还在魔改Transformer结构吗？微软&中山大学开源超强的视觉位置编码，涨点显著

原创 CV开发者都爱看的 极市平台 2021-08-02 22:00:00 手机阅读

收录于话题
#Transformer

作者丨小马
编辑丨极市平台

极市导读

本文重新思考并改进相对位置编码在视觉Transformer中的使用，并提出了 4 种专门用于视觉Transformer的方法，并通过实验证明了在检测和分类任务上较大的性能提升。 >>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

由于Transformer对于序列数据进行并行操作，所以序列的位置信息就被忽略了。因此，相对位置编码(Relative position encoding, RPE)是Transformer获取输入序列位置信息的重要方法，RPE在自然语言处理任务中已被广泛使用。

但是，在计算机视觉任务中，相对位置编码的有效性还没有得到很好的研究，甚至还存在争议。因此，作者在本文中先回顾了现有的相对位置编码方法，并分析了它们在视觉Transformer中应用的优缺点。接着，作者提出了新的用于二维图像的相对位置编码方法（iRPE）。iRPE考虑了方向，相对距离，Query的相互作用，以及Self-Attention机制中相对位置embedding。作为一个即插即用的模块，本文提出的iREP是简单并且轻量级的。

实验表明，通过使用iRPE，DeiT和DETR在ImageNet和COCO上，与原始版本相比，分别获得了1.5%（top-1 Acc）和1.3%（mAP）的性能提升（无需任何调参）。

论文和代码地址

Rethinking and Improving Relative Position Encoding for Vision Transformer

Kan Wu[1,2,*], Houwen Peng[2,*,†], Minghao Chen[2], Jianlong Fu[2], Hongyang Chao[1]
[1] Sun Yat-sen University [2] Microsoft Research Asia

论文地址：
https://arxiv.org/abs/2107.14222

代码地址：
https://github.com/microsoft/AutoML/tree/main/iRPE

研究动机

Transformer最近在计算机视觉领域引起了极大的关注，因为它具有强大的性能和捕获Long-range关系的能力。然而，Transformer中的Self-Attention有一个固有的缺陷——它不能捕获输入token的顺序。因此，Transformer在计算的时候就需要显式的引入位置信息。

为Transformer编码位置表示的方法主要有两类。一个是绝对位置编码，另一个是相对位置编码。**绝对位置编码**将输入token的绝对位置从1编码到最大序列长度。也就是说，每个位置都有一个单独的编码向量。然后将编码向量与输入token组合，使得模型能够知道每个token的位置信息。**相对位置编码**对输入token之间的相对距离进行编码，从而来学习token的相对关系。

这两种编码方式在NLP任务中都被广泛应用，并且证明是非常有效的。但是在CV任务中，它们的有效性还没被很好的探索。因此，在本文中，**作者重新思考并改进相对位置编码在视觉Transformer中的使用**。

在本文中，作者首先回顾了现有的相对位置编码方法，然后提出了专门用于二维图像的方法iRPE。

方法

方法背景

绝对位置编码

由于Transformer不包含递归和卷积，为了使模型知道序列的顺序，需要注入一些关于token位置的信息。原始Self-Attention采用了绝对位置，并添加绝对位置编码$p = （ p_1 ， \ldots , p_n ）$到输入token，用公式表示如下：

$$\mathbf{x}_i = \mathbf{x}_i + \mathbf{p}_i,$$

相对位置编码

除了每个输入token的绝对位置之外，一些研究人员还考虑了token之间的相对关系。相对位置编码使得Transformer能够学习token之间的相对位置关系，用公式表示如下：

$$\mathbf{z}_i = \sum_{j=1}^{n} \alpha_{ij} (\mathbf{x}_j \mathbf{W}^V + \mathbf{p}_{ij}^V),$$

$$e_{ij} = \frac{(\mathbf{x}_i \mathbf{W}^Q + \mathbf{p}_{ij}^Q)(\mathbf{x}_j \mathbf{W}^K + \mathbf{p}_{ij}^K)^T}{\sqrt{d_z}}.$$

