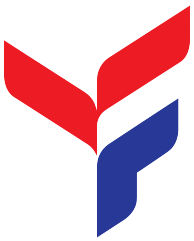
UNIVERSITY OF SCIENCE AND TECHNOLOGY HANOI



V&H.Corp

Data Science Department

Flood Area Segmentation using DeepLabV3+

November 4, 2025

Project Report

Nguyen Thanh Vinh - 23BI14453
Tran Ngoc Hung - 23BI14181
Chau Tuan Minh - 23BI14299
Vu Hoai Nam - 23BI14326
Pham Gia Bao - 23BI14061
Nguyen Trong Minh Duc - 22BA13082

Contents

Abstract

Flood segmentation is a crucial task in disaster management, enabling accurate identification of inundated areas from satellite or aerial imagery.This study is particularly relevant to Vietnam, where in recent years, three destructive tropical storms have caused severe flooding and widespread damage to residential areas. The results demonstrate that automated flood segmentation can significantly improve the speed and reliability of flood mapping, supporting early warning systems and disaster response planning in flood-prone regions such as Vietnam.

1 Introduction

Floods are among the most frequent and devastating natural disasters worldwide, posing serious threats to human life, infrastructure, and the environment. In this project, the **DeepLabV3+** architecture is employed to perform flood segmentation. **DeepLabV3+** is a state-of-the-art semantic segmentation model that combines atrous spatial pyramid pooling (ASPP) and an encoder–decoder structure to capture multi-scale contextual information while preserving spatial details. The model is trained on labeled flood imagery datasets to generate accurate flood masks that can support emergency response and disaster management activities.

2 Methology Approach

The dataset contains images of flood hit areas and corresponding mask images showing the water region.

There are 290 images and self annoted masks. The mask images were created using Label Studio, an open source data labelling software. The task is to create a segmentation model, which can accurately segment out the water region in a given picture of a flood hit area.

Such models cane be used for flood surveys, better decision-making and planning. Because of less data, pre-trained models and data augmentation may be used.

(a) Demo Image 1

(b) Demo Image 2

Figure 1: Images Data Preview

]

3 Dataset Overview

The dataset used in this project consists of **290 image–mask pairs** collected for flood segmentation. Each input image represents a scene captured from satellite or aerial imagery, showing regions affected by flooding under various lighting and environmental conditions. Corresponding to each image, there is a manually annotated **mask** that highlights flooded areas at the pixel level, where flooded regions are labeled distinctly from non-flooded regions.

These image–mask pairs serve as the foundation for training the **DeepLabV3+** model, enabling it to learn the spatial and contextual features necessary for accurate flood area detection. The dataset was preprocessed to ensure consistent resolution and normalized pixel values before training. This configuration allows the model to generalize effectively and perform reliable segmentation on unseen flood imagery.

