# Road Segmentation from Satellite Images Using PSPNet, U-Net, FCN-8 with VGG, SegNet and DeepLabv3

Artificial Intelligence department

King Abdullah || School of Information Technology

University of Jordan

Beesan Al-Attal

Layan Balbisi

Leen Samman

Zaina Abu-Naser

Supervised by

Professor Tamam Alsarhan

## Abstract

This report investigates the use of deep learning models for road segmentation from satellite imagery, aiming to enhance the accuracy and efficiency of automated road detection systems. We will evaluate several prominent architectures, including U-Net, DeepLabv3, SegNet, FCN-8 with VGG, and the Pyramid Scene Parsing Network (PSPNet), to determine their effectiveness in various urban scenarios. The study will involve a comprehensive analysis of each model's design, training methodologies, and performance metrics. Additionally, we will explore the impact of data augmentation techniques and loss functions on model performance. The findings from this research are expected to provide insights into the best practices for implementing deep learning in road segmentation tasks, ultimately contributing to improved urban planning and infrastructure management.

## Table of Contents

# 1 Introduction

## 1.1Background and Theory

Road segmentation plays a critical role in shaping transportation infrastructure, notably influencing multiple industries and sectors, starting from disaster management and mapping to traffic control. It helps in the process of urban development and planning by providing required data, which helps planners design and expand infrastructure appropriately. For instance, if there is a proper and successful mapping of the roads, then it clearly indicates areas where there is a need for connectivity, planned development with respect to growth factor in the coming years, and analyzes main pattern flows [1].

It provides good valuation in disaster response and recovery in identifying high-risk zones by road segmentation. During this disaster, it will enable rapid assessment of accessible routes to ensure timely aid delivery and improvement in the response times of aid [2]. It can also be utilized in creating large-scale and extremely accurate maps, particularly in regions where a traditional survey is impracticable or even outdated. This will allow for the timely updating of road networks and geographic information on such places [3] .

Another critical application relates to transportation management where automatic road segmentation of traffic images is employed for monitoring the flow of traffic, managing congestion, and detecting accidents in real time. It enables smart transportation systems that understand changing traffic conditions in real time and provide an added layer of road safety by leveraging automatic road segmentation of traffic images [4].

Given these diverse applications, understanding the underlying process used in road segmentation becomes essential. This process involves identifying road areas and outlining roads within images, mainly satellite and aerial imagery, by using techniques such as semantic segmentation and edge detection. It includes the identification of each pixel in an image - the semantic segmentation definition in general- [5]  as "road" or "non-road," clearly outlining the road networks for a variety of analysis requirements [6].

## 1.2 Problem Statement

Although road segmentation has several and diverse applications, it still faces significant challenges, whether due to real-time processing constraints, data quality issues, or difficulties in training the models. One major issue is the variability in environmental conditions, which can change completely with weather variations such as rain, snow, or sunlight. These changes affect the appearance of roads in images, resulting in inaccurate pixel values [7].

Another challenge occurs in unstructured environments, where roads may have unclear or inconsistent boundaries, making it difficult to outline roads accurately [8]. Also, there are problems during the training process, such as limited computational resources, low accuracy, and data related issues like imbalanced datasets and inaccurate labeling [9] [10].

These obstacles present important limitations for traditional road segmentation methods, often leading to inefficient solutions for critical applications. Therefore, this research aims to develop a novel approach that addresses these challenges effectively, delivering improved accuracy and robustness under varying conditions.

## 1.3 Motivation

Tackling Road segmentation problem is pivotal to understanding and analyzing the relationship between land usage type and land subsidence. Satellite and aerial images are critical data resource for updating road networks [11], which simultaneously supports urban development, disaster management, and traffic control. However, achieving high accuracy in the presence of unstructured environments [8] and with the data quality issues, remains a significant challenge. Our motivation is to achieve high F1 score, regardless of the complexities of varying image conditions, labels inconsistencies, and the lack of computational resources, encouraged us to explore and to work on implementing innovative solutions, that contributes to real-word applications.

## 1.4 Research Questions

Our aim in this project is to identify the ideal answers for the following research questions:

1. What is the best Convolutional Neural Networks (CNNs) architectures for solving the road segmentation problem in satellite and aerial images, considering the f1 score and computational resources constraints?

2. How does data augmentation impact the generalization ability of the model?

3. What strategies can be employed to optimize the utilization of the computation resources, while maintaining accuracy in road segmentation?

