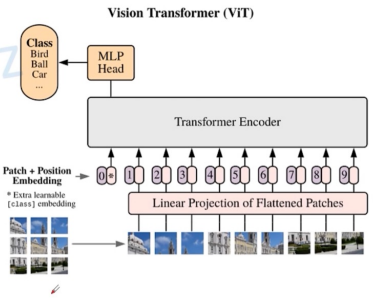
Vision Transformer

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

2020 CVPR



原文链接：https://arxiv.org/abs/2010.11929
博文链接：https://blog.csdn.net/qq_37541097/article/details/118242600

(1) paper

(2) CVPR2020

(3) Architecture

	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

Table 2: Comparison with state of the art on popular image classification benchmarks. We report mean and standard deviation of the accuracies, averaged over three fine-tuning runs. Vision Transformer models pre-trained on the JFT-300M dataset outperform ResNet-based baselines on all datasets, while taking substantially less computational resources to pre-train. ViT pre-trained on the smaller public ImageNet-21k dataset performs well too. *Slightly improved 88.5% result reported in Touvron et al. (2020).

(1) 卖结果

刷很多 dataset 88.55 { JFT pretrained (Google dataset)
ImageNet fine-tuned

(2) ViT-H/14 { Huge
14 patch size

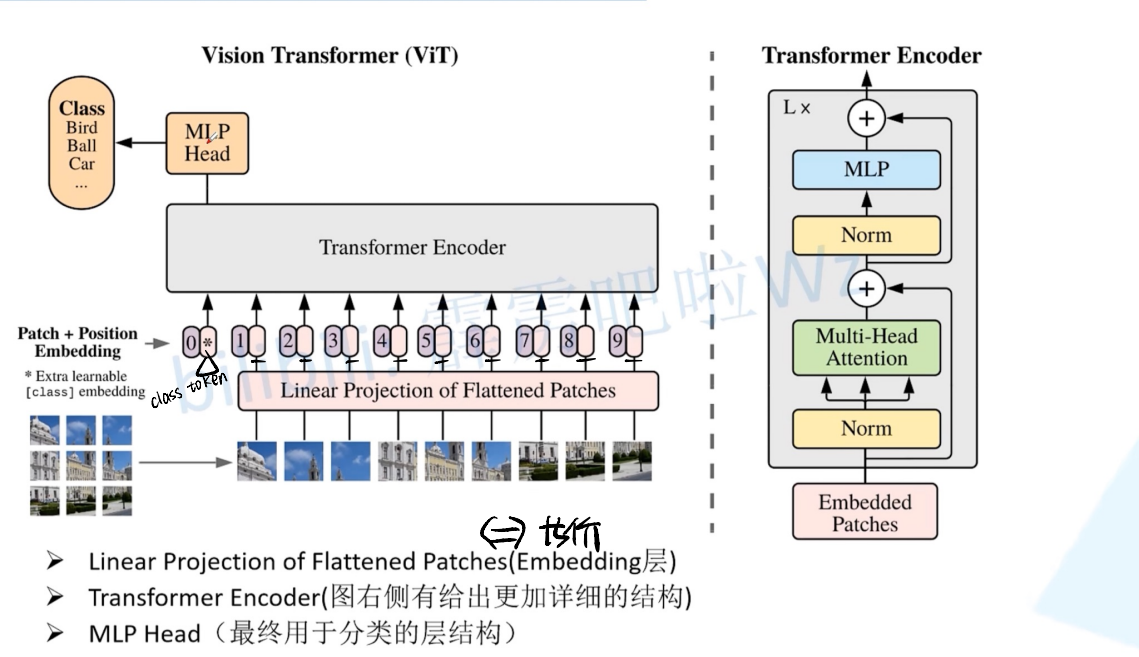
(3) ImageNet default ImageNet 1k

(4) Ours-I21k { I21k: ImageNet 21k pretrained
ImageNet 1k: Transfer Learning 85.30 (accuracy)

model: ViT-L/16: Large, 16x16 patch size

Vision Transformer

- ViT (纯 Transformer 模型)
- Hybrid (传统 CNN & Transformer 混合模型)