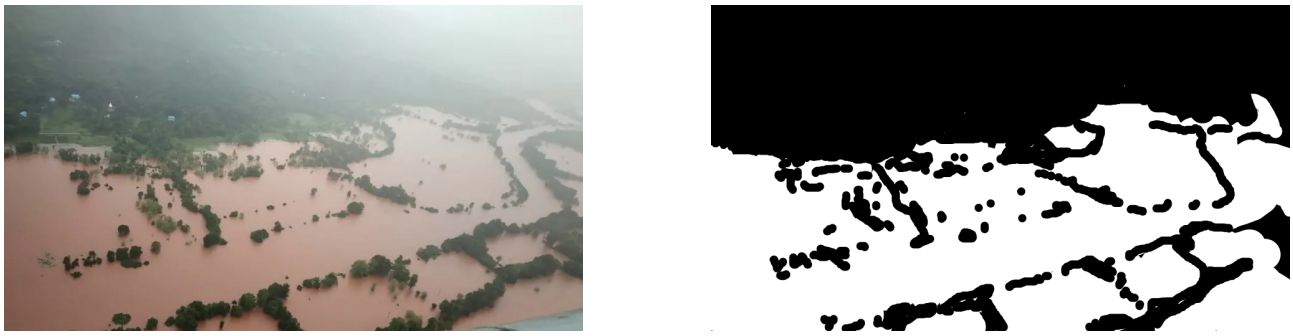
Table 1: Dataset Summary

Attribute	**Description**
Number of images	290
Image type	RGB satellite or aerial imagery
Annotation type	Pixel-level flood mask
Image resolution	1024 × 1024 pixels (resized)
Data format	PNG(Images)/ JPG(MASKS)
Train / Validation / Test split	70% / 20% / 10%
Purpose	Flood region segmentation

Each input image is paired with a corresponding mask that shows which areas are flooded and which are not. The image is used as the model's input, while the mask serves as the ground truth label. During training, the **DeepLabV3+** model takes an image and predicts a segmentation map — a pixel-by-pixel classification of the scene.

4 DeepLabV3

Semantic Segmentation classifying every pixel in an image into a category (e.g., sky, car, road, person, etc.).In semantic segmentation, the model must recognize what objects are present in the image and determine where they are by assigning a semantic label to every pixel, something that's tricky for normal CNNs



(a) Original Flood Image **(b)** Corresponding Mask

Figure 2: Example of an image and its corresponding mask used in the flood segmentation dataset.

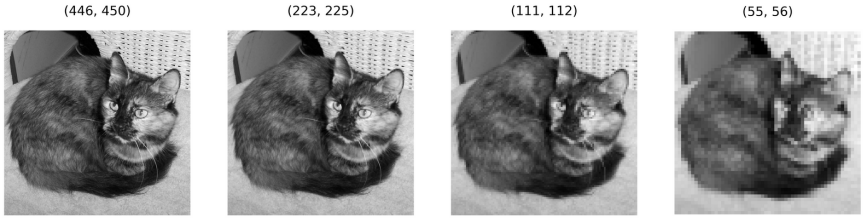


Figure 3: Pooling Process Reduced Image Detail

Deep CNNs (ResNet, VGG, etc.) are great for image classification but not directly suitable for segmentation. Most of CNNs reduce feature resolution from original image to smaller scaled-image, this might cause spartial information loss. This is called **Gloval label** which work smoothly in classification task but when the network becomes "invariant" to small local changes → **can't produce sharp pixel-level boundaries.**

4.1 Structure

To overcome the problem, a structure is called **Atrous Convolution** [36, 26, 74, 66], which effective for semantic image segmentation. Atrous Convolution also known as dilation filter, it reduces spatial size as stride increase, inserts `"hole"` dilation in kernel to enlarge the receptive field without downsampling the features map.

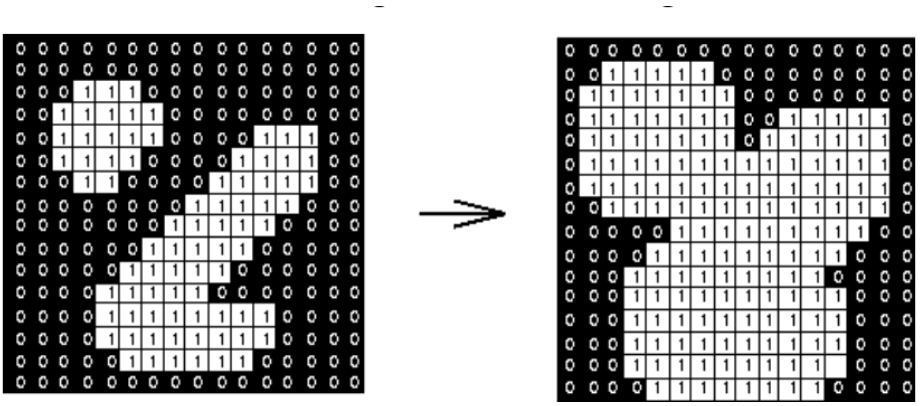


Figure 4: Dilation Mechanic

By increasing rate of 3x3 kernal, model can understand large context, in other words it see entire object without more computing-resource required. DeepLab uses hole convolution to expand the receptive field

