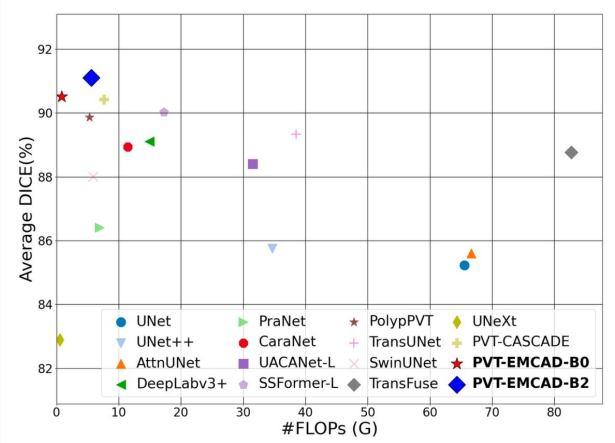
EMCAD: Efficient Multi-scale Convolutional Attention Decoding

Mohana Evuri - 23B1017
Gautam Siddharth K - 23B0957

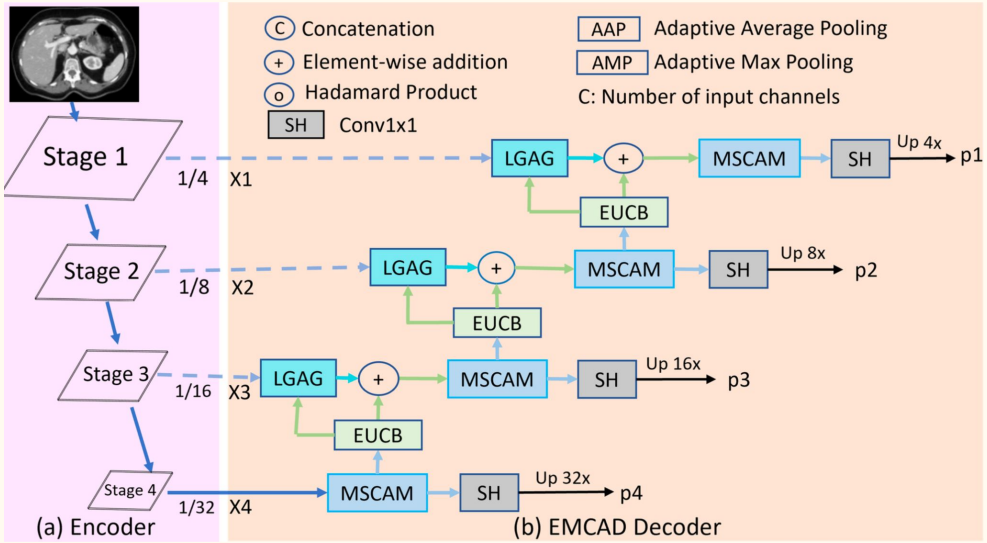
The paper: https://arxiv.org/abs/2405.06880

Introduction

- EMCAD is a new efficient multi-scale convolutional attention decoder, designed to optimize both performance and computational efficiency.
- It aims to achieve a better performance, with a great reduction in the number of parameters and FLOPS (FLoating point Operations Per Second).
- It claims to have a good performance on Medical Image Segmentation tasks.



Proposed Mechanism



Multi-scale Convolutional Attention Module - MSCAM

- Performs depth-wise convolutions at multiple scales; this refines the feature maps produced by vision encoders and enables capturing multi-scale salient features by suppressing irrelevant regions.
- **Channel Attention Block**: Focuses on which channels (features) to amplify.
- **Spatial Attention Block**: Focuses on where (i.e., which spatial locations) are important.
- **Multi-Scale Convolution Block**: Captures multi-resolution spatial features efficiently.
- MSCAM(x) = MSCB(SAB(CAB(x)))

