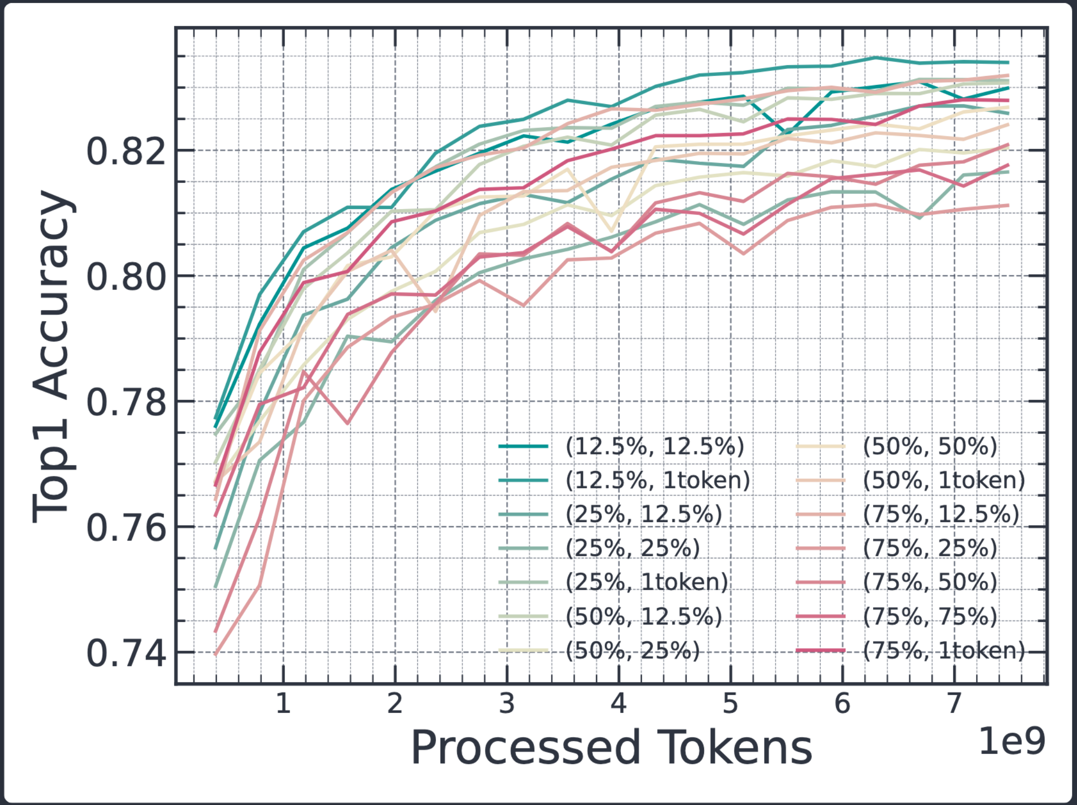
Effect of mask/pred ratio

- *Mask ratio* : (12.5%, 25%, 50%, 75%)
- *Pred ratio* : (1token, 12.5%, 25%, 50%, 75%)
- **MSE Loss** : $(\mathrm{noise}^{\mathrm{target}} - \mathrm{noise}^{\mathrm{predict}})^2$
- Higher mask ratio tend to have better MSE loss
- But how it effects on downstream task?
- 2 epochs for pretrainin

SFT(10 class classification)

- Using pretrained parameters
  - Global average pooing
  - Classification head are trained
  - CE loss for 10 class
- 2 epochs for classification

SSL for This Study : Effect of Mask/Pred Ratio



Accuracy for 10 class(test sample)

