www.aiengineer.tw

AI Study Session

An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale (*proposed by Dosovitskiy et al.*)

Reporter: Hung-Ming, Lin

Date: 2022/04/06

Outline

1. Introduction
2. Related Work
3. Method
4. Experiment
5. Conclusion
6. Code Demo
7. Reference

1. Introduction (1/2)

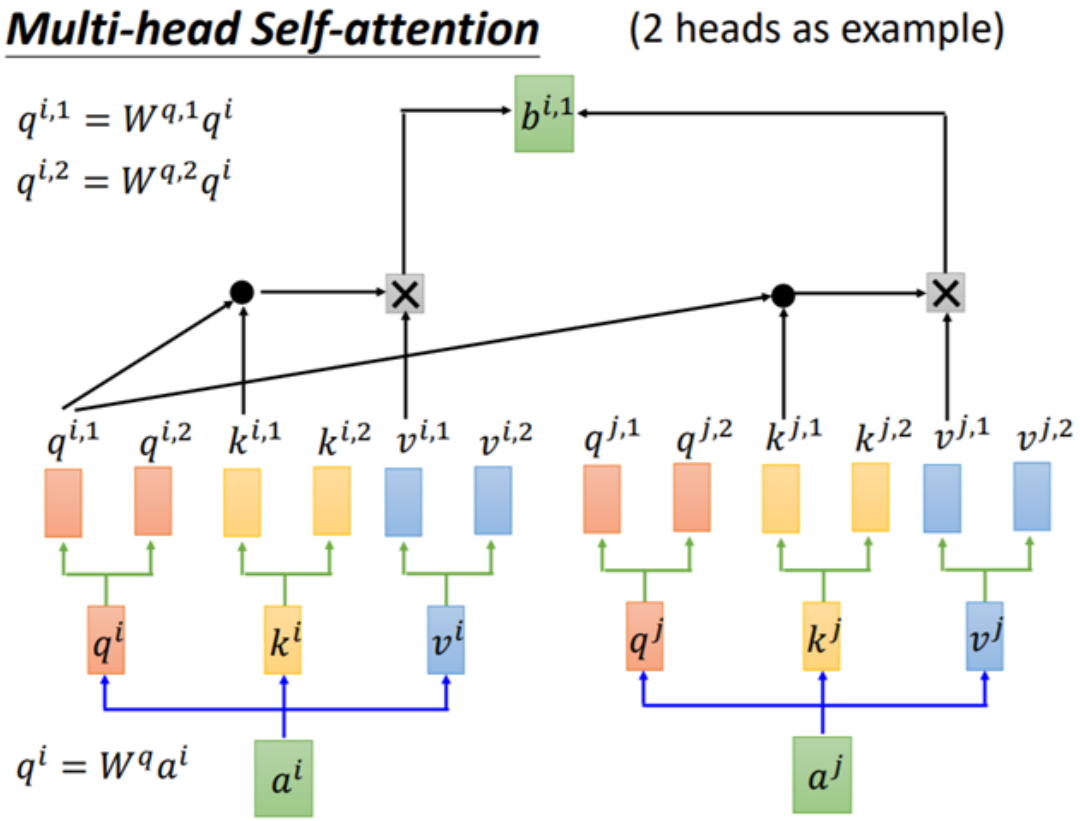
1. Difference between Language and Vision

	Language	Vision
1. Architectures	Tokens and positional embeddings	Convolution
2. Information density	Human-generated signals, Highly semantic and information-dense.	Natural signals, Heavy spatial redundancy
3. The autoencoder's decoder	missing *words* that contain rich semantic information	reconstructs *pixels* of a lower semantic level