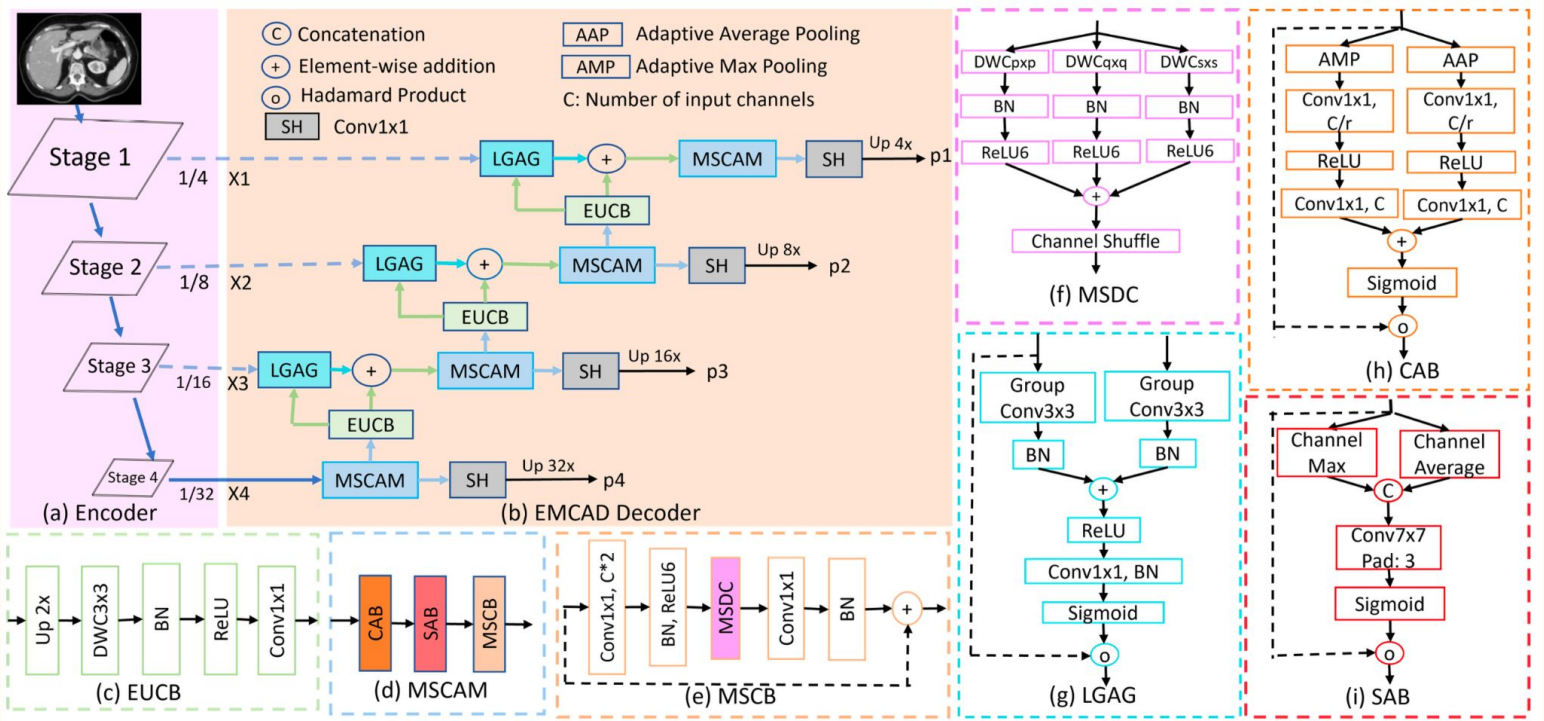
Efficient Up-Convolution Block - EUCB

- To upsample feature maps in the decoder path.
- **Structure**:
  - Up(x): Bilinear interpolation to double size.
  - Apply 3×3 depth-wise convolution → BN → ReLU.
  - Use 1×1 convolution to reduce channels for next stage fusion.
- $\mathrm{EUCB}(x) = \mathrm{C}_{1x1}(\mathrm{ReLU}(\mathrm{BN}(\mathrm{DWC}(\mathrm{Up}(x)))))$
- Using depth-wise convolution instead of a standard convolution makes EUCB very efficient, as each input channel is convolved separately with its own filter, and we just need to upsample and refine features, not mix channels.

Large-kernel Grouped Attention Gate (LGAG)

- To fuse refined features with the features from skip connections. By using larger kernel (3×3) group convolutions instead of point-wise convolutions in the design, we electively allow relevant spatial info from encoder to influence decoding, with minimal compute cost.
- **Structure:**
  - Apply 3×3 group convolutions separately on g (skip features) and x (upsampled decoder features).
  - Normalize + ReLU → fused → 1×1 conv → sigmoid → gating map.
  - Multiply gating map with decoder feature x.
- **Equations:**
  - $q_{att}(g, x) = \mathrm{ReLU}(\mathrm{BN}(GC_g(g) + \mathrm{BN}(GC_x(x))))$
  - $\mathrm{LGAG}(g, x) = x \circledast \sigma(\mathrm{BN}(C(q_{att}(g, x))))$

Overall Pipeline



Overall Architecture

- Two versions (mentioned in the paper) - PVT (Pyramid Vision Transformer): B0 and B2.
- We have replicated the code for **PVTv2b2**.

