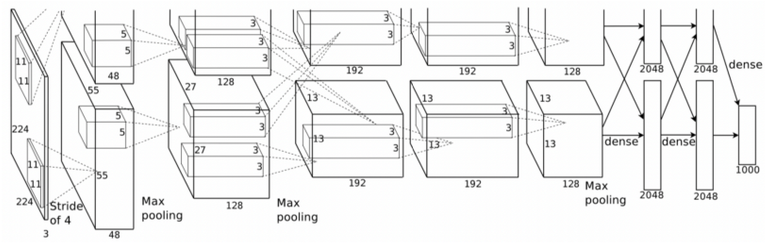
人工画像を用いた
Vision Transformerの超大規模事前学習

東京工業大学
学術国際情報センター
横田理央

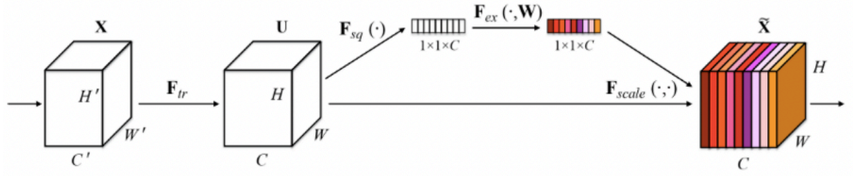
ABCI グランドチャレンジ 2021 成果発表会
2022年1月19日

主要な深層ニューラルネットモデルの変遷

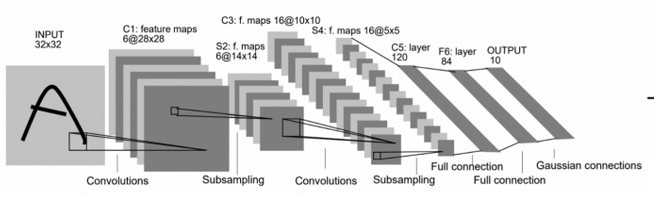
AlexNet: ReLU, Dropout, GPU



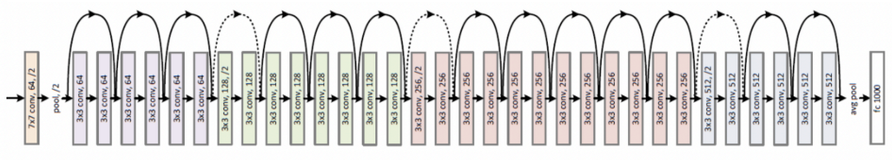
MobileNet: Squeeze and excite



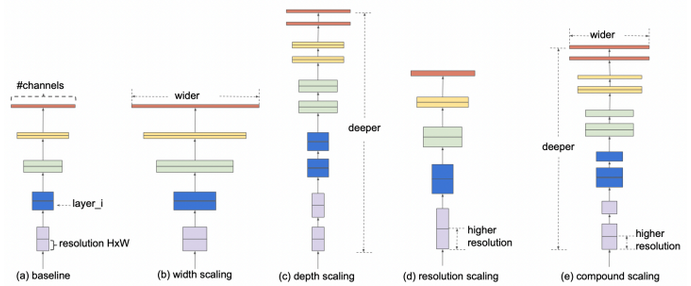
LeNet:畳み込み



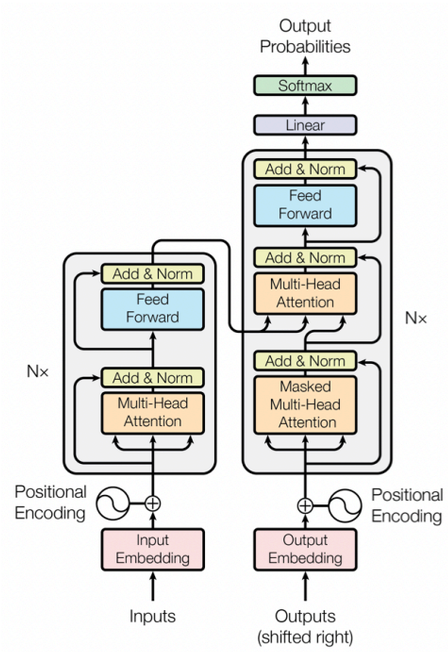
ResNet: Skip connection



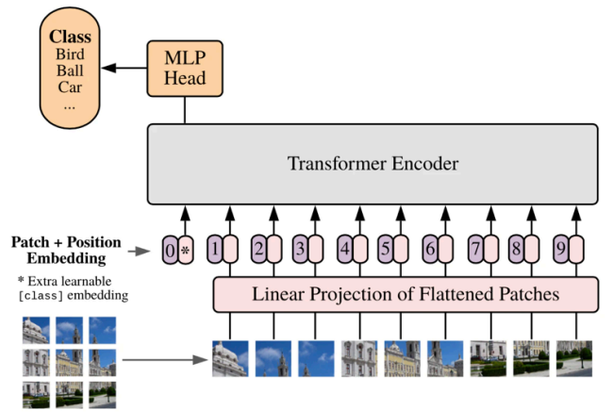
EfficientNet: Neural architecture search



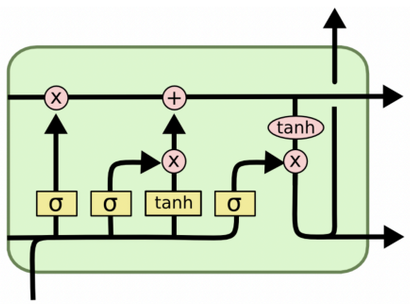
Transformer: 注意機構



Vision Transformer: 画像パッチ



LSTM



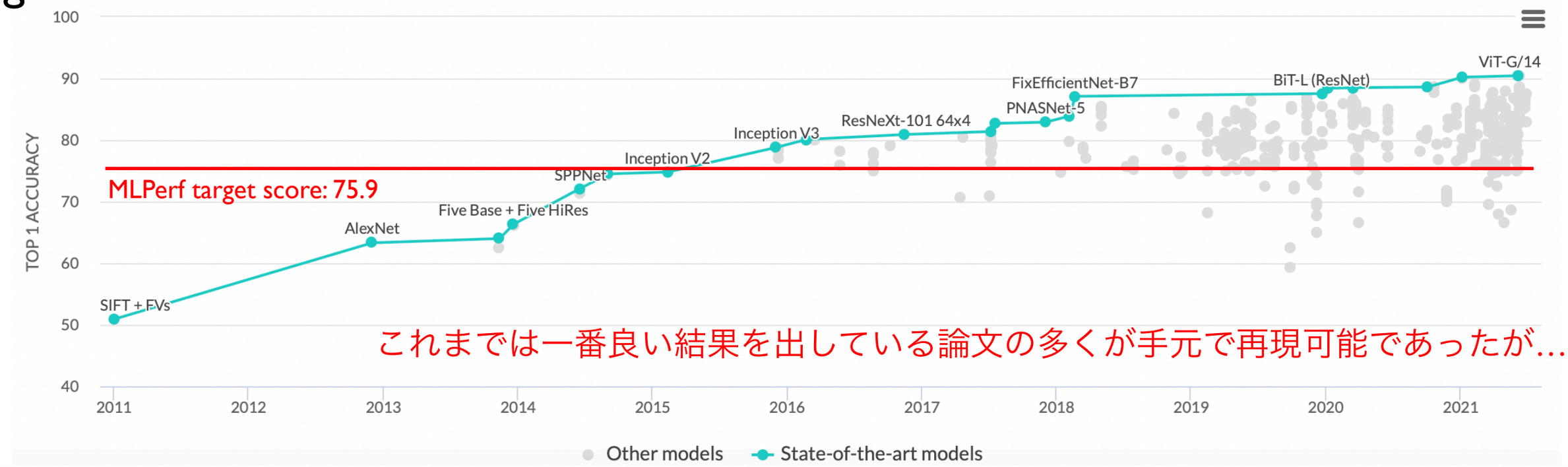
1995 2012 2015 2017 2019 2021

https://towardsdatascience.com/from-lenet-to-efficientnet-the-evolution-of-cnns-3a57eb34672f