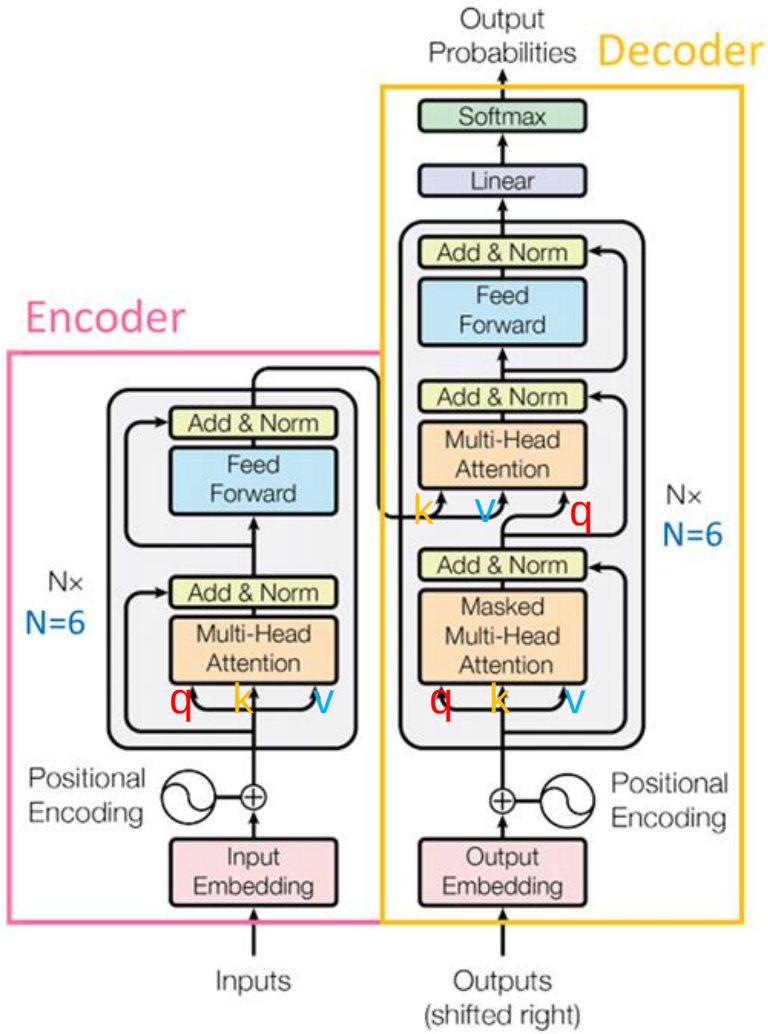
Source: He et al., (2021)
<Masked Autoencoders Are Scalable Vision Learners> https://arxiv.org/abs/2111.06377

