



# **Khoa Công Nghệ Thông Tin**

Chuyên ngành: **Khoa Học Máy Tính**

Bộ Môn: **Thị Giác Máy Tính**

Lớp 67CS2

## **Báo Cáo Đồ Án**

**Chủ Đề: Background Segmentation (Human)**

Giảng Viên Hướng Dẫn: Ths. Nguyễn Đình Quý

### **Nhóm 6:**

Hoàng Quốc Vũ	4005667
Tôn Văn Dân	0113167
Nguyễn Duy Hiếu	0178167
Bạch Hưng Bảo	0016867

# Giới Thiệu về Phân Đoạn Nền

Phân đoạn nền (Background Segmentation) là một nhiệm vụ quan trọng trong lĩnh vực thị giác máy tính (Computer Vision), với mục tiêu chính là tách biệt đối tượng tiền cảnh (foreground) như con người, xe cộ, đồ vật khỏi phần nền phía sau (background) trong ảnh hoặc video. Khác với các bài toán phân đoạn ngữ nghĩa thông thường, phân đoạn nền nhấn mạnh vào việc loại bỏ thông tin không quan trọng để làm nổi bật đối tượng chính, từ đó giúp các hệ thống xử lý ảnh tập trung hiệu quả hơn vào vùng quan trọng.

Trong đề tài này, nhóm đã triển khai một hệ thống phân đoạn nền sử dụng mô hình DeepLabV3+, một mô hình mạng nơ-ron sâu tiên tiến kết hợp giữa kiến trúc Encoder-Decoder và các module trích xuất đặc trưng đa tỉ lệ (ASPP - Atrous Spatial Pyramid Pooling). Mô hình sử dụng ResNet50 làm backbone nhằm tận dụng khả năng trích xuất đặc trưng mạnh mẽ, giúp cải thiện độ chính xác trong các bối cảnh phức tạp như nền đa dạng, ánh sáng không đồng đều hoặc hình ảnh bị nhiễu.

Thông qua quá trình huấn luyện, mô hình DeepLabV3+ với ResNet50 đã cho kết quả tốt, với chỉ số mIoU đạt trên 96% trên tập huấn luyện và khoảng 93% trên tập kiểm tra, thể hiện khả năng học đặc trưng và phân đoạn nền hiệu quả, phù hợp với các ứng dụng thực tế.

## 1.2 Đặt vấn đề

Bài toán phân đoạn nền tuy phổ biến nhưng vẫn gặp nhiều thách thức thực tế:

- Chất lượng ảnh đầu vào thấp hoặc nhiều ảnh khiến mô hình khó phân biệt được foreground và background.
- Yêu cầu tốc độ xử lý cao trong các ứng dụng thời gian thực như AR, camera giám sát, đòi hỏi mô hình vừa nhanh vừa chính xác.
- Độ phức tạp của nền: các họa tiết hoặc vật thể trong nền có thể gây nhầm lẫn với đối tượng chính.
- Điều kiện ánh sáng thay đổi, góc chụp nghiêng có thể làm sai lệch hình dạng và màu sắc của đối tượng.
- Kích thước và hình dạng đa dạng của đối tượng yêu cầu mô hình phải linh hoạt và tổng quát tốt.
- Độ phân giải ảnh thấp dẫn đến mất chi tiết biên, gây khó khăn cho việc phân đoạn chính xác.

Chính vì những khó khăn trên, việc áp dụng các mô hình học sâu như DeepLabV3+ có khả năng học được các đặc trưng sâu và trừu tượng là cần thiết. Dự án này hướng tới mục tiêu tách phần nền và phần tiền cảnh trong ảnh sao cho hệ thống có thể tập trung vào vùng chứa thông tin quan trọng, từ đó nâng cao chất lượng các bước xử lý phía sau như nhận diện, theo dõi, hoặc chỉnh sửa ảnh.

## Ứng dụng

Một hệ thống phân đoạn nền hiệu quả có thể mang lại nhiều lợi ích thiết thực: - Tách rõ vùng đối tượng và nền: xác định chính xác đâu là vật thể chính (con người, vật dụng...) và đâu là nền, làm nền tảng cho nhiều bài toán xử lý ảnh cao hơn. - Tiền xử lý cho các hệ thống khác: như nhận diện hành động, phân loại ảnh, nhận dạng khuôn mặt, theo dõi đối tượng (object

tracking). - Xóa bỏ thông tin nền không cần thiết: giúp mô hình học sâu tập trung hơn vào vùng quan trọng, giảm nhiễu và tăng hiệu quả học. - Chỉnh sửa ảnh/video: như xóa phông, thay nền, làm mờ nền, đặc biệt hữu ích trong nhiếp ảnh, thiết kế đồ họa và truyền thông. - Ứng dụng thời gian thực: như camera an ninh thông minh, AR/VR, hội nghị trực tuyến (Zoom, Google Meet...), nơi cần tách người khỏi nền nhanh và chính xác.

## 1.4. Các Nghiên Cứu Liên Quan

Các nghiên cứu nổi bật liên quan đến phân đoạn nền và các mô hình tương tự: - Chen et al. (2018) giới thiệu DeepLabV3+, cải tiến phân đoạn ngữ nghĩa với Encoder-Decoder, đạt IoU 89% trên Cityscapes. - He et al. (2015) phát triển ResNet, giải quyết vanishing gradient, trở thành backbone phổ biến. - Ronneberger et al. (2015) đề xuất U-Net, tối ưu cho phân đoạn y tế với skip connections. - Long et al. (2015) phát triển FCN, đặt nền móng cho phân đoạn pixel-wise.

## Đóng Góp vào Đề Tài

Phát triển mô hình DeepLabV3+ sử dụng ResNet50: triển khai cụ thể mô hình trong môi trường PyTorch để giải quyết bài toán phân đoạn nền.

Xây dựng pipeline xử lý hoàn chỉnh: từ bước tải và tiền xử lý dữ liệu đến huấn luyện, đánh giá và thử nghiệm mô hình.

Tối ưu hóa hiệu suất và độ chính xác: cải thiện loss và tăng mIoU qua từng epoch, kết quả đạt trên 96% mIoU ở tập huấn luyện và trên 93% ở tập kiểm tra.

Ứng dụng kết quả vào thực tế: hệ thống có thể mở rộng để dùng cho AR, chỉnh sửa ảnh, xóa nền trong hội họp online hoặc hỗ trợ các bài toán thị giác máy tính khác.

Phân tích ảnh hưởng điều kiện môi trường: thử nghiệm mô hình trong điều kiện nền phức tạp, ánh sáng không đồng đều, chứng minh khả năng tổng quát của mô hình.