Papers with code

https://paperswithcode.com

ImageNet-1k

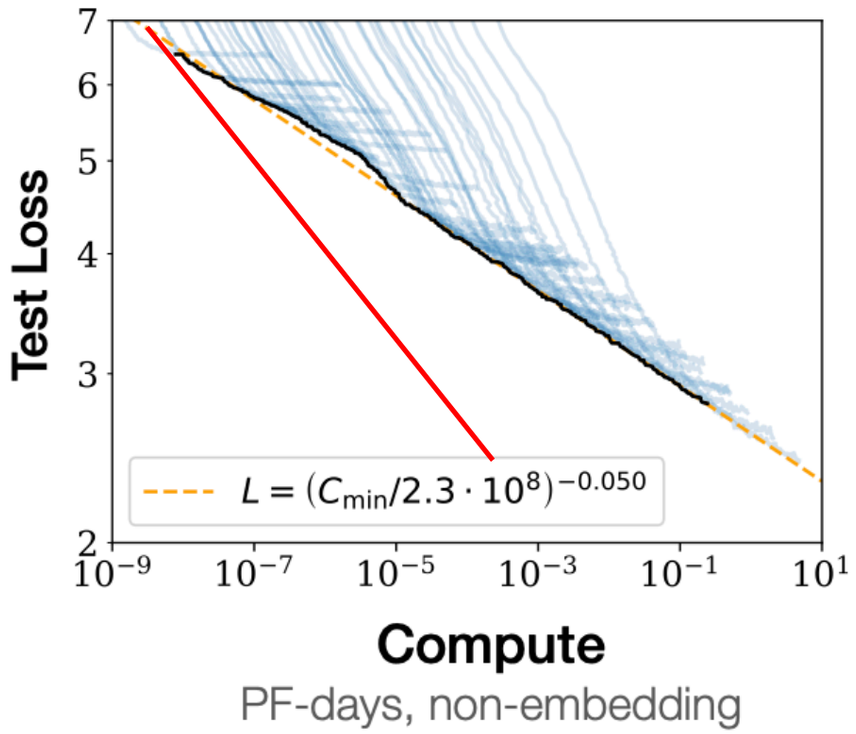


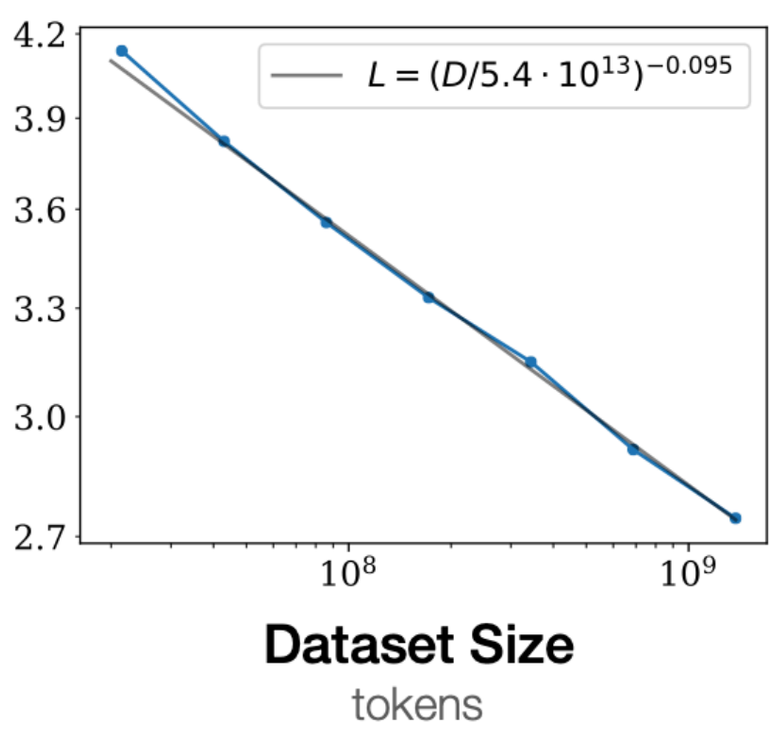
これまでは一番良い結果を出している論文の多くが手元で再現可能であったが…

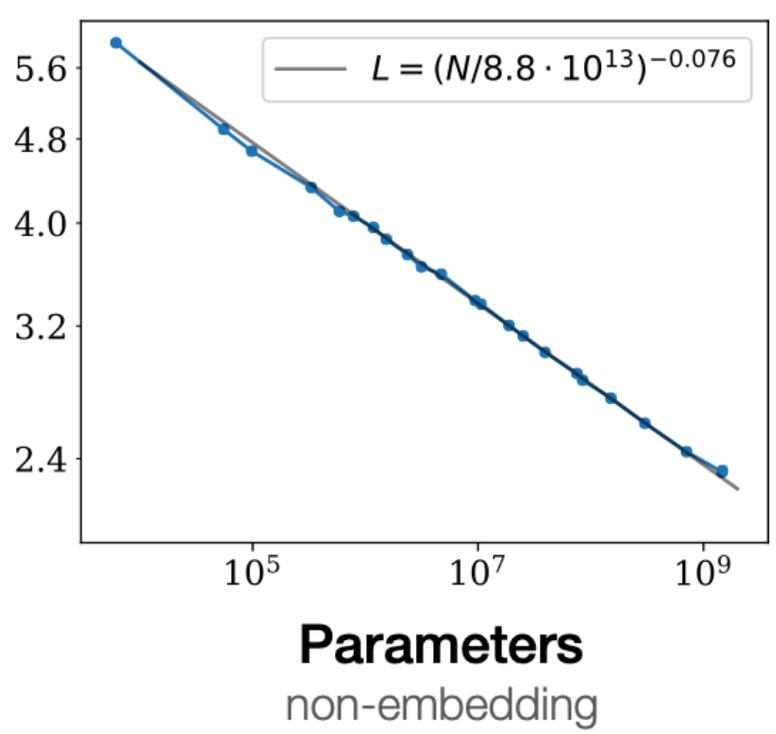
Rank	Model	Top 1 Accuracy	Top 5 Accuracy	Number of params	Extra Training Data	Paper	Code	Result	Year	Tags
1	**ViT-G/14**	90.45%		1843M	✓	Scaling Vision Transformers	未公開		2021	Transformer
2	**ViT-MoE-15B** (Every-2)	90.35%		14700M	✓	Scaling Vision with Sparse Mixture of Experts	未公開		2021	Transformer
3	**Meta Pseudo Labels** (EfficientNet-L2)	90.2%	98.8%	480M	✓	Meta Pseudo Labels			2021	EfficientNet
350	**ResNet-50 MLPerf v0.7 - 2512 steps**	75.92%			✗	A Large Batch Optimizer Reality Check: Traditional, Generic Optimizers Suffice Across Batch Sizes			2021	ResNet