回顾相对位置编码

Shaw's RPE

[1]提出一种Self-Attention的相对位置编码方法。输入token被建模为一个有向全连通图。每条边都代表两个位置之间的相对位置信息。此外，作者认为精确的相对位置信息在一定距离之外是无用的，因此引入了clip函数来减少参数量，公式表示如下：

$$\mathbf{z}_i = \sum_{j=1}^{n} \alpha_{ij} (\mathbf{x}_j \mathbf{W}^V + \mathbf{p}_{clip(i-j,k)}^V),$$

$$e_{ij} = \frac{(\mathbf{x}_i \mathbf{W}^Q)(\mathbf{x}_j \mathbf{W}^K + \mathbf{p}_{clip(i-j,k)}^K)^T}{\sqrt{d_z}},$$

$$clip(x,\kappa) = max(-\kappa, min(\kappa, x)),$$

RPE in Transformer-XL

[2]为query引入额外的bias项，并使用正弦公式进行相对位置编码，用公式表示如下：

$$e_{ij} = \frac{(\mathbf{x}_i\mathbf{W}^Q + \mathbf{u})(\mathbf{x}_j\mathbf{W}^K)^T + (\mathbf{x}_i\mathbf{W}^Q + \mathbf{v})(\mathbf{s}_{i-j}\mathbf{W}^R)^T}{\sqrt{d_z}},$$

Huang's RPE

[3]提出了一种同时考虑query、key和相对位置交互的方法，用公式表示如下：

$$e_{ij} = \frac{(\mathbf{x}_i\mathbf{W}^Q + \mathbf{p}_{ij})(\mathbf{x}_j\mathbf{W}^K + \mathbf{p}_{ij})^T - \mathbf{p}_{ij}\mathbf{p}_{ij}^T}{\sqrt{d_z}},$$

RPE in SASA

上面的相对位置编码都是针对一维的序列，[4]提出了一种对二维特征进行相对位置编码的方法，用公式表示如下：

$$e_{ij} = \frac{(\mathbf{x}_i\mathbf{W}^Q)(\mathbf{x}_j\mathbf{W}^K + concat(\mathbf{p}^K_{\delta\tilde{x}}, \mathbf{p}^K_{\delta\tilde{y}}))^T}{\sqrt{d_z}},$$

> **相对位置编码的确定**

接下来，作者引入了多种相对位置编码方式，并进行了详细的分析。首先，为了研究编码是否可以独立于输入token，作者引入了两种相对位置模式：**Bias模式**和**Contextual模式** 。然后，为了研究方向性的重要性，作者设计了两种**无向方法**和两种**有向方法**。

Bias Mode and Contextual Mode

以前的相对位置编码方法都依赖于输入token，因此，作者就思考了，相对位置的编码信息能否独立于输入token来学习。基于此，作者引入相对位置编码的**Bias模式**和**Contextual模式**来研究这个问题。前者独立于输入token，而后者考虑了与query、key或value的交互。无论是哪种模式，相对位置编码都可以用下面的公式表示：

$$e_{ij} = \frac{(\mathbf{x}_i\mathbf{W}^Q)(\mathbf{x}_j\mathbf{W}^K)^T + b_{ij}}{\sqrt{d_z}},$$

对于Bias模式，编码独立于输入token，可以表示成：

$$b_{ij} = r_{ij},$$

对于Contextual 模式，编码考虑了与输入token之间的交互，可以表示成：

$$b_{ij} = (\mathbf{x}_i\mathbf{W}^Q)\mathbf{r}_{ij}^T,$$

$$b_{ij} = (\mathbf{x}_i\mathbf{W}^Q)(\mathbf{r}^K_{ij})^T + (\mathbf{x}_j\mathbf{W}^K)(\mathbf{r}^Q_{ij})^T,$$