⇔ 也称

- Linear Projection of Flattened Patches(Embedding层)
- Transformer Encoder(图右侧有给出更加详细的结构)
- MLP Head（最终用于分类的层结构）

(1) Input Image ⟶ patches

eg: ViT-L/16 patch size 16×16

(2) patches —输入→ embedding层 —得到→ vector/向量 = token

deal

接着 在这一系列token前面加个class token ——— 这些token dimension相同

then: position embedding ie. 0, 1, 2, ……

finally we have

① position embedding
② patch ⟶ token
③ class token

(3) Transformer Encoder | Lx

重复堆叠 L次

(4) Transformer Encoder 输入n个变量 输出n个变量

but our task: classification

只关 extract class token 对应的 output

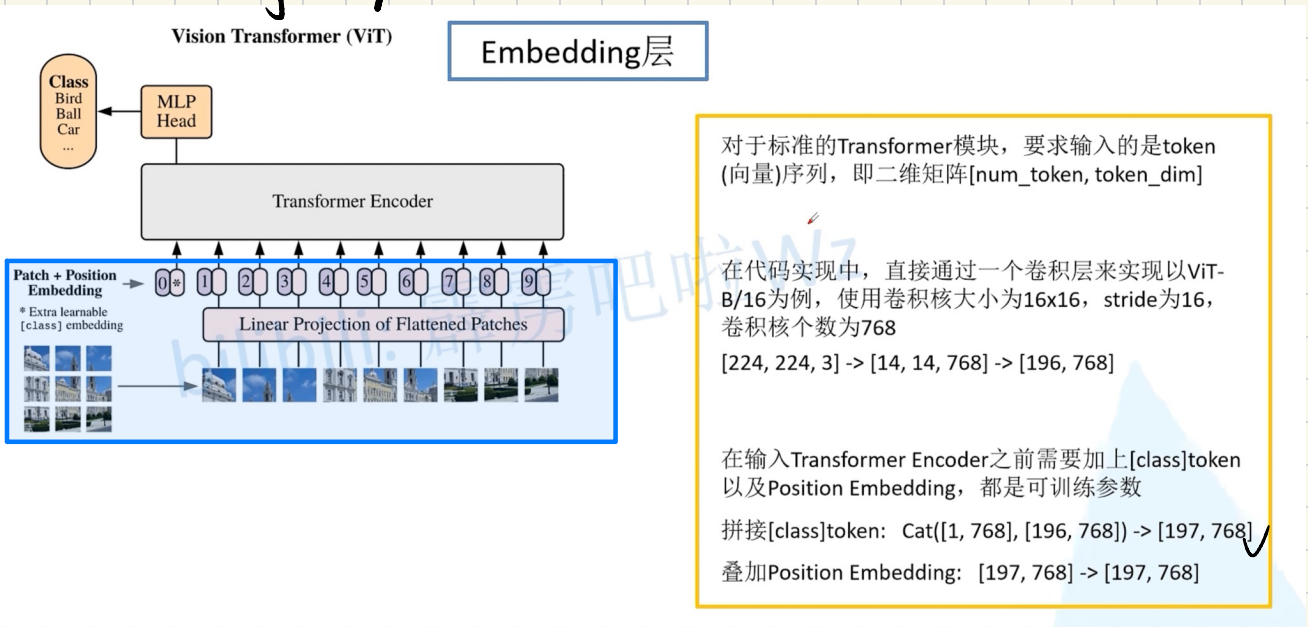
In the end, use MLP head —得到→ 分类结果

(5) ViT 可以分成3个部分

- Linear projection of flatten patches ⇔ Embedding层
- Transformer Encoder
- MLP Head 用于分类的层结构

模块细讲

Embedding Layer



(1) Transformer 的 input [num_token, token_dim]

