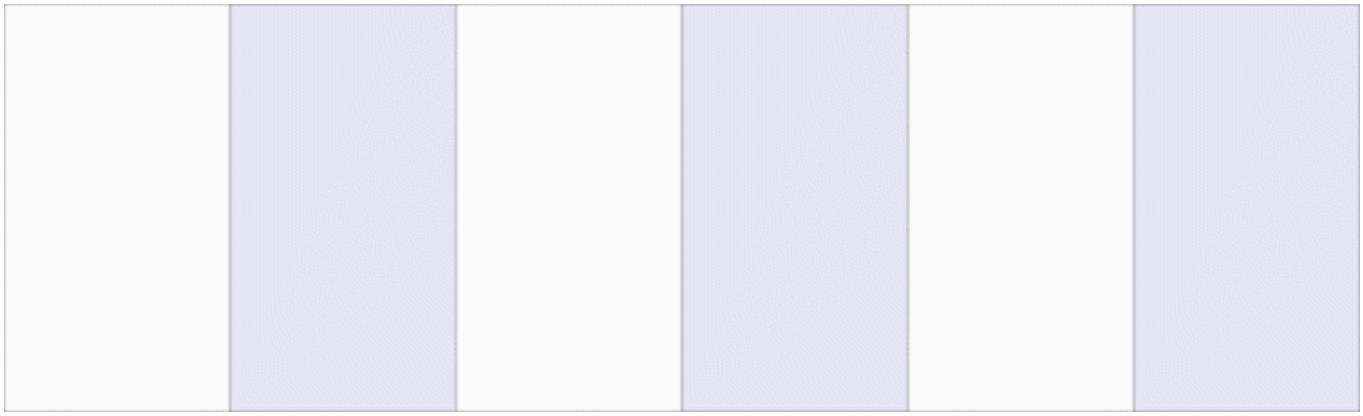
CVPR 2023 VAND Workshop Challenge零样本异常检测冠军方案

极市平台 2023-06-26 22:00:07 发表于广东 手机阅读

以下文章来源于皮皮嬉 ，作者嬉嬉皮

皮皮嬉

聚焦深度学习及其小样本问题



作者丨嬉嬉皮

来源丨皮皮嬉

编辑丨极市平台

极市导读

本文为CVPR 2023 VAND Workshop Challenge赛道一和赛道二分别获得第一和第四成绩的方案。 >>加入极市CV技术交流群，走在计算机视觉的最前沿

在计算机视觉领域，无监督异常检测(AD)旨在使用仅在无异常图像上训练的模型识别异常图像并定位异常区域， 广泛应用于工业缺陷检测。目前大多数方法都集中在为每个类别训练专用模型，这依赖大量正常图像集合作为参考。然而在实际应用中，需要检测的工业产品种类繁多，很难为每个类别收集大量的训练图像。因此，零样本/小样本设置在将AD带入实际应用中起着至关重要的作用。

A Zero-/Few-Shot Anomaly Classification and Segmentation Method for CVPR 2023 VAND Workshop Challenge Tracks 1&2: 1st Place on Zero-shot AD and 4th Place on Few-shot AD

Xuhai Chen[1*] Yue Han[1*] Jiangning Zhang[2*†]
[1]APRIL Lab, Zhejiang University [2]Youtu Lab, Tencent
{22232044, 22132041}@zju.edu.cn, vtzhang@tencent.com

对于工业视觉检测而言，在没有或只有少数正常参考图像的情况下，构建一个能够快速适应众多类别的单一模型是一个很有价值的研究方向。

在zero-shot任务中，所提解决方案在**CLIP模型上加入额外的线形层**，使图像特征映射到联合嵌入空间，从而使其能够与文本特征进行比较并生成异anomaly maps。

当有参考图像可用时（few-shot），所提解决方案利用**多个memory banks存储参考图像特征**，并在测试时与查询图像进行比较。

在ZS和FS两项挑战中，所提方案分别取得了第一和第四名的成绩。

具体来说，所提方案的几个要点是：

- 使用状态（state）和模板（template）的提示集成来制作文本提示。
- 为了定位异常区域，引入了额外的线性层，将从CLIP图像编码器提取的图像特征映射到文本特征所在的线性空间。
- 将映射的图像特征与文本特征进行相似度比较，从而得到相应的anomaly maps。
- few-shot中，保留zero-shot阶段的额外线性层并保持它们的权重。此外，在测试阶段使用图像编码器提取参考图像的特征并保存到memory banks中，以便与测试图像的特征进行比较。
- 为了充分利用浅层和深层特征，同时利用了图像编码器不同stage的特征。