4. How can the final model be deployed on real-life applications for road segmentation?

## 1.5 Related Works

Recent advancements in road detection from satellite imagery leverage innovative methods and deep learning architectures to enhance accuracy and efficiency. For instance, a combined approach integrating Tversky and cross-entropy loss functions significantly improved accuracy in road segmentation, achieving a mean Intersection over Union (mIoU) of 80.71% and Dice score of 89.82% while using fewer learnable parameters compared to traditional models like U-Net and SegNet. This approach addresses challenges like imbalanced datasets, improving performance in identifying road pixels efficiently [6]. Similarly, GeoPalette enhances model training by generating synthetic satellite images in diverse styles, which improved road detection accuracy by 3.5% compared to models trained only on real images, demonstrating the utility of synthetic data in augmenting limited datasets [19].

Other studies emphasize algorithmic and methodological innovations. A U-Net-based approach, trained on 6,226 RGB images with data augmentation, achieved an IoU of 85%, effectively identifying roads under varying conditions, including occlusions and lighting challenges [20]. Meanwhile, the Average Path Length Similarity (APLS) metric introduced a novel evaluation standard for road connectivity, highlighting performance disparities in urban and rural settings, with models like MobileNet-v2 achieving an F1 score of 71.9% in challenging environments [21]. Additionally, lane-level road extraction methods employing advanced descriptors and templates demonstrated exceptional accuracy, with completeness and correctness rates exceeding 98%, addressing issues like shadows and varying road widths [22]. Collectively, these studies underscore the potential of integrating innovative algorithms, loss functions, and synthetic data to advance road detection capabilities.

In the paper Research on road surface crack detection [12]. The authors developed a new neural network called SegCrackNet, which is based on the SegNet model, to improve the accuracy of detecting cracks in roads. They enhanced the network by adding dropout layers, optimizing the receptive fields, and using multi-level output fusion techniques, which helped the model better analyze complex crack patterns. The results showed that SegCrackNet outperformed other models: it achieved 3.18% higher accuracy, 3.10% better recall, and 3.11% improved F1 score compared to U-Net, and it surpassed ResUNet by 12.07% in accuracy, 10.70% in recall, and 11.43% in F1 score. When compared to DeepCrack, SegCrackNet also showed improvements of 2.70% in accuracy, 3.24% in recall, and 2.97% in F1 score. The study's findings emphasize the importance of thoughtful network design in enhancing crack detection performance, demonstrating that SegCrackNet is effective in identifying road cracks accurately and reliably across various challenging scenarios.

Also, in an Urban Paved Roads in RGB Satellite Images study [13] .The authors used the SegNet convolutional neural network to identify and segment paved roads in RGB satellite images. They trained SegNet on a dataset of these images, and it achieved an impressive accuracy of 83.97% and an Intersection over Union (IoU) of 71.93% for detecting roads. However, in some challenging cases, the model's accuracy dropped to 73.57% with an IoU of 44.96%, mainly due to confusion with other objects that had similar colors and textures, like fiber cement roofs. This research demonstrates the potential of using deep learning techniques like SegNet for effectively segmenting urban roads, especially since RGB images are widely available, making it easier to update geographic databases and support urban planning efforts.

An effectively applied model named Pyramid Scene Parsing Network or PSPNet demonstrate its capability for delineating road boundaries in complex urban environments. In evaluations on the Cityscapes dataset, PSPNet achieved a mean IoU of 79.20%, showcasing its proficiency in integrating local and global contextual information through its pyramid pooling module. That would be very important to segregate them out from other features such as vehicles, pedestrians, and applications in autonomous driving and urban planning. The PSPNet architecture captures both fine details while maintaining a broader contextual understanding. It has therefore been one of the strongest competitors among semantic segmentation models that have beaten earlier models in many comparative studies such as Fully Convolutional Networks and U-Net. [14] [15].

Recent advances in road segmentation have also witnessed new models being developed based on the very principles PSPNet was based on. The Seg-Road, which combined transformer and CNN architectures, realized the current state-of-the-art performances with an IoU of 68.38% on the Massachusetts dataset and 67.20% on the DeepGlobe dataset [10]. A number of limitations from the conventional segmentation schemes are overcome with the introduction of this model by enhancing connectivity structures that will reduce fragmentations in road extraction tasks. This represents the growing trend of using hybrid architectures that combine CNNs and transformers to improve performance on difficult segmentation scenarios. While PSPNet remains a robust model for road segmentation, such developments hint that the field is moving fast, with increasing solutions being tailored for application-specific needs [10] [16].

