Vision Transformer

서울대학교 IDEA 연구실
석사과정 신윤섭

An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale

Alexey Dosovitskiy[*,†], **Lucas Beyer**[*], **Alexander Kolesnikov**[*], **Dirk Weissenborn**[*], **Xiaohua Zhai**[*], **Thomas Unterthiner**, **Mostafa Dehghani**, **Matthias Minderer**, **Georg Heigold**, **Sylvain Gelly**, **Jakob Uszkoreit**, **Neil Houlsby**[*,†]

[*]equal technical contribution, [†]equal advising

Google Research, Brain Team

{adosovitskiy, neilhoulsby}@google.com

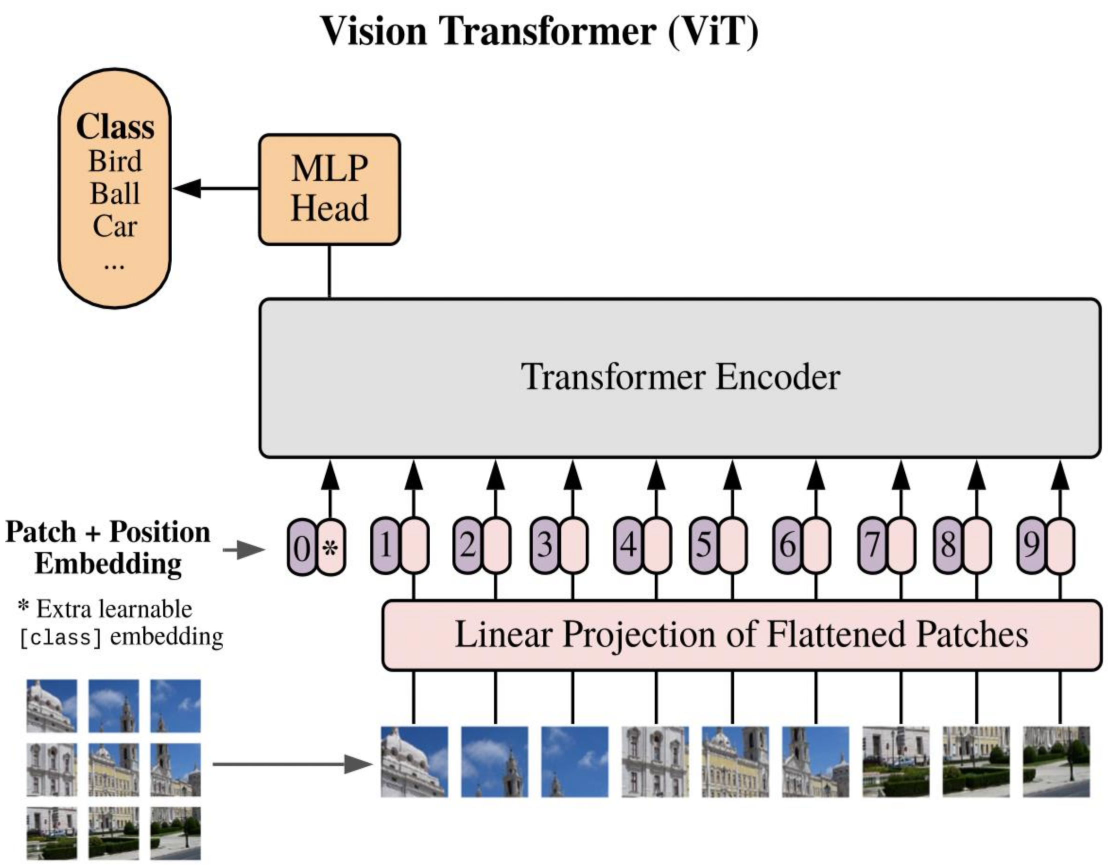
OUTLINE

01. Vision Transformer

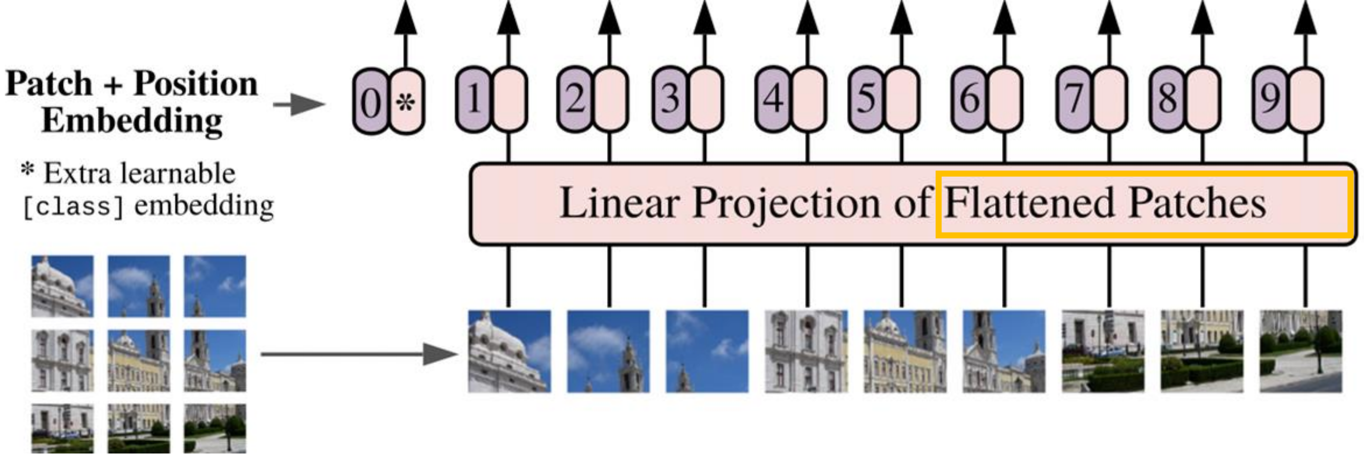
- The Transformer algorithm was first proposed in a paper published in 2021 by researchers from Google Brain.
- In vision, attention is either applied in conjunction with convolutional networks, or used to replace certain components of convolutional networks while keeping their overall structure in place.
- In this review, we will explore how images are applied to transformers and examine the differences between transformers in ViT and LLM. Additionally, we will briefly discuss self-supervised learning using ViT.