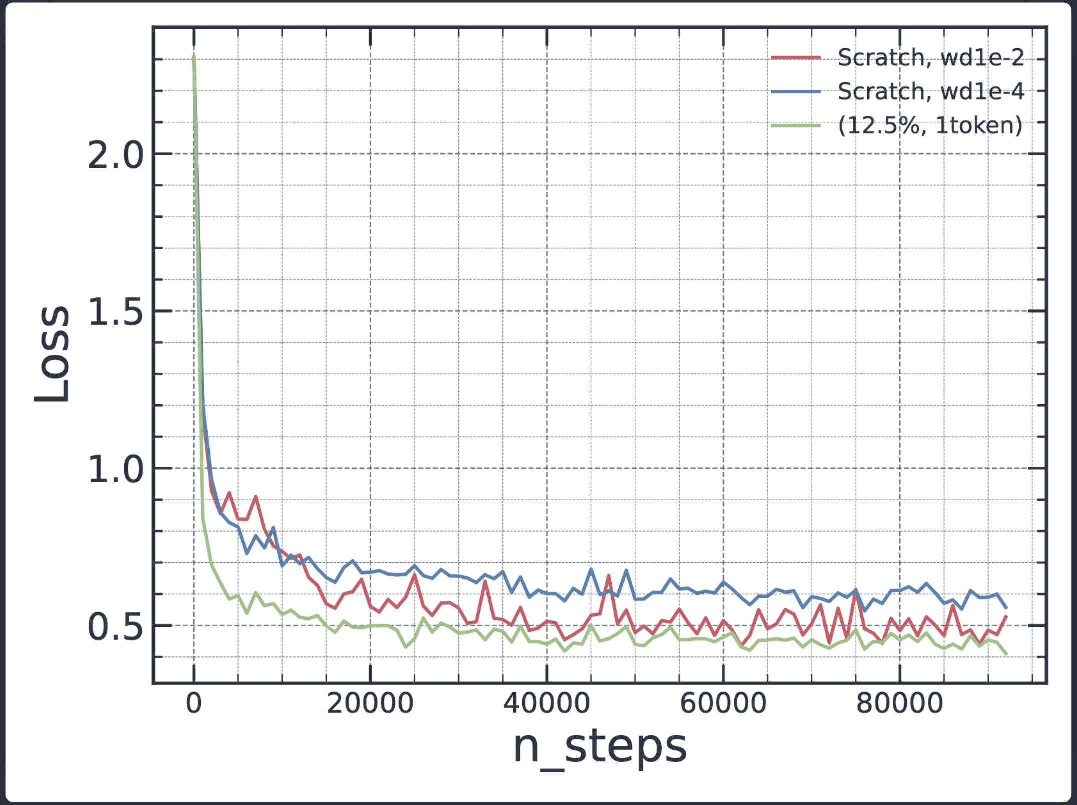
Effect of mask/pred ratio

- *Mask ratio* : (12.5%, 25%, 50%, 75%)
- *Pred ratio* : (1token, 12.5%, 25%, 50%, 75%)
- **MSE Loss** : $(\text{noise}^{\text{target}} - \text{noise}^{\text{predict}})^2$
- Higher mask ratio tend to have better MSE loss
- But how it effects on downstream task?
- 2 epochs for pretrainin

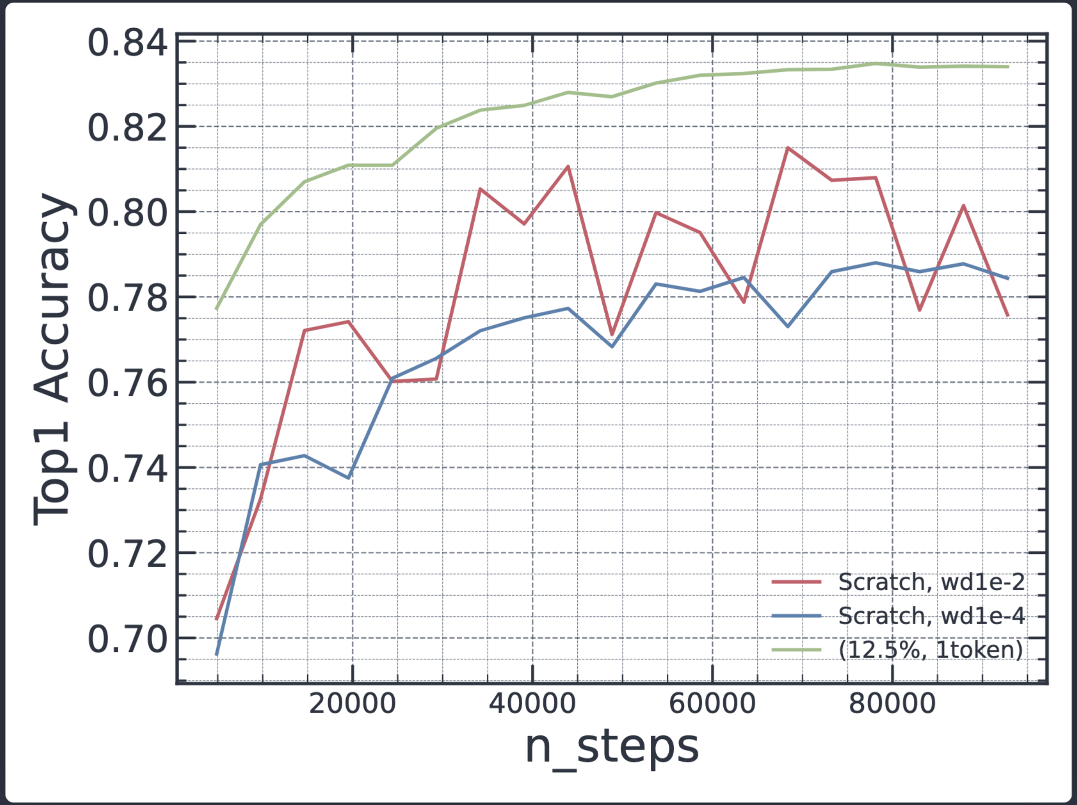
SFT(10 class classification)