Additionally, In the field of road segmentation from SAR satellite images, the Fully Convolutional Network (FCN) model, especially the FCN-8 architecture, has been widely studied. The FCN-8 model improves over earlier versions by combining information from different layers, which helps in capturing more details necessary for accurate road detection. This is especially important in SAR images where roads can look similar to other linear features like rivers. The paper Road Segmentation in SAR Satellite Images with Deep Fully Convolutional Neural Networks [17]studies the use of FCN-8 for this purpose. The authors found that while FCN-8 can struggle with the unique challenges of SAR images, it can still achieve good results with suitable adjustments and improvements.

Also, the integration of CNN-Transformer and U-Net architectures has shown promising results. The paper "Remote Sensing Image Road Segmentation Method Integrating CNN-Transformer and U-Net" [18] studies this hybrid approach. By combining the strengths of Convolutional Neural Networks (CNNs) for local feature extraction and Transformers for capturing long-range dependencies, this method aims to enhance the accuracy and robustness of road segmentation. The U-Net architecture further refines the segmentation process by providing exact localization through its encoder-decoder structure. The authors show that this integrated model can effectively address the challenges of low precision and poor robustness often faced in road segmentation tasks, leading to more accurate and reliable results.

The study provides basis for developing automated, end to end Pavement Distress detection and segmentation. The authors explore the use of You Only Look Once version 4 (YOLOv4) and DeeplabV3 for this task. DeeplabV3 achieved a 0.56 Mean intersection over Union (mloU) and 0.58 Dice coefficient, while YOLOv4 obtained a 0.272 Mean Average Precision (mAP) on original images and 0.286 mAP on pre-processed data. YOLOv4 exhibited a high false negative for cracks and scaling, while DeeplabV3 showed high false positives, classifying normal pavement surfaces as scaling [19].

This paper develops a modified Convolution Neural Network for road lane detection, named Lane Mark Detector, which was employed using Encoder-decoder architecture, Dilated convolution, and Fine-tuned modifications. The approach achieved a 65.2% Mean intersection over Union (mloU), and an improved testing speed to 34.4 fps producing in a stable model that performs well, handling many variations in road scenes effectively [20].

# 2 Methodology

## 2.1 Proposed Approach

The goal of our experiment was to optimize road segmentation performance by testing multiple models. To achieve that, each team member was assigned one to two models, focusing on experimenting with different optimization techniques and tuning. In total, we worked with five models: Pyramid Scene Parsing Network (PSPNet), U-Net Convolutional Network, Fully Convolutional Network with Visual Geometry Group 16 (FCN-8 with VGG-16), Segmentation Network (SegNet), and DeepLab Version 3 (DeeplabV3). For each experiment the dataset was split into training, validation, and test set. We began by training all the models on the same 100 images as a baseline. Thereafter, the models were trained on additional 600 new images, which were produced through various augmentation methods. All the models were evaluated using the F1 score on training, validation, and testing dataset. Additionally, the models were tested on unseen data provided by the AIcrowd, where the performance was evaluated visually. The model that achieved the highest F1 score on the test set was selected as the wining model.

## 2.2 Dataset Description

The data that will be used [21]specifically designed for road segmentation tasks. It provides a set of satellite and aerial imagery acquired from Google Maps. The size of the dataset is 32.6 MB (34,193,647 bytes). It represents a set of 100 RGB satellite road images and their corresponding 100 grayscale ground Truth or mask. where all pixels are labeled as either road or background. In fact, these pixel-wise annotations enable the model to learn to segment roads from their precise and fine level. A separate set of test images is included, which will be used to evaluate the generalization performance of the model. This form of pixel-based annotation will provide highly detailed and accurate identification of road networks present in the images, which is very critical to achieve high-performance segmentation. Figure 2 Below shows a small set of images from the training set. Under each image correspond its ground truth.