# Lý Thuyết Nền Tảng

## Giới Thiệu về ResNet50

ResNet50, một biến thể của Residual Network do Microsoft Research giới thiệu năm 2015, gồm 50 lớp và sử dụng Residual Connections để khắc phục vanishing gradient trong mạng sâu. Kiến trúc bao gồm 4 khối residual với các đặc điểm:

- Residual Connections truyền thông tin hiệu quả, giảm mất mát dữ liệu.
- khối (Stage 2-5) với Conv 7x7, BatchNorm, ReLU, và Identity Blocks.
- Sử dụng kernel 3x3 và 1x1, tối ưu hóa tính toán.
- Tích hợp Atrous Convolution trong DeepLabV3+ để mở rộng receptive field.

ResNet50 đóng vai trò backbone, trích xuất đặc trưng cấp thấp (128x128) và cấp cao (32x32x1024) cho phân đoạn nền.

Các đặc điểm chính ResNet50: - Sử dụng Residual Connections (Kết nối tắt): Cho phép truyền thông tin hiệu quả hơn giữa các lớp, giúp tránh mất mát thông tin khi mạng trở nên sâu hơn. - Hỗ trợ lan truyền gradient dễ dàng hơn, giúp giảm thiểu vấn đề gradient biến mất trong quá trình huấn luyện mô hình sâu. - Cải thiện khả năng học của mô hình mà không làm tăng đáng kể số lượng tham số, giữ được hiệu suất cao mà không gây quá tải tính toán.

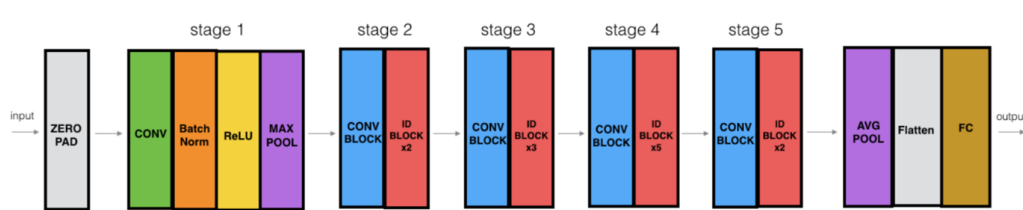
Kiến trúc 50 lớp với 4 khối residual chính: - Mạng bao gồm 4 khối residual (Residual Blocks), mỗi khối có 3 lớp convolution. - Các lớp trong mỗi khối residual bao gồm Batch Normalization và ReLU Activation giúp tăng tính ổn định trong quá trình huấn luyện, giảm hiện tượng overfitting. - Sử dụng các kernel có kích thước nhỏ (3x3 và 1x1) để tối ưu hóa hiệu suất tính toán, giúp mô hình hoạt động nhanh hơn nhưng vẫn giữ được độ chi tiết của ảnh.

Khả năng mở rộng receptive field với Atrous Convolution:

Khi tích hợp vào DeepLabV3+, ResNet-50 có thể sử dụng Atrous Convolution để tăng receptive field mà không làm tăng số tham số.

Điều này giúp mô hình có thể thu thập thông tin từ một vùng không gian rộng hơn, hỗ trợ tốt hơn cho các bài toán segmentation yêu cầu độ chính xác cao.

Xây dựng mạng ResNet50:



Hình 1: Kiến trúc mạng ResNet50

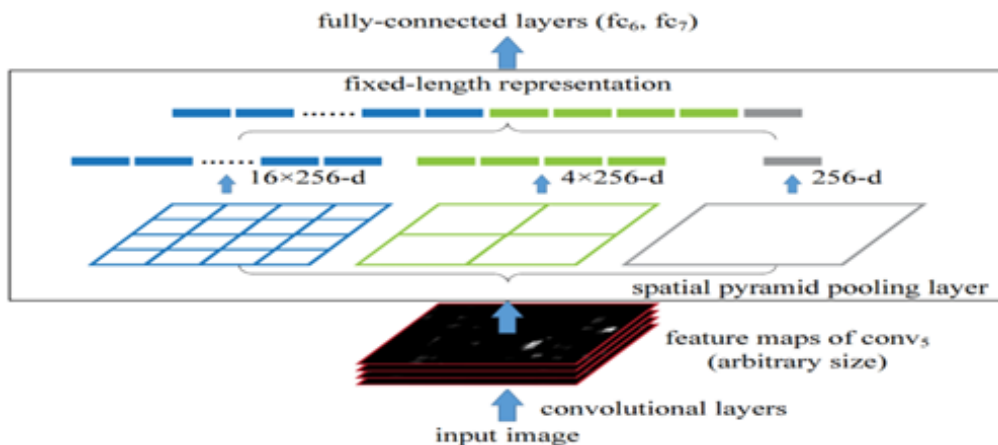
"ID BLOCK" trong hình trên là viết tắt của từ Identity block và ID BLOCK x3 nghĩa là có 3 khối Identity block chồng lên nhau. Nội dung hình trên như sau:

- Zero-padding : Input với (3,3)
- Stage 1 : Tích chập (Conv1) với 64 filters với shape(7,7), sử dụng stride (2,2). BatchNorm, MaxPooling (3,3).
- Stage 2 : Convolutional block sử dụng 3

filter với size  $64 \times 64 \times 256$ ,  $f=3$ ,  $s=1$ . Có 2 Identity blocks với filter size  $64 \times 64 \times 256$ ,  $f=3$ . - Stage 3 : Convolutional sử dụng 3 filter size  $128 \times 128 \times 512$ ,  $f=3, s=2$ . Có 3 Identity blocks với filter size  $128 \times 128 \times 512$ ,  $f=3$ . - Stage 4 : Convolutional sử dụng 3 filter size  $256 \times 256 \times 1024$ ,  $f=3, s=2$ . Có 5 Identity blocks với filter size  $256 \times 256 \times 1024$ ,  $f=3$ . - Stage 5 : Convolutional sử dụng 3 filter size  $512 \times 512 \times 2048$ ,  $f=3, s=2$ . Có 2 Identity blocks với filter size  $512 \times 512 \times 2048$ ,  $f=3$ . - The 2D Average Pooling : sử dụng với kích thước  $(2,2)$ . - The Flatten. - Fully Connected (Dense) : sử dụng softmax activation.

## Giới thiệu ASPP

ASPP giúp mở rộng trường nhìn mà không làm mất độ phân giải không gian. Nó bao gồm:



Hình 2: Spatial Pyramid Pooling (SPP) – Kỹ thuật quan trọng xử lý đầu vào có kích thước thay đổi

- $1 \times 1$  Convolution + Batch Normalization + ReLU.
- $3 \times 3$  Atrous Convolutions với các tỷ lệ khác nhau ( $\text{rate} = 6, 12, 18$ ) để thu thập thông tin ngữ cảnh ở nhiều tỷ lệ.