零样本异常检测设定

分类任务

state-level文本提示是使用通用的文本描述正常或异常的目标（比如flawless,damaged），而不会使用“chip around edge and corner”这种过于细节的描述；

所谓template-level文本提示，所提方案在CLIP中为ImageNet筛选了85个模板，并移除了“a photo of the weird [obj.]”等不适用于异常检测任务的模板。

这两种文本提示将通过CLIP的文本编码器提取为最终的文本特征: $F_t \in \mathbb{R}^{2\times C}$ 。

对应的图像特征经图像编码器为: $F_c \in \mathbb{R}^{1\times C}$ 。

state-level和template-level的集成实现如下，最后的两组向量分别描述了正常/异常目标的文本提示。

```
def encode\_text\_with\_prompt\_ensemble\(model, texts, device\):
    prompt\_normal = \['\{\}', 'flawless \{\}', 'perfect \{\}', 'unblemished \{\}', '\{\}
    prompt\_abnormal = \['damaged \{\}', 'broken \{\}', '\{\} with flaw', '\{\} with defec
    prompt\_state = \[prompt\_normal, prompt\_abnormal\]
    prompt\_templates = \['a bad photo of a \{\}.',
                          'a low resolution photo of the \{\}.',
                          'a bad photo of the \{\}.',
                          'a cropped photo of the \{\}.',
                          'a bright photo of a \{\}.',
                          'a dark photo of the \{\}.',
                          'a photo of my \{\}.',
                          'a photo of the cool \{\}.',
                          'a close-up photo of a \{\}.',
                          'a black and white photo of the \{\}.',
                          'a bright photo of the \{\}.',
                          'a cropped photo of a \{\}.',
                          'a jpeg corrupted photo of a \{\}.',
                          'a blurry photo of the \{\}.',
                          'a photo of the \{\}.',
                          'a good photo of the \{\}.',
                          'a photo of one \{\}.',
                          'a close-up photo of the \{\}.',
                          'a photo of a \{\}.',
                          'a low resolution photo of a \{\}.',
                          'a photo of a large \{\}.',
                          'a blurry photo of a \{\}.',
                          'a jpeg corrupted photo of the \{\}.',
                          'a good photo of a \{\}.',
                          'a photo of the small \{\}.',
                          'a photo of the large \{\}.',
                          'a black and white photo of a \{\}.',
                          'a dark photo of a \{\}.',
                          'a photo of a cool \{\}.',
```

```
                'a photo of a small \{\}.',
                'there is a \{\} in the scene.',
                'there is the \{\} in the scene.',
                'this is a \{\} in the scene.',
                'this is the \{\} in the scene.',
                'this is one \{\} in the scene.'\]

    text\_features = \[\]
    for i in range\(len\(prompt\_state\)\):
        prompted\_state = \[state.format\(texts\[0\]\) for state in prompt\_state\[i\]\]
        prompted\_sentence = \[\]
        for s in prompted\_state: # \[prompt\_normal, prompt\_abnormal\]
            for template in prompt\_templates:
                prompted\_sentence.append\(template.format\(s\)\)
        prompted\_sentence = tokenize\(prompted\_sentence\).to\(device\)
        class\_embeddings = model.encode\_text\(prompted\_sentence\)
        class\_embeddings /= class\_embeddings.norm\(dim=\-1, keepdim=True\)
        class\_embedding = class\_embeddings.mean\(dim=0\)
        class\_embedding /= class\_embedding.norm\(\)
        text\_features.append\(class\_embedding\)
    text\_features = torch.stack\(text\_features, dim=1\).to\(device\).t\(\)

    return text\_features
```

最后选择$\mathcal{S}$的第二维度作为异常检测分类问题的结果。

```
text\_probs = \(100.0 \* image\_features \@ text\_features.T\).softmax\(dim=\-1\)
results\['pr\_sp'\].append\(text\_probs\[0\]\[1\].cpu\(\).item\(\)\)
```