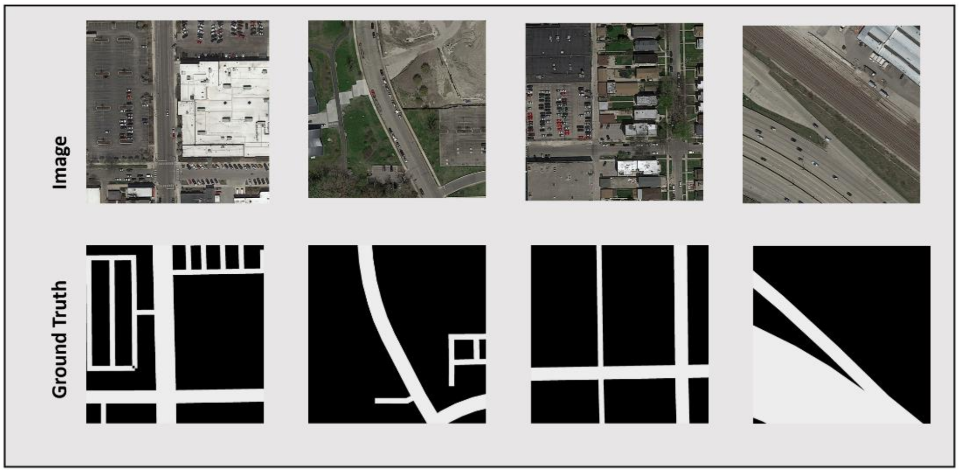

*Figure 1:training images with their labels*

## 2.3 Evaluation Metrics

The performance will be evaluated based on the F1 Score, which is quite a common metric in classification and segmentation tasks used in many researches [22] [23] [3]. The F1 score is the harmonic mean of precision and recall. It provides a well-balanced measure of accuracy, especially on very imbalanced datasets. Precision measures the percent of true positives among the ones predicted as positive, while Recall is the percent of true positives with respect to all actual positive cases. The F1 Score combines these two into a single number in such a way that the strengths of both aspects are captured in terms of the precision and completeness of the predictions. The F1 Score would then prove that segmentation was effective in targeting regions with both precision and recall taken into consideration. Below is F1, precision and recall formulas [23] [24].

**Recall** = TP / (TP + FN) **Precision** = TP / (TP + FP)

**F1** = 2 * (Precision * Recall) / (Precision + Recall)

## 2.4 Expected Outcomes

The primary expected outcome of this project is to find an efficient road segmentation model which segments roads from satellite images with high accuracy. It should have a high F1 score through a strongly efficient balance of precision and recall. It will classify each pixel properly into a label for road as **1** or background as **0**, based on the dataset provided. Through the implementation of the new approach or improvement, segmentation performance on the test set will surpass the benchmark models, particularly in areas where occlusions, shadows, and narrow roads may be present. Also, the work will effectively utilize the provided datasets and scripts such that model outputs will conform to the required format of submission and perform well on the test set. A rigorous evaluation will be done using the F1 score, precision, and recall to show the model's correctness. Quantitative results along with visual results are presented. The overall architecture, trainings, improvements, and the evaluation outcomes of the model are clearly documented in an extensive report, accompanied by a presentation overview of the project.

## 2.5 Methods of Validation and Analysis

The model of road segmentation proposed will be validated and analyzed on a train-validation split to ensure the most representative road types and backgrounds, and the final evaluation conducted on the provided unseen test set. Performance will be monitored mainly by the F1 score, which will be complemented by precision and recall, further complemented by visual inspection of segmentation masks overlaid on satellite images. While error analysis will point out occlusions, shadows, and other challenging situations. Improvements related to benchmarks and ablation studies of key components will be underlined through comparative studies. This ensures that the model validation is robust and reliable through quantitative metrics combined with qualitative analysis.

# 3 Feasibility Study

## 3.1 Technical feasibility

The project of extracting roads from satellite images is technically possible with the tools and resources available today. Models like U-Net, DeepLab, or SegNet are great for image segmentation, and transfer learning can help improve accuracy and save time during training. Frameworks such as TensorFlow and PyTorch offer the tools needed to build and train models, while OpenCV and NumPy handle image processing and data preparation. Platforms like Google Colab, with free access to GPUs, make it easier to manage the heavy processing required. Preprocessing involves resizing, normalizing, and augmenting the images. The models can use Binary Cross-Entropy loss and the Adam optimizer to learn effectively, and their performance will be measured using the F1 score to balance precision and recall. Challenges like small datasets and uneven class representation can be addressed with data augmentation, pre-trained models, and specialized loss functions. The project can also grow by expanding to other features in satellite images or by deploying the solution on cloud platforms for real-time use. With these tools and methods, the project is achievable and practical.

## 3.2 Technical Environment

The used environment in the study on road detection from high-resolution satellite images includes a combination of hardware and software resources. High-performance computing resources, particularly GPUs, are essential for accelerating the training and inference of deep learning models, along with sufficient RAM and storage to manage large datasets of satellite images. The implementation is typically carried out using popular deep learning frameworks like TensorFlow, and PyTorch which facilitate building and training neural networks. Additionally, image processing libraries such as OpenCV may be employed for tasks like resizing and normalizing the images. The programming is usually done in Python, often within Google Colab.

