# 去除冗余token的DETR效果怎么样？NUS颜水成团队提出端到端的PnP-DETR结构

原创 CV开发者都爱看的 极市平台 2021-09-25 22:00:00 手机阅读



收录于话题
#目标检测



作者丨小马
编辑丨极市平台

极市导读

颜水成团队将减少空间冗余的想法封装到一个投票和池化采样模块中，进而构建了一个端到端PnP-DETR结构，自适应地空间分配使其计算更有效率 。 >>加入极市CV技术交流群，走在计算机视觉的最前沿

## 写在前面

最近，DETR利用Transformer将图像特征图直接转换为目标检测的结果。虽然这种方法效果还不错，但将图像特征图转换为目标检测结果的过程中可能存在很多冗余的计算。**在这项工作中，作者将减少空间冗余的想法封装到一个投票和池化(PnP，poll and pool)采样模块中，进而构建了一个端到端PnP-DETR结构，自适应地空间分配使其计算更有效率** 。具体地说，PnP模块将图像特征映射转换为精细的前景特征向量和少量粗糙的背景上下文特征。Transformer对细粒度和粗粒度的特征信息进行交互建模，并将特征转化为检测结果。此外，PnP-augmented模型可以通过改变采样特征长度，实现性能和计算之间的trade-off。因此，它为在具有不同计算约束场景中的部署提供了更大的灵活性。此外，作者进一步验证了PnP模块在全景分割上和图像分类的泛化性，并都获得了一致的性能提升。

## 1. 论文和代码地址

**PnP-DETR: Towards Efficient Visual Analysis with Transformers**

Tao Wang[1,3*] Li Yuan[4*] Yunpeng Chen[2] Jiashi Feng[4] Shuicheng Yan[4]
[1] Institute of Data Science, National University of Singapore [2] Yitu Technology
[3] Integrative Science and Engineering Programme, NUS Graduate School, National University of Singapore
[4] Department of Electrical and Computer Engineering, National University of Singapore
twangnh@gmail.com ylustcnus@gmail.com yunpeng.chen@yitu-inc.com
jshfeng@gmail.com shuicheng.yan@gmail.com

PnP-DETR: Towards Efficient Visual Analysis with Transformers

论文地址：https://arxiv.org/abs/2109.07036

代码地址：https://github.com/twangnh/pnp-detr

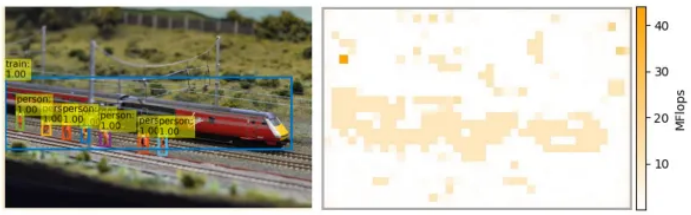
## 2. Motivation

目标检测任务旨在识别图像中的目标实例，并通过精确的边界框来定位它们。目前，DETR模型带来了一个新的目标检测范式，并且消除了原来CNN模型中的手工后处理设计（比如：NMS），实现了端到端目标检测。DETR将图像特征图在空间维度上flatten为一维特征向量。然后，用Transformer的注意力机制对其进行处理，生成最终的检测结果。

虽然这种结构简单而有效，但**将Transformer网络应用于图像特征图的计算代价比较大，因为Self-Attention的计算复杂度与输入特征的长度是呈二次相关的** 。此外，这些特征可能是冗余的：图像除了感兴趣的目标外，通常包含大量的背景区域，这些区域可能在相应的特征表示中占据很大一部分；此外，一些有代表性的特征向量可能已经足以检测目标了。

为了解决上述限制，**作者开发了一个可学习的投票和池化(PnP，poll and pool)采样模块，将图像特征映射压缩成一个由精细特征向量和少量粗粒度特征向量组成的抽象特征集** 。从输入的特征图中确定性地采样精细的特征向量，以捕获精细的前景信息，因此这对于检测目标是至关重要的。粗粒度特征向量聚合了背景位置的信息，所得到的上下文信息有助于更好地识别和定位目标。然后，Transformer对细-粗粒度特征空间内的信息交互进行建模，得到最终结果。由于抽象集特征比原来的图像特征图要少得多，因此Transformer的计算量显著减少，且主要分布在前景位置。