分割任务

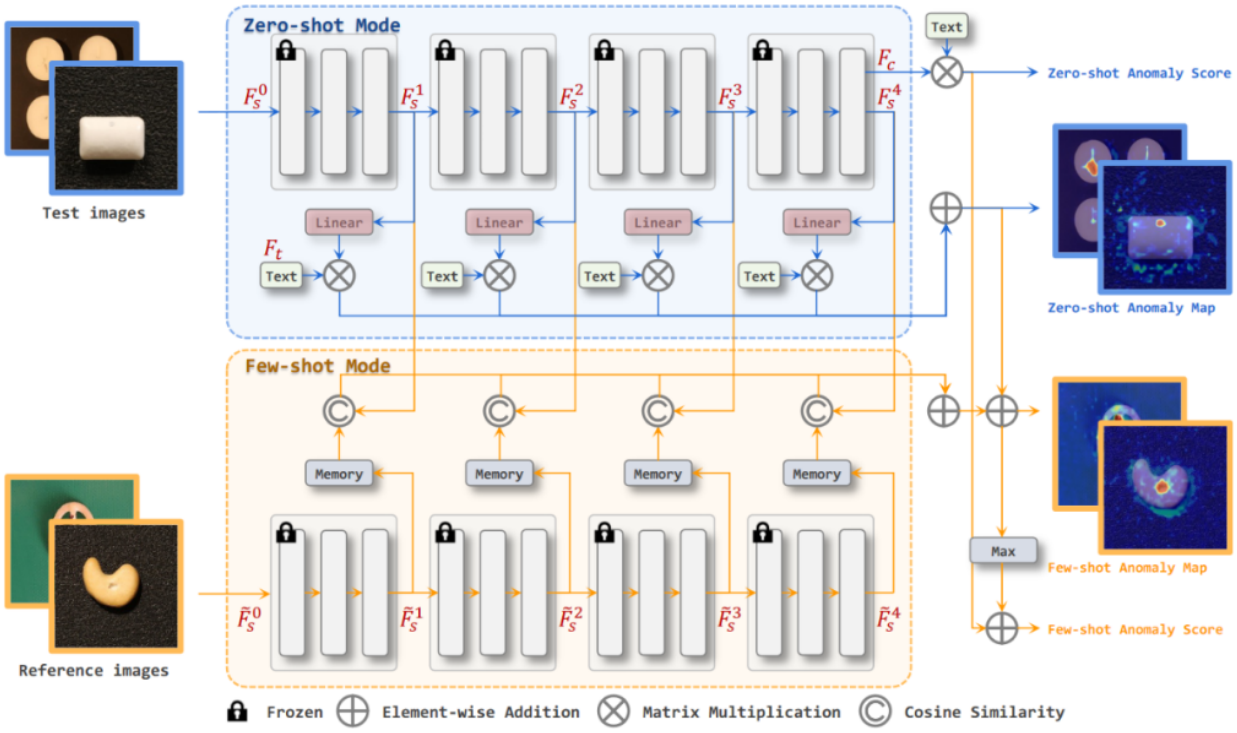


Figure 1. **Overall diagram of our solution**. *1)* The blue dashed box represents the pipeline for the zero-shot settiing. The "**Linear**" components denote additional linear layers and "**Text**" indicates the corresponding text features. Note that "**Text**" in this Figure is used to represent the same value. *2)* The orange dashed box represents the pipeline for the few-shot setting. The "**memory**" components represent the memory banks. The symbol with a letter C inside a circle denotes the calculation of cosine similarity.

在zero-shot分割任务中，因为这个赛道允许使用外部数据，所以这里额外引入了linear layer去映射patch_tokens，然后基于每个patch_token去和文本特征做相似度计算，从而得到anomaly map。

如上图中zero-shot Mode部分，这里将图像编码器拆分为n个stages，每个stage都分别计算了图像特征：

$$\mathbf{M} = \sum_{n} \text{softmax}\left(F_s^{n\prime} F_t^T\right)$$

具体实现如下代码段：

```
patch\_tokens = linearlayer\(patch\_tokens\)
anomaly\_maps = \[\]
for layer in range\(len\(patch\_tokens\)\):
  patch\_tokens\[layer\] /= patch\_tokens\[layer\].norm\(dim=\-1, keepdim=True\)
  anomaly\_map = \(100.0 \* patch\_tokens\[layer\] \@ text\_features.T\)
  B, L, C = anomaly\_map.shape
  H = int\(np.sqrt\(L\)\)
  anomaly\_map = F.interpolate\(anomaly\_map.permute\(0, 2, 1\).view\(B, 2, H, H\),
  size=img\_size, mode='bilinear', align\_corners=True\)
  anomaly\_map = torch.softmax\(anomaly\_map, dim=1\)\[:, 1, :, :\]
  anomaly\_maps.append\(anomaly\_map.cpu\(\).numpy\(\)\)
anomaly\_map = np.sum\(anomaly\_maps, axis=0\)
```