1. Introduction (2/2)

2. Transformer



Source: 李宏毅 (2021)
<Transformer & BERT PPT>

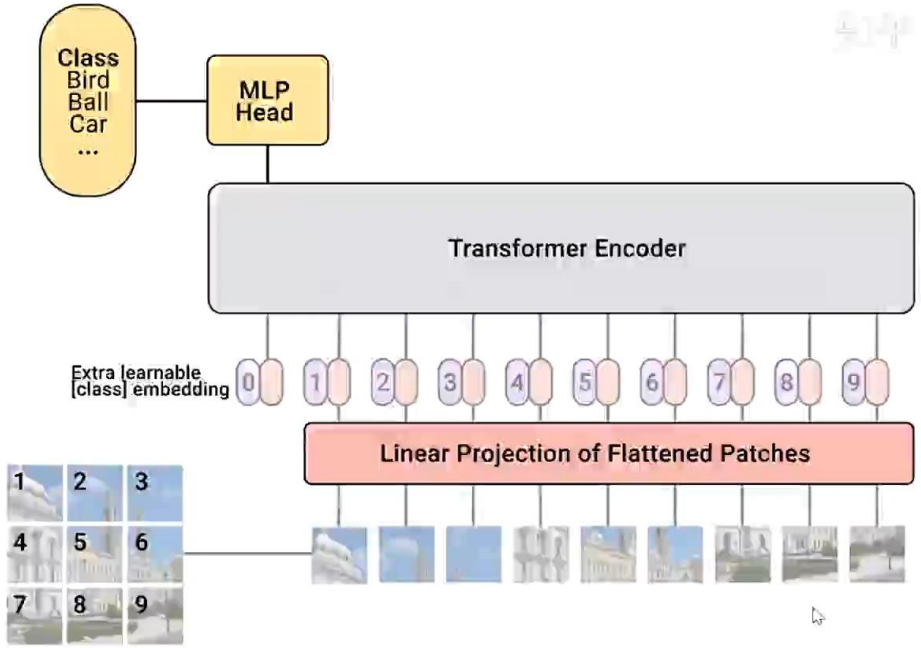


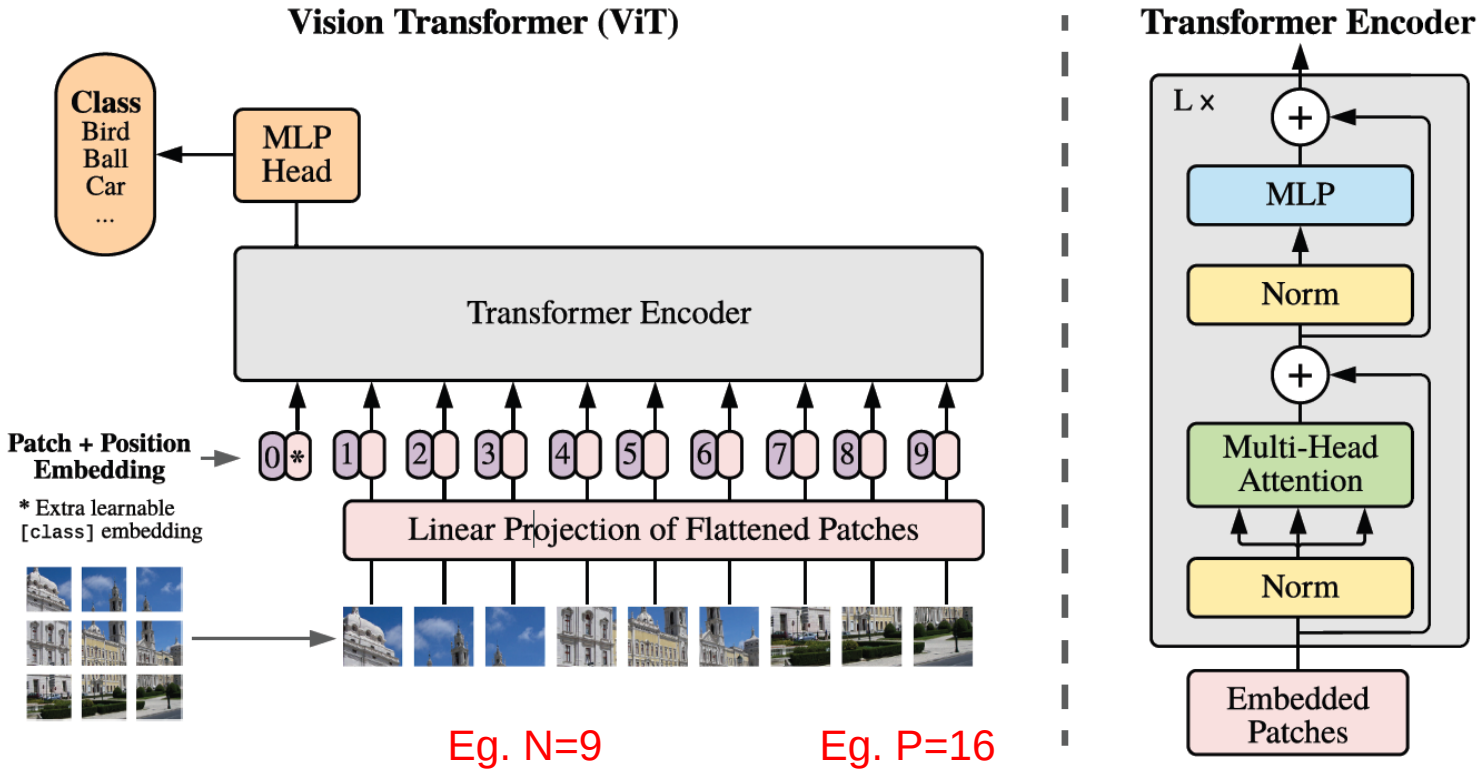
Source: Vaswani et al., (2017)
<Attention is all you need> https://arxiv.org/abs/1706.03762