(2) Input Image 224×224×3

ViT B/16 16 = patch_size

- 224×224×3 —conv (k=16 s=16 p=0, #kernel=768)→ 14×14×768 —H,W flatten→ 196×768

$$\frac{h-k+s+2p}{s} = \frac{224-16+16+0}{16} = 14$$

- 加1个类别的 token [1,768] class —concat→ 197×768
- 加上 position embedding (shape [197,768])

position embedding + [class] token shape不变 [197,768]

- now，对比 position embedding

Position Embedding

论文实验 对 pos_emb 的对比

Pos. Emb.	Default/Stem	Every Layer	Every Layer-Shared
No Pos. Emb.	0.61382	N/A	N/A
1-D Pos. Emb.	0.64206	0.63964	0.64292
2-D Pos. Emb.	0.64001	0.64046	0.64022
相对位置编码 Rel. Pos. Emb.	0.64032	N/A	N/A

↑ 3%

Table 8: Results of the ablation study on positional embeddings with ViT-B/16 model evaluated on ImageNet 5-shot linear.

☆ the differences in how to encode spatial information is less important

we can see

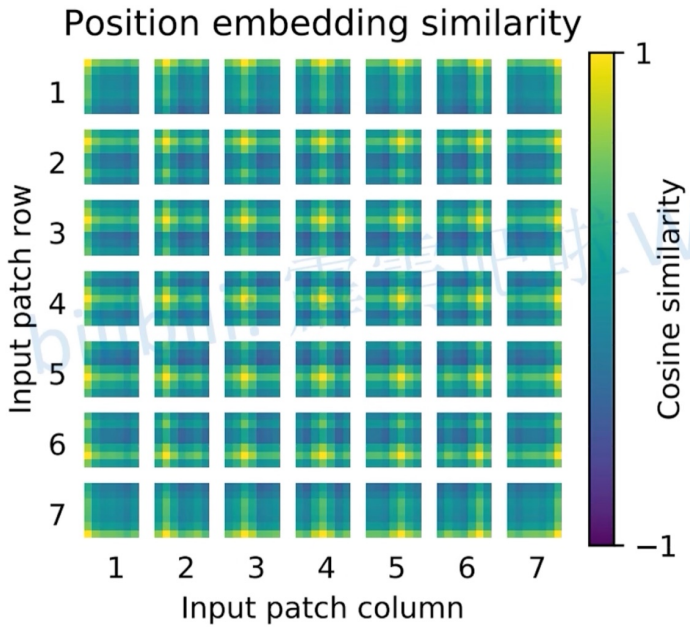
(1) 1D pos emb 比 No pos emb (不用位置编码) ↑ 3%