A Piecewise Index Function

由于实际距离到计算距离的关系是多对一的关系，所以首先需要定义一个实际距离到计算距离的映射函数。

先前有工作提出了采用clip函数来进行映射，如下所示：

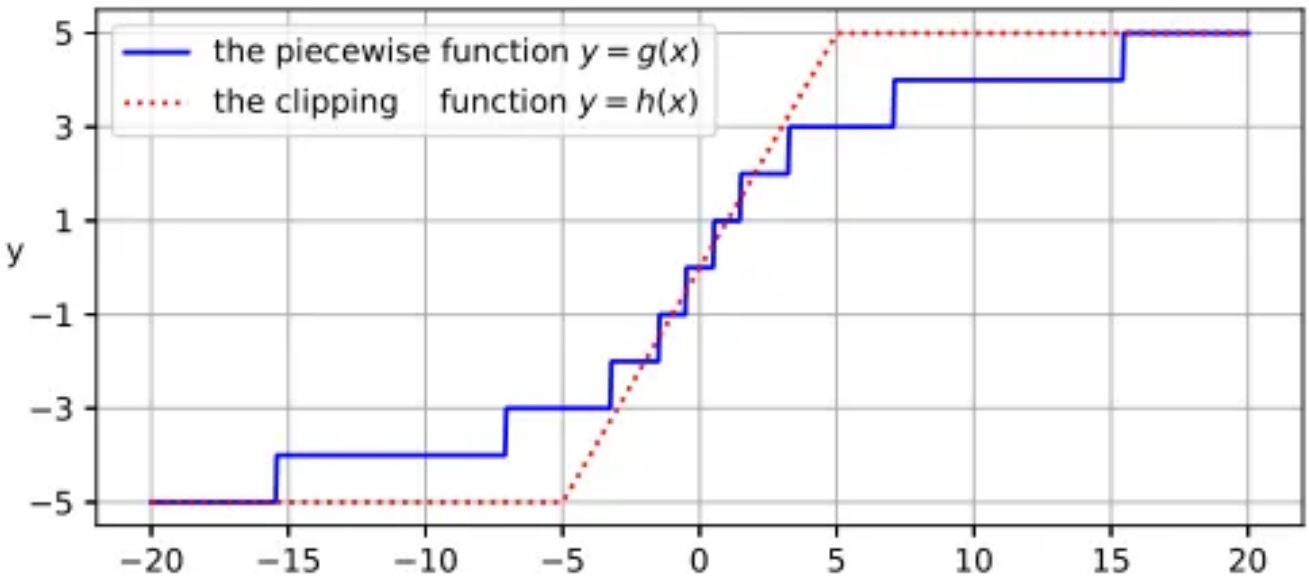
$$h(x) = max(-\beta, min(\beta, x))$$

在这种方法中，相对距离大于β的位置分配给相同的编码，因此丢失了远距离相对位置的上下文信息。

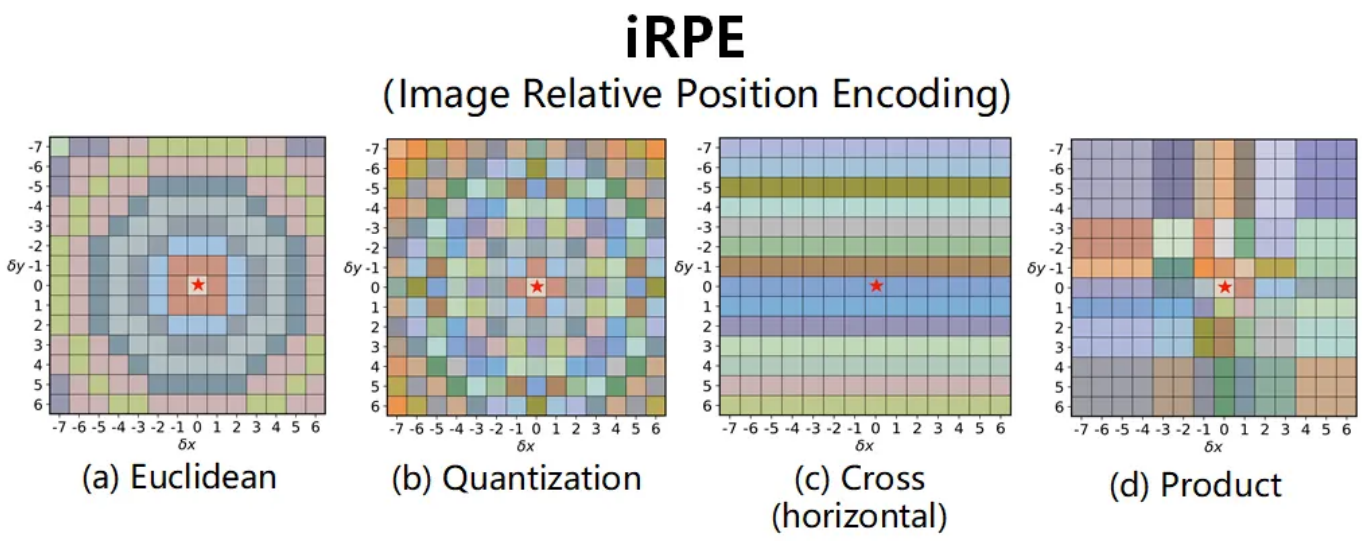
在本文中，作者采用了一种分段函数将相对距离映射到相应的编码。这个函数基于一个假设：越近邻的信息越重要，并通过相对距离来分配注意力。函数如下：

$$g(x) = \begin{cases} [x], & |x| \leq \alpha \\ sign(x) \times min(\beta, [\alpha + \frac{\ln(|x|/\alpha)}{\ln(\gamma/\alpha)}(\beta - \alpha)]), & |x| > \alpha \end{cases}$$