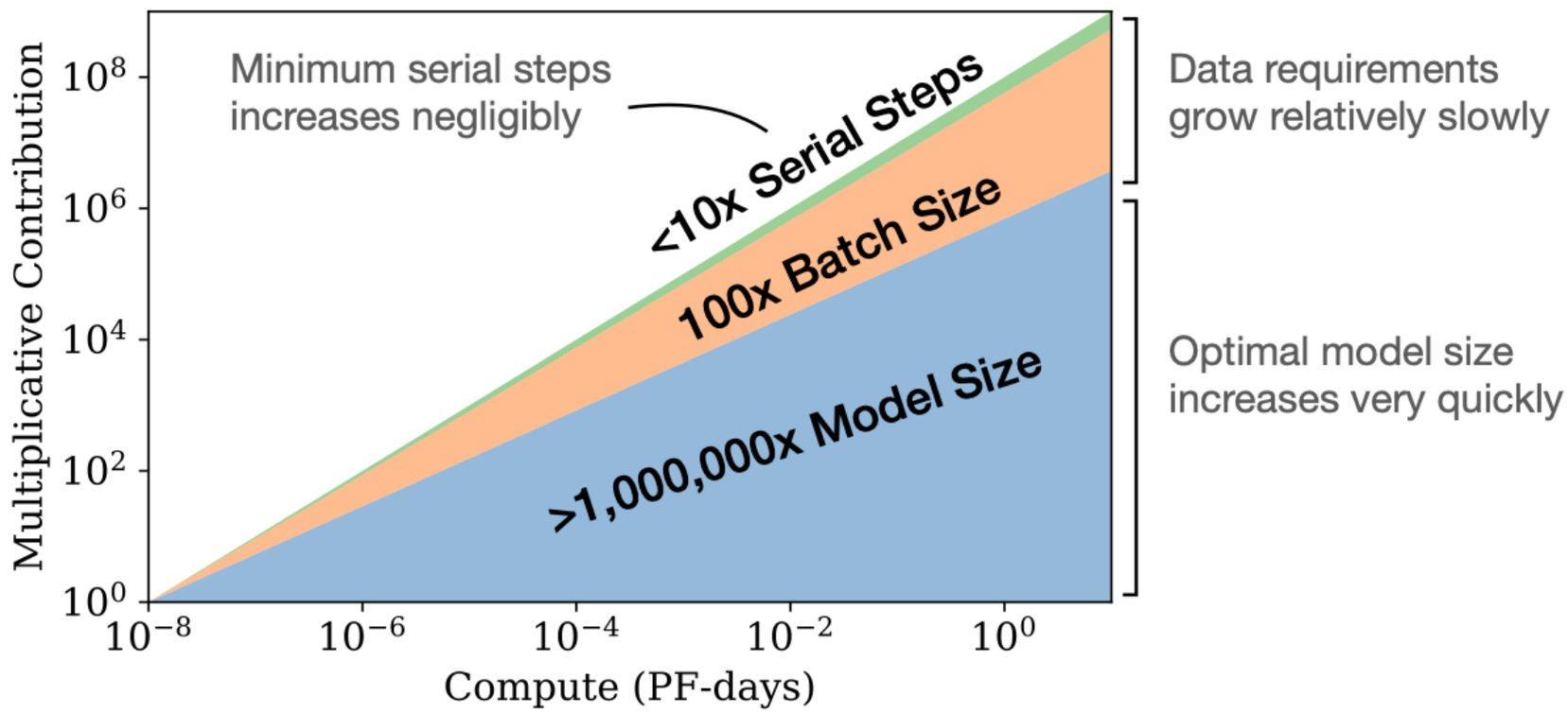
OpenAIのscaling laws論文

https://arxiv.org/pdf/2001.08361.pdf



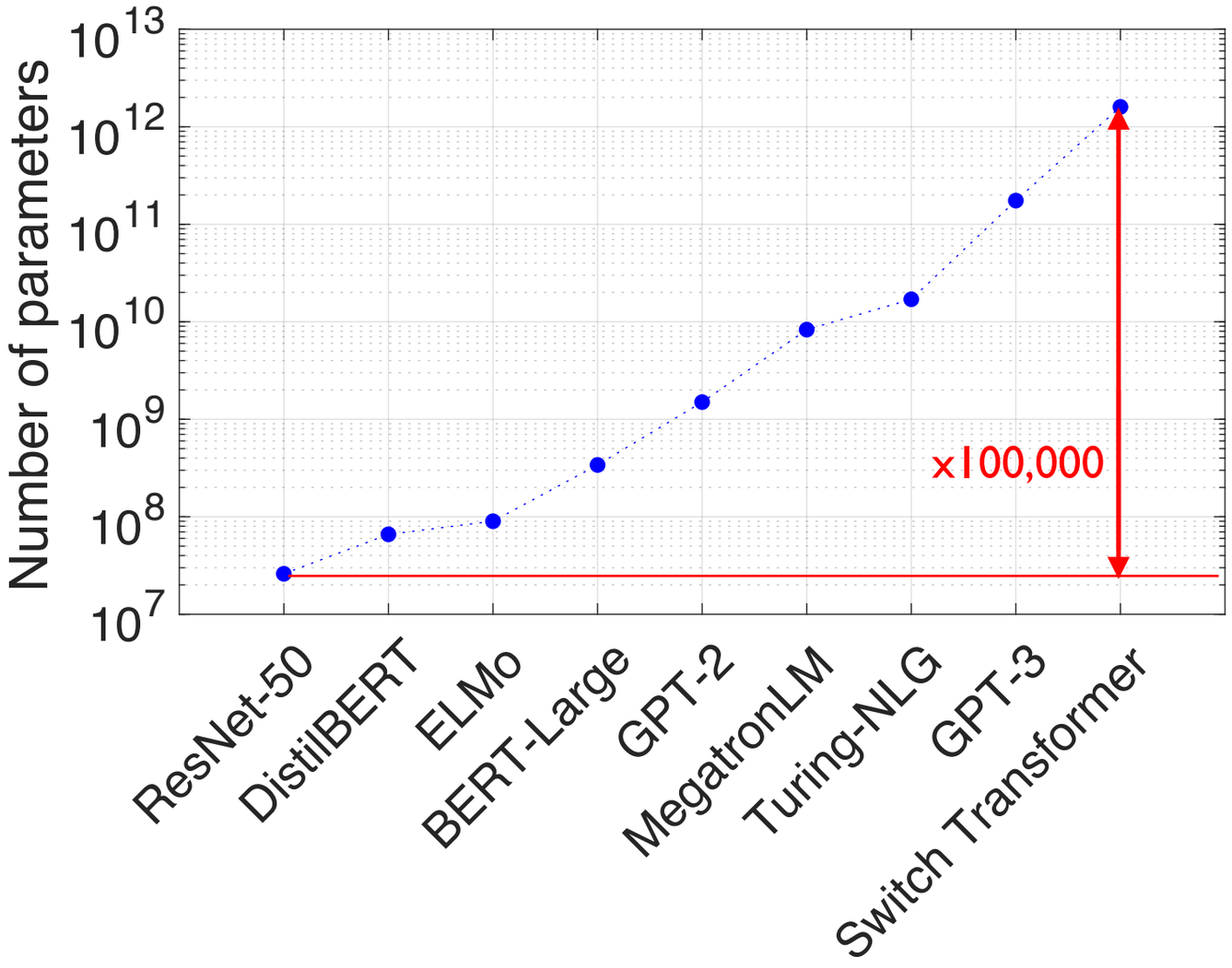






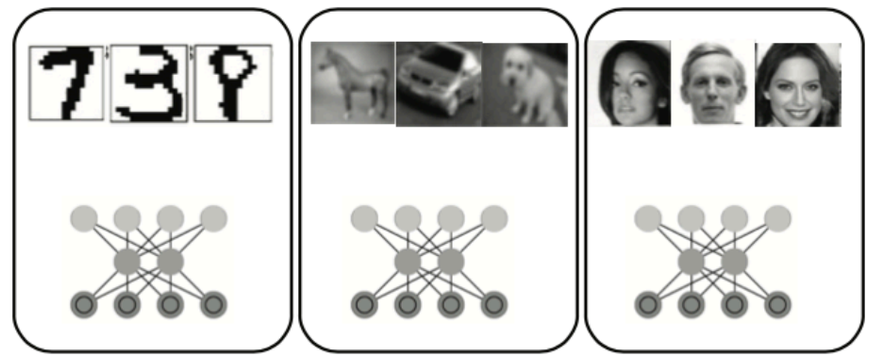
深層学習の大規模化

モデルはどんどん巨大化している



スパコンを用いないとできない
規模になってきている

