今天是我做了这么久基于Mamba视觉的核心模块——VMamba。

Day9: VMamba

先说一下我对VMamba的几个直观见解：

- VMamba的创新为NLP领域模型迁移到CV来提供了一个范本：使用各种方向展二维图像为一维序列——VMamba会先将输入图像分成多个补丁，但它不会像传统ViT那样简单地将这些补丁平展成一维序列。相反，它保留了补丁的二维结构，然后利用交叉扫描模块（CSM）从多个方向（如四个角到对角）扫描和重组这些补丁，以便更好地捕捉局部细节和全局上下文信息。
    - 之后我看到不论是在VMamba上魔改展开方向，还是xLSTM那边原作者水CV的贡献，都是类似于这种。
- 也算是一种即插即用模块，成为替代ViT的方案。比如Mamba-UNet还有VM-UNet都是直接封装套用。

1. 多方向平展图像增加因果联系

多向扫描，也是文章中提到的交叉扫描，在迁移NLP领域模型时是用必要的。由于视觉数据和文本不同，直观上是非因果性质，直接将这种策略应用于补丁化和展平的图像将不可避免地导致受限的感受野，因为无法估计相对于未扫描的补丁的关系。作者将这个问题称为"**方向敏感**"问题。

因此提出了**交叉扫描模块**（Cross-Scan Module，**CSM**）来解决它。CSM 不是以单向模式（列向或行向）遍历图像特征映射的空间域，而是采用四向扫描策略，即从特征映射的四个角到相对位置。这种策略确保特征映射中的每个元素从不同方向的所有其他位置集成信息，从而产生全局感受野，而不增加线性计算复杂性。

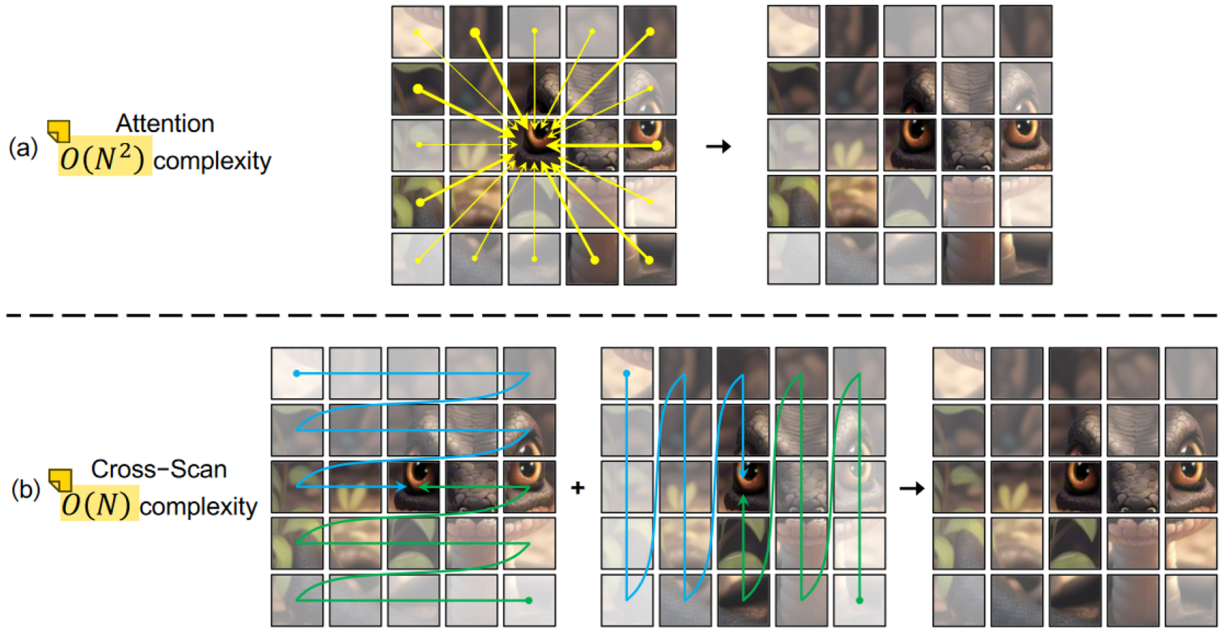
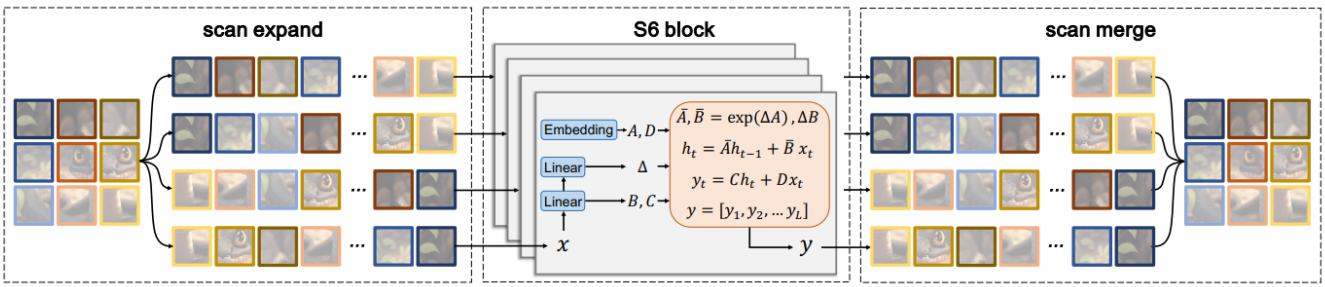


