中山大學
SUN YAT-SEN UNIVERSITY

弱光增强 小组会议

分享人：王雅恬

中山大學
SUN YAT-SEN UNIVERSITY

STAR: A Structure-aware Lightweight Transformer for Real-time Image Enhancement

ICCV 2021

图像视频增强算法存在的问题

- 1. 图像分辨率很高，而计算资源有限，需在模型灵活性和计算效率间做出取舍。
- 2. 为了得到高质量的稳定结果，需要结合输入图像的结构和全局信息。特别是颜色白平衡、弱光增强、色调映射这些任务。甚至是去噪、去马赛克这种有local support的任务，结构感知、基于区域的方法会产生更好的结果。

已有的三类方法

- 1. 使用堆叠的深层CNN。在处理过程中，为了保留高频信息，图像的空间分辨率通常保持不变，这就造成了大量计算资源损耗和内存占用。
- 2. 去计算一个全局的调整函数。不够灵活，难以应对真实世界图像的复杂性。
- 3. 显式地使用一个分割网络，将图像根据语义信息分割为几个部分，再分别处理每个区域。这种方法的局限性是需要逐像素标注的数据集。

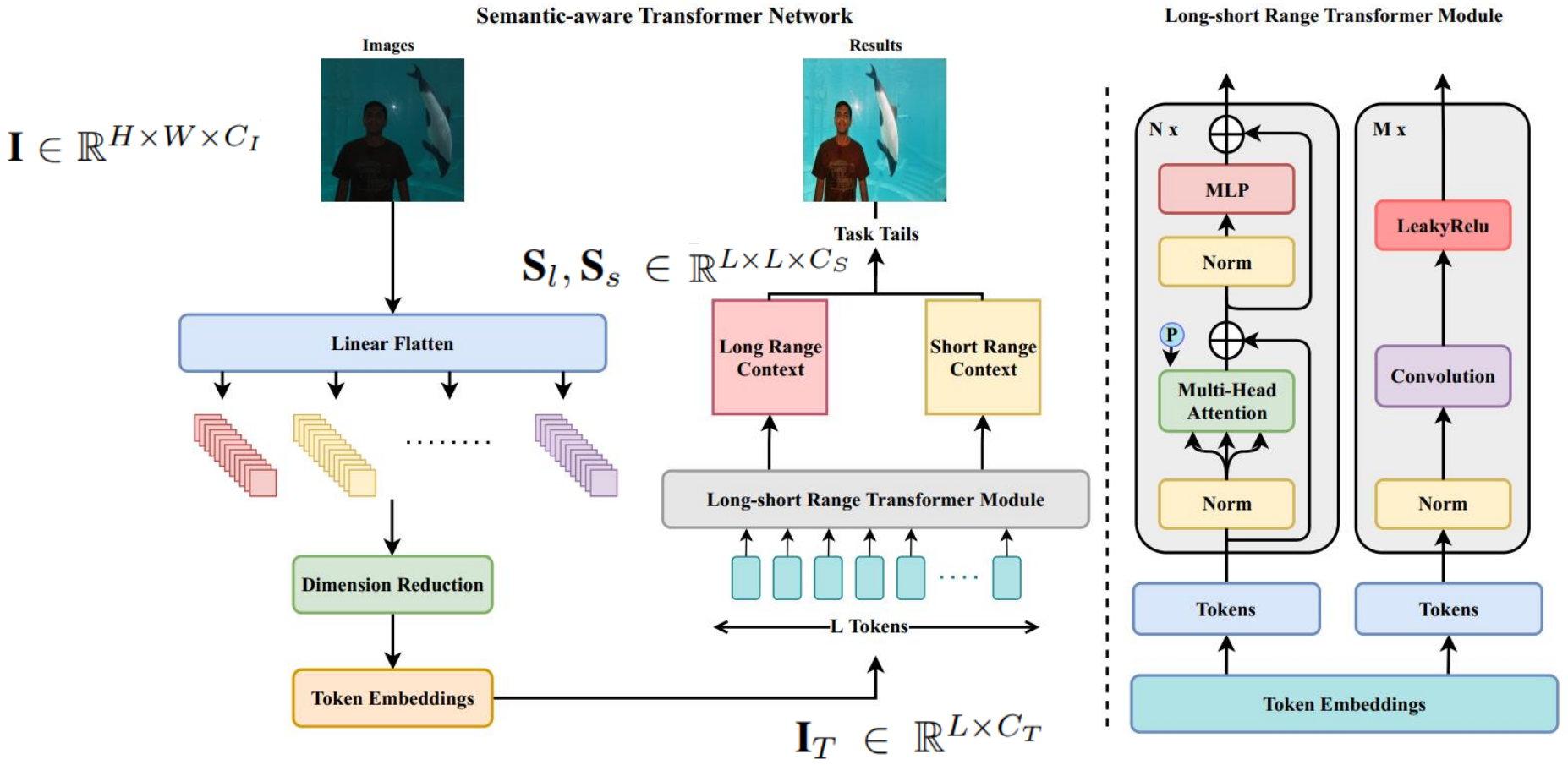
STAR (Structure-aware Transformer)