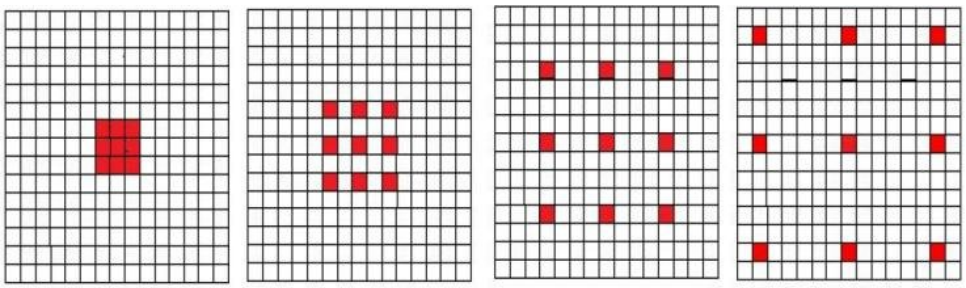


Figure 5: Dilation kernel rate change (r= (1, 3, 6, 9))

without pooling or stride.

4.2 Standard Convolution vs. Atrous (Hole) Convolution

In a standard convolution operation, each output feature $y[i]$ is computed as a weighted sum of its neighboring input pixels $x[i+k]$ using a kernel $w[k]$. The mathematical formulation is given as:

$$y[i] = \sum_{k=-K}^{K} x[i+k] \cdot w[k] \tag{1}$$

Here, the convolution kernel slides across the input feature map without skipping any pixel, hence it captures only local context within a limited receptive field.

In contrast, the **Atrous (or Dilated) Convolution** introduces a *dilation rate* r, which defines the spacing between kernel elements. The modified formula becomes:

$$y[i] = \sum_{k=-K}^{K} x[i+r\cdot k] \cdot w[k] \tag{2}$$

When $r = 1$, the operation reduces to a standard convolution. Increasing the dilation rate $r > 1$ effectively enlarges the receptive field without increasing the number of parameters or reducing the feature map resolution.

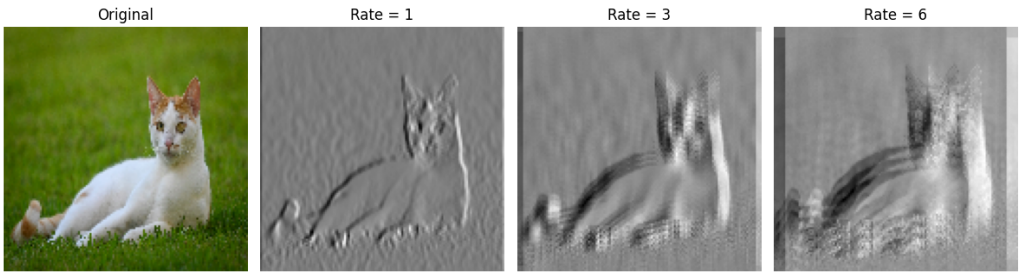


Figure 6: Rate Changing Comparision

4.3 ASPP

ASPP is a parallel module that applies several atrous convolutions (same kernel size, different dilation rates) plus an image-level pooling branch, then concatenates the results and fuses them with a 1×1 conv. The purpose is capture the multi-scale context (local to global) while keep spatial resolution and details by changing rate of adilation kernel and then stack them into a convolution network.