Figure 2: **Comparison of information flow: Attention vs. Cross-Scan Module (CSM).** (a) The attention mechanism uniformly integrates all pixels for the center pixel, resulting in $\mathcal{O}(N^2)$ complexity. (b) CSM integrates pixels from top-left, bottom-right, top-right, and bottom-left with $\mathcal{O}(N)$ complexity.

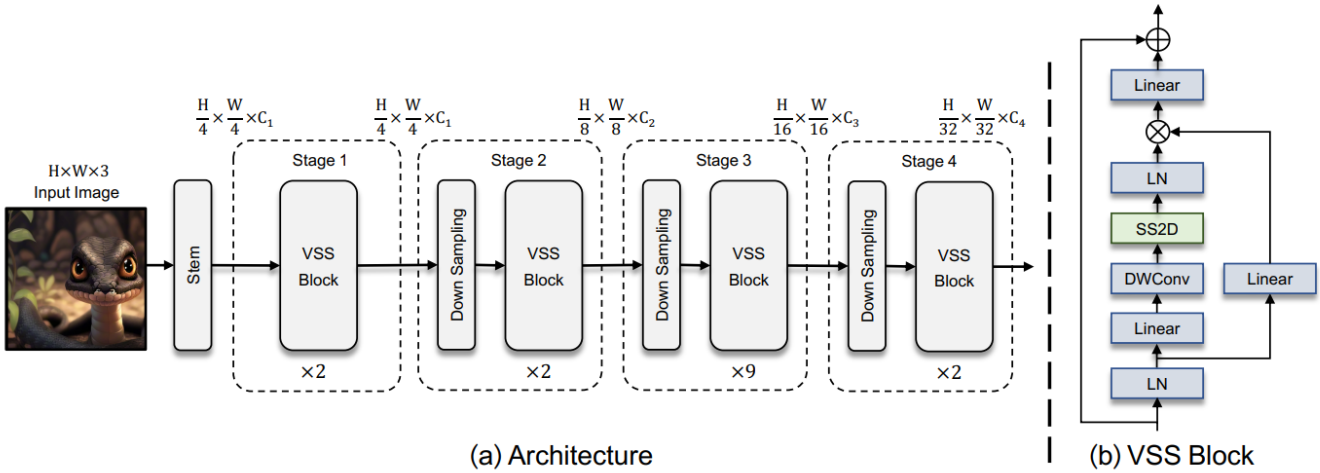
通过沿四个不同方向进行扫描：从左上到右下，从右下到左上，从右上到左下，从左下到右上。这样，任何像素都会从不同方向的所有其他像素中集成信息。然后将每个序列重新整形成单个图像，并将所有序列合并成一个新的序列。（VMamba通过交叉扫描模块对图像进行四个方向的扫描，生成四个方向的序列，然后对这些序列进行融合，以整合来自不同方向的上下文信息，从而获得全局的特征表达。）



二维选择性扫描机制SS2D的流程图↑

直接是用了一个4 * 4的卷积核心，步长为4，上来就是一顿分补丁，然后组合排序。

2. VMamba整体框架



VMamba-Tiny 的架构首先使用一个Stem节点将输入图像分割成多个patchs，类似于 ViTs，但没有将patchs进一步展平成 1-D 序列，这种修改保留了图像的 2D 结构。然后，堆叠多个 VSS 块，保持相同的维度，构成 “Stage 1”。之后通过patch合并操作对特征映射进行下采样，构建分层特征表示。随后再堆叠多个下采样和VSS块，创建“Stage 3” 和 “Stage 4”。这样就构成了一个类似CNN和ViT的基础模型，生成的架构可以在实际应用中作为对其它视觉模型的替代品。