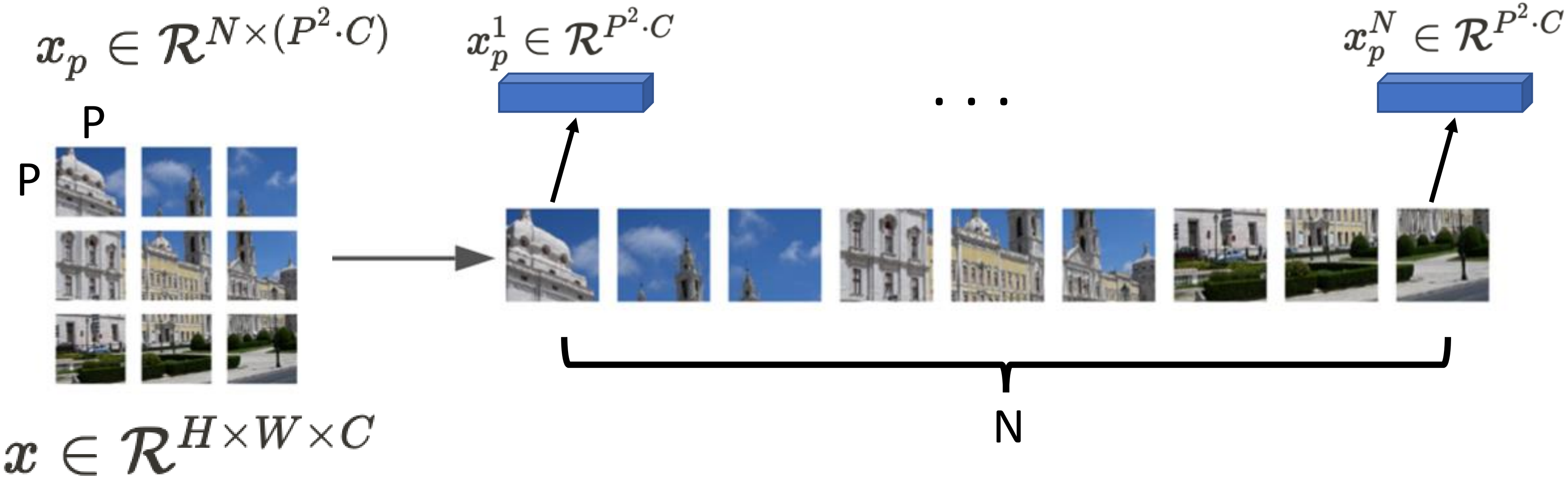
01. Vision Transformer

- ViT is used for image classification tasks.
- ViT use Encoder part of Transformer.
- ViT divides an image into multiple patches and then feeds them into the transformer as input.



- H and W represent the number of pixels in an image.
- C represent channel of image.
- The image is divided into patches and flattened into vectors.

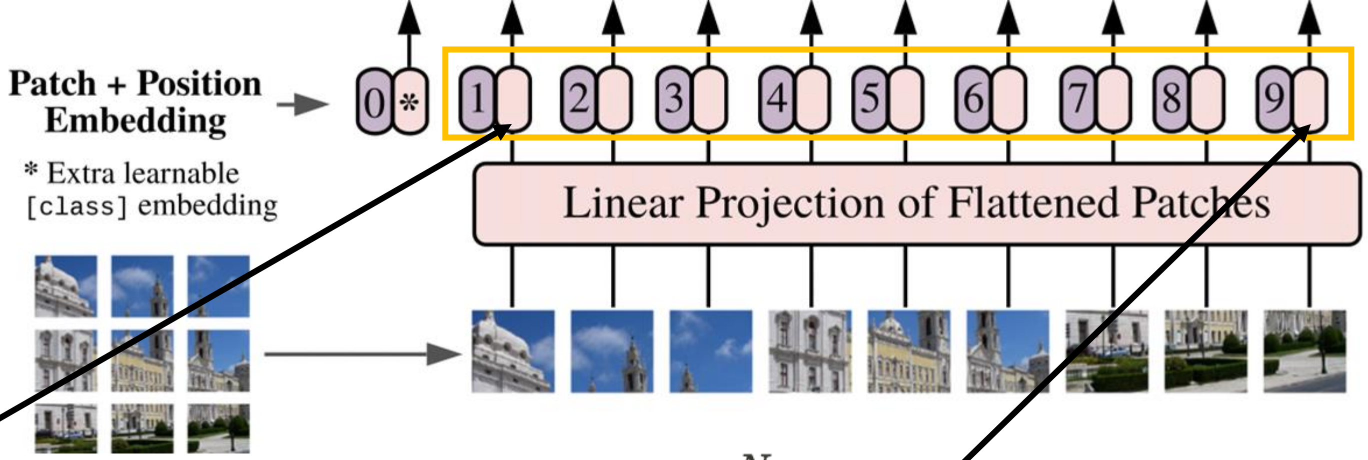




https://www.youtube.com/watch?v=7lxKirsixuQ

- The flattened vectors are linearly projected into D-dimensional vectors through a linear transformation.