具体来说，PnP模块由两个核心子模块组成：一个**轮询采样器（poll sampler）** 和一个**池化采样器（poll sampler）** 。**轮询采样器** 包含了一个内容感知的评分网络，该网络学习预测每个空间位置的特征向量的信息量得分。然后根据信息量得分对特征向量进行排序，并选择信息最丰富的特征向量的子集。随后的**池化采样器** 动态地预测非采样特征向量上的注意力权值，并将它们聚集成少量的特征向量来总结背景信息。




如上图所示（左边为原图，右边为计算密度图），作者利用PnP模块构建了一个PnP-DETR，该模块在细-粗粒度的特征空间上计算，并在空间域内自适应地分配Transformer的计算特征。因此，本文提出的PnP采样允许Transformer使用可变数量的输入特征向量，并实现计算和性能的trade-off。

作者在COCO基准数据集上进行了广泛的实验，结果表明PnP-DETR有效地降低了成本，实现了动态计算和性能权衡。此外，作者还在全景分割和图像分类任务上验证了本文方法的泛化性。

## 3. 方法

### 3.1. Preliminaries

DETR首先利用带有参数$\theta_c$的主干卷积网络$C$来提取图像特征图$F$：

$$\mathbf{F} = \mathcal{C}(\mathbf{I}, \theta_c)$$

其中$F$可以表示为：

$$\mathbb{F} = \{\mathbf{f}_{ij} \in \mathbb{R}^C | i = 1, \ldots, H, j = 1, \ldots, W\}$$

Grid结构特征集$F$可以看做一组具有强语义信息的高级视觉token，然后用$\theta_t$参数化的Transformer $T$将其转换为检测结果：

$$\{(cls_k, box_k)|k = 1, \ldots, D\} = \mathcal{T}(\mathbb{F}, \theta_t)$$

Grid结构特征集$F$的有一个缺点是它均匀地跨越了所有空间位置，并覆盖了大量的背景。虽然Transformer具有较强的attention能力，可以attend不同的区域，但计算并不符合这一优势，而是在均匀分布的空间域上均匀分布。因此，作者希望通过减少token的数量来减少计算量。

### 3.2. Feature Abstraction

作者提出了一个特征抽象方案来解决上述局限性。它包含两组用于紧凑特征表示的特征向量：

$$\mathbb{F}_f = \{\mathbf{f}_n \in \mathbb{R}^C | n = 1, \ldots, N\}$$
$$\mathbb{F}_c = \{\mathbf{f}_m \in \mathbb{R}^C | m = 1, \ldots, M\}$$

精细特征集$F_f$从完整的集合$F$中离散采样，包含对识别和检测至关重要的精细信息。粗粒度特征集$F_c$是通过聚合来自多个空间位置的信息并编码背景上下文信息而获得的。

它们一起形成了一个抽象集$F^*$：

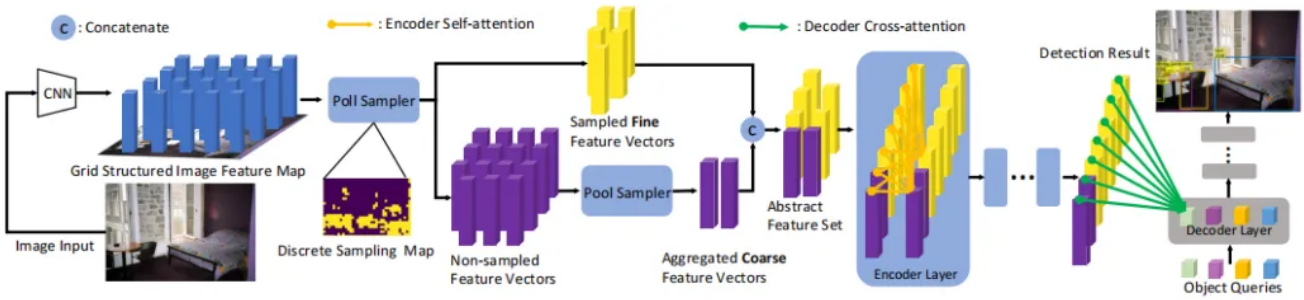
$$\mathbb{F}^* = \mathbb{F}_f \cup \mathbb{F}_c$$