Linear Layer的训练（CLIP部分的参数是冻结的）使用了focal loss和dice loss。

小样本异常检测设定

分类任务

对于few-shot设置，图像的异常预测来自两部分。第一部分与zero-shot设置相同。第二部分遵循许多AD方法中使用的常规方法，考虑anomaly map的最大值。所提方案将这两部分相加作为最终的异常得分。

分割任务

few-shot分割任务使用了memory bank，如图1中的黄色背景部分。

直白来说，就是查询样本和memory bank中的支持样本去做余弦相似度，再通过reshape得到anomaly map，最后再加到zero-shot得到的anomaly map上得到最后的分割预测。

另外在few-shot任务中没有再去fine-tune上文提到的linear layer，而是直接使用了zero-shot任务中训练好的权重。

实验

定性结果

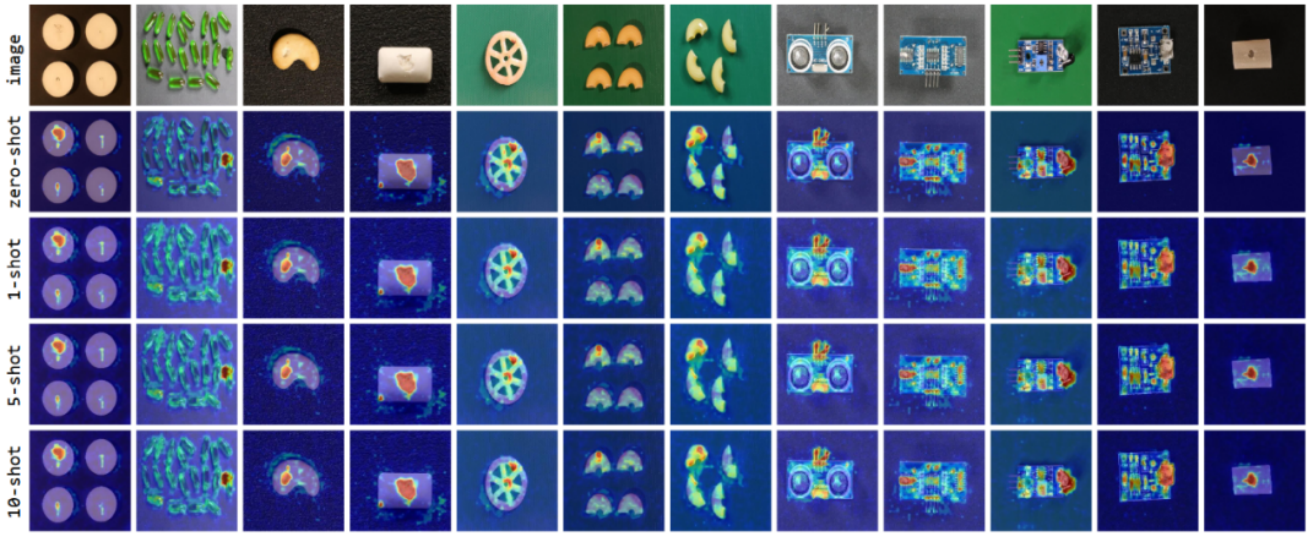


Figure 2. **Results visualizations on zero-/few-shot settings**. The first row shows the original image, the second row displays the zero-shot results, and the third to fifth rows present the results for 1-shot, 5-shot, and 10-shot, respectively.

简单来说，在简单一些的图像中zero-shot和few-shot上效果差不多，但面对困难任务时，few

-shot会改善一些。

定量结果

Table 1. Quantitative results of the top five participating teams on the **zero-shot** track leaderboard of the VAND 2023 Challenge.