- Note that this linear transformation is a learnable matrix that is trained.



$x_p^1 E$

. . .

$x_p^N E$

$E \in \mathcal{R}^{(P^2 \cdot C) \times D}$

$x_p \in \mathcal{R}^{N \times (P^2 \cdot C)}$

$x_p^1 \in \mathcal{R}^{P^2 \cdot C}$

$x_p^N \in \mathcal{R}^{P^2 \cdot C}$

P

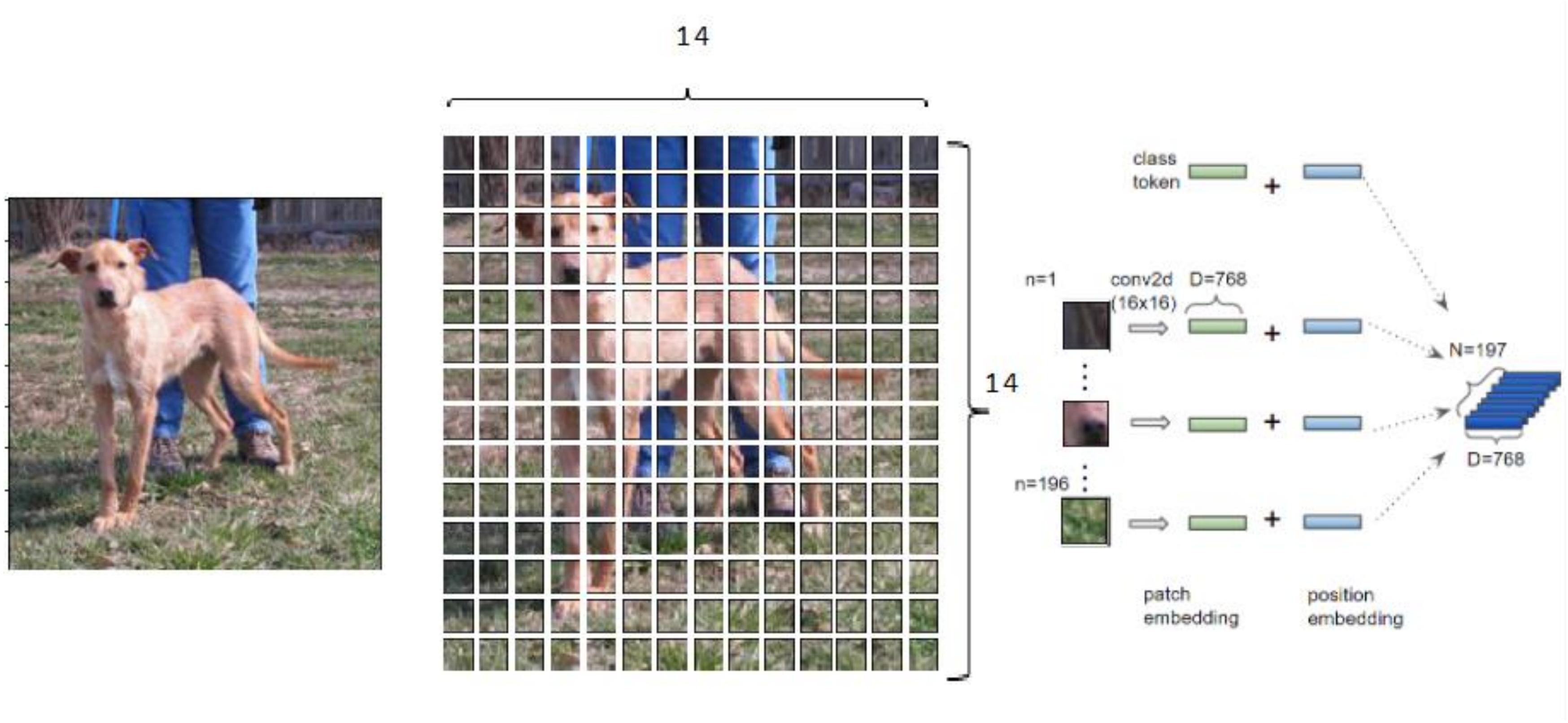
P

```
self.to_patch_embedding = nn.Sequential(
    Rearrange('b c (h p1) (w p2) -> b (h w) (p1 p2 c)', p1 = patch_height, p2 = patch_width),
    nn.LayerNorm(patch_dim),
    nn.Linear(patch_dim, dim),
    nn.LayerNorm(dim),
)
```