- 一个通用的轻量级backbone，可以应用到各种实时图像预处理任务上。
- 在图像块间捕捉long-range dependencies，自然、隐式地捕捉图像中的结构关系。图像块被转成token embeddings，STAR学习token间的关系，而不是像素间的关系。
- 基于Transformer模块，只包含多头自注意力和全连接层
- 隐式地学习语义结构，因此能比CNN传达出更加有意义的语义信息。

Transformer

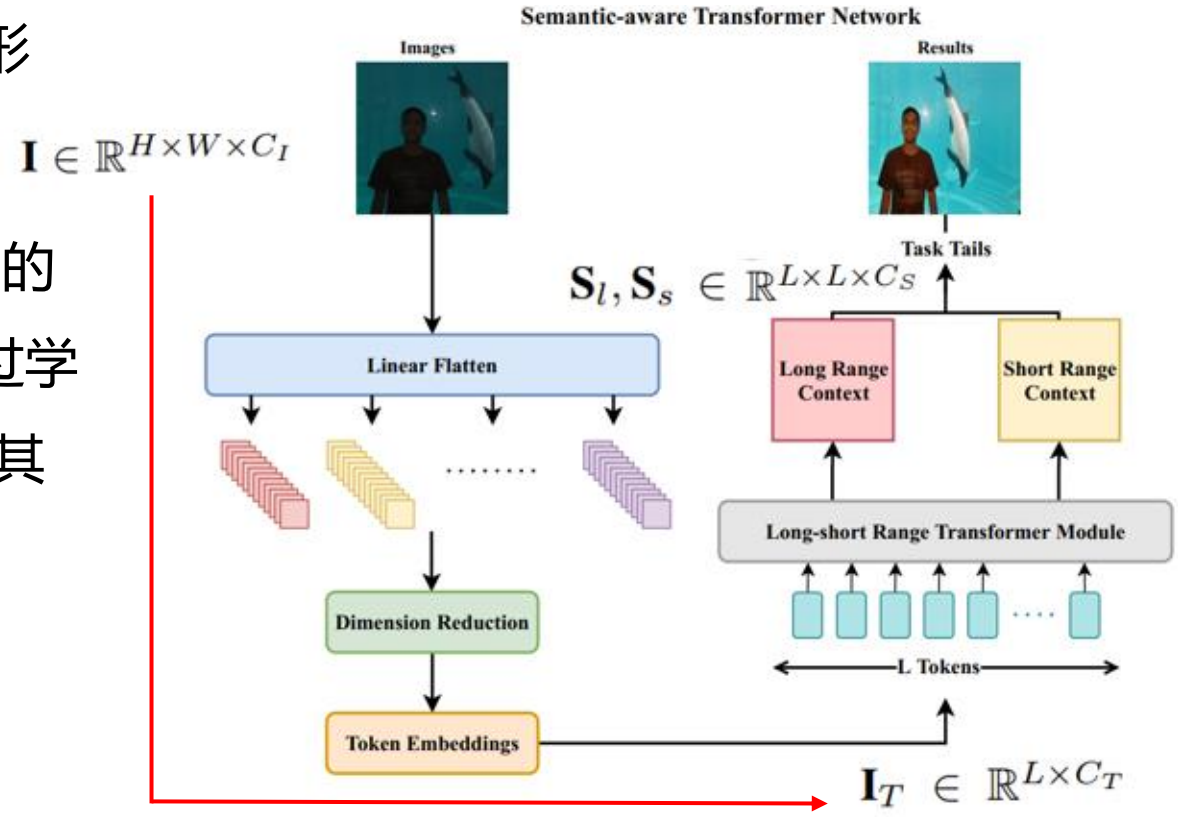
- Transformer是在2017年的《Attention is all you need》中被提出，用于解决机器翻译的任务。使用了多头自注意力机制（multi-head self-attention）和前馈多层感知机（feed-forward MLP）的堆叠，来学习长范围词语的相关性。
- Transformer的核心：用多头自注意力机制去特征化两个相隔的token间的依赖关系。这使得Transformer具有捕捉大而复杂的数据中隐含的相关性的潜能。