但是用 1-D pos_emb (or) 2-D pos emb (or) rel pos emb 区别不大

(2) 如何编码空间信息没那么重要

but need 编码

- About position embedding



closer patches tend to have more similar position embeddings

每一个 token 上都加 position embedding

针对每个 patches 的 位置编码 是一个向量

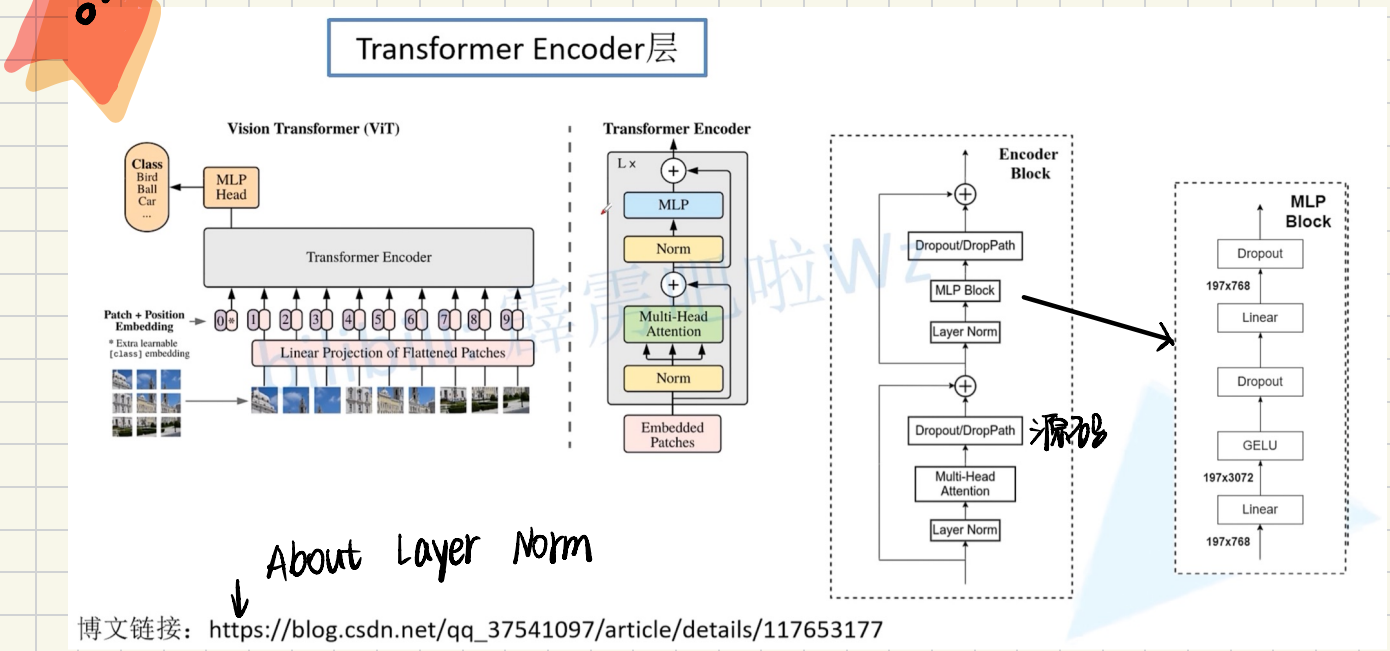
什么意思？不明白 token 与 patch 什么关系？

patch 是什么？ 224×224×3 ——(k=16 s=16 p=0, #k=768)——> 14×14×768 ——> 196×768

why 768?

Me: patch 更像是表达 16×16 这么大图像块的语义信息

Input patch row ? Input patch column?



About Layer Norm

博文链接：https://blog.csdn.net/qq_37541097/article/details/117653177

(1) Transformer Encoder 就是将 Encoder Block 重复堆叠 L次

(2) details: encoder block

① Norm 层 = Layer Normalization △ | (看blog) 什么是 Layer Normalization

② Multi-Head Attention

③ Dropout 层 { paper 没画出来 / code 源码有

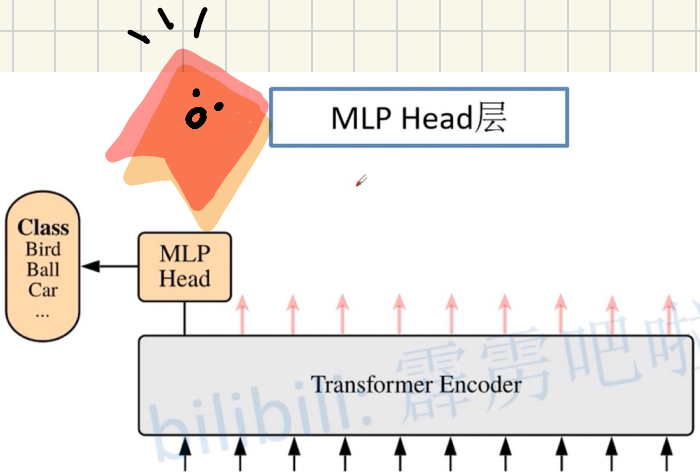
DropPath (大神实现用的) (差异不大)