$x \in \mathcal{R}^{H \times W \times C}$

https://github.com/lucidrains/vit-pytorch/blob/main/vit_pytorch/vit.py

01. Vision Transformer



https://dkamatblog.home.blog/2021/08/05/vision-transformers-vit/

01. Vision Transformer

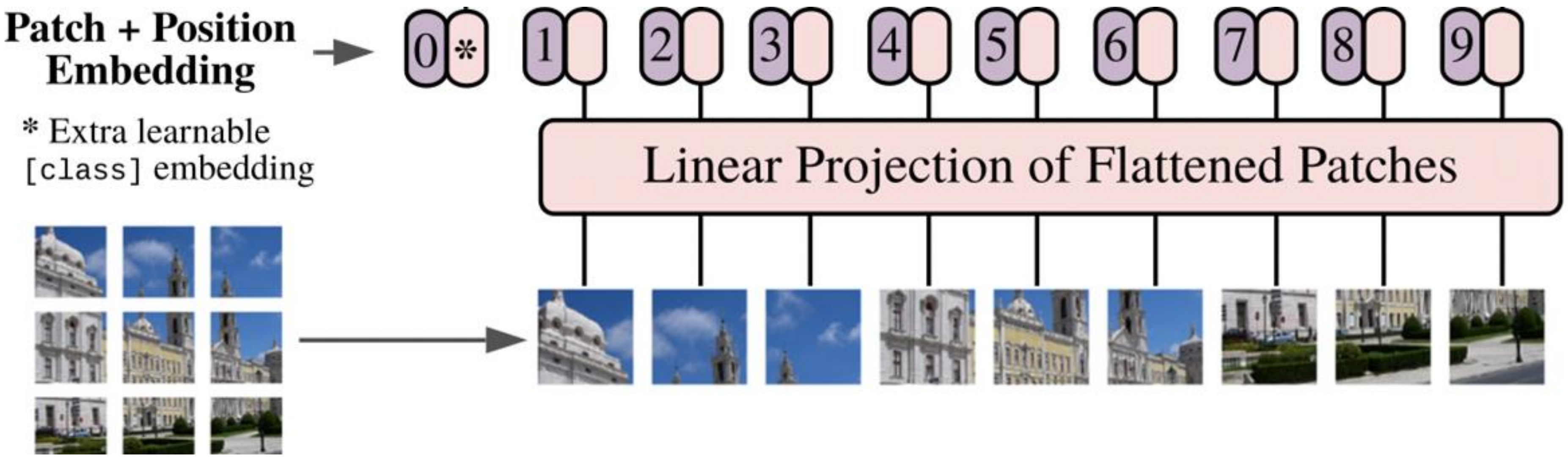
- Concatenate the class embedding, which is used to learn the representative characteristics of the entire image.

- And then add the position embedding that provides positional information.

$$\mathbf{E} \in \mathbb{R}^{(P^2 \cdot C) \times D}$$

$$\mathbf{E}_{pos} \in \mathbb{R}^{(N+1) \times D}$$

$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}};\, \mathbf{x}_p^1\mathbf{E};\, \mathbf{x}_p^2\mathbf{E};\, \cdots;\, \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos}$$



01. Vision Transformer

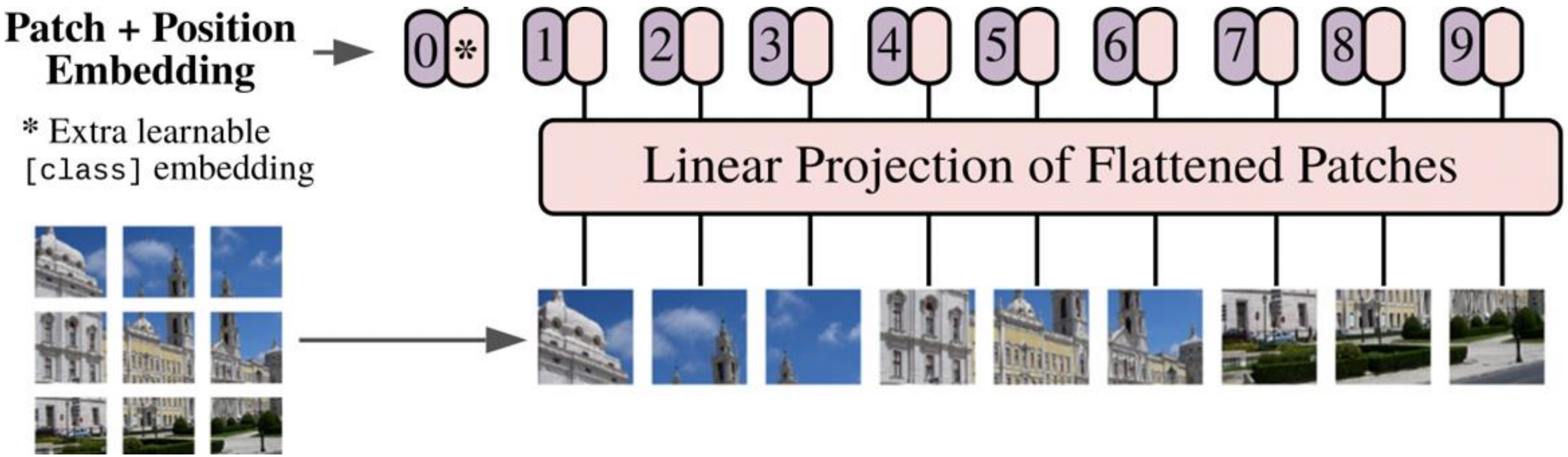
- Note that the class embedding and position embedding are also learnable.

```
self.pos_embedding = nn.Parameter(torch.randn(1, num_patches + 1, dim))
self.cls_token = nn.Parameter(torch.randn(1, 1, dim))
self.dropout = nn.Dropout(emb_dropout)
```