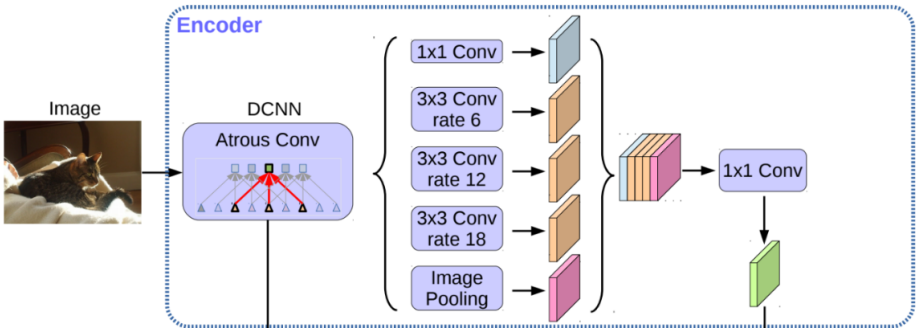


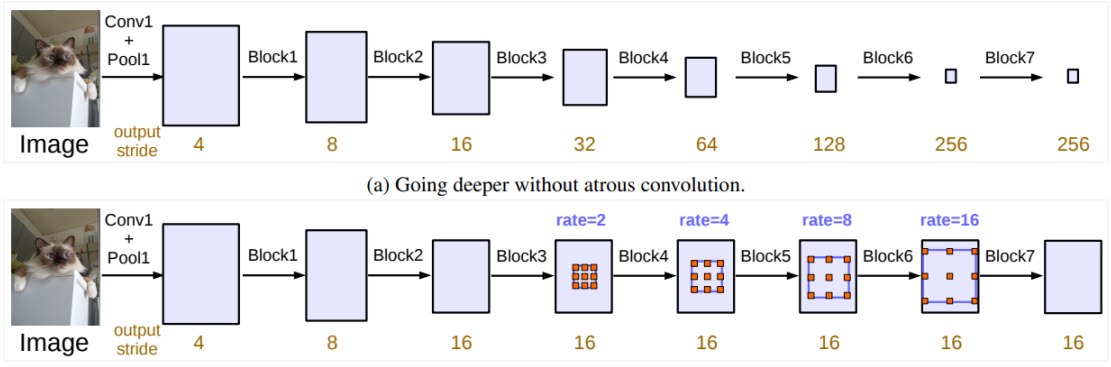
Figure 7: ASPP in Encoder (DeeplabV3+)

For 2D with 3×3 kernel same in both dimension, receptive field side length equals that R. Multiple atrous rates create filters that "see" windows of different sizes while keeping parameter count constant. Atrous convolution also allows us to explicitly control how densely to compute feature responses in fully convolutional networks. Inserting "holes" (zero spaces) between kernel elements, atrous convolution enables the network to capture multi-scale contextual information efficiently. T

$$R = k + (k-1)(r-1) \tag{3}$$

The backbone of ASPP structure is actually the resuse some of block from ResNet , several 3x3 convolutions and the last convolution contains **stride 2** , similar to original ResNet. The motivation behind this model is that the introduced striding makes it easy to capture long range information in the deeper blocks.For example, the whole image feature could be summarized in the last small resolution feature map.

However, consecutive striding is harmful for semantic segmentation since detail information is decimated, and thus we apply atrous convolution with rates determined by the desired output stride value, where output stride = 16.



(a) Going deeper without atrous convolution.

(b) Going deeper with atrous convolution. Atrous convolution with $rate > 1$ is applied after block3 when $output\_stride = 16$.

Figure 3. Cascaded modules without and with atrous convolution.