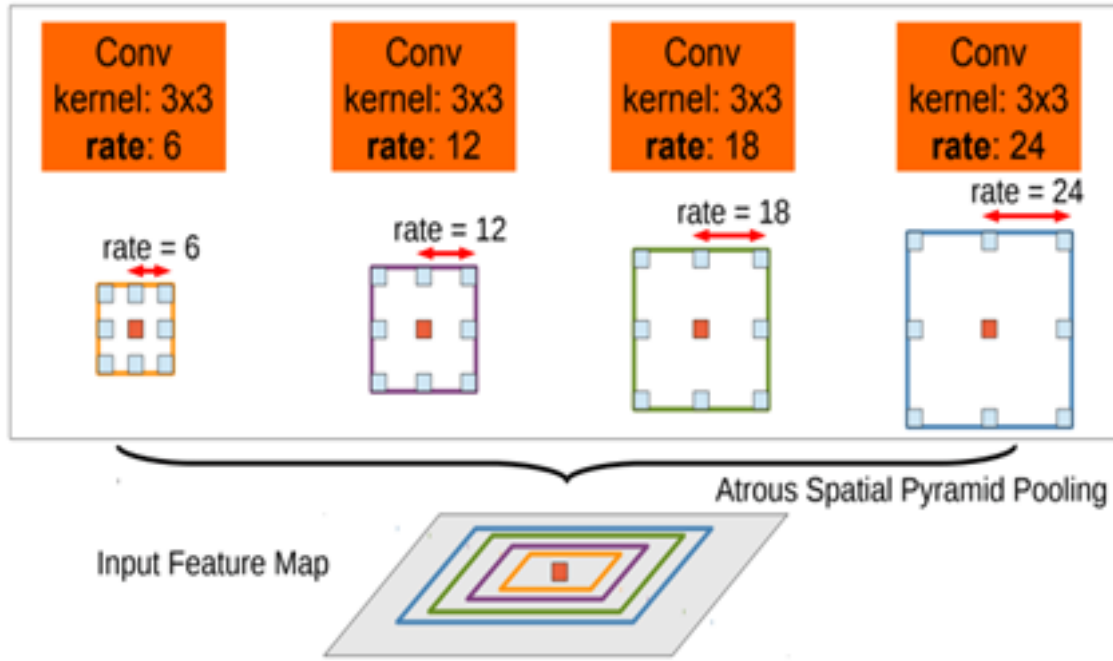
*Để phân loại điểm ảnh trung tâm (màu cam), ASPP khai thác các tính năng đa tỷ lệ bằng cách sử dụng nhiều bộ lọc song song với các tỷ lệ khác nhau. Trường nhìn (FOV) hiệu quả được hiển thị bằng các màu khác nhau.*

- Image Pooling: Global average pooling giúp nắm bắt thông tin toàn cảnh.
- Các đặc trưng từ ASPP được kết hợp lại và truyền qua một lớp  $1 \times 1$  Convolution + Batch Normalization + ReLU.

## Giới Thiệu về DeepLabV3+

DeepLabV3+, phát triển bởi Google năm 2018, là mô hình phân đoạn ngữ nghĩa cải tiến, kết hợp: - Encoder: ResNet50 trích xuất đặc trưng, ASPP (Atrous Spatial Pyramid Pooling) với dilation rates 6, 12, 18 thu thập ngữ cảnh đa tỷ lệ. - Decoder: Kết hợp đặc trưng cấp thấp và ASPP, sử dụng upsampling và Conv  $3 \times 3$  để phục hồi chi tiết, tạo mặt nạ  $512 \times 512 \times 1$ .

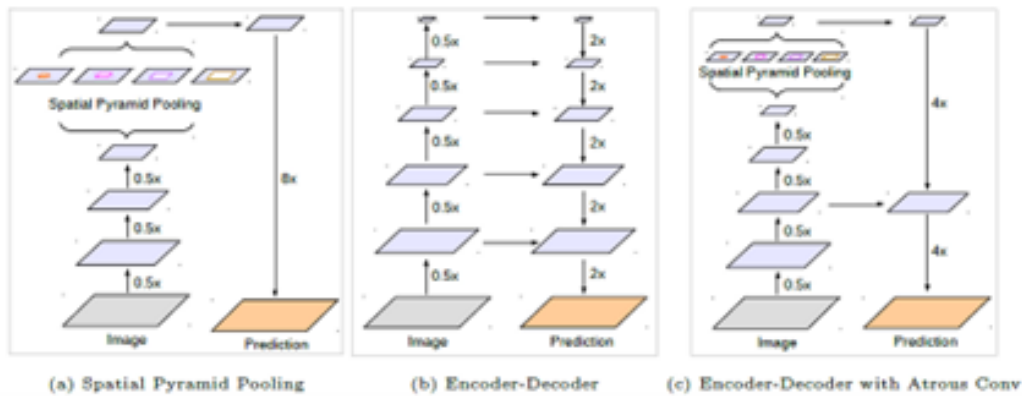
Mô hình này cải thiện ranh giới phân đoạn, phù hợp cho Background Segmentation.



Hình 3: Các nhánh atrous convolution với tỷ lệ khác nhau trong module ASPP (minh họa các trường nhìn khác nhau)

## Cấu trúc và lý thuyết của DeepLab v3+

DeepLab v3+ là sự kết hợp giữa DeepLab v3 (phiên bản trước đó) và một cấu trúc mã hóa-giải mã (Encoder-Decoder). Dưới đây là các thành phần chính:



Hình 4: a) DeepLabv3 b) Unet c) DeepLabv3+

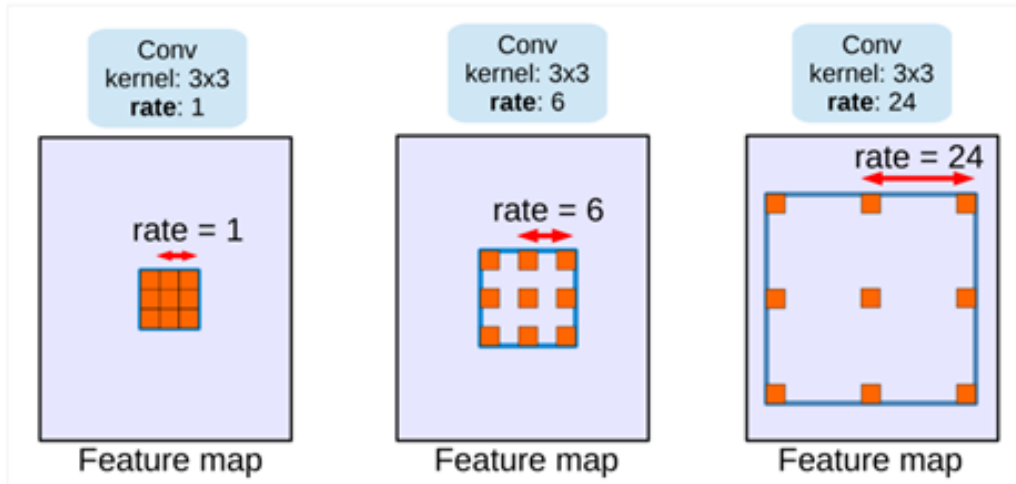
(a) : Các mạng sử dụng mô-đun SPP hoặc ASSP (Atrous Spatial Pyramid Pooling (ASPP), một dạng mở rộng của Spatial Pyramid Pooling (SPP)) ( có thể mã hóa thông tin theo ngữ cảnh đa thang đo bằng cách thăm dò các tính năng đầu vào bằng bộ lọc hoặc hoạt động gộp ở nhiều tốc độ và nhiều FOV (hiệu quả). (b) : Với Kiến trúc Encoder-Decoder, thông tin vị trí/không gian được phục hồi. Kiến trúc Encoder-Decoder đã được chứng minh là hữu ích trong các tài liệu như FPN, DSSD, TDM, SharpMask, RED-Net và U-Net cho các mục đích khác nhau. (c) : DeepLabv3+ sử dụng (a) và (b). DeepLabv3+ được đề xuất sử dụng những điểm tốt nhất của cả hai thế giới. Với các khối nâng cấp, nối và xử lý trung gian, mô hình có OS=4., mạng lưới nhanh hơn và mạnh hơn đã được phát triển.

DeepLabv3+ mở rộng DeepLabv3 bằng cách thêm một mô-đun giải mã đơn giản nhưng hiệu quả để tinh chỉnh kết quả phân đoạn, đặc biệt là dọc theo ranh giới đối tượng.

## Tích chập Atrous

Tích chập atrous, còn được gọi là tích chập giãn nở, là một kỹ thuật mạnh mẽ được sử dụng tích cực cho các tác vụ phân đoạn.

Nó cho phép chúng ta kiểm soát độ phân giải mà các tính năng được tính toán bởi DCNN và điều chỉnh FOV của bộ lọc để thu thập thông tin tầm xa.



Hình 5: So sánh tích chập tiêu chuẩn và tích chập atrous

Như được chỉ ra trong sơ đồ trên, sự giãn nở của hạt nhân tích chập tương đương với việc chèn các lỗ ('trous' trong tiếng Pháp) giữa các trọng số bộ lọc.

Sử dụng các hạt nhân tích chập giãn nở, người ta có thể dễ dàng sử dụng các mạng được đào tạo trước ImageNet để trích xuất các bản đồ đặc trưng dày đặc hơn. Nó có thể dễ dàng được áp dụng cho một mạng được đào tạo và tích hợp liền mạch trong quá trình đào tạo.

So với phép tích chập thông thường với bộ lọc lớn hơn, phép tích chập atrous cho phép chúng ta mở rộng trường nhìn (FOV) của bộ lọc một cách hiệu quả mà không cần tăng số lượng tham số hoặc lượng tính toán.

- Trích xuất đặc điểm thưa thớt với tích chập tiêu chuẩn trên bản đồ đặc điểm đầu vào có độ phân giải thấp
- Trích xuất đặc điểm dày đặc với tích chập atrous với tỷ lệ  $r=2$ , được áp dụng trên bản đồ đặc điểm đầu vào có độ phân giải cao.

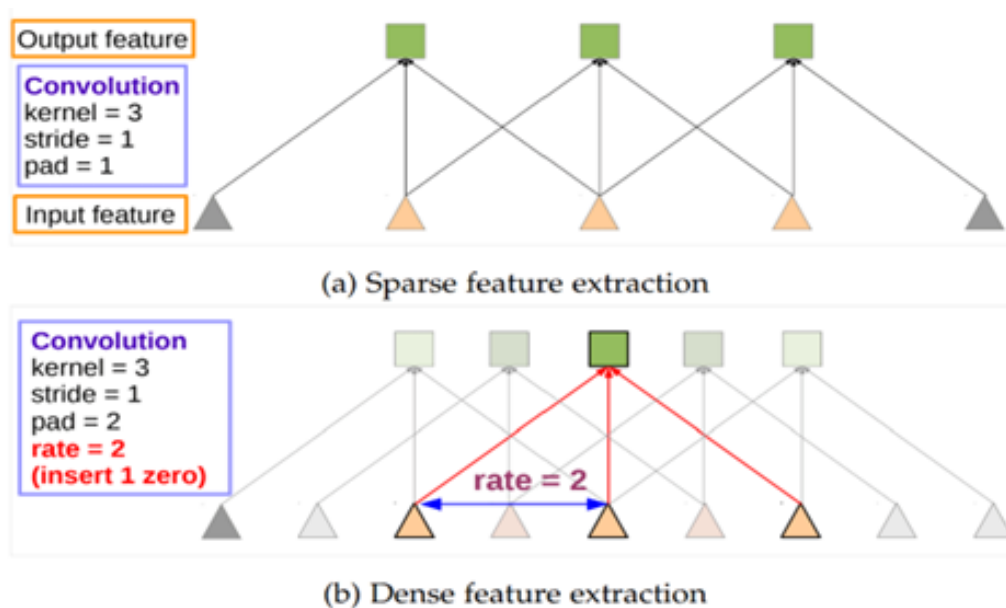
## Kiến trúc của DeepLabV3+

### 1. Encoder

Backbone Network: Phần đầu tiên là một mạng backbone (thường là ResNet hoặc Xception) để trích xuất đặc trưng từ ảnh đầu vào.

Đầu ra của backbone gồm hai phần: - Đặc trưng sâu (deep features) được truyền qua module ASPP. - Đặc trưng tầng thấp (low-level features) dùng trong decoder để giúp phục hồi chi tiết không gian.

Atrous Spatial Pyramid Pooling (ASPP):



Hình 6: (a) Trích xuất đặc điểm thưa thớt với tích chập tiêu chuẩn trên bản đồ đặc điểm đầu vào có độ phân giải thấp; (b) Trích xuất đặc điểm dày đặc với tích chập atrous với tỷ lệ  $r=2$ , được áp dụng trên bản đồ đặc điểm đầu vào có độ phân giải cao.

ASPP giúp mở rộng trường nhìn mà không làm mất độ phân giải không gian. Nó bao gồm:

- 1x1 Convolution + Batch Normalization + ReLU.
- 3x3 Atrous Convolutions với các tỷ lệ khác nhau (rate = 6, 12, 18) để thu thập thông tin ngữ cảnh ở nhiều tỷ lệ.