$$\mathbf{E} \in \mathbb{R}^{(P^2 \cdot C) \times D}$$

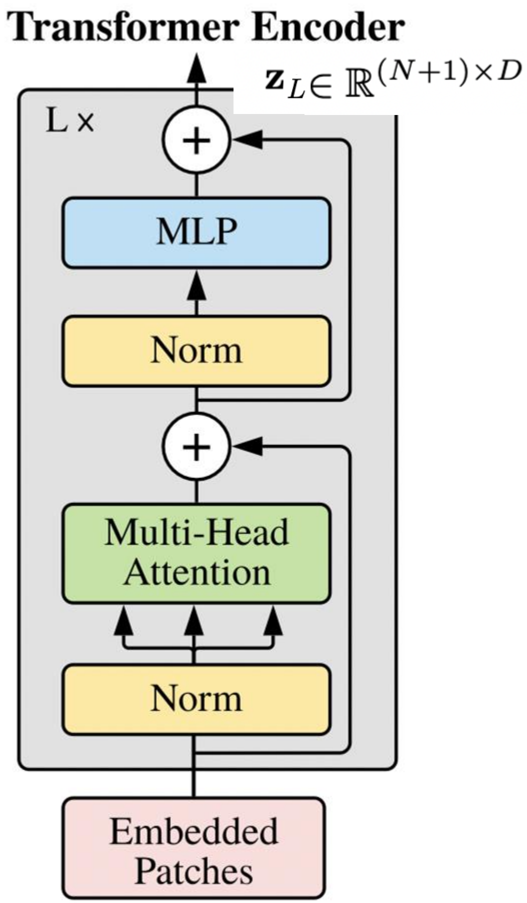
$$\mathbf{E}_{pos} \in \mathbb{R}^{(N+1) \times D}$$

$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}};\, \mathbf{x}_p^1\mathbf{E};\, \mathbf{x}_p^2\mathbf{E};\, \cdots;\, \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos}$$