其中VSS Block：输入经过初始线性嵌入层，输出分为两个信息流。一个流通过一个 3 × 3 的深度卷积层，然后通过 Silu 激活函数进入核心 SS2D 模块。SS2D 的输出通过一层标准化层，然后加到另一个信息流的输出上，该信息流经过 Silu 激活。由于 VMamba 的因果性质，不使用位置嵌入偏差。

```
VSSM(
  (patch_embed): PatchEmbed2D(
    (proj): Conv2d(3, 96, kernel_size=(4, 4), stride=(4, 4))
    (norm): LayerNorm((96,), eps=1e-05, elementwise_affine=True)
  )
  (pos_drop): Dropout(p=0.0, inplace=False)
  (layers): ModuleList(
    (0): VSSLayer(
      (blocks): ModuleList(
        (0): VSSBlock(
          (ln_1): LayerNorm((96,), eps=1e-05, elementwise_affine=True)
          (self_attention): SS2D(
            (in_proj): Linear(in_features=96, out_features=384, bias=False)
            (conv2d): Conv2d(192, 192, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=192)
            (act): SiLU()
            (out_norm): LayerNorm((192,), eps=1e-05, elementwise_affine=True)
            (out_proj): Linear(in_features=192, out_features=96, bias=False)
          )
          (drop_path): timm.DropPath(0.0)
        )
        (1): VSSBlock(
          (ln_1): LayerNorm((96,), eps=1e-05, elementwise_affine=True)
          (self_attention): SS2D(
            (in_proj): Linear(in_features=96, out_features=384, bias=False)
            (conv2d): Conv2d(192, 192, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=192)
            (act): SiLU()
            (out_norm): LayerNorm((192,), eps=1e-05, elementwise_affine=True)
            (out_proj): Linear(in_features=192, out_features=96, bias=False)
          )
          (drop_path): timm.DropPath(0.014285714365541935)
        )
      )
      (downsample): PatchMerging2D(
        (reduction): Linear(in_features=384, out_features=192, bias=False)
        (norm): LayerNorm((384,), eps=1e-05, elementwise_affine=True)
      )
    )
    (1): VSSLayer(
      (blocks): ModuleList(
        (0): VSSBlock(
          (ln_1): LayerNorm((192,), eps=1e-05, elementwise_affine=True)
          (self_attention): SS2D(
            (in_proj): Linear(in_features=192, out_features=768, bias=False)
            (conv2d): Conv2d(384, 384, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=384)
            (act): SiLU()
            (out_norm): LayerNorm((384,), eps=1e-05, elementwise_affine=True)
            (out_proj): Linear(in_features=384, out_features=192, bias=False)
```

```
          )
          (drop_path): timm.DropPath(0.02857142873108387)
        )
        (1): VSSBlock(
          (ln_1): LayerNorm((192,), eps=1e-05, elementwise_affine=True)
          (self_attention): SS2D(
            (in_proj): Linear(in_features=192, out_features=768, bias=False)
            (conv2d): Conv2d(384, 384, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=384)
            (act): SiLU()
            (out_norm): LayerNorm((384,), eps=1e-05, elementwise_affine=True)
            (out_proj): Linear(in_features=384, out_features=192, bias=False)
          )
          (drop_path): timm.DropPath(0.04285714402794838)
        )
      )
      (downsample): PatchMerging2D(
        (reduction): Linear(in_features=768, out_features=384, bias=False)
        (norm): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
      )
    )
    (2): VSSLayer(
      (blocks): ModuleList(
        (0): VSSBlock(
          (ln_1): LayerNorm((384,), eps=1e-05, elementwise_affine=True)
          (self_attention): SS2D(
            (in_proj): Linear(in_features=384, out_features=1536, bias=False)
            (conv2d): Conv2d(768, 768, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=768)
            (act): SiLU()
            (out_norm): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
            (out_proj): Linear(in_features=768, out_features=384, bias=False)
          )
          (drop_path): timm.DropPath(0.05714285746216774)
        )
        (1): VSSBlock(
          (ln_1): LayerNorm((384,), eps=1e-05, elementwise_affine=True)
          (self_attention): SS2D(
            (in_proj): Linear(in_features=384, out_features=1536, bias=False)
            (conv2d): Conv2d(768, 768, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=768)
            (act): SiLU()
            (out_norm): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
            (out_proj): Linear(in_features=768, out_features=384, bias=False)
          )
          (drop_path): timm.DropPath(0.0714285746216774)
        )
        (2): VSSBlock(
          (ln_1): LayerNorm((384,), eps=1e-05, elementwise_affine=True)
          (self_attention): SS2D(
            (in_proj): Linear(in_features=384, out_features=1536, bias=False)
```

```
            (conv2d): Conv2d(768, 768, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=768)
            (act): SiLU()
            (out_norm): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
            (out_proj): Linear(in_features=768, out_features=384, bias=False)
          )
          (drop_path): timm.DropPath(0.08571428805589676)
        )
        (3): VSSBlock(
          (ln_1): LayerNorm((384,), eps=1e-05, elementwise_affine=True)