Để phân loại điểm ảnh trung tâm (màu cam), ASPP khai thác các tính năng đa tỷ lệ bằng cách sử dụng nhiều bộ lọc song song với các tỷ lệ khác nhau. Trường nhìn (FOV) hiệu quả được hiển thị bằng các màu khác nhau.

- Image Pooling: Global average pooling giúp nắm bắt thông tin toàn cảnh.
- Các đặc trưng từ ASPP được kết hợp lại và truyền qua một lớp 1x1 Convolution + Batch Normalization + ReLU.

## 2. Decoder

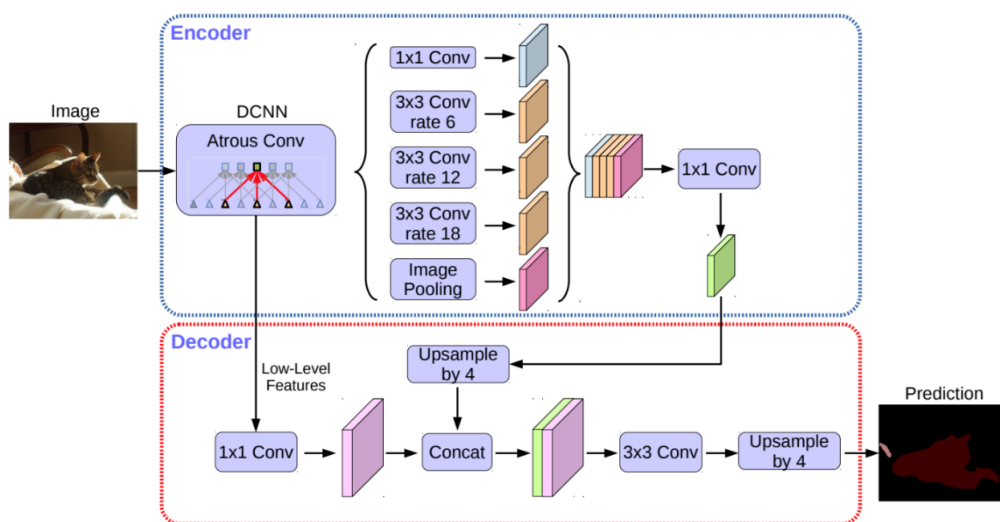
Kết hợp đặc trưng tầng thấp và đặc trưng từ ASPP:

- Đặc trưng tầng thấp  $L_y$  (có độ phân giải cao hơn) được giảm chiều bằng một lớp 1x1 Convolution + Batch Normalization + ReLU.
- Đặc trưng từ ASPP được upsample (phóng to) lên gấp 4 lần và kết hợp với đặc trưng tầng thấp qua thao tác Concat.

Phục hồi độ phân giải:

- Kết quả kết hợp được truyền qua hai lớp 3x3 Convolution + Batch Normalization + ReLU để tinh chỉnh các đặc trưng.
- Cuối cùng, thực hiện upsample thêm lần nữa (gấp 4 lần) và áp dụng một lớp 1x1 Convolution để tạo ra bản đồ phân đoạn đầu ra.





Hình 7: Kiến trúc chi tiết của DeepLab v3+

# Phương pháp đề xuất

## Encoder

3.1.1. DCNN (Deep Convolutional Neural Network): trích xuất đặc trưng - Sử dụng ResNet50 làm backbone chính với 4 khối chính giúp trích xuất features ở nhiều scale khác nhau. Là bước tiền xử lý đặc trưng cho ảnh đầu vào. - Cung cấp cả low-level (được lấy sau khối 2) và high-level features

Atrous Spatial Pyramid Pooling (ASPP): Thu thập đặc trưng không gian – ngữ cảnh ở nhiều độ phân giải. - 5 nhánh song song xử lý đa tỷ lệ - Image Pooling: Global context qua AveragePooling2D -  $1 \times 1$  Convolution: Giảm chiều và tích hợp thông tin -  $3 \times 3$  Conv với dilation rate 6 -  $3 \times 3$  Conv với dilation rate 12 -  $3 \times 3$  Conv với dilation rate 18

3.1.3 Atrous Convolution: - Mở rộng field of view mà không tăng tham số - Nắm bắt context ở nhiều tỷ lệ khác nhau - Dilation rates: 6, 12, 18 cho các vùng nhận thức khác nhau

Feature Fusion: - Concatenate output từ 5 nhánh ASPP: Các bản đồ đặc trưng từ ASPP được ghép lại theo chiều kênh (depth-wise concatenation). Các bản đồ đặc trưng đầu ra từ các nhánh ASPP có cùng kích thước không gian (H, W). Sắp xếp các nhánh theo thứ tự cố định. Các nhánh có vai trò khác nhau—có nhánh bắt đặc trưng cục bộ, có nhánh bắt đặc trưng toàn cục.

$$\text{Output} = \text{concat}(F1, F2, F3, F4, F5)$$

Với: - F1 là đầu ra từ  $1 \times 1$  convolution - F2, F3, F4 là đầu ra từ  $3 \times 3$  Atrous Convolution với các rate khác nhau - F5 là đầu ra từ Image Pooling

Sau khi ghép, số lượng kênh của tensor tăng lên 5 lần so với một bản đồ đơn lẻ

Nếu mỗi nhánh có 256 kênh, tổng số kênh sau khi ghép là:

$$C_{\text{new}} = 256 + 256 + 256 + 256 + 256 = 1280$$

- Conv  $1 \times 1$  để tích hợp thông tin: Sau khi ghép xong, ta dùng  $1 \times 1$  Convolution để giảm số kênh xuống một giá trị hợp lý (thường là 256) trước khi đưa vào Decoder (chỉ thay đổi số kênh, không ảnh hưởng không gian)

$$W = \text{Ma trận trọng số kích thước } (1280, 256)$$

Rồi với mỗi vị trí  $(x, y)$ , ta thực hiện phép nhân ma trận:

$$F'_{(x,y)} = W^T \cdot F_{(x,y)} + b$$

- BatchNorm: Ngăn gradient vanishing, cho phép sử dụng learning rate lớn hơn. Đặc biệt quan trọng trong mạng sâu như ResNet50 và ReLU. Giúp model học được các pattern phức tạp. Tính toán đơn giản và hiệu quả. Thêm tính phi tuyến vào mạng. Gradient không bị triệt tiêu với  $x > 0$ .

## 3.2 Decoder

3.2.1 Bilinear Upsampling: phương pháp nội suy song tuyến tính dùng để phóng to feature map dùng tính toán toán học giữa các pixel lân cận. Khôi phục kích thước gần như ảnh gốc, giúp kết quả phân vùng bám sát hình dạng thật giúp concat giữa các bước.