④ skip connection

⑤ MLP Block

(1) 1st 全连接层 197×768 ——Linear——> 197×3072 (4倍)

(2) 2nd 全连接层 197×3072 ——Linear——> 197×768 (还原)



注意，在Transformer Encoder前有个Dropout层，后有一个Layer Norm

训练ImageNet21K时是由Linear+tanh激活函数+Linear

但是迁移到ImageNet1K上或者你自己的数据上时，只有一个Linear

(1) Note:

Transformer
- ↑ Layer Norm层
- ↑ Dropout层

paper中未给置
code

(2) MLP Head 3个模块组成

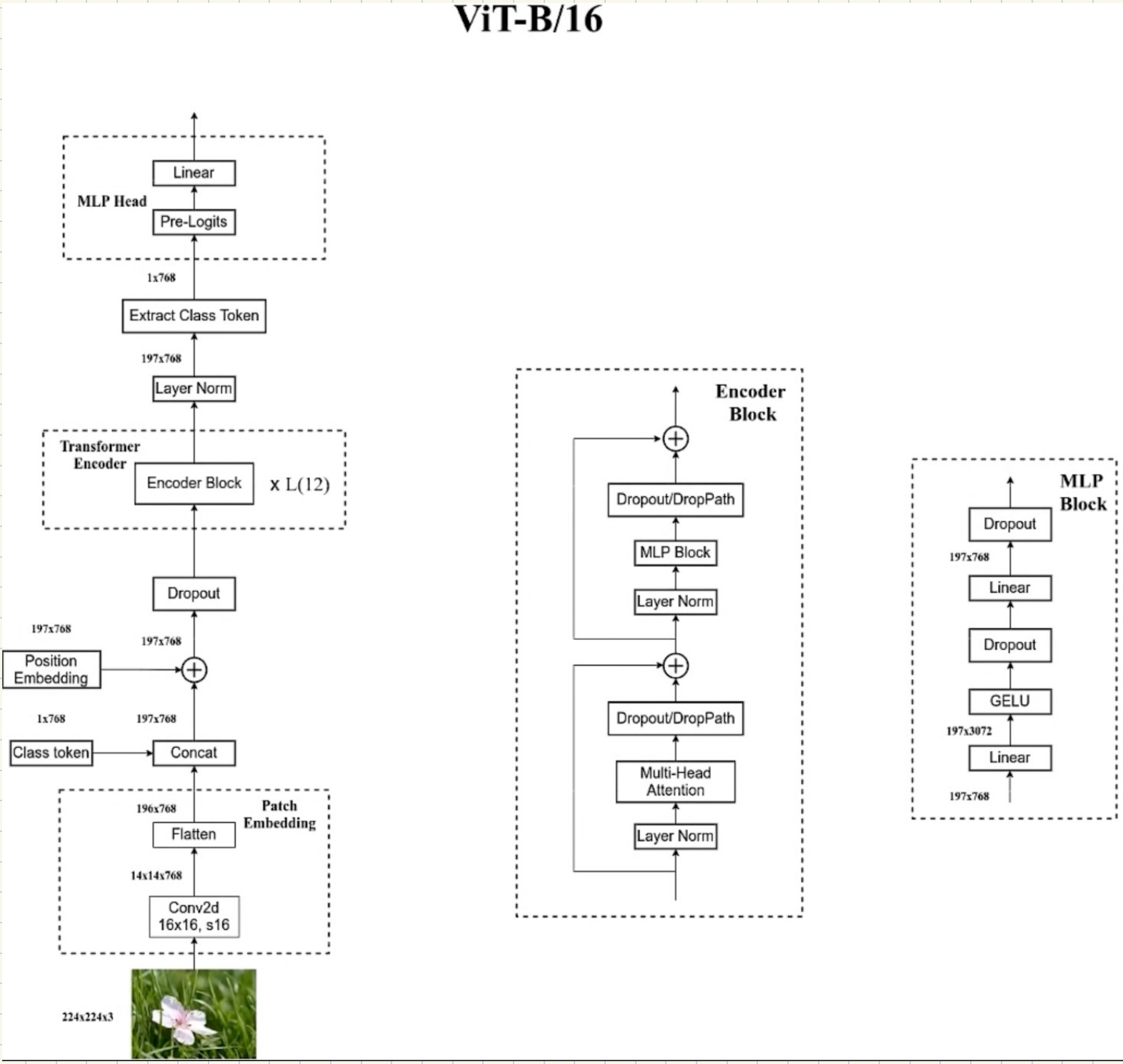
① Linear 全连接层
② tanh activation function
③ Linear 全连接层