因此，相比于原始特征集，$F^*$的表示更加紧凑，并且也包含了完整的前景信息。

### 3.3. Poll and Pool (PnP) Sampling

上述抽象方案需要解决两个挑战：1)**精细的特征集需要确定性的抽样，这是不可微的** 。手工制作的采样器可以通过一些中间目标来学习，但这与端到端的学习目标不兼容，并且手工采样规则可能不是最优的。2)**要提取一个只关注背景上下文信息的紧凑的特征集也是很困难的** 。

作者将抽象方案分为两个步骤，并开发了一个**轮询采样器** 和一个**池化采样器** 来实现。轮询采样器首先从特征集F中对特征向量进行采样；然后，池化采样器将剩余的非采样特征向量动态聚合成少量的粗粒度特征向量。下图给出了本文方法的示意图。




#### Poll Sampler

轮询采样器旨在获得一个细粒度前景特征集$F_f$。作者使用一个小型的评分网络来预测每个空间特征位置$(i\ ,\ j)$的信息量得分：

$$s_{ij} = \text{ScoringNet}(\mathbf{f}_{ij}, \theta_s)$$

分数越大，特征向量$f_{ij}$的信息就越大。根据分数$s_{ij}$进行排序：

$$[s_l, |l = 1, \ldots, L], \aleph = \text{Sort}(\{s_{ij}\})$$

然后，取前N个得分向量，形成细粒度前景特征集：

$$\mathbb{F}_f = [\mathbf{f}_l, |l = 1, \ldots, N]$$

为了实现具有反向传播的得分网络的学习，作者将预测的信息量分数作为采样的细粒度特征集的加权因子：

$$\mathbb{F}_f = [\mathbf{f}_l * s_l, |l = 1, \ldots, N]$$

在实现的时候，作者先对特征进行归一化再加权，这样可以稳定得分网络的训练：

$$\mathbb{F}_f = [LayerNorm(\mathbf{f}_l) * s_l, |l = 1, \ldots, N]$$

N可能会随着图像内容的变化而变化，但作者也观察到固定量的采样已经产生了良好的性能了，即N=αL，其中α是一个恒定的数，称之为轮询比（poll ratio）。

**Pool Sampler**

上面的采样器提取了精细的特征集。其余的特征向量主要对应于背景区域。为了将它们压缩成一个总结上下文信息的小特征集，作者设计了一个池化采样器，它对剩余的特征向量执行加权池化，以获得固定数量的M个背景上下文特征向量，该步骤用于生成全局描述符。

首先，剩下的特征集可以表示为：

$$\mathbb{F}_r = \mathbb{F} \setminus \mathbb{F}_f = \{\mathbf{f}_r, |r = 1, \ldots, L - N\}$$