overview


Semantic-aware Transformer Network
Images
Results
$\mathbf{I} \in \mathbb{R}^{H \times W \times C_I}$
Task Tails
$\mathbf{S}_l, \mathbf{S}_s \in \mathbb{R}^{L \times L \times C_S}$
Linear Flatten
Long Range Context
Short Range Context
Long-short Range Transformer Module
Dimension Reduction
L Tokens
Token Embeddings
$\mathbf{I}_T \in \mathbb{R}^{L \times C_T}$
Long-short Range Transformer Module
N x
MLP
Norm
P
Multi-Head Attention
Norm
M x
LeakyRelu
Convolution
Norm
Tokens
Tokens
Token Embeddings

Tokenization

- 目的：将图像转为Transformer能输入的token形状。
- STAR的方法：先将全尺寸图像flatten成一序列的图像块。对每个图像块进行维度压缩。然后通过学习线性embedding来对每个图形块提取token。其中维度压缩的操作是模型节省内存的关键。
- 提到三种tokenization的策略： Linear Head，Conv Head，Mean Head



Linear Head

最简单的方法：将图像flatten成图像块，例如

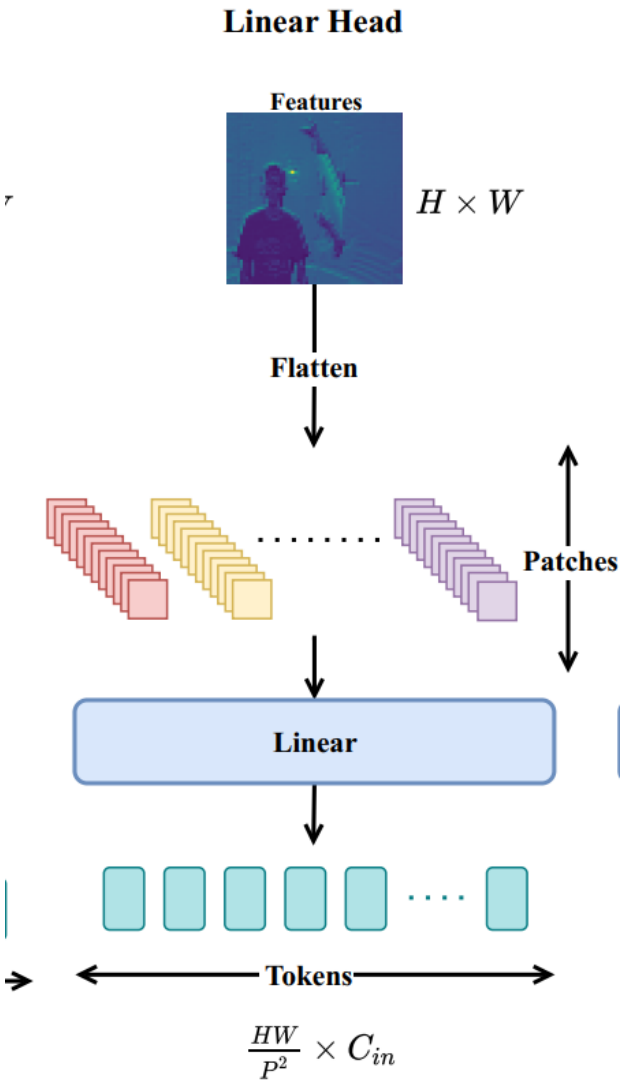
$$\mathbf{I} \in \mathbb{R}^{H \times W \times C_I} \longrightarrow \mathbf{T} \in \mathbb{R}^{(\frac{H}{P} \times \frac{W}{P})(P^2 \times C_I)}$$