如下图所示，相比于先前的方法，本文提出的方法感知距离更长，并且对不同的距离分布施加了不同程度的注意力。



2D Relative Position Calculation



为了衡量二维图像上两个点的相对距离，作者提出了两种**无向方法**（Euclidean method，Quantization method）和两种**有向方法**（Cross method，Product method），如上图所示。

- **Euclidean method**

在Euclidean method中，作者采用了欧氏距离来衡量两个点之间的距离，如上图a所示：

$$I(i,j)=g(\sqrt{(\tilde{x}_i-\tilde{x}_j)^2+(\tilde{y}_i-\tilde{y}_j)^2}),$$

- **Quantization method**

在上述的Euclidean method中，具有不同相对距离的两个距离可能映射到同一距离下标（比如二维相对位置（1,0）和（1,1）都映射到距离下标1中）。因此，作者提出Quantization method，如上图b所示，公式如下所示：

$$I(i,j)=g(quant(\sqrt{(\tilde{x}_i-\tilde{x}_j)^2+(\tilde{y}_i-\tilde{y}_j)^2})).$$

$quant$（·）函数可以映射一组实数0，1，1.41，2，2.24，...到一组整数0，1，2，3，4，...。

- **Cross method**

像素的位置方向对图像理解也很重要，因此作者又提出了有向映射方法。Cross method分别计算水平方向和垂直方向上的编码，然后对它们进行汇总。编码信息如上图c所示，公式如下：

$$\mathbf{r}_{ij}=\mathbf{p}^{\tilde{x}}_{I^{\tilde{x}}(i,j)}+\mathbf{p}^{\tilde{y}}_{I^{\tilde{y}}(i,j)},$$

$$I^{\tilde{x}}(i,j)=g(\tilde{x}_i-\tilde{x}_j),$$

$$I^{\tilde{y}}(i,j)=g(\tilde{y}_i-\tilde{y}_j),$$

- **Product method**

如果一个方向上的距离相同（水平或垂直），Cross method将会把不同的相对位置编码到相同的embedding中。因此，作者又提出了Product method，如上图d所示，公式如下所示：

$$\mathbf{r}_{ij}=\mathbf{p}_{I^{\tilde{x}}(i,j),I^{\tilde{y}}(i,j)}.$$

> **高效实现**

对于Contextual模式的相对位置编码，编码信息可以通过下面的方式得到：

$$y_{ij}=(\mathbf{x}_i\mathbf{W})\mathbf{p}_{I(i,j)}{}^T.$$

但是这么做的计算复杂度是$O(n^2d)$，所以作者在实现的时候就只计算了不同映射位置的位置编码，如下所示：

$$z_{i,t}=(\mathbf{x}_i\mathbf{W})\mathbf{p}_t{}^T, t\in\{I(i,j)|i,j\in[0,n)\},$$

$$y_{ij}=z_{i,I(i,j)}.$$

这样做就可以将计算复杂度降低到$O(nkd)$，对于图像分割这种任务，k是远小于n的，就可以大大降低计算量。

4.实验

> **相关位置编码分析**

- **Directed-Bias v.s. Undirected-Contextual**

Method based on DeiT-S [22]	Is Directed	Mode	Top-1 Acc(%)	Δ Acc(%)
Original [22]	-	-	79.9	–
Euclidean	×	bias	80.1	+0.2
		contextual	80.4	+0.5
Quantization	×	bias	80.3	+0.4
		contextual	80.5	+0.6
Cross	✓	bias	80.5	+0.6
		contextual	80.8	+0.9
Product	✓	bias	80.5	+0.6
		contextual	80.9	+1.0