- Using pretrained parameters
  - Global average pooing
  - Classification head are trained
  - CE loss for 10 class
- 2 epochs for classification
- ***~83%*** accuracy in the best case

Comparison with SFT and Scratch



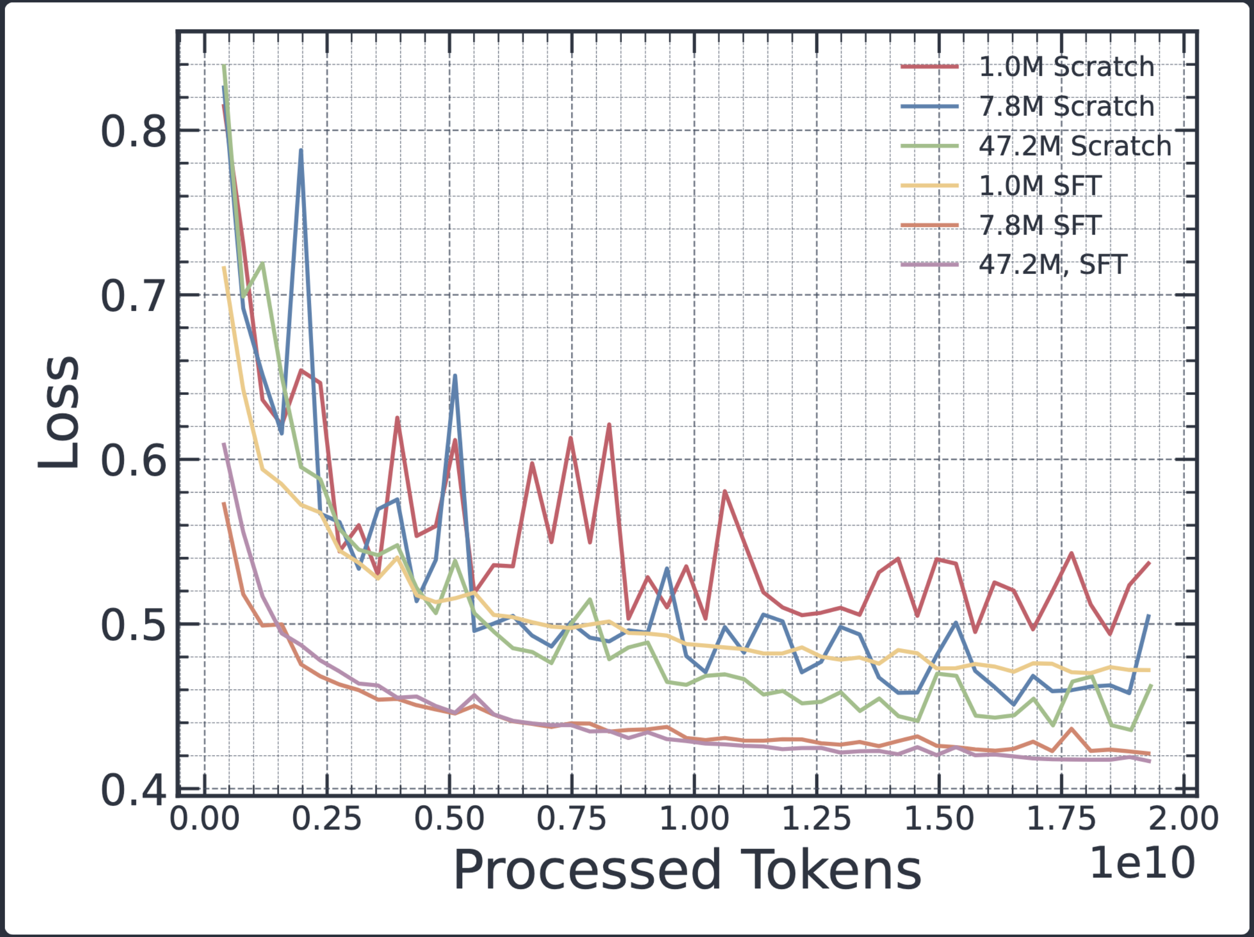
Loss for 10 class(train sample)



Accuracy for 10 class(test sample)

- 2 epochs for SSL, 5 epochs for SFT or scratch, SSL : mask 12.5% ttokens and pred just 1 token
- SSL + SFT result show *~83.5%*, from scratch show *81.5%* ⟶ 2% gain!
  - Scratch training is sensitive for weight decay, higher value tends to have better performance, but unstable.