图像尺寸：128*128*3，patch_size取16，就把图像分为了8*8块，

T的尺寸：(8*8)*（256*3）=64*768

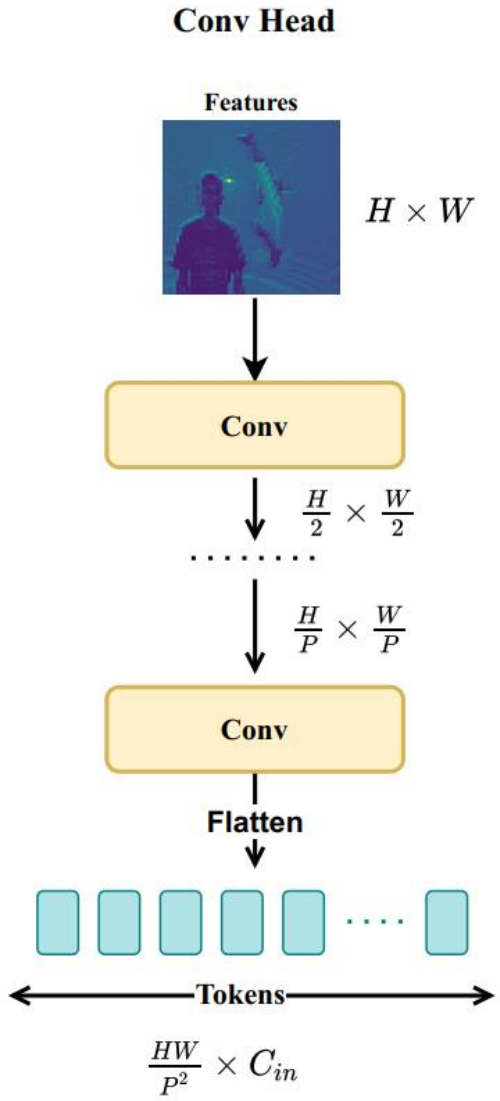
$$\mathbf{T} \in \mathbb{R}^{(\frac{H}{P} \times \frac{W}{P})(P^2 \times C_I)} \xrightarrow[P^2C \times C_T]{\text{线性映射计算量}} \mathbf{T} \in \mathbb{R}^{(\frac{H}{P} \times \frac{W}{P}) \times C_T}$$

但是，这种方法会消耗大量内存，需要巨大的训练参数量。



Conv Head

用一个预训练好的CNN，逐步减小输入图像的空间尺寸，到预定的尺寸。这时，可将输入矩阵的每一个数据，视作1*1的图像块。再进行flatten，刚好能输入Transformer。