ImageNet 21K

but ImageNet 1K
MLP Head 只有1个 Linear (全连接层)

若识别类别 则 接一个 softmax

大佬手绘 ViT-B/16 的 Architecture　　B: Base　patch_size 16



(1) Input image 224x224x3 —conv (k=16 s=16 p=0, #k=768)→ 14x14x768 —flatten→ 196x768 (14x14=196)

patch embedding

(2) concat { patch embedding 196x768 ; class token 1x768 } → 197x768

(3) add ⊕ { position embedding (参数可训练) 197x768 ; concat 197x768 } → shape不变

(4) Dropout 层

(5) Encoder Block X L(12)

重复堆叠12次 构成 Encoder Transformer

(6) Layer Norm → 197×768

(7) extract class token → 1×768

(只提取 class token 的输出)

(8) MLP Head 得到最终输出

若在ImageNet 21K上 pretrained 的话，则 Pre-Logits

= Linear + tanh + Linear

若在ImageNet 1K 则 Pre-Logits = Linear 全连接层

以上 ViT-B/16 model

Topic: ViT 模型的一些参数

Model	Patch Size	Layers	Hidden Size D	MLP size	Heads	Params
ViT-Base	16x16	12	768	3072	12	86M
ViT-Large	16x16	24	1024	4096	16	307M
ViT-Huge	14x14	32	1280	5120	16	632M

- Layers是Transformer Encoder中重复堆叠Encoder Block的次数
- Hidden Size是通过Embedding层后每个token的dim（向量的长度） ViT-B/16 768 有eg
- MLP size是Transformer Encoder中MLP Block第一个全连接的节点个数（是Hidden Size的四倍）
- Heads代表Transformer中Multi-Head Attention的heads数