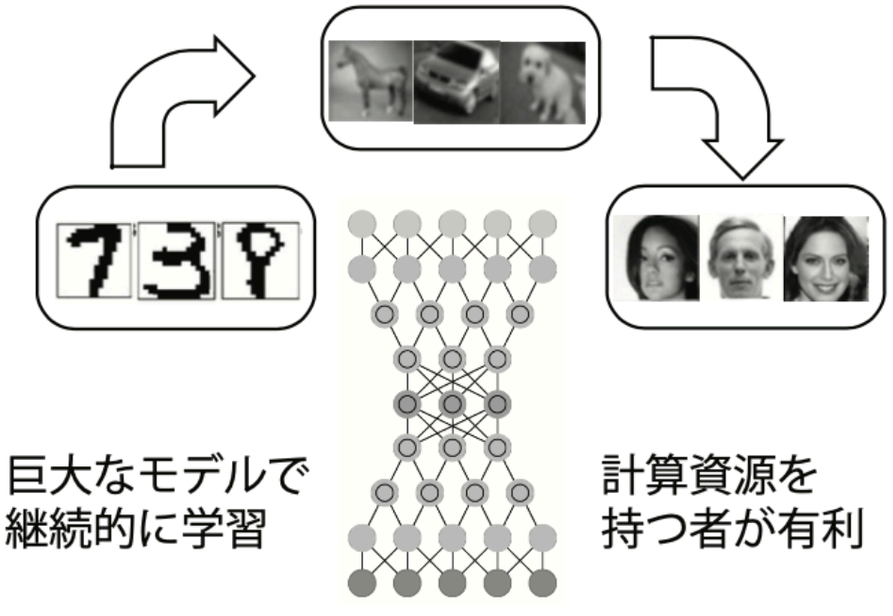
今までの深層学習



皆が冗長に似たような学習

データを大量に持つ者が有利

これからの深層学習



巨大なモデルで
継続的に学習

計算資源を
持つ者が有利

ViTは事前学習に膨大なデータが必要

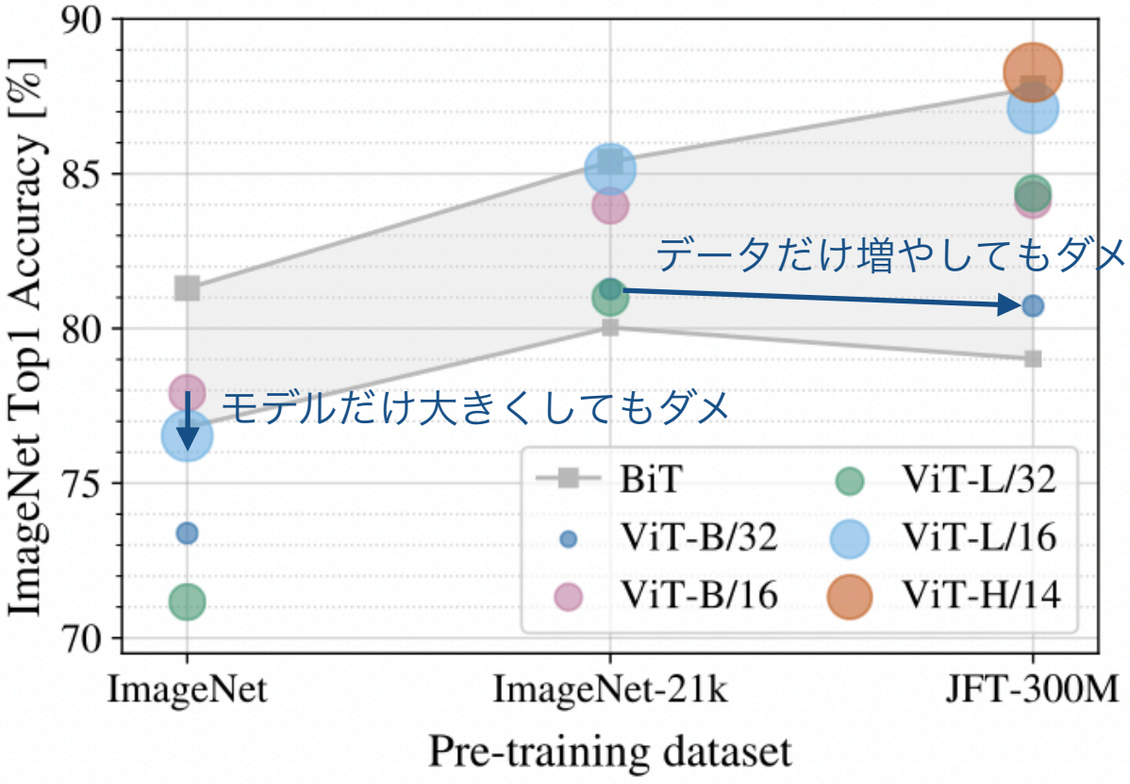
ViTはTiny, Small, Base, Large, Huge,Giant
と規模の異なるものが何種類もある