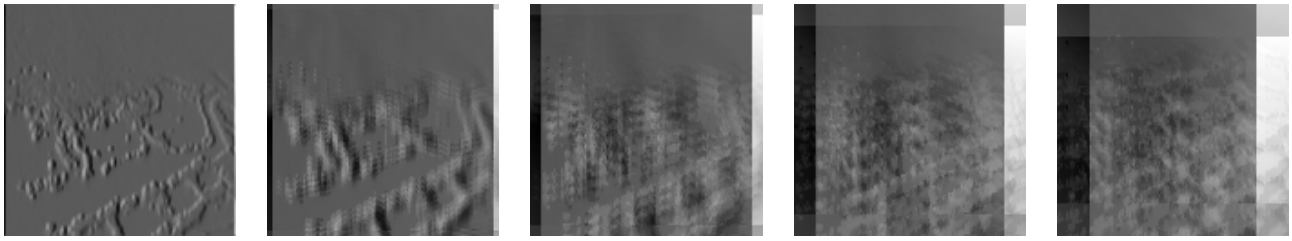
Figure 8: Backbone of ASPP

4.4 Demo Images

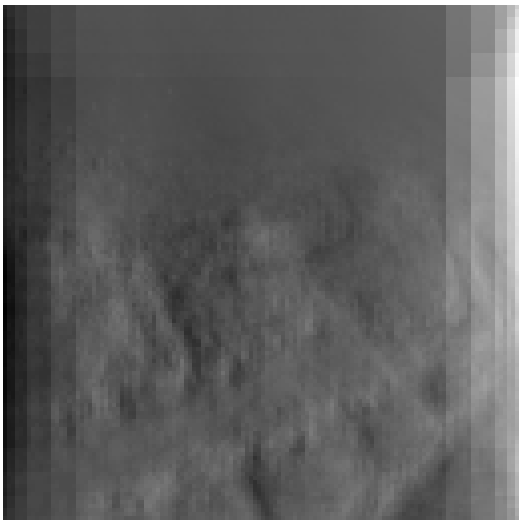
Figure 10 illustrates the effect of applying atrous (dilated) convolutions with different dilation rates on the same input image. When the dilation rate is small (e.g., rate = 1), the convolution behaves like a standard 3 × 3 kernel, focusing on fine local details.

As the rate increases (e.g., 3, 6, 12, 18), At rate = 1, edges are well recognized, move to rate = 3, edges are more blurry, move to rate 6 and 12 , the edges are no longer sharp, but background details are informed. After using rate =18, all images are stacked to process final scaled image which still contain most of detailed.

Figure 9: Original Image



(b) Rate = 1 (c) Rate = 3 (d) Rate = 6 (e) Rate = 12 (f) Rate = 18



(g) ASPP Combined Feature Map

Figure 10: The bottom image demonstrates the ASPP combined output, where all dilation levels are fused to enhance flood region detection accuracy.

5 Model Training

5.1 Data Preparation

The dataset consists of paired 290 satellite images and their corresponding flood masks. Each input image represents a geographical region affected by flooding, while the mask highlights flooded and non-flooded areas using binary labeling with Label Studio. Before training, all images were resized to a fixed resolution of 256×256 and normalized to the range $[0, 1]$, and the `meatadata.csv` was obtained as the guiding map for each pair of image and mask.

Table 2: DeepLab-FloodArea-Custom Model Summary

Layer (type)	Output Shape	Param #	Connected to
InputLayer (InputLayer)	(None, 256, 256, 3)	0	[]
conv1_pad (ZeroPadding2D)	(None, 262, 262, 3)	0	['InputLayer[0][0]']
conv1_conv (Conv2D)	(None, 128, 128, 64)	9472	['conv1_pad[0][0]']
conv1_bn (BatchNormalization)	(None, 128, 128, 64)	256	['conv1_conv[0][0]']
conv1_relu (Activation)	(None, 128, 128, 64)	0	['conv1_bn[0][0]']
pool1_pad (ZeroPadding2D)	(None, 130, 130, 64)	0	['conv1_relu[0][0]']
pool1_pool (MaxPooling2D)	(None, 64, 64, 64)	0	['pool1_pad[0][0]']
conv2_block1_1_conv (Conv2D)	(None, 64, 64, 64)	4160	['pool1_pool[0][0]']
Total params		12,510,528	
Trainable params		12,477,280	
Non-trainable params		33,248	