2. Related Work

Authors	Paper	Method
1. Parmer et al. (2018)	**Image transformer** https://arxiv.org/abs/1802.05751	Applied the self-attention only in local neighborhoods for each query pixel.
2. Child et al. (2019)	**Generating long sequences with sparse Transformers** https://arxiv.org/abs/1904.10509	Employ scalable approximations to global self-attention in order to be applicable to images.
3. Weissenborn et al. (2019)	**Scaling autoregressive video models** https://arxiv.org/abs/1906.02634	Apply self-attention in blocks of varying sizes in the extreme case only along individual axes.
4. Cordonnier et al. (2020)	**On the relationship between self-attention and convolutional layers** https://arxiv.org/abs/1911.03584	Extracts patches of size 2x2 from the input image and applies full self-attention on top
5. Chen et al. (2020)	**Generative pretraining from pixels** http://proceedings.mlr.press/v119/chen20s.html	Image GPT (iGPT): applies Transformers to image pixels after reducing image resolution and color space
6. Dosovitskiy et al. (2020)	**An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale** https://arxiv.org/abs/2010.11929	ViT (discussed on next page)

3. Method





Eg. N=9 Eg. P=16

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

$$\mathbf{z}'_\ell = \text{MSA}(\text{LN}(\mathbf{z}_{\ell-1})) + \mathbf{z}_{\ell-1}, \quad \ell = 1 \ldots L \tag{2}$$

$$\mathbf{z}_\ell = \text{MLP}(\text{LN}(\mathbf{z}'_\ell)) + \mathbf{z}'_\ell, \quad \ell = 1 \ldots L \tag{3}$$

$$\mathbf{y} = \text{LN}(\mathbf{z}_L^0) \tag{4}$$

Source:科技猛兽 (2021)
<Vision Transformer 超详细解读 (原理分析+代码解读) (二)>

4. Experiment

Pre-training Datasets：

- ILSVRC-2012 ImageNet dataset：1000 classes、**1.3M** images
- ImageNet-21k：21k classes, **14M** images
- JFT：18k classes, **303M** High Resolution Images

Transfer Models on Datasets：

- CIFAR-10/100
- Oxford-IIIT Pets
- Oxford Flowers-102
- VTAB

For these datasets, pre-processing follows Kolesnikov et al. (2020)

Big Transfer (BiT): General Visual Representation Learning

https://arxiv.org/abs/1912.11370

4. Experiment

Model Variants：

Model	Layers	Hidden size D	MLP size	Heads	Params
ViT-Base	12	768	3072	12	86M
ViT-Large	24	1024	4096	16	307M
ViT-Huge	32	1280	5120	16	632M

Eg. ViT-L/16 means the “Large” variant with 16x16 input patch size.

Training & Fine-tuning：

- training: Adam ($\beta_1 = 0.9,\ \beta_2 = 0.999$), batch size = 4096
- fine-tuning: SGD with momentum, batch size = 512