meaning (1) ViT-B/16 ViT-L/16 ViT-H/14

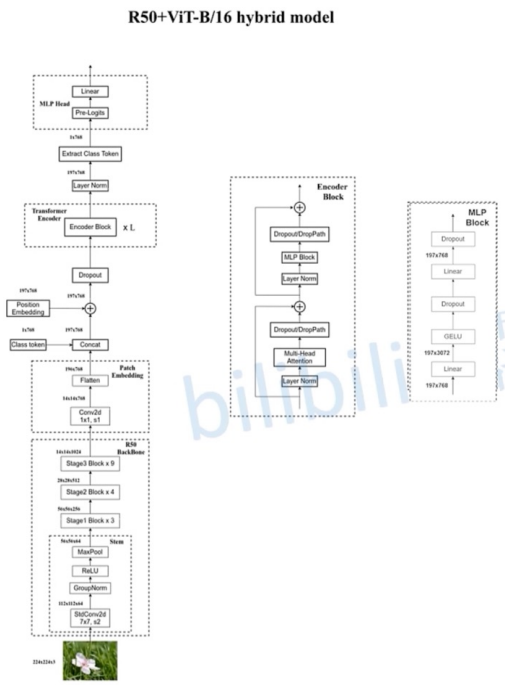
(2) Layers

(3) embedding 以后的 向量长度

(4) MLP size = 4 × Hidden Size

(5) Heads

Topic: Hybrid 混合模型



Hybrid混合模型

note

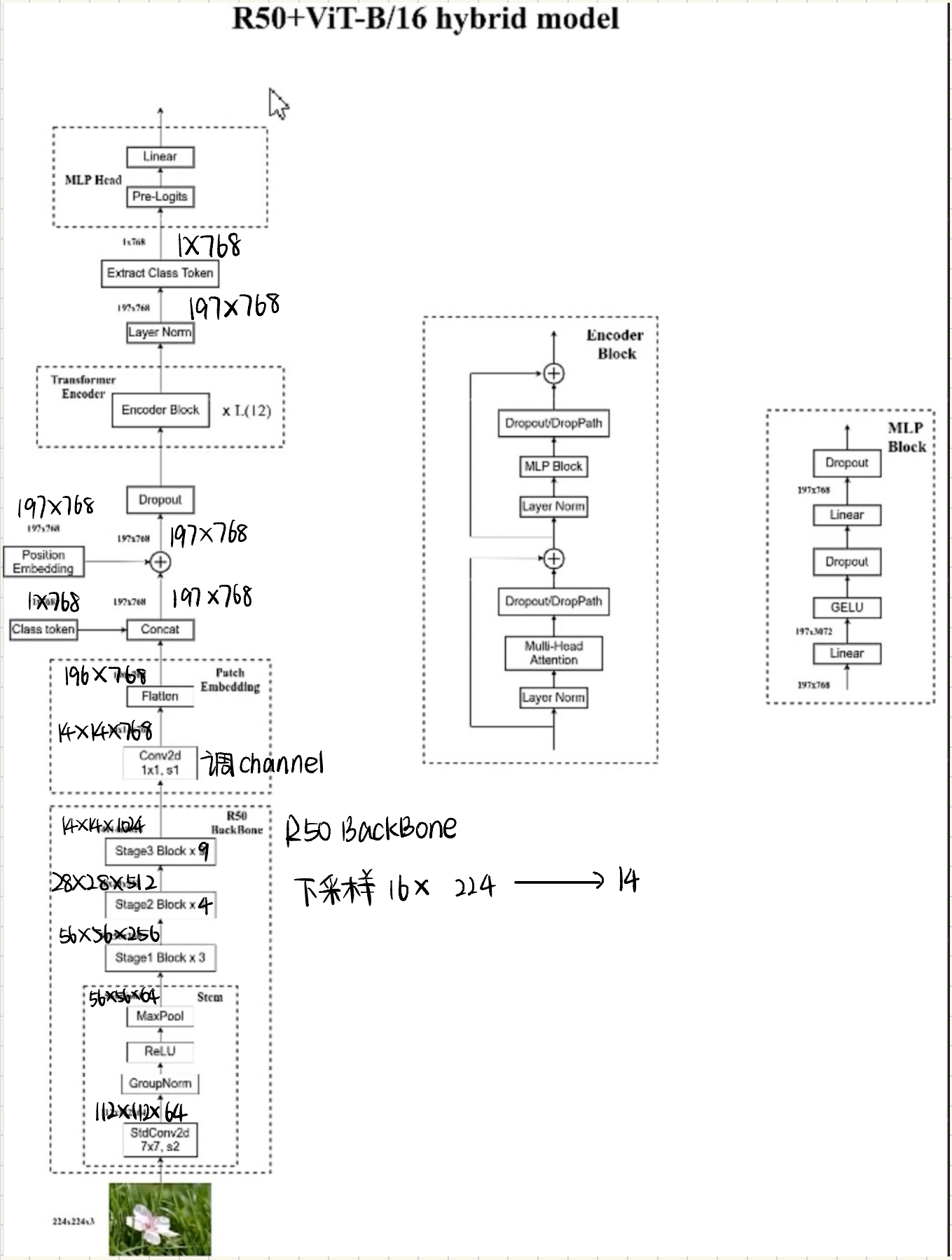
(1) R50的卷积层采用的StdConv2d 不是传统的Conv2d

(2) 将所有的BatchNorm层替换成 GroupNorm层

(3) 把stage4中的3个Block移至 stage3中

博文链接：https://blog.csdn.net/qq_37541097/article/details/118016048 (About GroupNormalization)