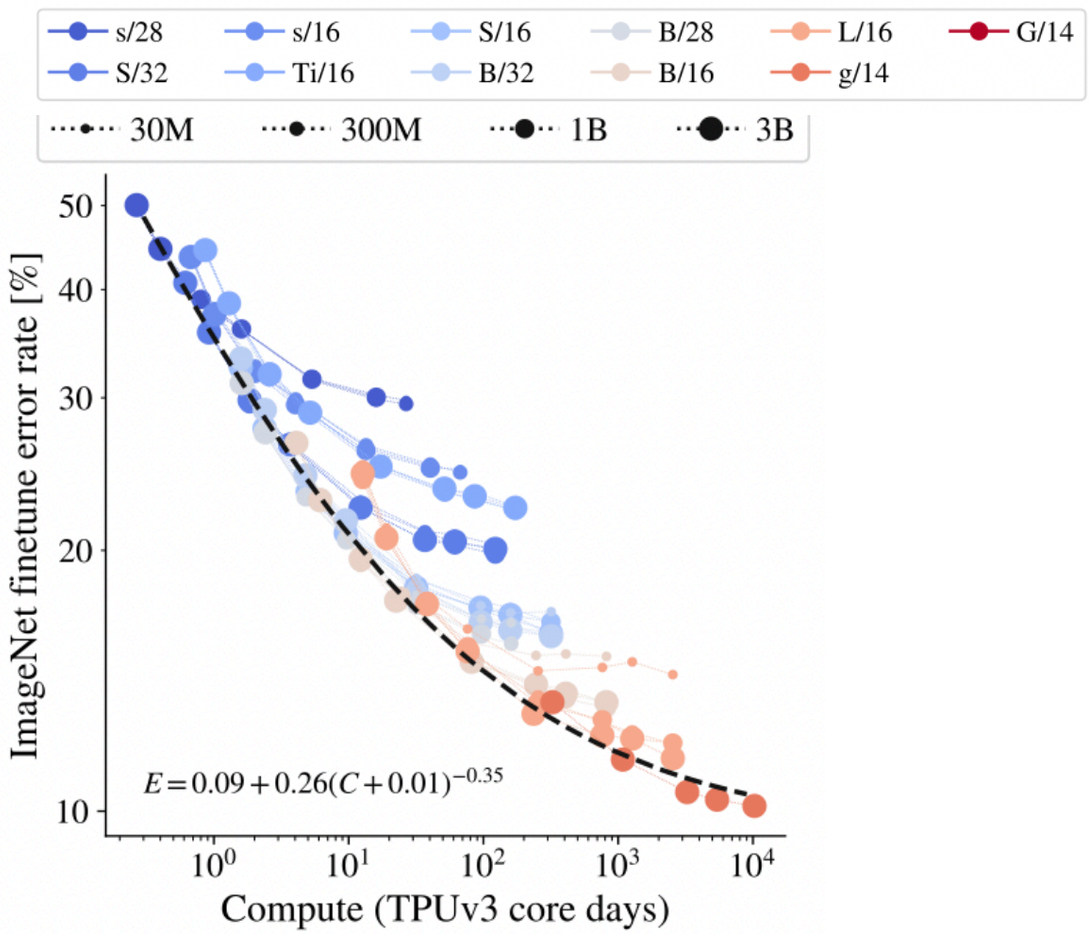
大規模なものほど高精度になるが、
その事前学習に必要なデータも増大

JFT-300MやJFT-3Bはそもそも未公開
なので追試ができない

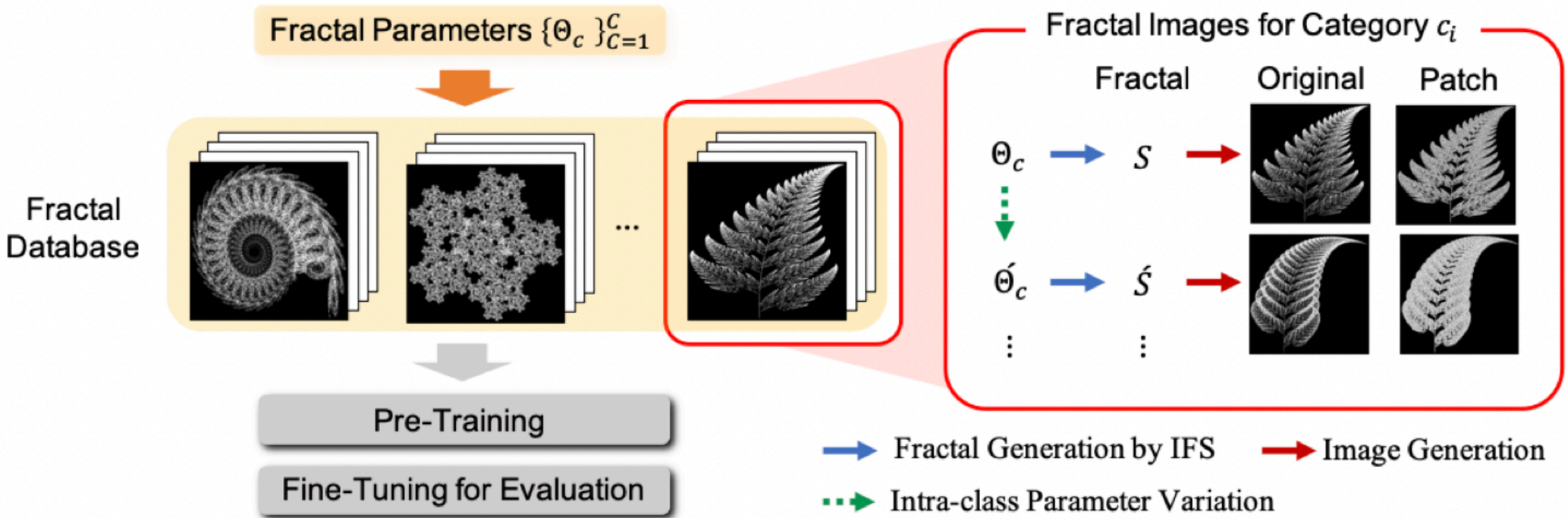
Table 2: Model architecture details.

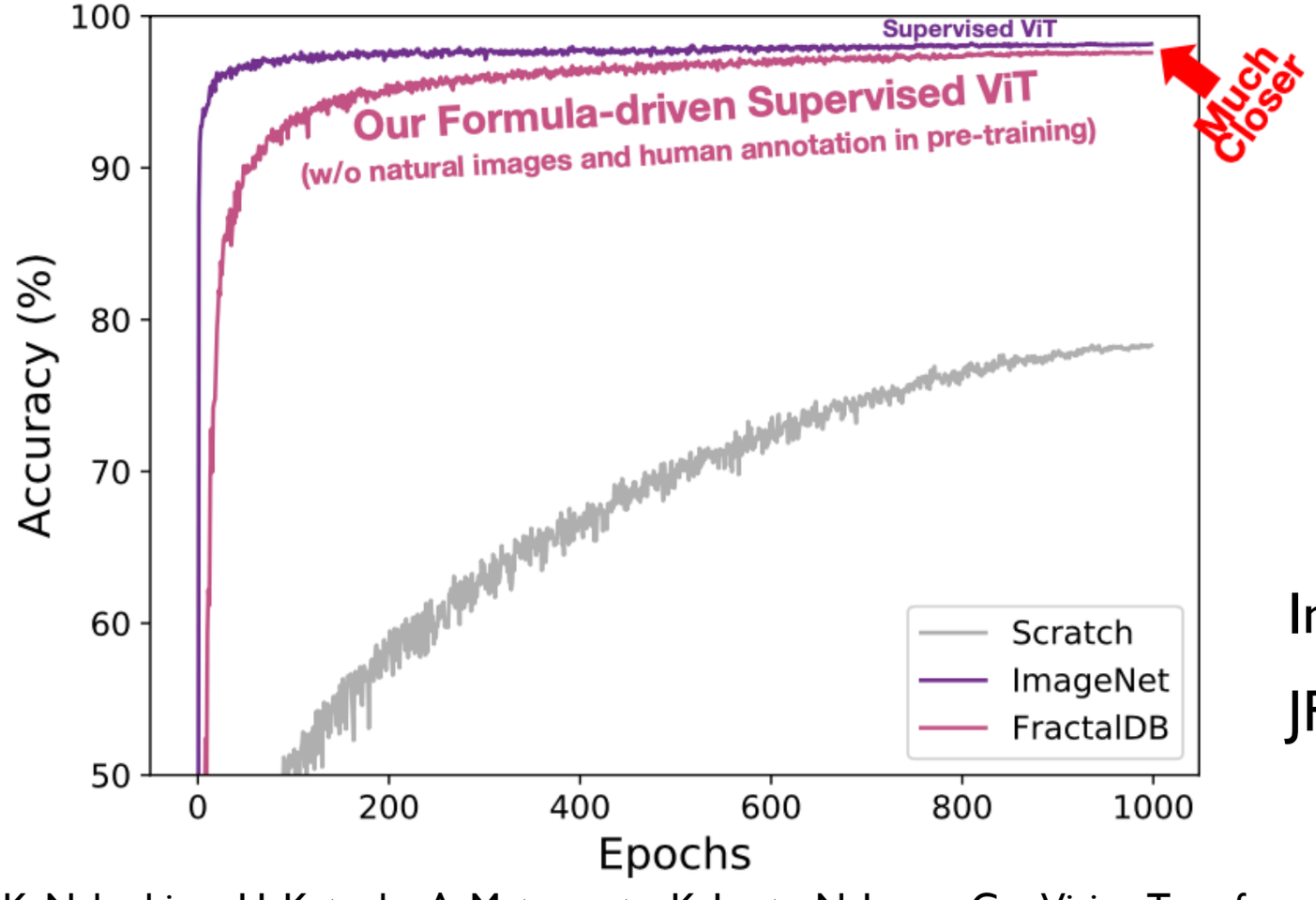
Name	Width	Depth	MLP	Heads	Mio. Param	GFLOPs	
						224^2	384^2
s/28	256	6	1024	8	5.4	0.7	2.0
s/16	256	6	1024	8	5.0	2.2	7.8
S/32	384	12	1536	6	22	2.3	6.9
Ti/16	192	12	768	3	5.5	2.5	9.5
B/32	768	12	3072	12	87	8.7	26.0
S/16	384	12	1536	6	22	9.2	31.2
B/28	768	12	3072	12	87	11.3	30.5
B/16	768	12	3072	12	86	35.1	111.3
L/16	1024	24	4096	16	303	122.9	382.8
g/14	1408	40	6144	16	1011	533.1	1596.4
G/14	1664	48	8192	16	1843	965.3	2859.9