https://github.com/lucidrains/vit-pytorch/blob/main/vit_pytorch/vit.py

01. Vision Transformer

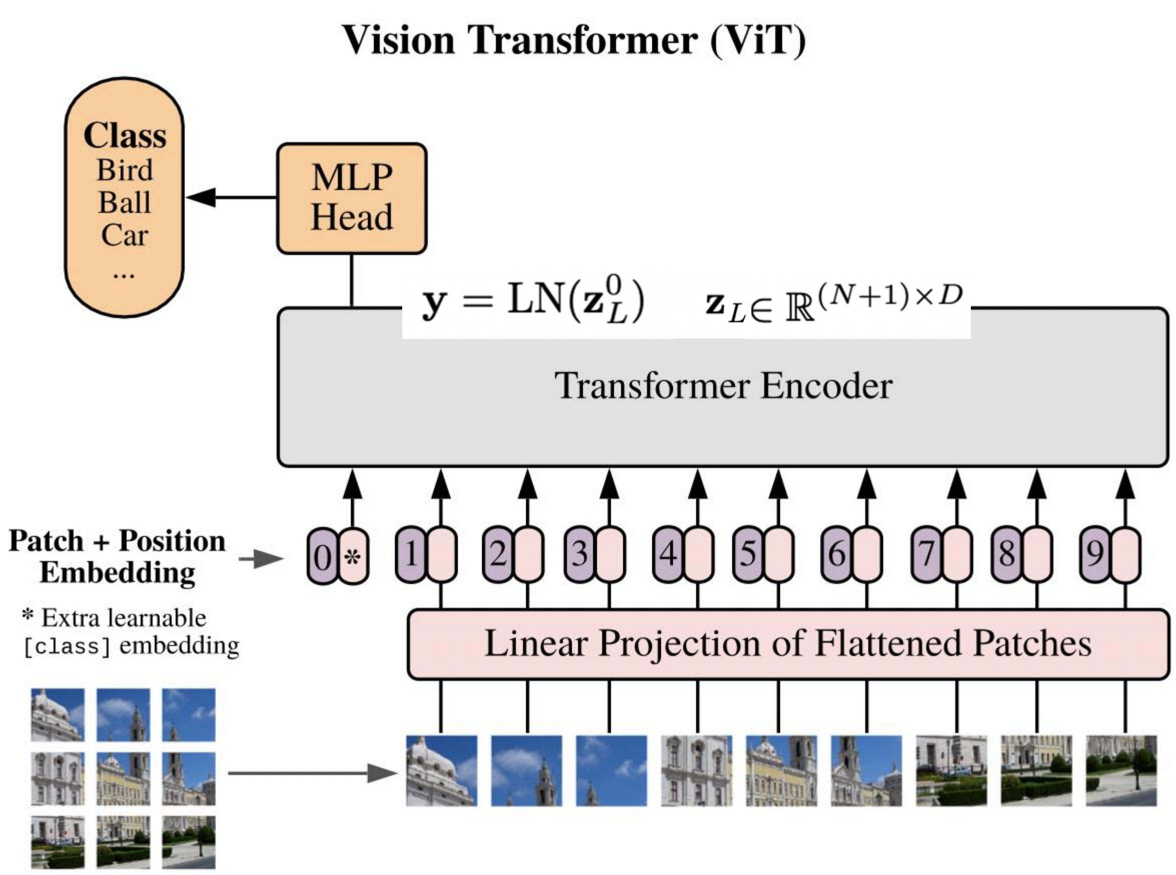


$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}};\, \mathbf{x}_p^1\mathbf{E};\, \mathbf{x}_p^2\mathbf{E};\cdots;\, \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos}, \qquad \mathbf{E} \in \mathbb{R}^{(P^2\cdot C)\times D},\ \mathbf{E}_{pos} \in \mathbb{R}^{(N+1)\times D}$$

$$\mathbf{z'}_\ell = \mathrm{MSA}(\mathrm{LN}(\mathbf{z}_{\ell-1})) + \mathbf{z}_{\ell-1}, \qquad \ell = 1 \ldots L$$

$$\mathbf{z}_\ell = \mathrm{MLP}(\mathrm{LN}(\mathbf{z'}_\ell)) + \mathbf{z'}_\ell, \qquad \ell = 1 \ldots L$$

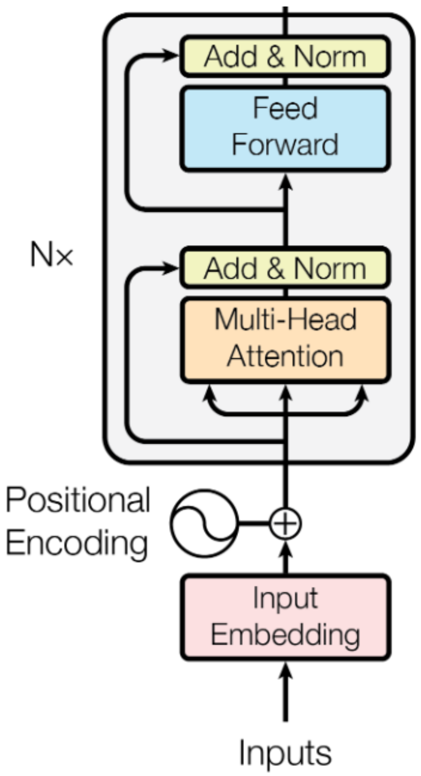
01. Vision Transformer



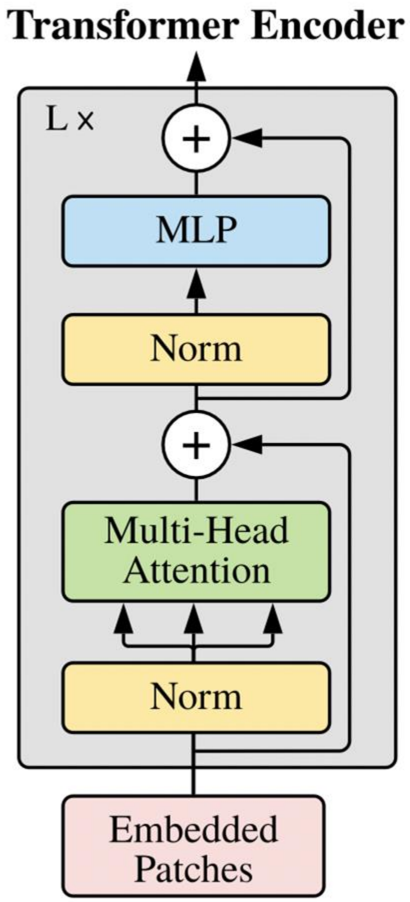
- We use only the transformer output token corresponding to the class embedding position for the classification task.

- If you are interested in using all tokens, refer to 'All Tokens Matter: Token Labeling for Training Better Vision Transformers' by Zihang Jiang (2021).