作者通过一个可学习的权重$W^a \in R^{C \times M}$获得聚合权重$a_r \in R^M$：

$$\mathbf{a}_r = \mathbf{f}_r \mathbf{W}^a$$

并利用可学习的权重$W^v \in R^{C \times C}$投影特征向量，得到投影特征：

$$\mathbf{f}_r^{'} = \mathbf{f}_r \mathbf{W}^v$$

然后，用softmax对所有剩余的非采样位置的聚合权重$a_r \in R^M$进行归一化：

$$a_{rm} = \frac{e^{a_{rm}}}{\sum_{r'=1}^{N-L} e^{a_{r'm}}}$$

利用归一化聚合权值，对投影的特征向量进行聚合，获得一个新的特征向量：

$$\mathbf{f}_m = \sum_{r=1}^{L-N} \mathbf{f}_r^{'} * a_{rm}$$

通过聚合所有的M个聚合权值，可以得到了总结出来的粗粒度背景上下文特征集：

$$\mathbb{F}_c = \{\mathbf{f}_m, |r = 1, \ldots, M\}$$

#### Reverse Projection for Dense Prediction Tasks

PnP模块将图像特征映射从二维坐标空间简化为抽象空间，而抽象空间不能用于图像分割等密集的预测任务。为了解决这一问题，作者将编码器输出特征向量投影回二维坐标空间。具体来说，精细特征向量放置回采样位置；粗特征向量首先根据聚合权重拓展回原始二维空间：

$$\hat{\mathbf{f}}_r = \sum_{m=1}^{M} \hat{\mathbf{f}_m} * a_{rm}$$

然后放置回的原始的非采样位置。再利用所得到的二维特征图进行密集预测。

### 3.4. PnP-augmented Models

#### PnP-DETR and PnP-ViT

作者评估了本文的方法在ViT模型上的泛化性，通过在Transformer网络之前插入PnP模块来构建PnP-DETR和PnP-ViT。所得到的模型是端到端学习的，其他设置与原始模型相同。与原始的DETR和ViT直接在整个图像特征空间上运行不同，PnP增强的Transformer在精-粗粒度特征空间上对信息交互进行建模，并在空间域自适应地分配计算，以获得更好的效率。

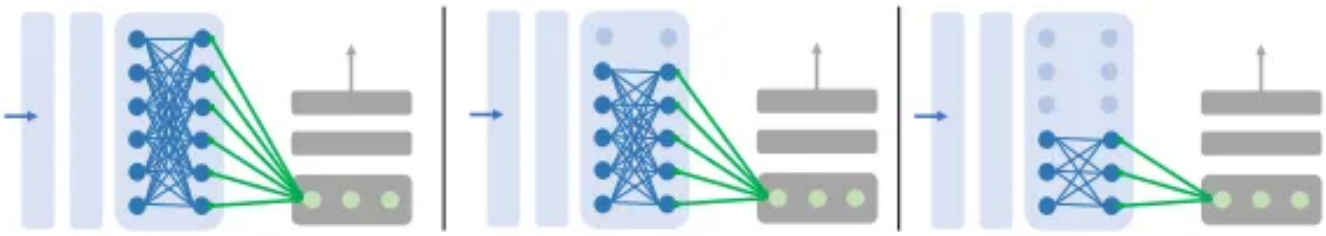
#### Instant Computation and Performance Trade-off

为了实现不同的计算和性能权衡，现有的提高Transformer效率的方法通常训练多个不同复杂度的模型。与它们不同的是，一个配备了PnP模块的模型通过改变采样率α可以实现单个模型计算和性能权衡。α越大，获得特征向量越好，总体性能越高；α越小，性能可能越低，但节省更多的计算。