フラクタル画像を用いた事前学習





$$\theta_i = (a_i, b_i, c_i, d_i, e_i, f_i)$$

$$w_i(\boldsymbol{x}; \theta_i) = \begin{bmatrix} a_i & b_i \\ c_i & d_i \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} e_i \\ f_i \end{bmatrix}$$

ImageNet ⇒ 100万画像規模

JFT-3B ⇒ 30億画像規模

3000倍

K. Nakashima, H. Kataoka, A. Matsumoto, K. Iwata, N. Inoue, Can Vision Transformers Learn without Natural Images? arXiv:2103.13023

フラクタルの描画方法

クラス数

100

充填率

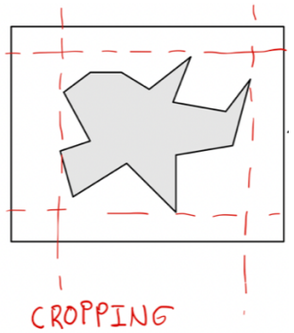
80%

描画の範囲

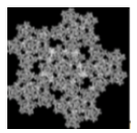
10,000

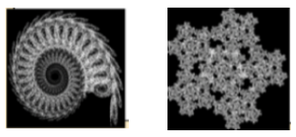
画像の生成

$$w_i(\boldsymbol{x};\theta_i) = \begin{bmatrix} a_i & b_i \\ c_i & d_i \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} e_i \\ f_i \end{bmatrix}$$