Công thức nội suy song tuyến tính:

$$I_{(x,y)} = \sum_{i=0}^1 \sum_{j=0}^1 w_{ij} \cdot I_{(x_i,y_j)}$$

Trong đó: -  $I_{(x,y)}$  là giá trị pixel nội suy tại vị trí  $(x, y)$  -  $I_{(x_i,y_j)}$  là 4 pixel lân cận quanh  $(x, y)$  -  $w_{ij}$  là hệ số trọng số, phụ thuộc khoảng cách  $(x, y)$  và  $(x_i, y_j)$

3.2.2 Skip Connection: Lấy feature từ các tầng sớm (thường là sau một vài tầng conv đầu tiên của backbone, như ResNet hoặc Xception) và đưa vào decoder. Những feature sâu thường giàu thông tin ngữ nghĩa nhưng mất chi tiết; các feature cạn thì giữ được chi tiết. Bảo toàn thông tin rìa, kết cấu, hình dạng.

3.2.3 Feature Fusion: Sau khi upsample ASPP output và xử lý low-level feature, concat hai feature maps lại với nhau (trục kênh).

3.2.4 Convolution Refinement: Sau khi concat feature maps, cho qua một chuỗi  $3 \times 3$  convolution  $\rightarrow$  loại bỏ nhiễu, làm rõ thông tin.

## Các API và Công Cụ Sử dụng

### 4.1 Framework Deep Learning

- **PyTorch (1.10+):**

- Framework chính cho việc xây dựng và huấn luyện mô hình
- Hỗ trợ tính toán GPU và tối ưu hóa
- Ecosystem phong phú và cộng đồng lớn

- **TorchVision:**

- Thư viện hỗ trợ xử lý ảnh và mô hình pre-trained
- Cung cấp các transforms chuẩn hóa dữ liệu
- Tích hợp sẵn nhiều kiến trúc mạng phổ biến

- **TensorBoard:**

- Công cụ theo dõi và trực quan hóa quá trình huấn luyện
- Hiển thị metrics và loss function
- So sánh kết quả giữa các thử nghiệm

### 4.2 Thư viện xử lý ảnh

- **OpenCV:**

- Xử lý ảnh cơ bản và tiền xử lý dữ liệu
- Hỗ trợ đọc/ghi nhiều định dạng ảnh

- Các phép biến đổi hình học và màu sắc
- **Pillow (PIL):**
  - Thao tác với ảnh và augmentation
  - Tương thích tốt với PyTorch
  - Xử lý metadata và EXIF
- **Albumentations:**
  - Tăng cường dữ liệu nâng cao
  - Tối ưu hóa cho bài toán phân đoạn
  - Pipeline linh hoạt và hiệu quả

## 4.3 Công cụ phụ trợ

- **NumPy:**
  - Xử lý dữ liệu số và tính toán ma trận
  - Tối ưu hóa cho các phép toán đại số tuyến tính
  - Tích hợp tốt với các thư viện khác
- **Matplotlib:**
  - Trực quan hóa kết quả và biểu đồ
  - Tùy chỉnh linh hoạt và chuyên nghiệp
  - Hỗ trợ nhiều định dạng xuất
- **tqdm:**
  - Hiển thị tiến trình xử lý
  - Ước tính thời gian hoàn thành
  - Tích hợp với notebook và CLI

## 5. Thực nghiệm kiểm tra mô hình

### 5.1 Input

- Đưa hình ảnh cần thực hiện semantic segmentation. Mình sẽ đưa ảnh của mình để thực hiện semantic segmentation với size ảnh là 512x512x3.
- Chuyển kiểu dữ liệu về float32 để tính toán chính xác, chuẩn hóa giá trị pixel từ [0,255] về [0,1], đảm bảo ảnh có đúng kích thước và formatEncoder.

## 5.2 Encoder

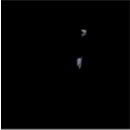

- Tiền xử lí ảnh qua Resnet50: đưa ảnh qua backbone resnet50, giai đoạn đầu giảm kích thước từ  $512 \times 512$  xuống  $256 \times 256$  qua conv  $7 \times 7$ . Tiếp tục giảm xuống  $128 \times 128$  khi qua MaxPooling (bắt đầu trích xuất các đặc trưng cơ bản như cạnh, góc).
- Tại  $128 \times 128$  lưu lại low-level features cho chi tiết cục bộ (dùng ở bước decoder).
- Tại  $64 \times 64$  ảnh được xử lí lấy các pattern phức tạp hơn.
- Tại  $32 \times 32$ : Tạo high-level features cho ngữ cảnh toàn cục.
- Xử lí qua ASPP: Sau khi ảnh qua bước Resnet50 thì sẽ được đưa ASPP xử lí song song 5 nhánh:
  - Image pooling: Nắm bắt context toàn cục của ảnh. Chứa thông tin tổng quan về nội dung ảnh. Giúp model hiểu được bối cảnh tổng thể.
  - Conv  $1 \times 1$ : Xử lý thông tin tại từng điểm. Giảm chiều và tổng hợp features. Học mối quan hệ tuyến tính giữa các channels.
  - Atrous Convolution 6, 12, 18: Nắm bắt thông tin ở nhiều mức độ.
- Các nhánh này sẽ được ghép lại theo chiều kênh để tổng hợp thông tin tạo ra 1 feature map với ngữ cảnh đầy đủ bao quát, giữ nguyên kích thước không gian ( $32 \times 32$ ) và tổng hợp channels. Sau đó sẽ được giảm channels qua conv  $1 \times 1$ , giảm độ phức tạp của features, tích hợp thông tin giữa các channels mà vẫn giữ nguyên không gian.

## 5.3 Decoder

- ASPP output sẽ được upsampling lên 4 lần bằng phương pháp bilinear để chuẩn bị cho quá trình fusion với low-level features mà vẫn giữ nguyên các thông tin quan trọng.
- Low-level features được lưu lại ở phần trên sẽ được xử lí để phù hợp với ASPP output.
- Kết hợp ASPP output với low-level features theo chiều kênh, upsampling lên 4 lần cho bằng với kích thước ban đầu.

## 5.4 Tạo mask cuối cùng

- Chuyển đổi thành probability map.
- Mỗi pixel có giá trị từ 0 đến 1.
- Thể hiện xác suất pixel thuộc vùng cần segment.
- Quá trình này kết hợp cả thông tin ngữ cảnh và chi tiết không gian để tạo ra mask phân đoạn chính xác.