02. Comparison



Original Transformer Encoder



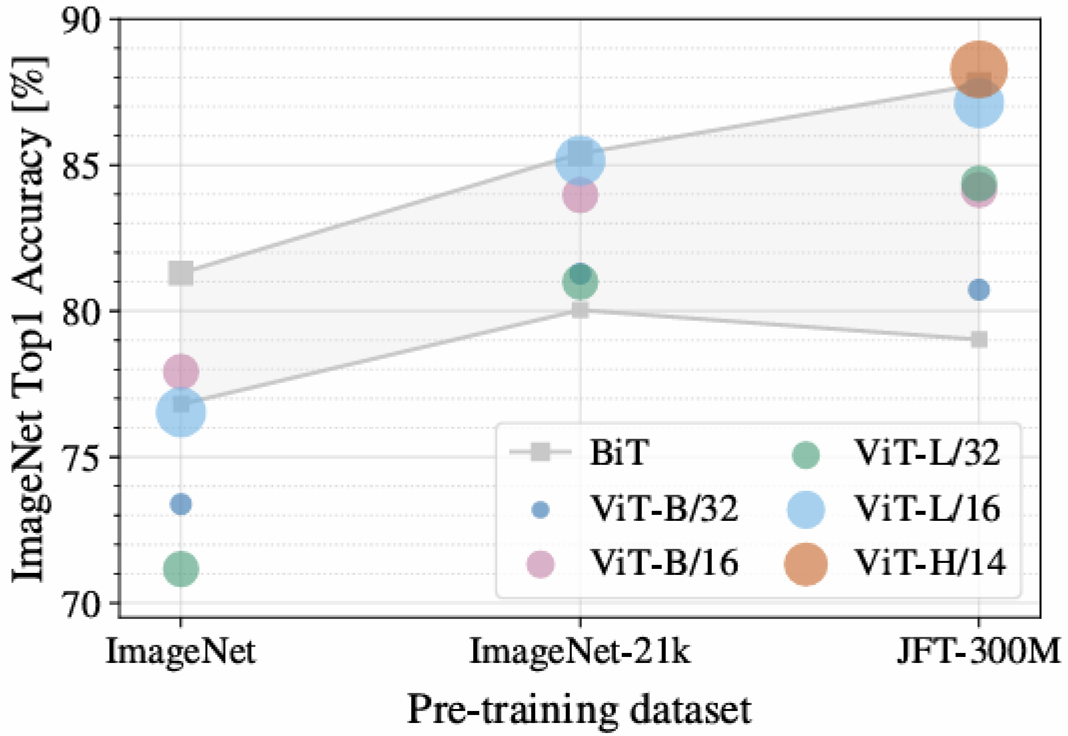
ViT Transformer Encoder

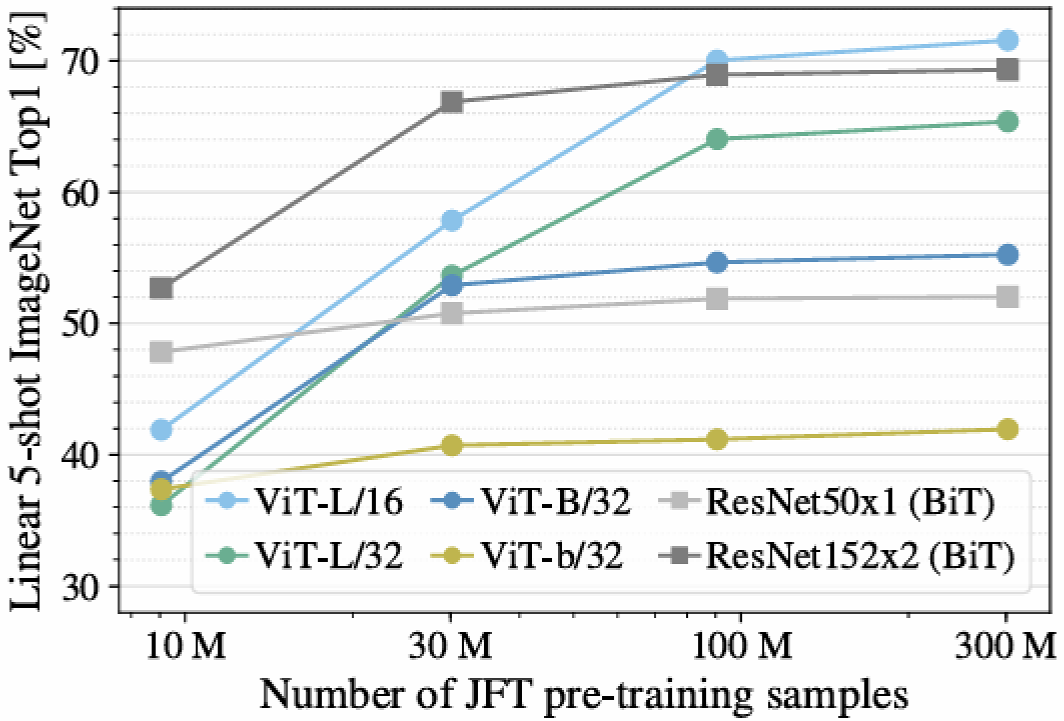
- While the original transformer applies normalization after the attention block, ViT applies normalization before the attention block.
- While the original transformer uses the ReLU function in the MLP process, ViT uses GeLU.
- In the original transformer, positional embeddings are fixed vectors, but in ViT, they are learnable parameters.

03. Experiment

	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L(JFT) (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	**88.55** $\pm$ 0.04	87.76 $\pm$ 0.03	85.30 $\pm$ 0.02	87.54 $\pm$ 0.02	88.4/88.5*
ImageNet ReaL	**90.72** $\pm$ 0.05	90.54 $\pm$ 0.03	88.62 $\pm$ 0.05	90.54	90.55
CIFAR-10	**99.50** $\pm$ 0.06	99.42 $\pm$ 0.03	99.15 $\pm$ 0.03	99.37 $\pm$ 0.06	–
CIFAR-100	**94.55** $\pm$ 0.04	93.90 $\pm$ 0.05	93.25 $\pm$ 0.05	93.51 $\pm$ 0.08	–
Oxford-IIIT Pets	**97.56** $\pm$ 0.03	97.32 $\pm$ 0.11	94.67 $\pm$ 0.15	96.62 $\pm$ 0.23	–
Oxford Flowers-102	99.68 $\pm$ 0.02	**99.74** $\pm$ 0.00	99.61 $\pm$ 0.02	99.63 $\pm$ 0.03	–
VTAB (19 tasks)	**77.63** $\pm$ 0.23	76.28 $\pm$ 0.46	72.72 $\pm$ 0.21	76.29 $\pm$ 1.70	–
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

03. Experiment





04. Self-Supervised Learning

- ViT benefits from being trained on large datasets and is used as a pre-trained model for transfer learning.
- However, in reality, do large datasets always have labels? No!!
- In such cases, how can we train ViT?