上表的结果表明了：

1）无论使用哪种方法，Contextual模式都比Bias模式具有更好的性能。

2）在视觉Transformer中，有向方法通常比无向方法表现更好。

- **Shared v.s. Unshared**

Mode	Shared	#Param. (M)	MACs (M)	Top-1 Acc(%)
Bias	×	22.05	4613	80.54 ± 0.06
	✓	22.05	4613	80.05 ± 0.04
Contextual	×	22.28	4659	80.99 ± 0.16
	✓	22.09	4659	80.89 ± 0.04

对于bias模式，在head上共享编码时，准确度会显著下降。相比之下，在contextual模式中，两种方案之间的性能差距可以忽略不计。

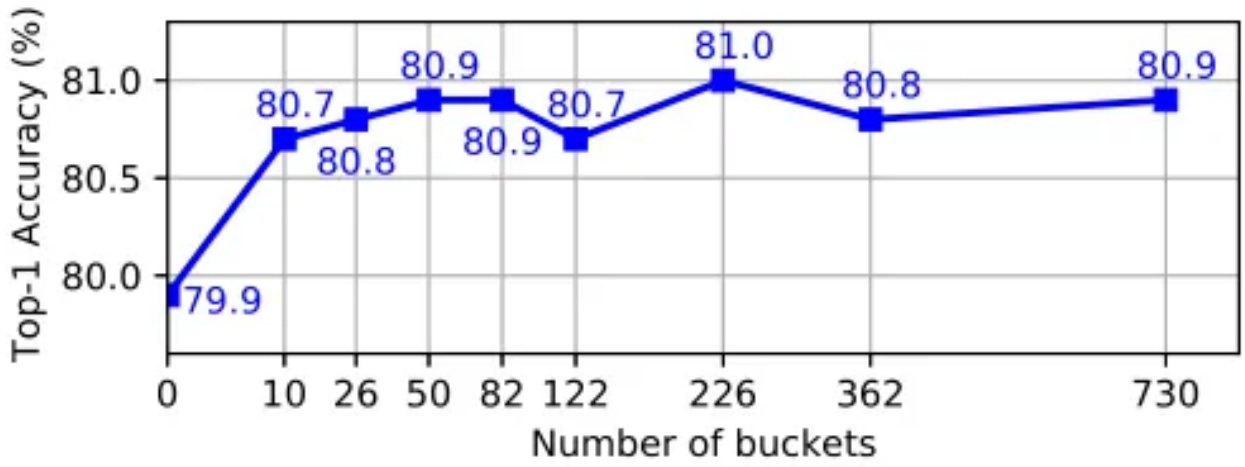
- **Piecewise v.s. Clip.**

Function	Mode	Top-1 Acc(%)	Top-5 Acc(%)
clip	bias	80.1	94.9
	contextual	80.9	95.5
piecewise	bias	80.0	95.0
	contextual	80.9	95.5

上表比较了clip函数和分段函数的影响，在图像分类任务中，这两个函数之间的性能差距非常小，甚至可以忽略不计。但是从下表中可以看出，在检测任务中，两个函数性能还是有明显差距的。

#	Abs Pos.	Rel Pos.	#buckets	epoch	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
1 [1]	sinusoid	none	-	150	39.5	60.3	41.4	17.5	43.0	59.1
2	none	none	-	150	30.4(-9.1)	52.5	30.2	9.4	31.2	50.5
3	sinusoid	bias	9×9	150	40.6(+1.1)	61.2	42.8	19.0	43.9	60.2
4	none	contextual	9×9	150	38.7(-0.8)	60.1	40.4	18.2	41.8	56.7
5	sinusoid	ctx clip	9×9	150	40.4(+0.9)	60.9	42.4	19.1	43.7	59.8
6	sinusoid	contextual	9×9	150	40.8(+1.3)	61.5	42.5	18.5	44.4	60.5
7	sinusoid	contextual	15×15	150	40.8(+1.3)	61.7	42.6	18.5	44.2	61.2
8 [1]	sinusoid	none	-	300	40.6	61.6	-	19.9	44.3	60.2
9	sinusoid	contextual	9×9	300	42.3(+1.7)	62.8	44.3	20.7	46.2	61.1