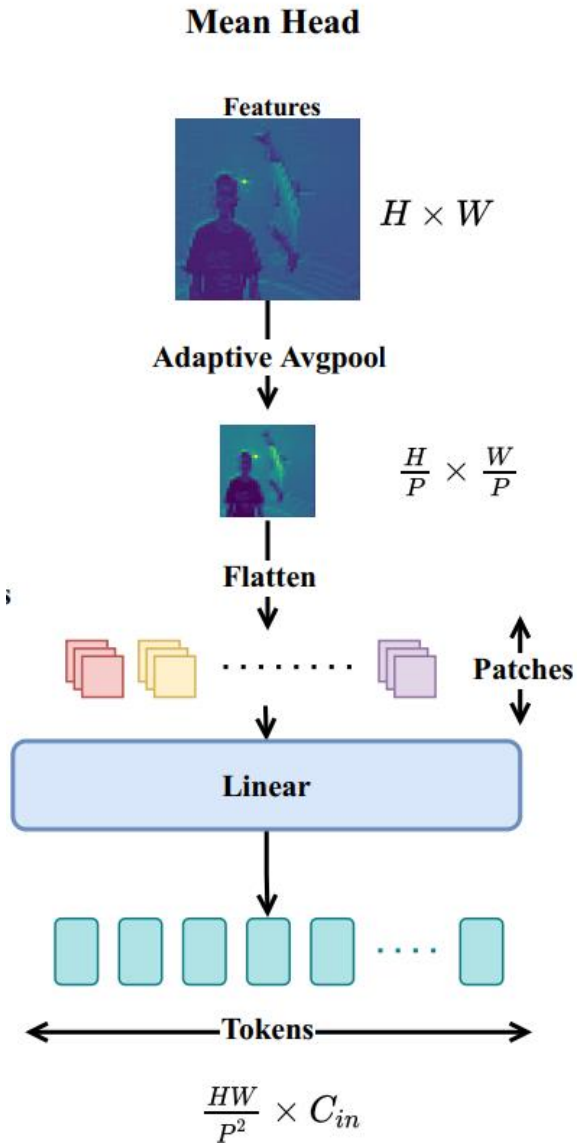


Mean Head

先用适应性均值池化来减小图像的空间大小，再进行线性映射。

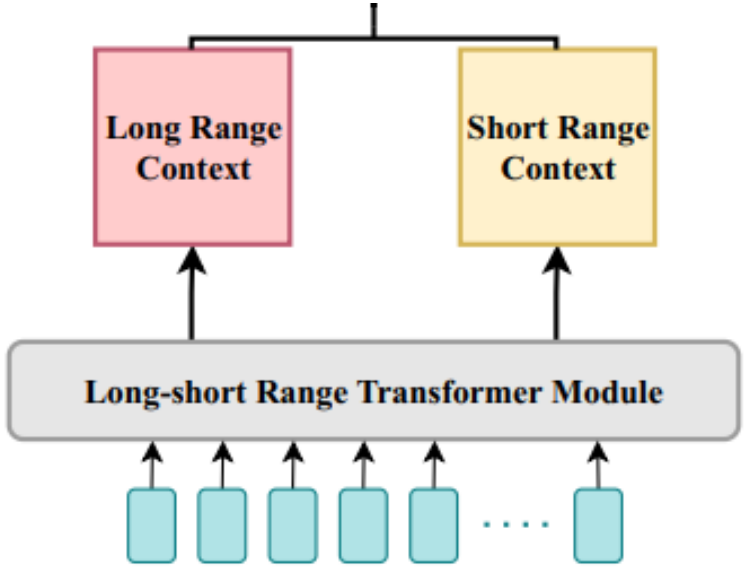
$$\mathbf{T} \in \mathbb{R}^{(\frac{H}{P}\times\frac{W}{P})(P^2\times C_I)} \xrightarrow[P^2C_I \times C_T]{\text{线性映射计算量}} \mathbf{T} \in \mathbb{R}^{(\frac{H}{P}\times\frac{W}{P})\times C_T}$$

最大程度上减小tokenization的复杂度，但是这个做法基于“纹理信息（例如图片角落）对任务没那么重要”的假设。



Long-short Range Transformer Module

- 两分支Transformer设计。
- 输入 $\mathbf{T} \in \mathbb{R}^{\frac{HW}{P^2} \times C_T}$ ，在维度上分为两个部分 $\{\mathbf{T}_{long}, \mathbf{T}_{short}\} \in \mathbb{R}^{\frac{HW}{P^2} \times \frac{C_T}{2}}$ 分别输入到两个分支中。这样的操作能有效降低整体计算量。
- 左侧长范围分支：基础Transformer模型
- 右侧短范围分支：CNN
- （与之前的工作把CNN嵌入Transformer不同，本文将两者设为平行处理）



Long-short Range Transformer Module

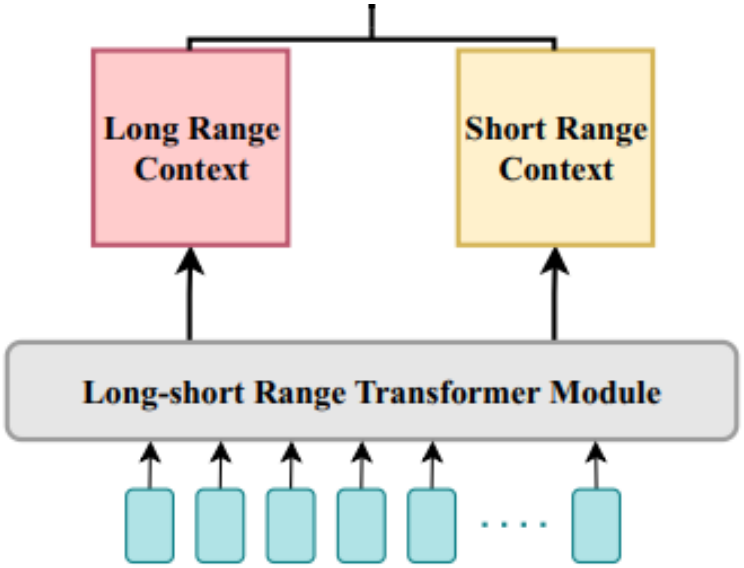
$$\mathbf{T}_0 = \mathbf{T}_{long} + p$$