5.2 Model Buidling

5.3 Results and Evaluation

5.3.1 Learning Curve

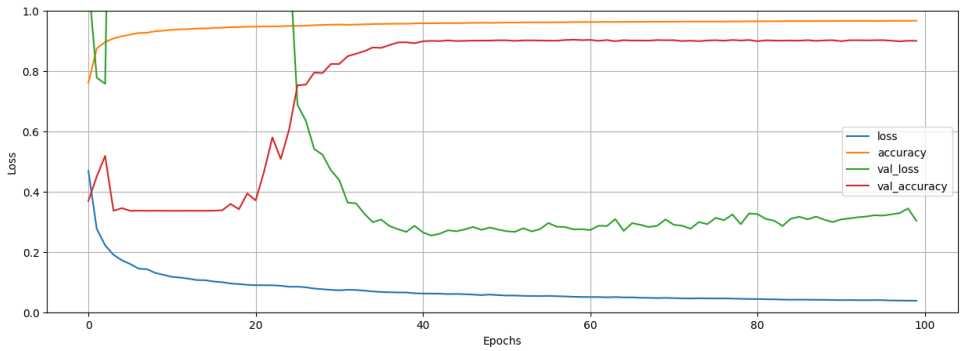


Figure 11: Learning Curve

A learning curve shows how a model's performance, typically measured by loss or accuracy, changes over training iterations or epochs. When the learning curve converges, it indicates that the model has reached a stable state where further training does not significantly improve performance. `Accuracy` and `loss` are reverting from each other tell the thing that model training work pretty well, valid loss and accuracy start converge from epoch 26 27.

5.3.2 Evaluation

For the metrics method, we will use **Mean Intersection over Union**(mIoU). The metric is defined by the overlap between the predicted segmentation and the ground truth, divided by the total area covered by the union of the two. The metric can be computed for each class separately or for all classes at once. The metric is optimal at a value of 1 and worst at a value of 0, -1 is returned if class is completely absent both from prediction and the ground truth labels.

$$\text{mIoU} = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i + FP_i + FN_i} \tag{4}$$

where TP_i, FP_i, and FN_i denote the number of true positive, false positive, and false negative pixels for class i, respectively. A higher mIoU value indicates better segmentation accuracy.

Table 3: Comparison of ASPP dilation rates and corresponding mIoU.

Model	Dilation Rates (d)	mIoU (%)
ASPP Convs=4	1, 6, 12, 18	**72.99**
ASPP Convs=5	1, 3, 6, 12, 18	**75.48**

Two different metrics show that choosing and defining the right amount of convolution layers and dilation rate can have an effect on the performance of the model, with `ASPP[1, 6, 12, 18]` layers, the model can run fast and converge early, but the big jump from dilation rate 1 from 6 causes loss of spatial information. Unlikely, `ASPP[1, 3, 6, 12, 18]` may not the fastest model but it keep almost all spatial information from original images.

References

Research Resources

[1] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam, *Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation (DeepLabV3+), Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 801–818, 2018. Available at: https://arxiv.org/abs/1802.02611

[2] TorchMetrics Documentation, *Mean Intersection over Union (mIoU)*, Available at: https://lightning.ai/docs/torchmetrics/stable/segmentation/mean_iou.html

Working and Implementation Resources

[3] Sik-Ho Tsang, *Review: DeepLabv3+ — Atrous Separable Convolution (Semantic Segmentation)*, Medium, 2020. Available at: https://sh-tsang.medium.com/review-deeplabv3-atrous-separable-convolution-semantic-segmentation-a625f6e83b90

[4] Utkarsh Saxena, *Flood Area Segmentation using DeepLabV3+*, Kaggle Notebook, 2023. Available at: https://www.kaggle.com/code/utkarshsaxenadn/flood-area-segmentation-deeplabv37

[5] Vit Hoàng, *X lý nh: Erosion, Dilation, Opening, Closing*, Viblo Blog, 2021. Available at: https://viblo.asia/p/xu-ly-anh-erosion-dilation-opening-closing-4dbZNpWq5YM

[6] Faizal Karim · Krish Sharma · Niyar R Barman, *Flood Area Segmentation, Segment the flooded area*, Available at: https://www.geeksforgeeks.org/python/erosion-dilation-images-using-opencv-python/

[7] GeeksforGeeks, *Flood Area Segmentation*, Available at: https://www.kaggle.com/code/k214744nizamuldin/unet-flood-segmentation/input