Scaling The Model



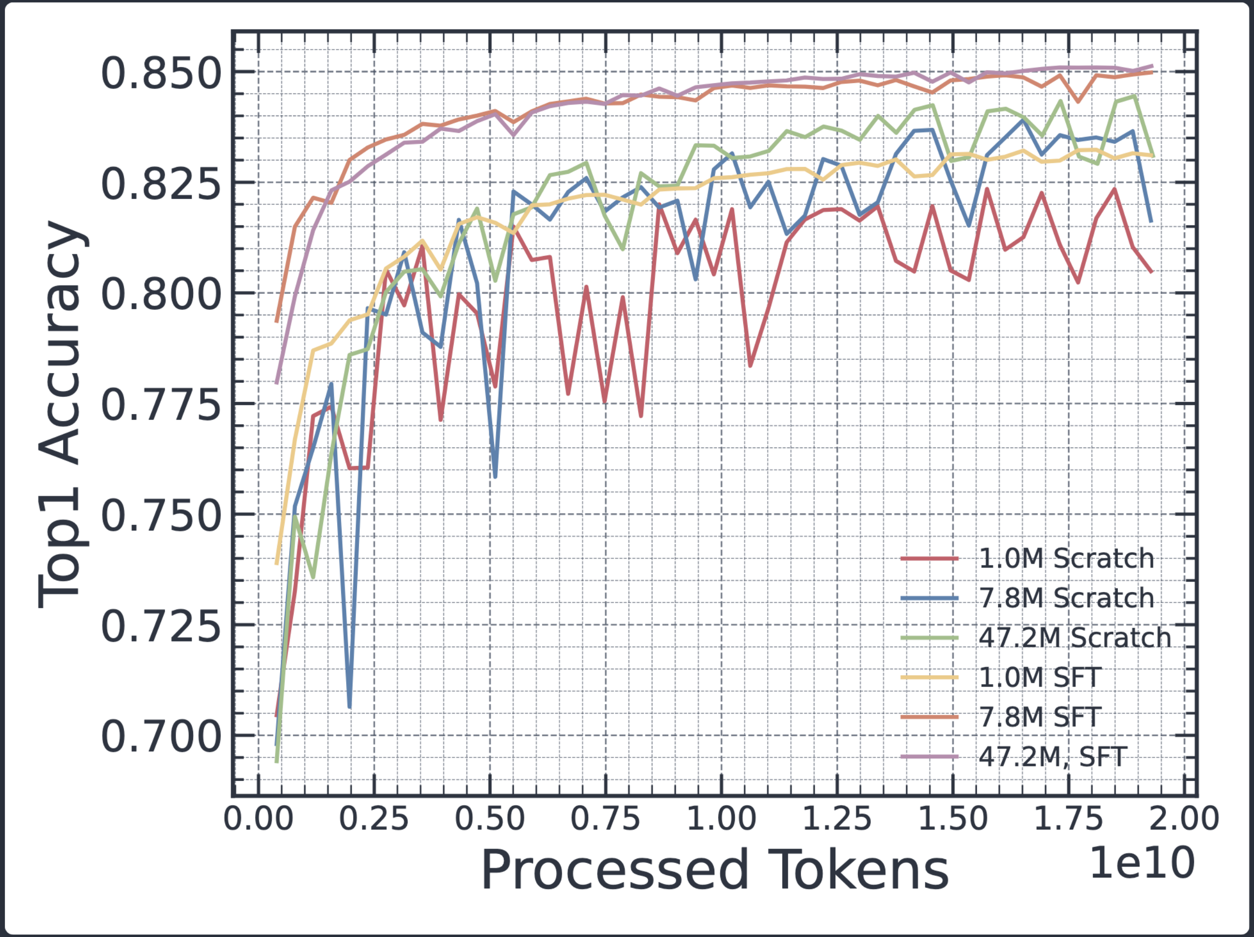
Test Loss Curve

Model Parameters

	depth	dim	Group	head(n,dim)	#of params
1M	4	128	2	(2, 32)	987.6K
8M	8	256	2	(4,32)	7.9M
47M	12	512	2	(8, 32)	47.2M

- 7.8M model show good improvement for test loss.
- SFT models show stable loss curves

Scaling The Model



Test Loss Curve

Model Parameters

	1M	8M	47M
Scratch	82.3%	83.5%	84.2%
SFT	83.2%	84.9%	85.2%

- SFT model show good improvement for test loss.
- 85% accuracy for SFT model, 83% for scratch
  - ~2% gain
- 47M model $\longrightarrow$ limit of this data?
  - bug? mistaken?
- SSL show clear improvement!