Topic: 看大佬手绘的模型结构



traing process

(1) 首先将 input image 大小为 224×224×3 输入到卷积层 stdConv2d
卷积核 k=7 s=32 p=0 #k=64 输出为 112×112×64
紧接着 GroupNorm、ReLU、MaxPool
MaxPool卷积核 k=2 s=2 (尺寸/2, channel不变)
输出 56×56×64

(2) 接着依次通过 stage1, stage2, stage3
在原来的 ResNet 50 网络中，stage3 应该重复 6次
在 Hybrid 设计中 将 stage4 的3个block 移至 stage3当中
∴ stage3一共重复 6+3=9次
经过这一系列的 shape之外 数据 shape 变成 14×14×1024
刚好由 Input Image size 224×224×3 到 14×14×1024 下采样 16倍

(3) then 数据进入 embedding 层 此时 size=14×14×1024
不需要进行下采样，接下来进行 1×1 的卷积 s=1 p=0 #k=768
这个卷积层主要调整 channel 从 1024 调整到 768 why 768

(4) 后面和 ViT 是一样的 concat 上 1个token 加上一个 position embedding
然后 Dropout、Transformer Encoder、LayerNorm、extract class token
提取 class token、通过 MLP Head 得到输出 why 768?

topic: comparison ViT & Hybrid

Model	Epochs	ImageNet	ImageNet ReaL	CIFAR-10	CIFAR-100	Pets	Flowers	exaFLOPs
ViT-B/32	7	80.73	86.27	98.61	90.49	93.40	99.27	164
ViT-B/16	7	84.15	88.85	99.00	91.87	95.80	99.56	743
ViT-L/32	7	84.37	88.28	99.19	92.52	95.83	99.45	574
ViT-L/16	7	86.30	89.43	99.38	93.46	96.81	99.66	2586
ViT-L/16	14	87.12	89.99	99.38	94.04	97.11	99.56	5172
ViT-H/14	14	88.08	90.36	99.50	94.71	97.11	99.71	12826
ResNet50x1	7	77.54	84.56	97.67	86.07	91.11	94.26	150
ResNet50x2	7	82.12	87.94	98.29	89.20	93.43	97.02	592
ResNet101x1	7	80.67	87.07	98.48	89.17	94.08	95.95	285
ResNet152x1	7	81.88	87.96	98.82	90.22	94.17	96.94	427
ResNet152x2	7	84.97	89.69	99.06	92.05	95.37	98.62	1681
ResNet152x2	14	85.56	89.89	99.24	91.92	95.75	98.75	3362
ResNet200x3	14	87.22	90.15	99.34	93.53	96.32	99.04	10212
R50x1+ViT-B/32	7	84.90	89.15	99.01	92.24	95.75	99.46	315
R50x1+ViT-B/16	7	85.58	89.65	99.14	92.63	96.65	99.40	855
R50x1+ViT-L/32	7	85.68	89.04	99.24	92.93	96.97	99.43	725
R50x1+ViT-L/16	7	86.60	89.72	99.18	93.64	97.03	99.40	2704
R50x1+ViT-L/16	14	87.12	89.76	99.31	93.89	97.36	99.11	5165

看纯transformer & 混合hybrid模型

如ViT-L/16 训练迭代7个epochs 在ImageNet1K上 达到准确率86.30

对比ResNet50+ViT-L/16 同样迭代7个epochs 同样 ImageNet1K 达到acc=86.60

比纯Transformer ViT-L/16的效果好

若迭代14个epochs (ViT-L/16) acc=87.12

同样 ResNet50+ViT-L/16 epochs=14 acc=87.12

效果一样

在训练次数epoch较小时 训练精度更高一点

随着训练更多epoch之后 ViT精度会超过混合模型