Aspect	PVT-EMCAD-B0	PVT-EMCAD-B2
Encoder	PVTv2-B0 (Tiny)	PVTv2-B2 (Standard)
Params	3.92M	26.76M
FLOPs	0.84G	5.6G
Use-case	Lightweight, real-time, edge	High-accuracy, server/GPU
Accuracy (Avg DICE)	Lower than B2 but better than most SOTA	Highest across all benchmarks
Speed	Faster	Slower
Best for	Mobile or limited-compute settings	High-accuracy clinical apps

Multi-stage Loss and Output Aggregation

- The loss:

$$L_{\text{total}} = \alpha L_{p_1} + \beta L_{p_2} + \gamma L_{p_3} + \zeta L_{p_4} + \delta L_{p_1+p_2+p_3+p_4}$$
$$\alpha = \beta = \gamma = \zeta = \delta = 1.0$$

- Multi-stage loss encourages learning at every stage.
- **Post-processing:**
  - Sigmoid → for binary segmentation
  - Softmax → for multi-class segmentation

Loss Function - Cross Entropy + DICE

- Weighted Sum of cross-entropy and dice loss.
- Cross-entropy treats each pixel independently.
- Due to tumour size being small compared to the rest of the image, most pixel values are easily deduced.

```
w_ce, w_dice = 0.3, 0.7

for s in ss:
    iout = 0.0
    if(s==[]):
        continue
    for idx in range(len(s)):
        iout += P[s[idx]]
    loss_ce = ce_loss(iout, label_batch[:].long())
    loss_dice = dice_loss(iout, label_batch, softmax=True)
    loss += (w_ce * loss_ce + w_dice * loss_dice)
```

Different Loss Function - Focal + Dice - Unified

- Focal Loss + Dice Loss
- Focal Loss cares a lot more about hard samples where it is unsure i.e., (1-P_t) is high.
- This aligns with our case where the model needs to decide the border of the tumor + the tumor size itself is small
- Reference - https://www.sciencedirect.com/science/article/pii/S0895611121001750

$$L_f = -\alpha_t \ (1 - P_t)^{\gamma} \ log(P_t)$$

$$\alpha_t = \begin{cases} \alpha & if\ y = 1 \\ 1 - \alpha & otherwise \end{cases}$$

```python
def unified_focal_loss(logits, target, smooth=1e-6, gamma=2.0):
    if logits.shape != target.shape:
        target = target.unsqueeze(1)

    logits = logits.view(-1)
    target = target.view(-1)

    probs = torch.sigmoid(logits)
    intersection = (probs * target).sum()
    uni = (2. * intersection + smooth) / (probs.sum() + target.sum() + smooth)

    bce_logits = nn.BCEWithLogitsLoss(reduction='none')(logits, target)
    focal = ((1 - torch.exp(-bce_logits)) ** gamma) * bce_logits
    focal = focal.mean()

    return 0.5 * (1 - uni) + 0.5 * focal
```

Complexity of our Model

- For **DICE:**
  - FLOPs: 5.28 GMac
  - Params: 29.13 M
- For **Unified:**
  - FLOPs: 5.28 GMac
  - Params: 29.13 M

Parameters

- Learning Rate - 5e-5
- Focal Loss
  - Gamma - 2 (quadratic, gives high weight to hard samples)
  - w_fe = w_dice = 0.5
- Dice Loss
  - w_dice = 0.7, w_ce = 0.3
- Batch Size - 8
- Train-Test Split - 80 - 20

Final Results

Our Implementation

- Dice
  - F1: 0.9184
  - Dice: 0.8491
- Unified
  - F1: 0.9230
  - Dice: 0.8569

Their Implementation

- PVT-EMCAD-B0 (Basic)
  - F1: 0.9171

  PVT-EMCAD-B2 (Complex)
  - F1: 0.9231

F1 Score: Harmonic Mean of Precision and Recall
IoU: The overlap between predicted and true regions relative to their Union