Fractal画像





画像の大きさ

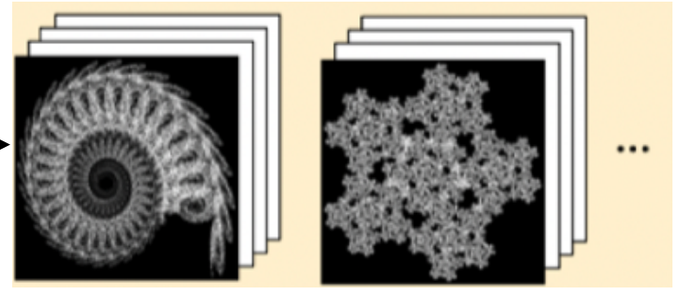
512 x 512

クラスあたりの画像数

1000

データセットの生成

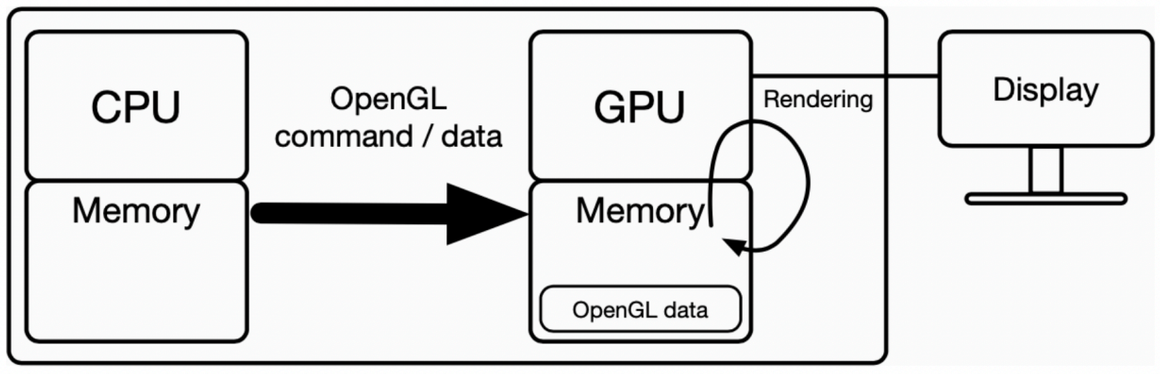
- 回転
- 反転
- 縮小・拡大



1000画像　1000画像

フラクタル描画の高速化

Original FractalDB



CPU上でFractalの計算を行い

GPU上でOpenGLの描画を行う場合

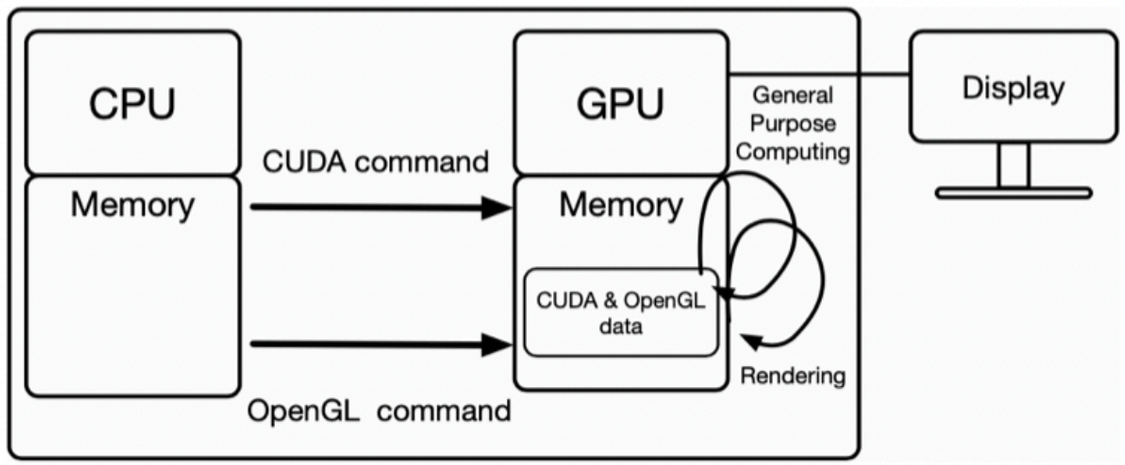
CPU-GPU間通信が大量に発生

1秒に5.4枚だと30億枚の描画には17.6年かかる

↓

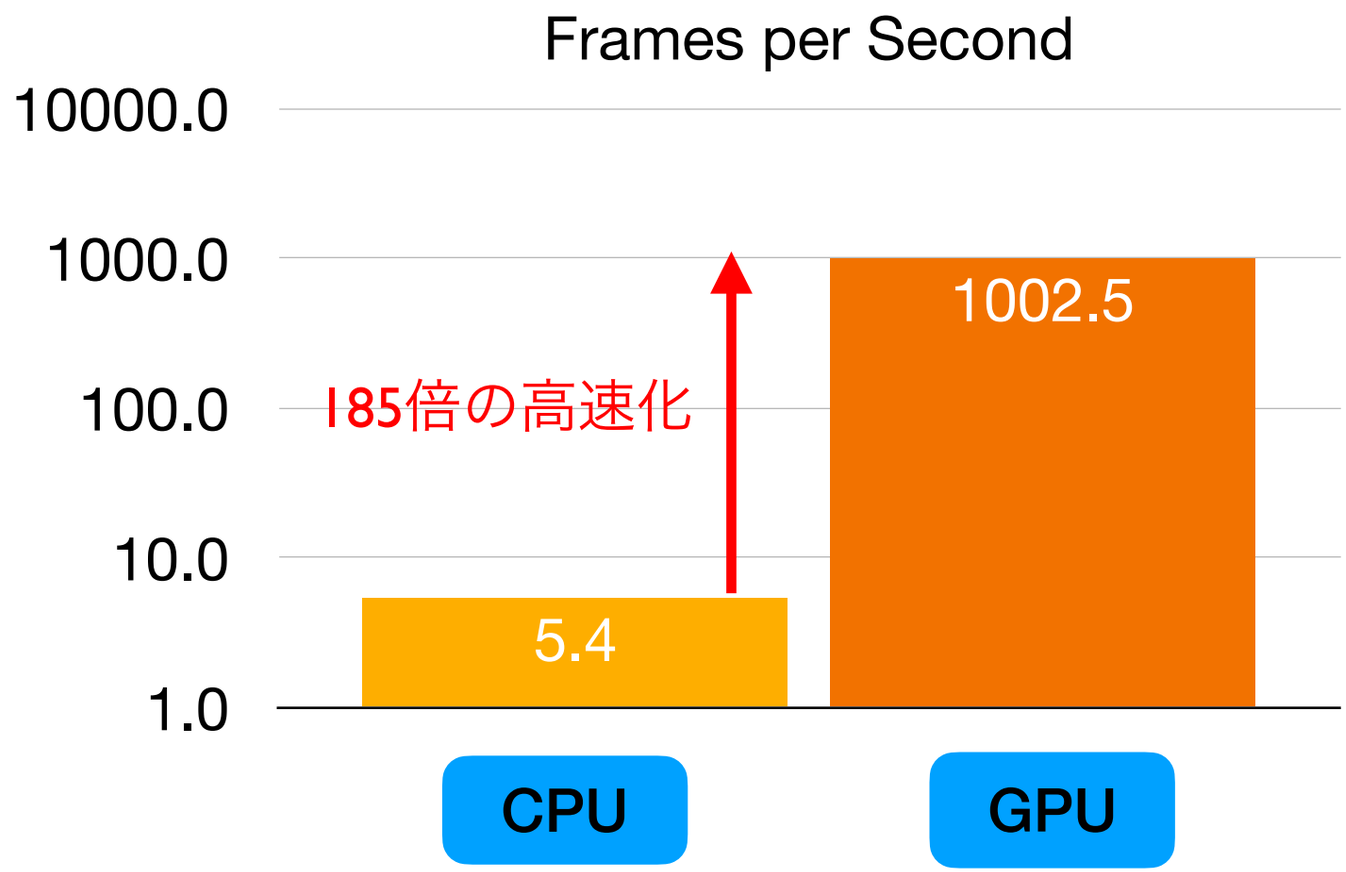
185倍の高速化によりこれが34.6日に短縮

Ours



Fractalの計算もOpenGLの描画も

両方GPU上で行う場合



ABCI上での高速描画

通常描画にはディスプレイが必要

↓

ディスプレイを介さずに直接画像を
生成して保存できないか？

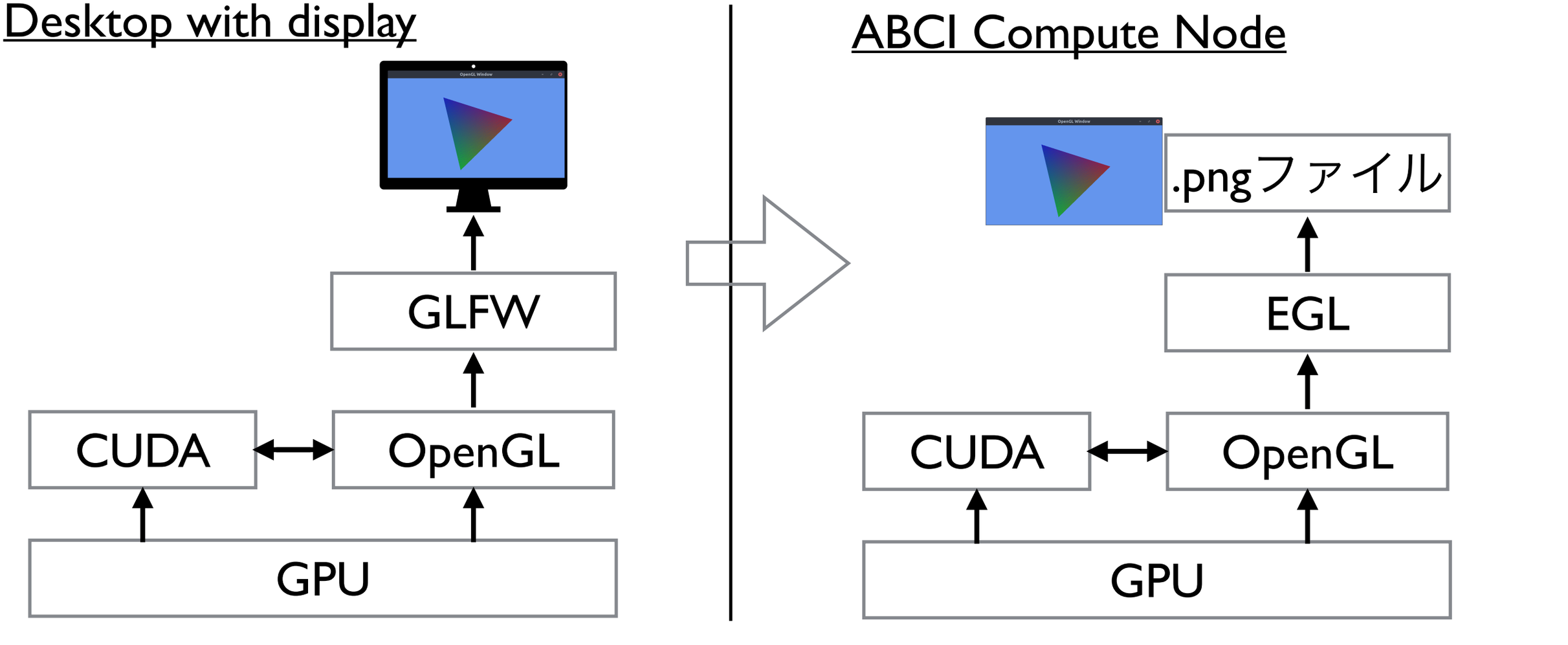
1秒に5.4枚だと30億枚の描画には17.6年かかる

↓

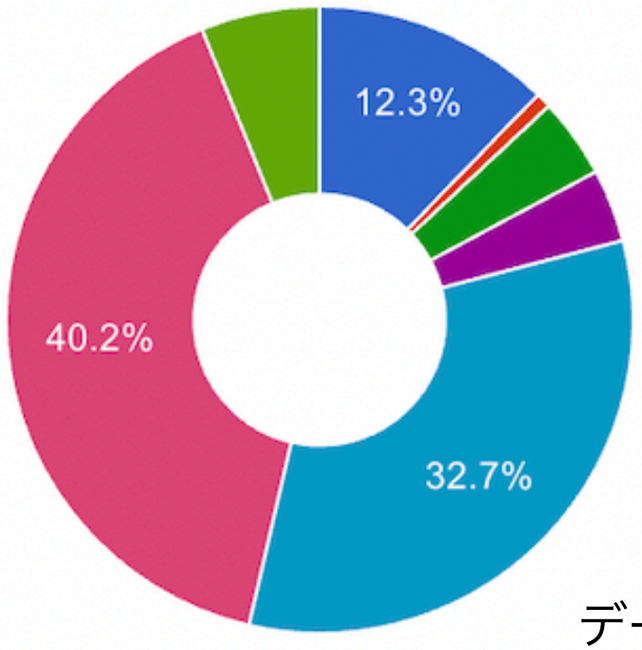
185倍の高速化によりこれが34.6日に短縮

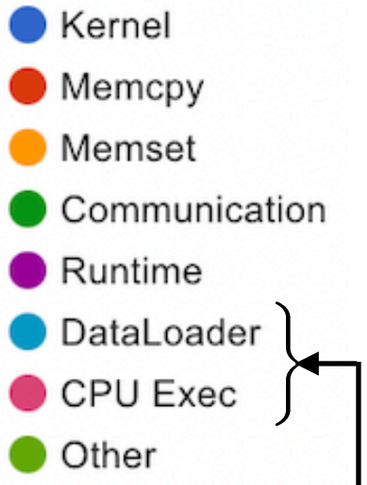
↓

128GPUを用いれば6.5時間で描画可能

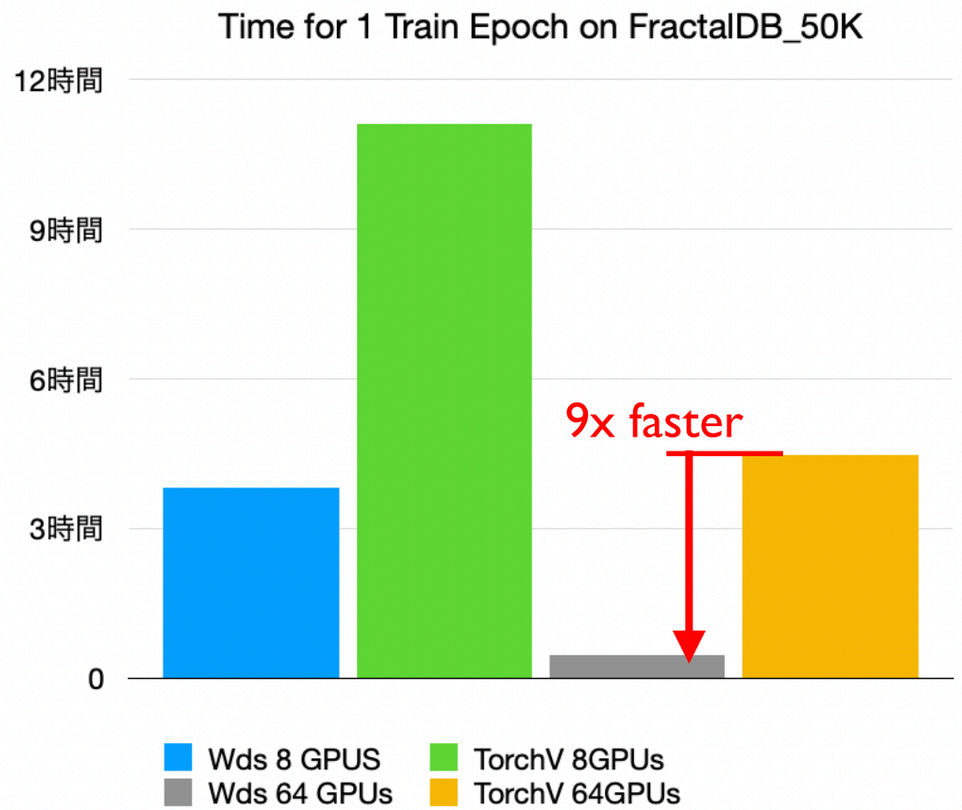


Tarファイルから直接学習





データの読み込みにかかる時間



PyTorch DataLoader

```
train/n01440764/n01440764_0.png
                n01440764_1.png
                n01440764_2.png
                n01440764_3.png
                      ⋮
     /n01443537/*
           ⋮
     /n01507514/*
```

クラス毎のフォルダに画像をpngで保存

大規模になると大量のファイルアクセス

でメタデータサーバが過負荷に…

５千万画像の学習は３時間経っても始まらなかった

WebDataset