- **Number of buckets**



bucket数量影响了模型的参数，上图展示了不同bucket数量下，模型准确率的变化。

- **Component-wise analysis**

#	Abs Pos.	p_{ij}^Q	p_{ij}^K	p_{ij}^V	Top-1	Top-5
1 [22]	learnable	×	×	×	79.9	95.0
2	×	×	×	×	77.6(-2.3)	93.8
3	×	✓	×	×	80.9(+1.0)	95.4
4	×	×	✓	×	80.9(+1.0)	95.3
5	×	×	×	✓	80.2(+0.3)	95.0
6	×	✓	✓	×	81.0(+1.1)	95.5
7	×	✓	✓	✓	81.3(+1.4)	**95.7**
8	learnable	✓	×	×	80.9(+1.0)	95.5
9	learnable	×	✓	×	80.9(+1.0)	95.5

10	learnable	×	×	✓	80.2(+0.3)	95.1
11	learnable	✓	✓	×	81.1(+1.2)	95.4
12	learnable	✓	✓	✓	**81.4**(+1.5)	95.6

从上表可以看出，相对位置编码和绝对位置编码对DeiT模型的精度都有很大帮助。

- **Complexity Analysis**

上图表明，本文方法在高效实现的情况下最多需要1%的额外计算成本。

> **在图像分类任务上的表现**

Model	#Param.	Input	MACs (M)	Top-1 Acc (%)
Convnets				
ResNet-50 [10]	25M	224^2	4121	79.0
RegNetY-4.0GF [15]	21M	224^2	4012	79.4
EfficientNet-B1 [21]	8M	240^2	712	79.1
EfficientNet-B5 [21]	30M	456^2	10392	83.6
Transformers				
ViT-B/16 [6]	86M	384^2	55630	77.9
ViT-L/16 [6]	307M	384^2	191452	76.5
DeiT-Ti [22]	5M	224^2	1261	72.2
CPVT-Ti(0-5) [2]	6M	224^2	1262	73.4
DeiT-Ti with iRPE-K(**Ours**)	6M	224^2	1284	73.7
DeiT-S [22]	22M	224^2	4613	79.9
CPVT-S(0-5) [2]	23M	224^2	4616	80.5
DeiT-S(Shaw's) [22, 18]+	22M	224^2	4659	80.9
DeiT-S(Trans.-XL's) [22, 3]+	23M	224^2	4828	80.8
DeiT-S(Huang's) [22, 11]+	22M	224^2	4706	81.0

DeiT-S(SASA's) [22, 17]*	22M	224^2	4639	80.8
DeiT-S with iRPE-K(**Ours**)	22M	224^2	4659	80.9
DeiT-S with iRPE-QK(**Ours**)	22M	224^2	4706	81.1
DeiT-S with iRPE-QKV(**Ours**)	22M	224^2	4885	81.4
DeiT-B [22]	86M	224^2	17592	81.8
CPVT-B(0-5) [2]	86M	224^2	17598	81.9
DeiT-B with iRPE-K(**Ours**)	87M	224^2	17684	82.4

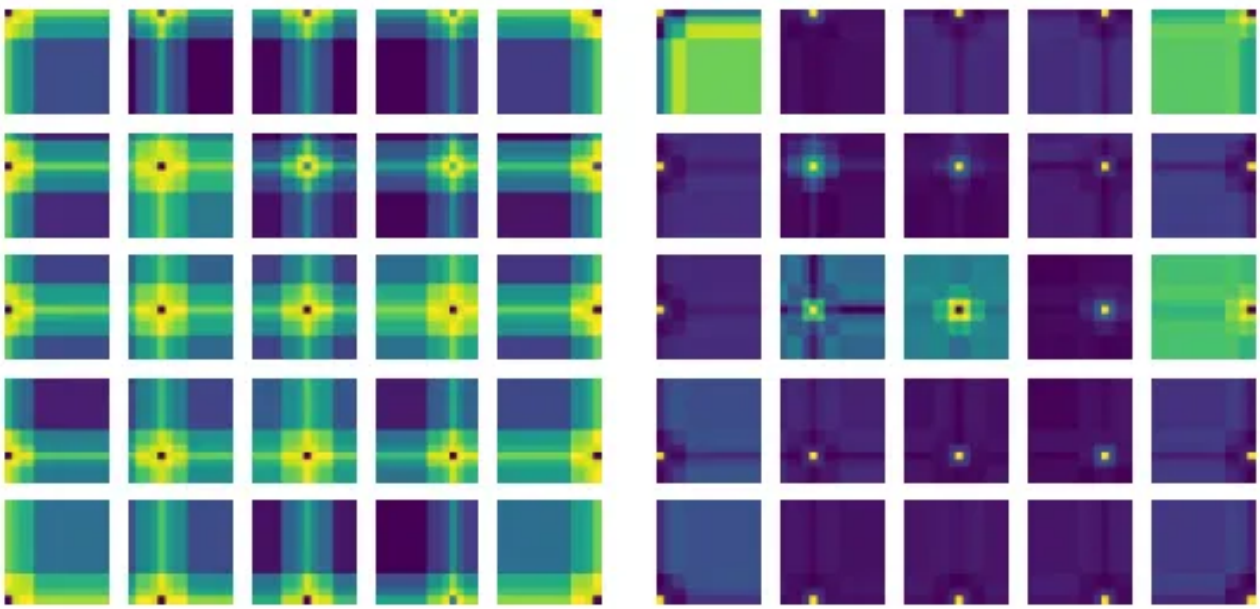
通过仅在key上添加相对位置编码，将DeiT-Ti/DeiT-S/DeiT-B模型分别提升了1.5%/1.0%/0.6%的性能。