$$\tilde{\mathbf{T}}_n = MSA(LN(\mathbf{T}_{n-1})) + \mathbf{T}_{n-1}, n = 1...\mathbf{N}$$

$$\mathbf{T}_n = MLP(LN(\tilde{\mathbf{T}}_n)) + \tilde{\mathbf{T}}_n, n = 1...\mathbf{N}$$

$$\mathbf{y_{long}} = LN(\mathbf{T}_N)$$

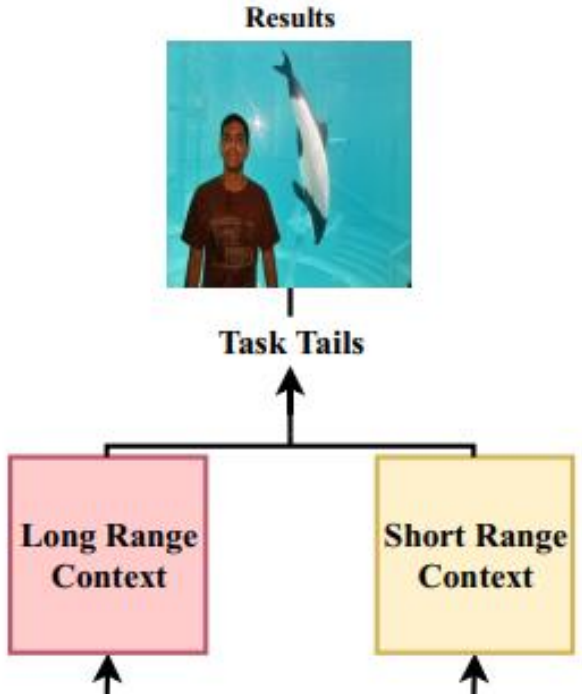
$$\mathbf{y_{short}} = CNN(LN(\mathbf{T}_{short}))$$



- MSA：多头自注意力
- MLP：多层感知机
- LN：layer normalization，同一个样本的不同通道做归一化

Long-short Range Transformer Module

- 得到结果： $\mathbf{y}_{\mathbf{long}}$ and $\mathbf{y}_{\mathbf{short}}$
- 增加一个task tail，就能运用于不同任务。



在三个任务上检验STAR效果

- **弱光增强中的曲线预测**
- 自动白平衡
- 修图

弱光增强的应用

- Zero-DCE思想：预测多条曲线 $\alpha_i \in [-1,1]^{H\times W}$

  对输入图像 $I(\mathbf{x})$，迭代多次进行以下计算：

$$LE(I(\mathbf{x});\alpha) = I(\mathbf{x}) + \alpha I(\mathbf{x})(1 - I(\mathbf{x}))$$

$$LE_i(I(\mathbf{x});\alpha_i) = LE_{i-1}(\mathbf{x}) + \alpha_i LE_{i-1}(\mathbf{x})(1 - LE_{i-1}(\mathbf{x}))$$