Conclusion

Summary

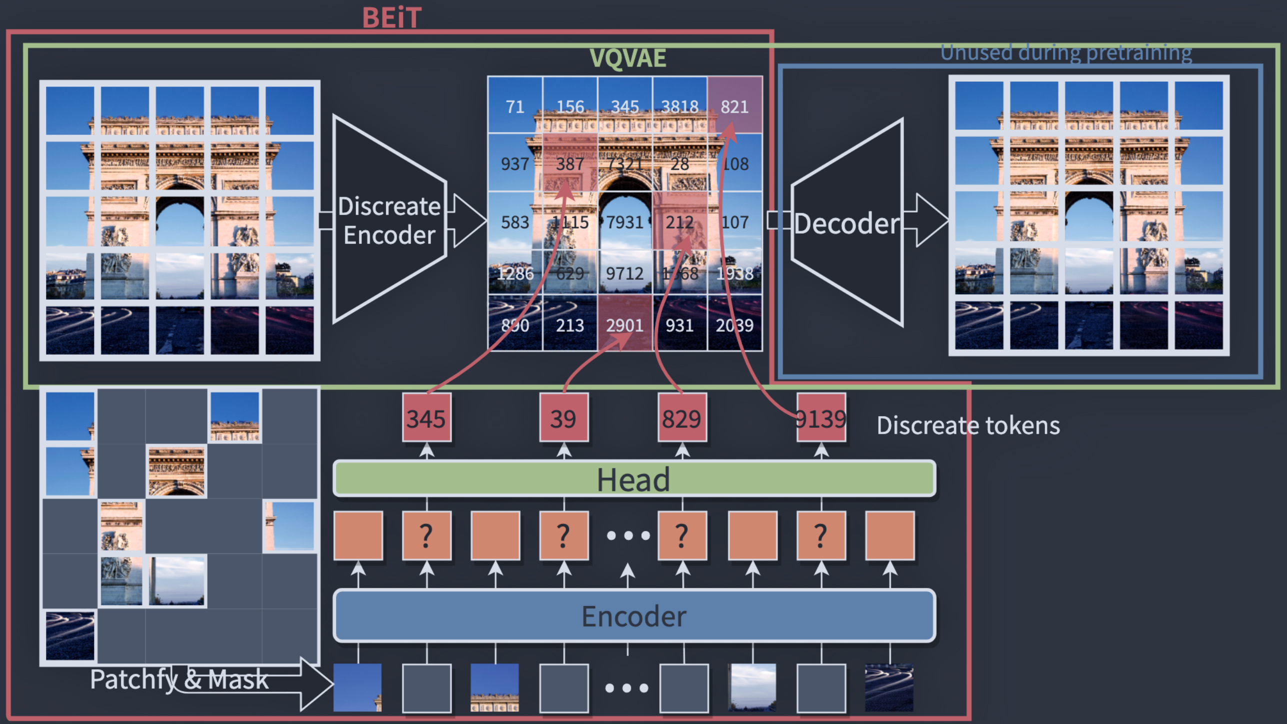
- Using the foundation model can improve various jet related tasks.
- To train the foundation model is a one of difficult problem
- Developed a novel method without vector quantization
- Confirmed SoTA level performance with the self-supervised learning
    - **It's done by pure transformer without any jet related knowldge**

Plan

- Confirm performance with other tasks $\longrightarrow$ creating new benchmark datasets
- Try larger models, confirm the scaling law
- A lot of techniques for improvements to be tried
    - MoE, ToMe, differential attention

backup

Self-Supervised Learning: BEiT



BEiT : Masked Image Modeling with Vector-Quantized Visual Tokenizers

- BERT style with Vector Quantizatized VAE, Predict VQVAE index for masked tokens
- **Good**: Reducing feature collaspes,
- **Bad** : Painful training of Discreate VAE(VQVAE), not clear for performance after SFT.
  - A lot of techniques and methods, need to optimize hyparparameters
  - Need three steps : VQVAE training → SSL → SFT