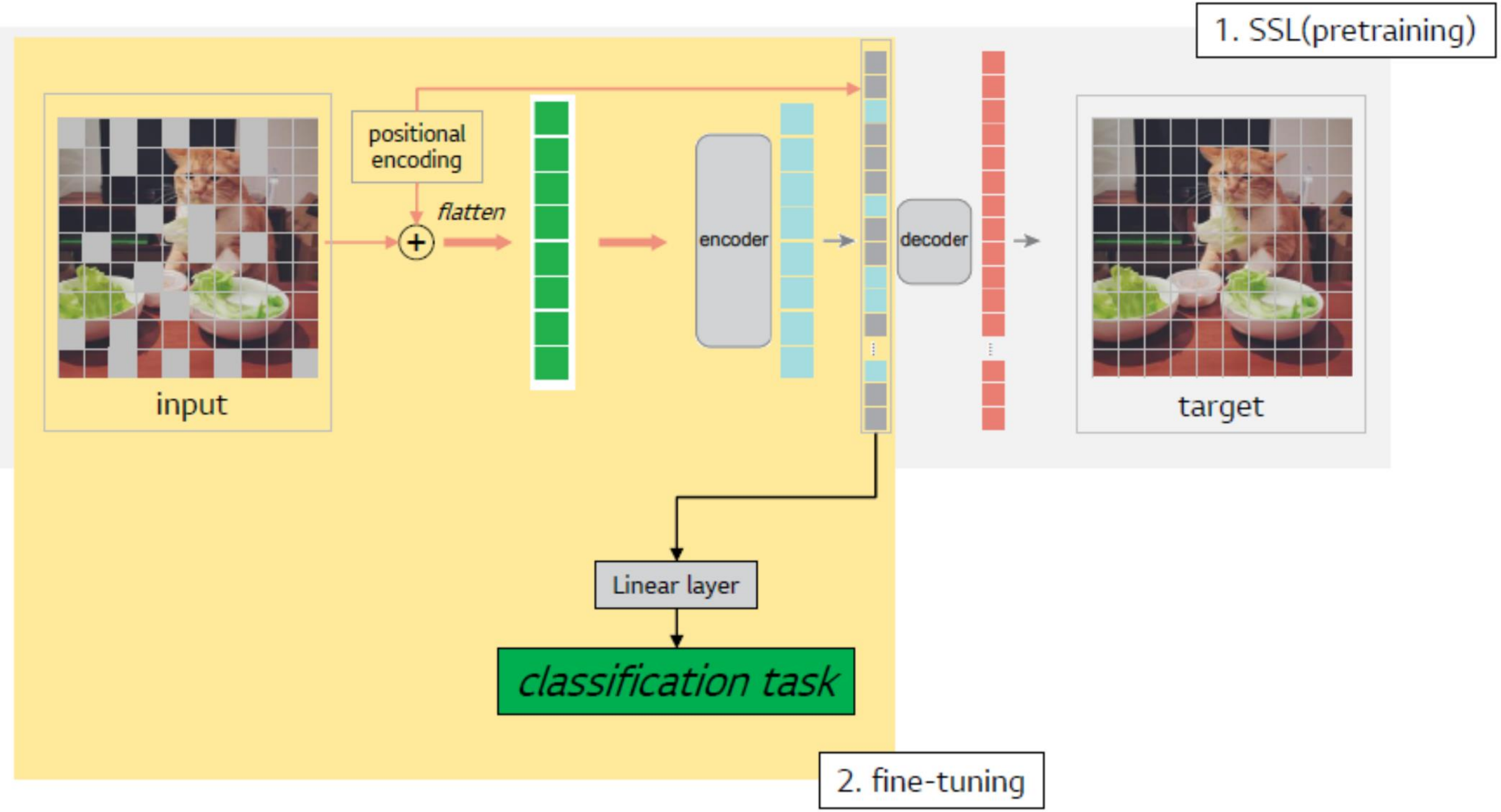
Masked Autoencoders Are Scalable Vision Learners

Kaiming He[*,†] Xinlei Chen[*] Saining Xie Yanghao Li Piotr Dollár Ross Girshick

[*]equal technical contribution [†]project lead

Facebook AI Research (FAIR)

04. Self-Supervised Learning



https://developers-shack.tistory.com/13

04. Self-Supervised Learning

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

WHAT DO SELF-SUPERVISED VISION TRANSFORMERS LEARN?

Namuk Park[1*] **Wonjae Kim**[2] **Byeongho Heo**[2] **Taekyung Kim**[2] **Sangdoo Yun**[2]
[1]Prescient Design, Genentech [2]NAVER AI Lab
park.namuk@gene.com {wonjae.kim,bh.heo,taekyung.k,sangdoo.yun}@navercorp.com

End