Metrics：

- few-shot or fine-tuning accuracy

4. Experiment

Comparison with SOTA on popular image classification benchmarks

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

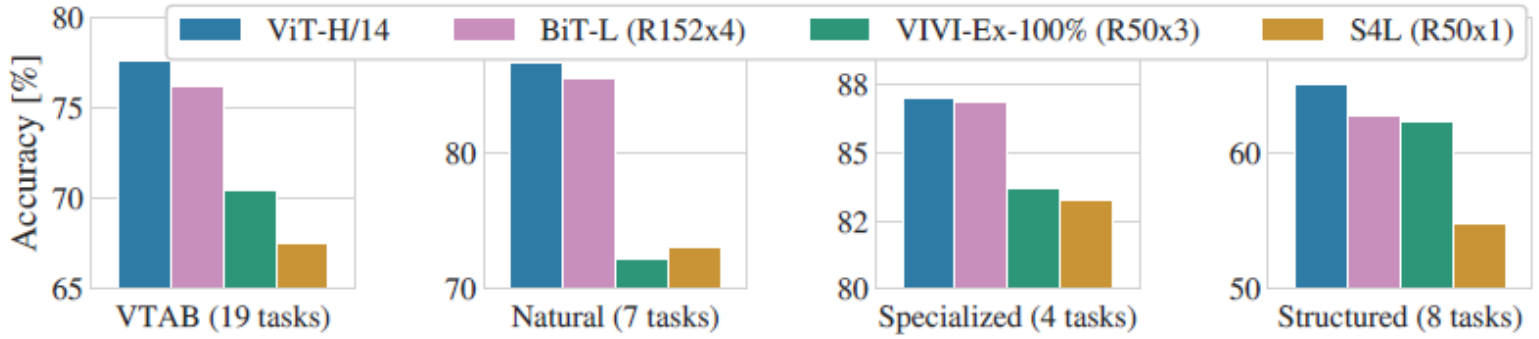


Figure 2: Breakdown of VTAB performance in *Natural*, *Specialized*, and *Structured* task groups.

4. Experiment

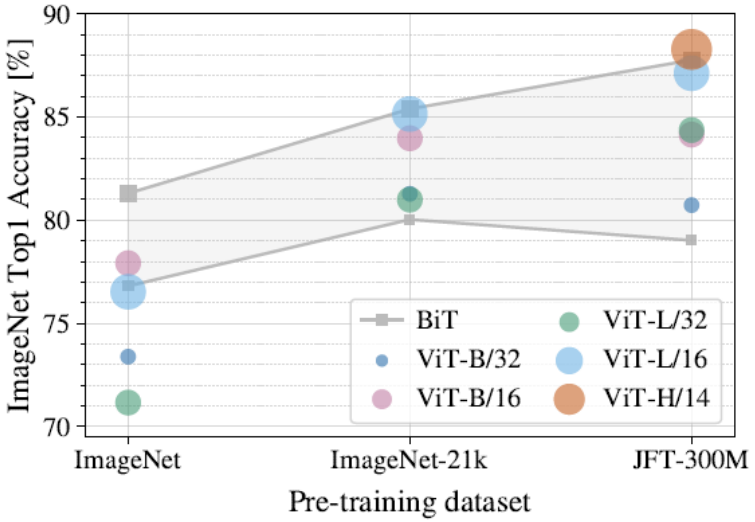


Figure 3: Transfer to ImageNet. While large ViT models perform worse than BiT ResNets (shaded area) when pre-trained on small datasets, they shine when pre-trained on larger datasets. Similarly, larger ViT variants overtake smaller ones as the dataset grows.

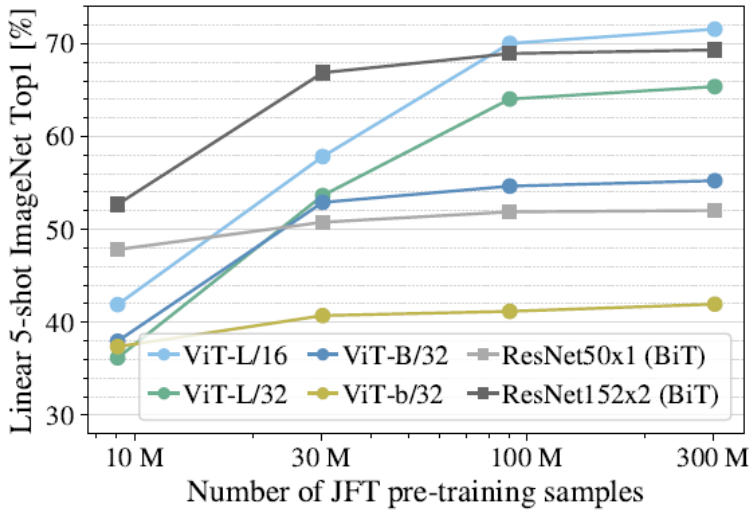


Figure 4: Linear few-shot evaluation on ImageNet versus pre-training size. ResNets perform better with smaller pre-training datasets but plateau sooner than ViT, which performs better with larger pre-training. ViT-b is ViT-B with all hidden dimensions halved.