ID_pictu	Ảnh	Mong Muốn	Kết quả thực tế	Nhận Xét
P_1	Người và nền	Nền bị xóa tách biệt với người		Pass
P_2	Người và nền chứa người	Lấy cả người ở nền		Pass
P_3	Vật thể và nền	Nền bị xóa tách biệt với vật thể		Fail Không tách được nền
P_4	Người và vật thể	Nền bị xóa tách biệt với người ở cạnh vật thể		Fail Vẫn tách được người nhưng vật thể đi cùng thì không
P_5	Vật thể và nền (được làm mờ)	Nền bị xóa tách biệt với vật thể		Fail Không tách được dù nền đã được làm mờ

Bảng 1: Kết quả kiểm tra phân đoạn nền trên các trường hợp điển hình

## Các trường hợp ảnh sau khi qua mô hình

### Nhận xét

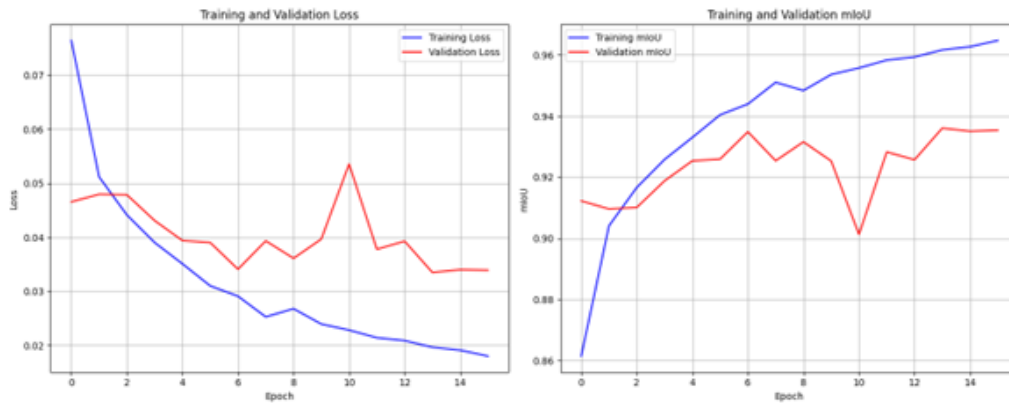
- Mô hình có thể hoạt động tốt nếu như mục đích chỉ để tách nền với người (ảnh chân dung,...). Còn các trường hợp khác mô hình sẽ hoạt động không đúng hoặc không hoạt động.
- Lý do có thể là do dataset chuyên dùng để human\_segmentation. Nếu tổng hợp được dataset lớn hơn thì % chính xác có thể sẽ cao hơn.
- Hoặc cũng có thể mô hình được xây dựng để tách nền với vật thể chứ không phải là phân đoạn ảnh.

# Đánh giá hiệu quả mô hình

Dựa vào các chỉ số train của 5 epoch đầu của U Net và DeepLab V3+:

## Hình 1 – Loss & mIoU

- Train loss (xanh) của DeepLab giảm đều tới  $\approx 0.02$ , val loss (đỏ) tiệm cận  $\approx 0.03$ .
- mIoU train > val  $\sim 0.03 - 0.04$ ; không over fit nhưng còn “gap”.
- Giảm LR sớm hoặc tăng augmentation có thể thu hẹp gap.



Hình 8: Loss & mIoU

## Chú thích màu

- Data 1 (nét liền)  $\equiv$  mô hình U Net (cùng tập dữ liệu)
- Data (nét đứt)  $\equiv$  mô hình DeepLab V3+ (cùng tập)
- Trục hoành = Epoch, trục tung = chỉ số tương ứng.

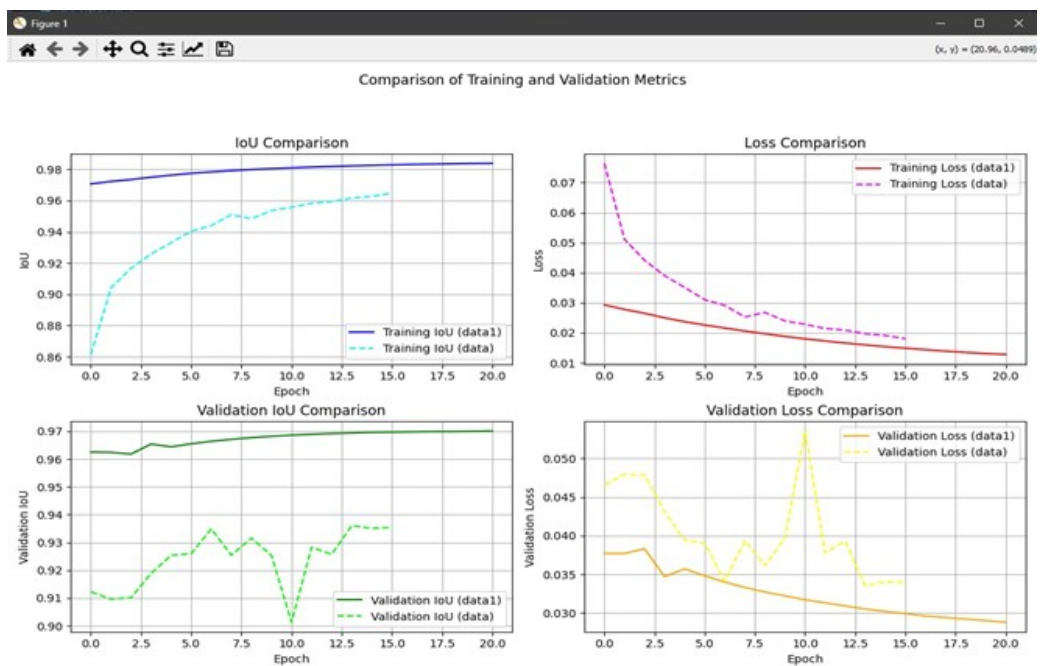
## Hình 2 – 4 biểu đồ

Tổng kết từ 4 biểu đồ:

Chỉ tiêu	U Net	DeepLab V3+	Nhận xét nhanh
Số epoch đã chạy	5	5	DeepLab thường cần $\geq 20$ epoch mới “vọt” mạnh nhờ ASPP; U Net hội tụ rất sớm.
Thời gian/epoch	$\approx 4\,300\text{ s}$	$\approx 3\,000\text{ s}$	U Net mất nhiều VRAM hơn vì giữ full res; DeepLab dùng stride 16 nên nhanh hơn.
Train IoU (cuối)	0.976	0.951	U Net cao hơn $\approx +2,5\%$ .
Val IoU (cuối)	0.964	0.924	Khoảng cách $\approx +4\%$ .
Best Val IoU	0.965 (E4)	0.924 (E4)	U Net dẫn.
Train Loss (cuối)	0.0237	0.0253	Tương đương, U Net nhỉnh nhẹ.
Val Loss (cuối)	0.0357	0.0410	Phù hợp chênh lệch IoU.
Xu hướng loss	Giảm đều, dao động nhẹ (xem hình 1 – trái)	Giảm đều hơn (đường xanh đậm), song cao hơn ban đầu.	–
Xu hướng IoU	Cao ngay từ đầu, dao động $< 1\%$ (hình 1 – phải)	Tăng ổn định; vẫn còn dư địa nếu train lâu hơn.	–
Over fit / Under fit	Không rõ rệt: Train IoU – Val IoU $\approx +1\%$	Chênh $\approx +2,7\% \rightarrow$ chấp nhận được; chưa over fit.	–
Số tham số $\approx$	17 M	45 M	DeepLab nặng hơn $\times 2,6$ .
Kích thước receptive field	Hẹp hơn; chủ yếu nhờ skip connection	Rất rộng nhờ Atrous + ASPP $\rightarrow$ lợi thế ảnh cảnh rộng/phức tạp.	–

Bảng 2: Bảng đánh giá hiệu quả mô hình U Net và DeepLabV3+





Hình 9: 4 biểu đồ Training/Validation IoU, Loss

Biểu đồ	Quan sát chi tiết	Giải thích & Hệ quả	
Training IoU Comparison	<ul style="list-style-type: none"> <li>U Net (xanh đậm) khởi động rất cao <math>\approx 0.97</math>, sau <math>\approx 15</math> epoch tiến sát 0.983 rồi phẳng.</li> <li>DeepLab (xanh nhạt) bắt đầu thấp <math>\approx 0.86</math> nhưng leo dốc ổn định, tiệm cận 0.965 lúc epoch 18–20.</li> <li>Khoảng cách hai đường thu hẹp dần.</li> </ul>	<ul style="list-style-type: none"> <li>U Net học nhanh nhờ skip connection full resolution.</li> <li>DeepLab cần thời gian để khai thác ASPP + atrous, nhưng vẫn tiến gần.</li> <li>Không có dấu hiệu overfit ở train IoU (không vượt 0.99).</li> </ul>	
Loss Comparison	<ul style="list-style-type: none"> <li>Cả hai đường mất mát cùng giảm lũy tiến.</li> <li>U Net (đỏ) khởi đầu <math>\approx 0.03</math>, xuống dưới 0.015 ở epoch 20.</li> <li>DeepLab (hồng) cao hơn <math>\sim 0.05</math> khi bắt đầu; dốc giảm mạnh trong 10 epoch đầu, sau đó chậm lại và còn <math>\sim 0.02</math>.</li> </ul>	<ul style="list-style-type: none"> <li>Loss tương quan nghịch với IoU: U Net thấp hơn, phù hợp IoU cao hơn.</li> <li>Hai đường hội tụ <math>\Rightarrow</math> optimizer, LR schedule phù hợp, không diverge.</li> </ul>	
Validation IoU Comparison	<ul style="list-style-type: none"> <li>Đường xanh lá đậm (U Net): dao động rất hẹp (0.962<math>\rightarrow</math>0.971), tăng dần tới gần 0.97.</li> <li>Xanh lá nhạt (DeepLab): vượt 0.90 ở epoch 1, lên tối đa <math>0.935 \pm 0.005</math>; có "rãnh" giảm mạnh tại epoch 10 (<math>\sim 0.90</math>) <math>\rightarrow</math> nhiều dữ liệu / batch đặc thù.</li> <li>Khoảng cách bền vững <math>\approx 3\text{--}4\%</math>.</li> </ul>	<ul style="list-style-type: none"> <li>U Net generalize tốt ngay từ đầu; biên độ dao động <math>&lt; 0.01 \rightarrow</math> mô hình ổn định.</li> <li>DeepLab còn "gap" generalization; cần: <ol style="list-style-type: none"> <li>Epoch nhiều hơn,</li> <li>data augmentation bổ sung,</li> <li>scheduler giảm LR sau epoch 15.</li> </ol> </li> </ul>	
Validation Loss Comparison	<ul style="list-style-type: none"> <li>Cam đậm (U Net) giảm mượt từ <math>\sim 0.037</math> xuống <math>\sim 0.029</math>, không tăng đột ngột.</li> <li>Cam nhạt (DeepLab) dao động 0.033–0.05; spike tại epoch 10 (<math>&gt; 0.05</math>) cùng chỗ rơi của val IoU <math>\rightarrow</math> batch "khó" hoặc LR quá lớn lúc đó.</li> <li>Sau spike, đường quay lại xu hướng giảm nhưng vẫn cao hơn U Net <math>\sim 0.003\text{--}0.005</math>.</li> </ul>	<ul style="list-style-type: none"> <li>Spike xác nhận điểm nhiễu – khuyến nghị ghi log batch hard case hoặc dùng Gradient Accum/Lookahead.</li> <li>Khoảng cách loss khớp khoảng cách IoU.</li> </ul>	

Bảng 3: Phân tích chi tiết 4 biểu đồ huấn luyện/đánh giá

Tiêu chí	U Net (Data 1)	DeepLab V3+ (Data)	Hàm ý
Tốc độ hội tụ	Rất nhanh – đạt 0.965 val IoU trước epoch 5	Chậm hơn, nhưng vẫn tăng đều	U Net lợi về thời gian huấn luyện ngắn
Độ ổn định	Đường val IoU & val loss dao động thấp	Dao động nhiều hơn; spike @ epoch 10	DeepLab cần regularization / scheduler
Khoảng cách Train–Val	< 1% (IoU) → fit tốt	2-3% → chưa khai thác hết tiềm năng nhưng chưa overfit	DeepLab tiềm năng cải thiện nếu train dài
Hiệu năng hiện tại (epoch 20)	Val IoU $\approx 0.97$ , Loss $\approx 0.029$	Val IoU $\approx 0.935$ , Loss $\approx 0.033$	U Net vẫn dẫn đầu $\approx +3-3.5\%$ IoU

Bảng 4: So sánh tổng kết mô hình

## 7. Tài Liệu Tham Khảo

- Chen, L.-C., et al. “Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation.” (2018).
- He, K., et al. “Deep Residual Learning for Image Recognition.” (2015).
- Ronneberger, O., et al. “U-Net: Convolutional Networks for Biomedical Image Segmentation.” (2015).
- Long, J., et al. “Fully Convolutional Networks for Semantic Segmentation.” (2015).
- Sh-tsang. “Review: DeepLabV3 - Atrous Separable Convolution - Semantic Segmentation.” Medium.
- LearnOpenCV. “DeepLabV3+: Ultimate Guide.”