Team Name	F1-max	F1-max-segm	F1-max-cls	Rank
AaxJIjQ	0.2788	0.2019	0.7742	5
MediaBrain	0.2880	0.1866	**0.7945**	4
Variance Vigilance Vanguard	0.3217	0.2197	0.7928	3
SegmentAnyAnomaly	0.3956	0.2942	0.7517	2
APRIL-GAN (Ours)	**0.4589**	**0.3431**	0.7782	1

Table 2. Quantitative results of the top five participating teams on the **few-shot** track leaderboard of the VAND 2023 Challenge.

Team Name	F1-max	F1-max-segm	F1-max-cls	Rank
VAND-Organizer (WinCLIP)	0.5323	0.4118	0.8114	5
PatchCore+	0.5742	0.4542	0.8423	3
MediaBrain	0.5763	0.4515	0.8480	2
Scortex	**0.5909**	**0.4706**	0.8399	1
APRIL-GAN (Ours)	0.5629	0.4264	**0.8687**	4

Table 3. Quantitative comparisons on the **MVTec AD** [2] dataset. We report the mean and standard deviation over 5 random seeds for each measurement. Bold indicates the best performance, while underline denotes the second-best result.

Setting	Method	AUROC-segm	F1-max-segm	AP-segm	PRO-segm	AUROC-cls	F1-max-cls	AP-cls
zero-shot	WinCLIP [8]	85.1	31.7	-	**64.6**	**91.8**	**92.9**	**96.5**
	Ours	**87.6**	**43.3**	**40.8**	44.0	86.1	90.4	93.5
1-shot	SPADE [4]	92.0±0.3	44.5±1.0	-	85.7±0.7	82.9±2.6	91.1±1.0	91.7±1.2
	PaDiM [5]	91.3±0.7	43.7±1.5	-	78.2±1.8	78.9±3.1	89.2±1.1	89.3±1.7
	PatchCore [14]	93.3±0.6	53.0±1.7	-	82.3±1.3	86.3±3.3	92.0±1.5	93.8±1.7
	WinCLIP [8]	**95.2±0.5**	**55.9±2.7**	-	87.1±1.2	**93.1±2.0**	**93.7±1.1**	**96.5±0.9**
	Ours	95.1±0.1	54.2±0.0	**51.8±0.1**	**90.6±0.2**	92.0±0.3	92.4±0.2	95.8±0.2
2-shot	SPADE [4]	91.2±0.4	42.4±1.0	-	83.9±0.7	81.0±2.0	90.3±0.8	90.6±0.8
	PaDiM [5]	89.3±0.9	40.2±2.1	-	73.3±2.0	76.6±3.1	88.2±1.1	88.1±1.7
	PatchCore [14]	92.0±1.0	50.4±2.1	-	79.7±2.0	83.4±3.0	90.5±1.5	92.2±1.5
	WinCLIP [8]	**96.0±0.3**	**58.4±1.7**	-	88.4±0.9	**94.4±1.3**	**94.4±0.8**	**97.0±0.7**
	Ours	95.5±0.0	55.9±0.5	**53.4±0.4**	**91.3±0.1**	92.4±0.3	92.6±0.1	96.0±0.2
4-shot	SPADE [4]	92.7±0.3	46.2±1.3	-	87.0±0.5	84.8±2.5	91.5±0.9	92.5±1.2
	PaDiM [5]	92.6±0.7	46.1±1.8	-	81.3±1.9	80.4±2.5	90.2±1.2	90.5±1.6
	PatchCore [14]	94.3±0.5	55.0±1.9	-	84.3±1.6	88.8±2.6	92.6±1.6	94.5±1.5
	WinCLIP [8]	**96.2±0.3**	**59.5±1.8**	-	89.0±0.8	**95.2±1.3**	**94.7±0.8**	**97.3±0.6**
	Ours	95.9±0.0	56.9±0.1	**54.5±0.2**	**91.8±0.1**	92.8±0.2	92.8±0.1	96.3±0.1

Table 4. Quantitative comparisons on the **VisA** [19] dataset. We report the mean and standard deviation over 5 random seeds for each measurement. Bold indicates the best performance, while underline denotes the second-best result.