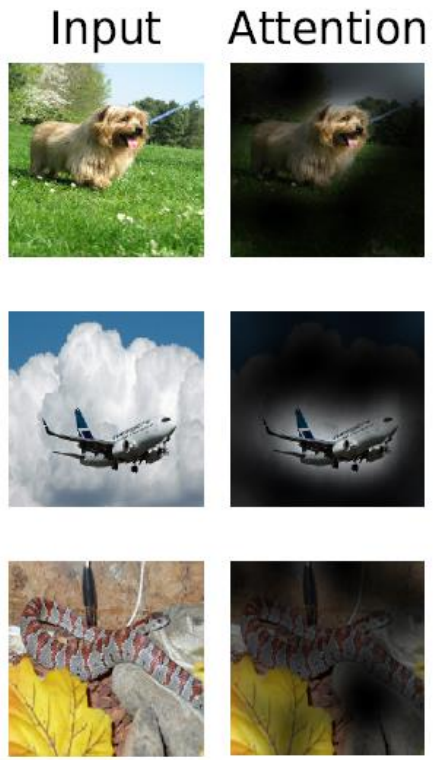


Figure 6: Representative examples of attention from the output token to the input space. See Appendix D.7 for details.

5. Conclusion

1. **ViT** interprets an image as **a sequence of patches** and process it by a standard **Transformer encoder** as used in NLP.
2. This simple, yet scalable, strategy works surprisingly well when coupled with **pre-training** on **large datasets**.
3. Challenge: **objection detection** and **segmentation**. continue exploring **self-supervised pre-training** methods.

6. Code Demo

```python
class ViT(nn.Module):
    def __init__(self, *, image_size, patch_size, num_classes, dim, transformer, pool = 'cls', channels = 3):
        super().__init__()
        image_size_h, image_size_w = pair(image_size)
        assert image_size_h % patch_size == 0 and image_size_w % patch_size == 0, 'image dimensions must be divisible by the patch size'
        assert pool in {'cls', 'mean'}, 'pool type must be either cls (cls token) or mean (mean pooling)'
        num_patches = (image_size_h // patch_size) * (image_size_w // patch_size)
        patch_dim = channels * patch_size ** 2

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

        self.pos_embedding = nn.Parameter(torch.randn(1, num_patches + 1, dim))
        self.cls_token = nn.Parameter(torch.randn(1, 1, dim))
        self.transformer = transformer

        self.pool = pool
        self.to_latent = nn.Identity()

        self.mlp_head = nn.Sequential(
            nn.LayerNorm(dim),
            nn.Linear(dim, num_classes)
        )
```

```python
def forward(self, img):
    x = self.to_patch_embedding(img)
    b, n, _ = x.shape

    cls_tokens = repeat(self.cls_token, '() n d -> b n d', b = b)
    x = torch.cat((cls_tokens, x), dim=1)
    x += self.pos_embedding[:, :(n + 1)]
    x = self.transformer(x)

    x = x.mean(dim = 1) if self.pool == 'mean' else x[:, 0]

    x = self.to_latent(x)
    return self.mlp_head(x)
```

6. Code Demo

Install

```
$ pip install vit-pytorch
```

Usage

```
import torch
from vit_pytorch import ViT

v = ViT(
    image_size = 256,
    patch_size = 32,
    num_classes = 1000,
    dim = 1024,
    depth = 6,
    heads = 16,
    mlp_dim = 2048,
    dropout = 0.1,
    emb_dropout = 0.1
)

img = torch.randn(1, 3, 256, 256)

preds = v(img) # (1, 1000)
```

7. Reference

1. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., & Gelly, S. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.
2. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., & Girshick, R. (2021). Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*.
3. Kolesnikov, A., Beyer, L., Zhai, X., Puigcerver, J., Yung, J., Gelly, S., & Houlsby, N. (2020). Big transfer (bit): General visual representation learning. *arXiv preprint arXiv:1912.11370*
4. Yannic Kilcher (2020). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale (Paper Explained). Retrieved from https://youtu.be/TrdevFK_am4
5. 科技猛兽 (2021). Vision Transformer 超详细解读 (原理分析+代码解读) (二) Retrieved from https://zhuanlan.zhihu.com/p/342261872
6. AppleHank (2020). ViT — Vision Transformer : Convolution is dead, long live Transformers! Retrieved from https://applehank.medium.com/vit-vision-transformer-convolution-is-dead-long-live-the-self-attention-bbced72a8487
7. https://github.com/google-research/vision_transformer/
8. https://github.com/lucidrains/vit-pytorch