然而，作者发现使用与训练时不同的α进行推理会严重降低性能。因此，作者提出在训练期间使用一个随机的采样率α：

$$\alpha = uniform(\alpha_{low}, \alpha_{high})$$

其中 $\alpha_{low}$和$\alpha_{high}$为区间的上下界。α在每次迭代中都会进行更新。示意图如下：



## 4.实验

### 4.1. Experiments on Object Detection

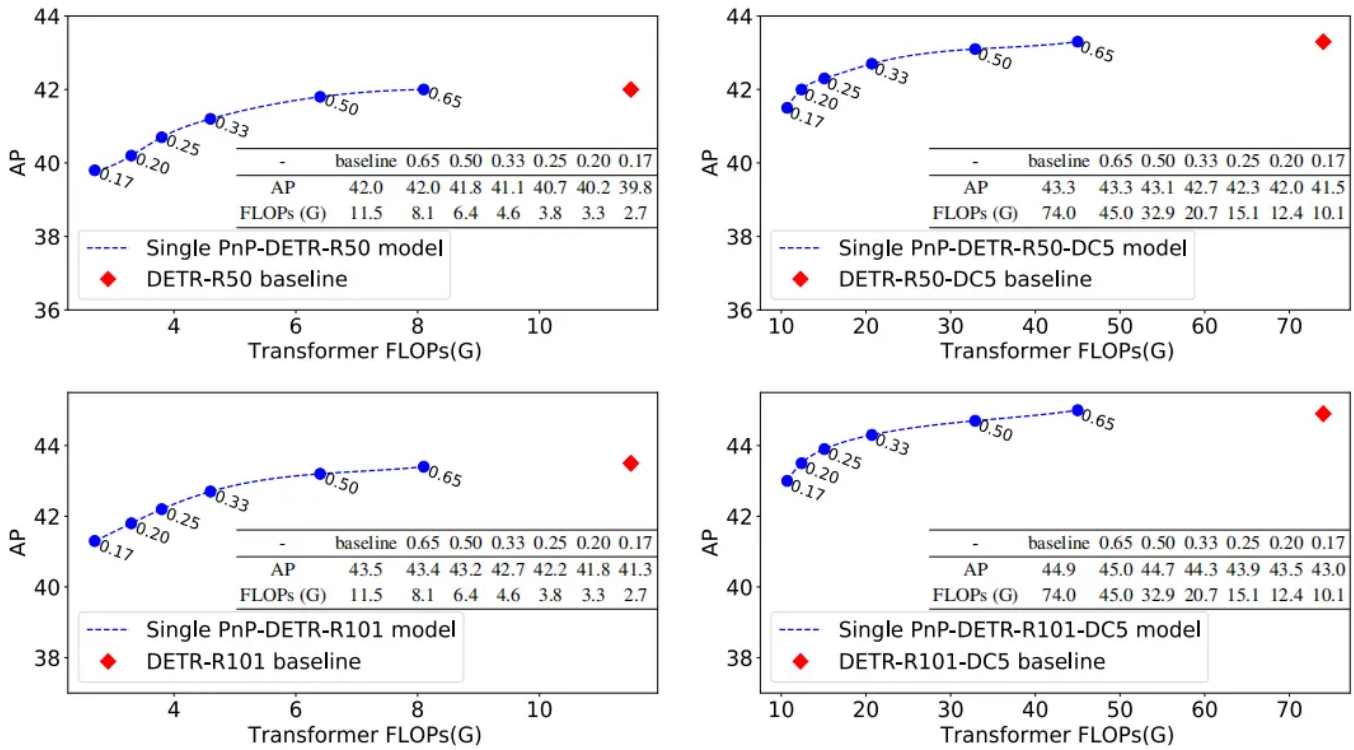
#### Fixed Poll Ratio Training

| Model | AP | $AP_{50}$ | $AP_{75}$ | $AP_s$ | $AP_m$ | $AP_l$ | F-encoder | F-decoder | F-sampler | F-total |
|---|---|---|---|---|---|---|---|---|---|---|
| DETR-R50 [3] | 42.0 | 62.4 | 44.2 | 20.5 | 45.8 | 61.1 | 9.6G | 1.9G | - | 11.5G |
| Deformable-DETR [37] | 40.4 | 60.5 | 43.4 | 21.3 | 44.6 | 57.8 | - | - | - | 5.5G (-52%) |
| PnP-DETR-R50-$\alpha$-0.33 | 41.1 | 61.5 | 43.7 | 20.8 | 44.6 | 60.0 | 3.2G | 1.3G | 0.1G | 4.6G (**-60%**) |
| Inference-$\alpha$-0.5 | 36.1 | 59.8 | 36.1 | 13.9 | 38.7 | 57.7 | - | - | - | - |
| PnP-DETR-R50-$\alpha$-0.5 | 41.8 | 62.1 | 44.4 | 21.2 | 45.3 | 60.8 | 4.8G | 1.5G | 0.1G | 6.4G (-45%) |
| DETR-R50-DC5 [3] | 43.3 | 63.1 | 45.9 | 22.5 | 47.3 | 61.1 | 69.2G | 4.8G | - | 74.0G |
| ACT+MTKD(L=32) [35] | 43.1 | - | - | 22.2 | 47.1 | 61.4 | - | - | - | 58.2 (-21%) |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| ACT+MTKD(L=24) [35] | 42.3 | - | - | 21.3 | 46.4 | 61.0 | - | - | - | 53.1 (-28%) |
| Deformable-DETR-DC5 [37] | 42.1 | 62.3 | 45.6 | 24.3 | 45.6 | 57.3 | - | - | - | 26.4G (-64%) |
| PnP-DETR-R50-DC5-$\alpha$-0.33 | 42.7 | 62.8 | 45.1 | 22.4 | 46.2 | 60.0 | 17.8G | 2.5G | 0.4G | 20.7G (**-72%**) |
| PnP-DETR-R50-DC5-$\alpha$-0.5 | 43.1 | 63.4 | 45.3 | 22.7 | 46.5 | 61.1 | 29.1G | 3.1G | 0.7G | 32.9G (-56%) |