  在原方法中，曲线的参数是用CNN，通过特征级联，逐像素预测出来的。

- **STAR-DCE**：用上一步得到的 $\mathbf{y_{long}}$ and $\mathbf{y_{short}}$ 预测曲线：

$$\tilde{\alpha}_i = FC([y_{short}, y_{long}])$$

$$\alpha_i = \psi(tanh(\tilde{\alpha}_i))$$

  （其中FC是一个学习到的全连接层映射；φ是一个插值函数，将逐token的曲线映射到逐像素的曲线（因为token数和像素数不一致））

对比实验

- 其他实验设置均与原模型保持一致。
- DCE-Net的最大输出通道数为32，所以将Transformer的输入维度也设为32。

  在STAR中，每张图片被转化为32*32个token（32个图像块，每个图像块对应32维的token）

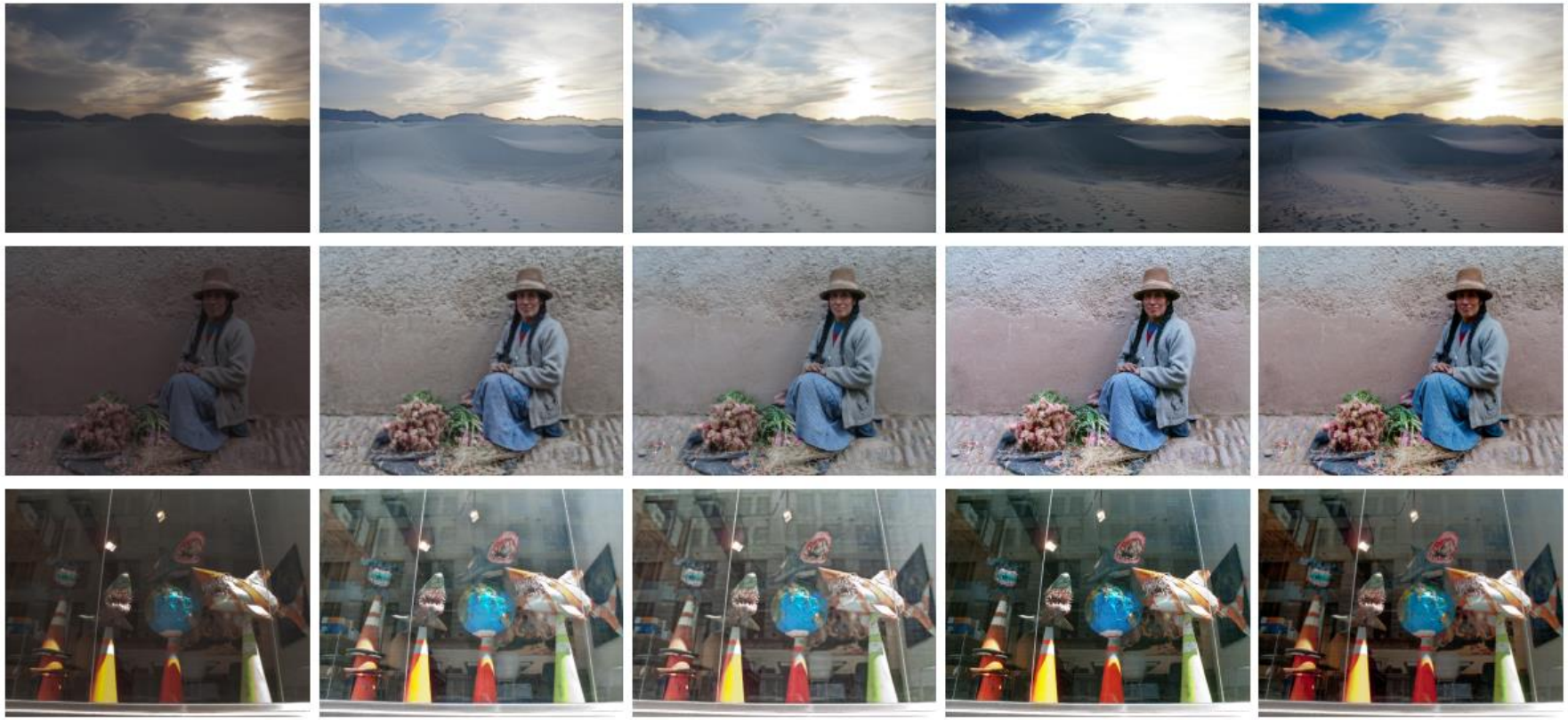
对比实验

	Tokenization	Branches	Parameters (K)	FLOPS (G)	PSNR (dB)	SSIM
DCE-Net	-	-	79.4	5.20	22.7	0.870
DCE-Net$_D$	-	-	79.4	0.51	22.2	0.866
STAR-DCE	Linear	1	79.6	0.17	24.0	0.882
STAR-DCE	Conv	1	32.6	0.10	24.5	0.894
STAR-DCE	Mean	1	23.3	0.07	24.5	0.892
STAR-DCE	Linear	2	77.8	0.15	24.1	0.885
STAR-DCE	Conv	2	30.8	0.08	24.4	0.894
STAR-DCE	Mean	2	**20.1**	**0.05**	24.5	0.893

注：DCE-NetD是在训练时下采样到32*32，与下面基于32*32token的方法作对比。

- 采用不同的tokenization策略，实验效果接近，但参数量和flops明显降低。
- 观察到，STAR模型几乎不会受益于复杂的tokenization方法，即使是运用最广泛的Linear Head。反而简单地均值池化，能节省超过一半的消耗，而不会造成效果下降
- 用两支路的设计，能进一步压缩模型。
- 结果印证了前文的假设：在这种调节任务中，纹理信息不是最重要的。

对比实验


Inputs
DCE-Net (5.2 GFLOPS)
DCE-Net_D (0.31 GFLOPS)
STAR-DCE (0.04 GFLOPS)
Reference