```
train/shard00.tar
     /shard01.tar
           ⋮
     /shard99.tar
```

```
shard00.tar: n01440764_4.png
             n01440764_4.cls
             n01443537_2.png
             n01443537_2.cls
             n01445623_7.png
             n01445623_7.cls
                   ⋮
```

シャッフルされた画像とラベルのペアを

任意の粒度でtarファイルに固める

実験条件

事前学習

ImageNet-21k: 21,000クラスのImageNet

FractalDB: 2次元のフラクタル画像

ExFractalDB: 3次元のフラクタル画像

RadialContourDB: 輪郭のみからなる画像

様々なサイズのデータセットを用意

FractalDB-10k, 21k, 50k, 100k, 300k←JFT-300Mと同規模

ABCI環境

python 3.8.6
cuda/11.1
cudnn/8.0 nccl/2.7
openmpi/3.1.6
gcc/7.4.0

ファインチューニング

ImageNet-1k, CIFAR-100, CIFAR-10, Cars, Flowers, VOC12, P30, IN100

モデル

ViT-Ti (5M), ViT-S (20M), ViT-B (80M)

ハイパーパラメータ

基本的にはDeiT [https://arxiv.org/abs/2012.12877]と同じ設定

学習率 {ViT-Ti: 8e-4, ViT-S: 4e-4, ViT-B: 1e-4}

Epochs {10k: 300, 21k: 90, 50k: 40, 100k: 20, 300k, 7}

実験結果

H. Kataoka, R. Hayamizu, R. Yamada, K. Nakashima, N. Inoue, S. Takashima, X. Zhang, E. J. Martinez Noriega, R. Yokota, Replacing Labeled Real-image Datasets with Auto-generated Contours, CVPR 2022 (査読中, 3月に採否通知)

CVPR採択後にプレスリリースを予定しており、

現時点では公表できません。