上表显示了在COCO基准数据集上使用固定的α训练的结果。可以看出，提高α可以提高性能，但也会带来计算量的提升。

### Dynamic Poll Ratio Training




| - | baseline | 0.65 | 0.50 | 0.33 | 0.25 | 0.20 | 0.17 |
|---|---|---|---|---|---|---|---|
| AP | 42.0 | 42.0 | 41.8 | 41.1 | 40.7 | 40.2 | 39.8 |
| FLOPs (G) | 11.5 | 8.1 | 6.4 | 4.6 | 3.8 | 3.3 | 2.7 |

| - | baseline | 0.65 | 0.50 | 0.33 | 0.25 | 0.20 | 0.17 |
|---|---|---|---|---|---|---|---|
| AP | 43.3 | 43.3 | 43.1 | 42.7 | 42.3 | 42.0 | 41.5 |
| FLOPs (G) | 74.0 | 45.0 | 32.9 | 20.7 | 15.1 | 12.4 | 10.1 |

| - | baseline | 0.65 | 0.50 | 0.33 | 0.25 | 0.20 | 0.17 |
|---|---|---|---|---|---|---|---|
| AP | 43.5 | 43.4 | 43.2 | 42.7 | 42.2 | 41.8 | 41.3 |
| FLOPs (G) | 11.5 | 8.1 | 6.4 | 4.6 | 3.8 | 3.3 | 2.7 |

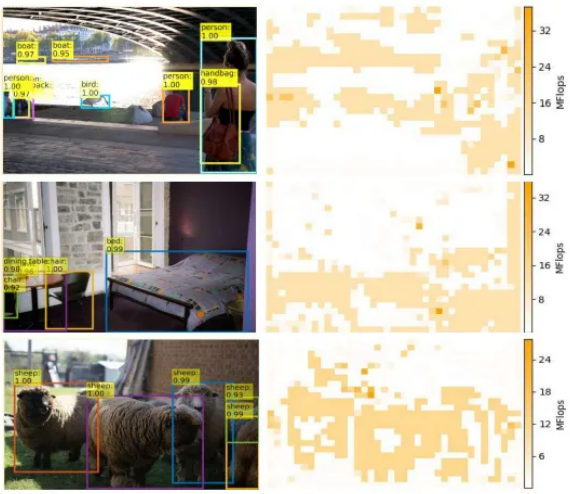
| - | baseline | 0.65 | 0.50 | 0.33 | 0.25 | 0.20 | 0.17 |
|---|---|---|---|---|---|---|---|
| AP | 44.9 | 45.0 | 44.7 | 44.3 | 43.9 | 43.5 | 43.0 |
| FLOPs (G) | 74.0 | 45.0 | 32.9 | 20.7 | 15.1 | 12.4 | 10.1 |

上表显示了比率范围为（0.15,0.8）的训练结果，得到的模型可以通过改变推理时的α来实现动态计算和性能权衡。

| Methods | Encoder | Decoder | PnP-sampler |
|---|---|---|---|
| DETR (baseline) | 72.4 | 11.1 | - |
| PnP-DETR-$\alpha$-0.5 | **28.4** | 10.5 | 2.1 |
| PnP-DETR-$\alpha$-0.33 | **17.4** | 10.3 | 2.0 |