Setting	Method	AUROC-segm	F1-max-segm	AP-segm	PRO-segm	AUROC-cls	F1-max-cls	AP-cls
zero-shot	WinCLIP [8]	79.6	14.8	-	56.8	**78.1**	**79.0**	81.2
	Ours	**94.2**	**32.3**	**25.7**	**86.8**	78.0	78.7	**81.4**
1-shot	SPADE [4]	95.6±0.4	35.5±2.2	-	84.1±1.6	79.5±4.0	80.7±1.9	82.0±3.3
	PaDiM [5]	89.9±0.8	17.4±1.7	-	64.3±2.4	62.8±5.4	75.3±1.2	68.3±4.0
	PatchCore [14]	95.4±0.6	38.0±1.9	-	80.5±2.5	79.9±2.9	81.7±1.6	82.8±2.3
	WinCLIP [8]	**96.4±0.4**	**41.3±2.3**	-	85.1±2.1	83.8±4.0	83.1±1.7	85.1±4.0
	Ours	96.0±0.0	38.5±0.3	**30.9±0.3**	**90.0±0.1**	**91.2±0.8**	**86.9±0.6**	**93.3±0.8**
2-shot	SPADE [4]	96.2±0.4	40.5±3.7	-	85.7±1.1	80.7±5.0	81.7±2.5	82.3±4.3
	PaDiM [5]	92.0±0.7	21.1±2.4	-	70.1±2.6	67.4±5.1	75.7±1.8	71.6±3.8
	PatchCore [14]	96.1±0.5	41.0±3.9	-	82.6±2.3	81.6±4.0	82.5±1.8	84.8±3.2
	WinCLIP [8]	**96.8±0.3**	**43.5±3.3**	-	86.2±1.4	84.6±2.4	83.0±1.4	85.8±2.7
	Ours	96.2±0.0	39.3±0.2	**31.6±0.3**	**90.1±0.1**	**92.2±0.3**	**87.7±0.3**	**94.2±0.3**
4-shot	SPADE [4]	96.6±0.3	43.6±3.6	-	87.3±0.8	81.7±3.4	82.1±2.1	83.4±2.7
	PaDiM [5]	93.2±0.5	24.6±1.8	-	72.6±1.9	72.8±2.9	78.0±1.2	75.6±2.2
	PatchCore [14]	96.8±0.3	43.9±3.1	-	84.9±1.4	85.3±2.1	84.3±1.3	87.5±2.1
	WinCLIP [8]	**97.2±0.2**	**47.0±3.0**	-	87.6±0.9	87.3±1.8	84.2±1.6	88.8±1.8
	Ours	96.2±0.0	40.0±0.1	**32.2±0.1**	**90.2±0.1**	**92.6±0.4**	**88.4±0.5**	**94.5±0.3**

极市干货

极视角动态：2023GCVC全球人工智能视觉产业与技术生态伙伴大会在青岛圆满落幕！ | 极视角助力构建城市大脑中枢，芜湖市湾沚区智慧城市运行管理中心上线！

数据集：面部表情识别相关开源数据集资源汇总 | 打架识别相关开源数据集资源汇总（附下载链接） | 口罩识别检测开源数据集汇总

经典解读：多模态大模型超详细解读专栏

极视角
EXTREME VISION
开创AI视觉算法商城

高空抛物识别

基于人工智能视觉分析技术，通过仰拍摄像头对高楼楼面状态进行实时监控，追踪物品的运行轨迹，若在检测区域内检测到物品从上往下掉落，且超过设定的触底边界线，则告警高空抛物；记录整个抛物过程，并将抛物轨迹可视化，从而定位抛物位置，提高工作人员管理效率。

应用场景

智慧城市 / 智慧安防 / 智慧楼宇 / 高空抛物检测 / 高空抛物识别

相关算法

徘徊识别　人员闯入识别

◀ 扫二维码查看
了解更多算法详情

点击阅读原文进入CV社区

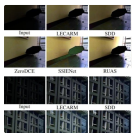
收获更多技术干货

阅读原文

喜欢此内容的人还喜欢

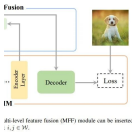
ICCV23｜将隐式神经表征用于低光增强，北大张健团队提出NeRCo

极市平台



ICCV 2023｜Pixel-based MIM: 简单高效的多级特征融合自监督方法

极市平台



ICCV 2023｜南开程明明团队提出适用于SR任务的新颖注意力机制（已开源）

极市平台