          (self_attention): SS2D(
            (in_proj): Linear(in_features=384, out_features=1536, bias=False)
            (conv2d): Conv2d(768, 768, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=768)
            (act): SiLU()
            (out_norm): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
            (out_proj): Linear(in_features=768, out_features=384, bias=False)
          )
          (drop_path): timm.DropPath(0.10000000149011612)
        )
        (4): VSSBlock(
          (ln_1): LayerNorm((384,), eps=1e-05, elementwise_affine=True)
          (self_attention): SS2D(
            (in_proj): Linear(in_features=384, out_features=1536, bias=False)
            (conv2d): Conv2d(768, 768, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=768)
            (act): SiLU()
            (out_norm): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
            (out_proj): Linear(in_features=768, out_features=384, bias=False)
          )
          (drop_path): timm.DropPath(0.11428571492433548)
        )
        (5): VSSBlock(
          (ln_1): LayerNorm((384,), eps=1e-05, elementwise_affine=True)
          (self_attention): SS2D(
            (in_proj): Linear(in_features=384, out_features=1536, bias=False)
            (conv2d): Conv2d(768, 768, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=768)
            (act): SiLU()
            (out_norm): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
            (out_proj): Linear(in_features=768, out_features=384, bias=False)
          )
          (drop_path): timm.DropPath(0.12857143580913544)
        )
        (6): VSSBlock(
          (ln_1): LayerNorm((384,), eps=1e-05, elementwise_affine=True)
          (self_attention): SS2D(
            (in_proj): Linear(in_features=384, out_features=1536, bias=False)
            (conv2d): Conv2d(768, 768, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=768)
            (act): SiLU()
```

```
            (out_norm): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
            (out_proj): Linear(in_features=768, out_features=384, bias=False)
          )
          (drop_path): timm.DropPath(0.1428571492433548)
        )
        (7): VSSBlock(
          (ln_1): LayerNorm((384,), eps=1e-05, elementwise_affine=True)
          (self_attention): SS2D(
            (in_proj): Linear(in_features=384, out_features=1536, bias=False)
            (conv2d): Conv2d(768, 768, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=768)
            (act): SiLU()
            (out_norm): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
            (out_proj): Linear(in_features=768, out_features=384, bias=False)
          )
          (drop_path): timm.DropPath(0.15714286267757416)
        )
        (8): VSSBlock(
          (ln_1): LayerNorm((384,), eps=1e-05, elementwise_affine=True)
          (self_attention): SS2D(
            (in_proj): Linear(in_features=384, out_features=1536, bias=False)
            (conv2d): Conv2d(768, 768, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1), groups=768)
            (act): SiLU()
            (out_norm): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
            (out_proj): Linear(in_features=768, out_features=384, bias=False)
          )
          (drop_path): timm.DropPath(0.17142857611179352)
        )
      )
      (downsample): PatchMerging2D(
        (reduction): Linear(in_features=1536, out_features=768, bias=False)
        (norm): LayerNorm((1536,), eps=1e-05, elementwise_affine=True)
      )
    )
    (3): VSSLayer(
      (blocks): ModuleList(
        (0): VSSBlock(
          (ln_1): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
          (self_attention): SS2D(
            (in_proj): Linear(in_features=768, out_features=3072, bias=False)
            (conv2d): Conv2d(1536, 1536, kernel_size=(3, 3), stride=(1, 1), padding=
(1, 1), groups=1536)
            (act): SiLU()
            (out_norm): LayerNorm((1536,), eps=1e-05, elementwise_affine=True)
            (out_proj): Linear(in_features=1536, out_features=768, bias=False)
          )
          (drop_path): timm.DropPath(0.18571428954601288)
        )
        (1): VSSBlock(
          (ln_1): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
```

```
          (self_attention): SS2D(
            (in_proj): Linear(in_features=768, out_features=3072, bias=False)
            (conv2d): Conv2d(1536, 1536, kernel_size=(3, 3), stride=(1, 1), padding=
(1, 1), groups=1536)
            (act): SiLU()
            (out_norm): LayerNorm((1536,), eps=1e-05, elementwise_affine=True)
            (out_proj): Linear(in_features=1536, out_features=768, bias=False)
          )
          (drop_path): timm.DropPath(0.20000000298023224)
        )
      )
    )
  )
  (norm): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
  (avgpool): AdaptiveAvgPool1d(output_size=1)
  (head): Linear(in_features=768, out_features=1000, bias=True)
)
```

- **文章的缺点**

  没有指定方向的消融，因此之后出现的很多魔改最基本的都是添加了各种方向。

但整个模型还是很好用的，也确实给NLP迁移CV提供了范本。