上表为不同α的推理时间，可以看出α越小，推理时间越短。

### Visualization of Computation Density Map



上图显示了α为0.33时，检测结果和相关的计算密度图的可视化。

### 4.2. Experiments on Other Tasks

**Panoptic Segmentation**

| - | DETR | $\alpha$-0.65 | $\alpha$-0.5 | $\alpha$-0.33 | $\alpha$-0.25 | $\alpha$-0.2 |
|---|---|---|---|---|---|---|
| PQ | 43.4 | 43.5 | 43.2 | 42.8 | 42.4 | 41.8 |
| SQ | 79.3 | 79.2 | 79.1 | 78.9 | 78.7 | 78.4 |
| RQ | 53.8 | 53.8 | 53.4 | 53.0 | 52.4 | 51.7 |
| FLOPs (G) | 11.6 | 8.3 | 6.6 | 4.8 | 4.0 | 3.5 |

作者基于DETR模型在全景分割任务上进行了实验，可以看出，本文的方法在α比较大的时候，能够在没有明显降低性能的同时，显著降低计算量。

**Image Recognition**

| - | ViT | $\alpha$-0.7 | $\alpha$-0.5 | $\alpha$-0.33 | $\alpha$-0.25 | $\alpha$-0.2 |
|---|---|---|---|---|---|---|
| Top1-Acc | 82.2 | 82.1 | 81.9 | 81.6 | 81.4 | 81.2 |
| FLOPs (G) | 10.0 | 7.3 | 5.5 | 3.9 | 3.2 | 2.8 |

作者基于ViT模型在图像分类任务上进行了实验，可以看出，与全景分割相似，本文的方法在α比较大的时候，能够在没有明显降低性能的同时，显著降低计算量。

## 5. 总结

这篇文章的Motivation是通过总结背景token来使得Transformer中的token尽量都是有用的，从而来加速运算。在计算机视觉的领域中，还有一些文章也是通过这样的方法来实现加速的，比如Visual Parser [1],DynamicViT [2],Evo-ViT [3]。虽然motivation一样，但是这几篇文章的具体实现方法还是有挺大的区别的。其中**Visual Parser** 是通过几个可学习的查询向量，在原来的token中学习可以代表整张图片的信息，从而降低计算复杂度；**DynamicViT** 是在训练的时候，根据这些token的重要性进行排序，每次只取重要的token进行训练，从而减少图片的token；**Evo-ViT** 同样是将token分为两类，信息token和占位符token，作者将所有的占位符token总结为一个总结性token，对总结性token和信息token进行精细的计算，对占位符token进行简单的计算，从而减少计算量。

个人觉得，目前减少Transformer计算量的方法主要也可以分为两类：（1）**直接降低Transformer中Self-Attention的计算量，比如提出一个新的attention方式代替Self-Attention来降低计算量**；（2）**仍然使用Self-Attention，但是降低输入特征的长度**。而本文的方法和上面提到的方法都属于第二类。

**参考文献**

[1]. Visual Parser: Representing Part-whole Hierarchies with Transformers

[2]. DynamicViT: Effificient Vision Transformers with Dynamic Token Sparsifification

[3]. Evo-ViT: Slow-Fast Token Evolution for Dynamic Vision Transformer

## 极市干货

**神经网络：**视觉神经网络模型优秀开源工作：timm库使用方法和最新代码解读

**技术综述：**综述：神经网络中 Normalization 的发展历程｜CNN轻量化模型及其设计原则综述

**算法技巧（trick）：**8点PyTorch提速技巧汇总｜图像分类算法优化技巧

Moco-ML 开源四足机器人 项目教程4：Moco-Minitaur LTS机器人的机架设计

古月居

大众欲购华为自动驾驶部门；马斯克陷虐猴风波；闪存又要涨价丨时光机

果壳硬科技

Linux内核之旅祝大家虎虎生风，无忧无Bug

Linux内核之旅