在目标检测任务上的表现

#	Abs Pos.	Rel Pos.	#buckets	epoch	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
1 [1]	sinusoid	none	-	150	39.5	60.3	41.4	17.5	43.0	59.1
2	none	none	-	150	30.4(-9.1)	52.5	30.2	9.4	31.2	50.5
3	sinusoid	bias	9×9	150	40.6(+1.1)	61.2	42.8	19.0	43.9	60.2
4	none	contextual	9×9	150	38.7(-0.8)	60.1	40.4	18.2	41.8	56.7
5	sinusoid	ctx clip	9×9	150	40.4(+0.9)	60.9	42.4	19.1	43.7	59.8
6	sinusoid	contextual	9×9	150	40.8(+1.3)	61.5	42.5	18.5	44.4	60.5
7	sinusoid	contextual	15×15	150	40.8(+1.3)	61.7	42.6	18.5	44.2	61.2
8 [1]	sinusoid	none	-	300	40.6	61.6	-	19.9	44.3	60.2
9	sinusoid	contextual	9×9	300	42.3(+1.7)	62.8	44.3	20.7	46.2	61.1

在DETR中绝对位置嵌入优于相对位置嵌入，这与分类中的结果相反。作者推测DETR需要绝对位置编码的先验知识来定位目标。

可视化



(a) block 0 (b) block 10

上图展示了Contextual模式下相对位置编码（RPE）的可视化。

5. 总结

本文作者回顾了现有的相对位置编码方法，并提出了四种专门用于视觉Transformer的方法。作者通过实验证明了通过加入相对位置编码，与baseline模型相比，在检测和分类任务上都有比较大的性能提升。此外，作者通过对不同位置编码方式的比较和分析，得出了下面几个结论：

1）相对位置编码可以在不同的head之间参数共享，能够在contextual模式下实现与非共享相当的性能。

2）在图像分类任务中，相对位置编码可以代替绝对位置编码。然而，绝对位置编码对于目标检测任务是必须的，它需要用绝对位置编码来预测目标的位置。

3）相对位置编码应考虑位置方向性，这对于二维图像是非常重要的。

4）相对位置编码迫使浅层的layer更加关注局部的patch。

参考文献

[1]. Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. ACL, 2018.
[2]. Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell,Quoc Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. In ACL,2019.
[3]. Zhiheng Huang, Davis Liang, Peng Xu, and Bing Xiang. Improve transformer models with better relative position embeddings. In EMNLP, 2020
[4]. Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Standalone self-attention in vision models. arXiv preprint arXiv:1906.05909, 2019.

极市平台签约作者#

小马
知乎：努力努力再努力

厦门大学人工智能系20级硕士。
研究领域：多模态内容理解，专注于解决视觉模态和语言模态相结合的任务，促进Vision-Language模型的实地应用。

作品精选

CVPR2021最佳学生论文提名：Less is More
Transformer一作又出新作！HaloNet：用Self-Attention的方式进行卷积
超越Swin，Transformer屠榜三大视觉任务！微软推出新作：Focal Self-Attention

投稿方式：

添加小编微信Fengcall（微信号：fengcall19），备注：**姓名-投稿**

△长按添加极市平台小编

觉得有用麻烦给个在看啦~

阅读原文

喜欢此内容的人还喜欢

15个目标检测开源数据集汇总
极市平台