# 4 Data preprocessing

Effective data preprocessing is crucial for training robust and accurate deep learning models. In this section, we describe the preprocessing steps applied to the images and masks used in our semantic segmentation task [25].

## 4.1 Normalization

Normalization is a common preprocessing step that helps in regularizing the training process, improving the convergence speed of the model and making it more stable. In our case, both the images and masks were normalized by dividing their pixel values by 255.0. This scales the pixel values to the range [0, 1], which is beneficial for neural network training [26].

## 4.2 Resizing

To ensure consistency in input dimensions and to reduce computational load, the images and masks were resized from their original size of 400x400 pixels to 256x256 pixels. This resizing step is essential for fitting the data into the models, which most of them expect a fixed input size [27].

## 4.3 Data Augmentation

Data augmentation refers to a collection of techniques used to artificially increase the size of a dataset. These techniques involve applying small, random transformations to existing samples in a way that preserves their original labels. By creating variations of the data, augmentation helps address the challenge of needing large datasets for training deep neural networks, and as a result improving the performance and robustness of the models [28].In this project, a range of augmentation methods were applied to generate additional training samples. This process increased the dataset size from 100 images to 700 images, combining the augmented samples with the original dataset.

### 4.3.1 Data Augmentation Techniques

In this section, the used data augmentation techniques will be explained

**A. Geometric Transformation**

Geometric transformation is one of the most powerful and easiest methods to expand the dataset size by applying geometric manipulations. One common transformation was used is **rotation**, which is rotating the image by different angles, using rotation helps the model generalize better to different orientations of objects, reducing overfitting and improving performance on unseen data. Second method used is **reflection** or vertical/horizontal flipping ,which involves mirroring an image along the vertical axis (y-axis) for horizontal flipping or horizontal axis (x-axis) for vertical flipping, this help the model to focus on the essential characteristics of the object rather than its specific orientation .**Scaling** where also used , which is changing the size of an image while maintaining its aspect ratio , this helps the model in recognizing objects regardless of their size in the image, which is crucial for some applications [29] [30]. The figure 2 below shows an example of a geometric transformation.

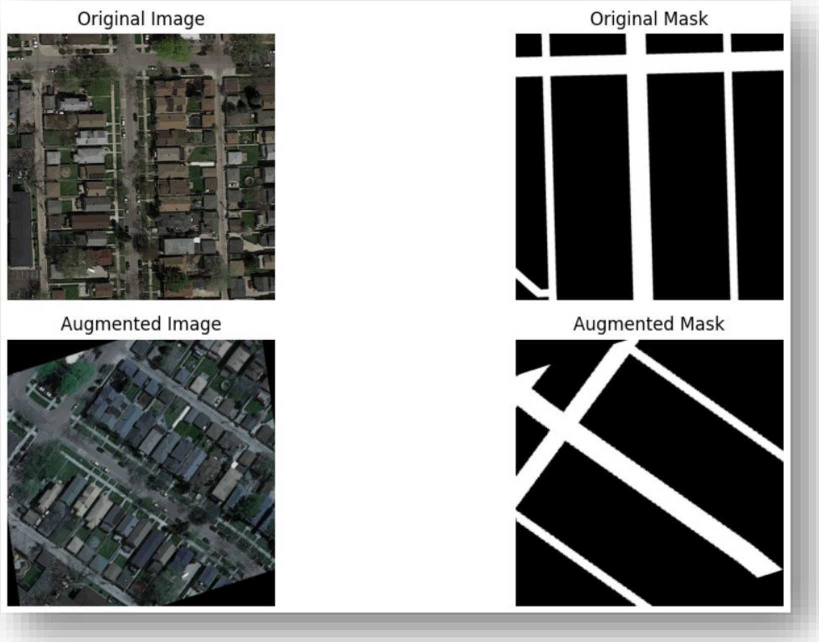

*Figure 2: geometric augmentation*

### B. Noise Addition

adding different types of noise (as Gaussian, salt-and-pepper, speckle) to images creates a more varied training dataset. This helps the model generalize better to different conditions and reduces overfitting and can perform well even if the input image is noisy or lower quality [31] [30].Gaussian noise is  characterized by a normal distribution, with a specified mean (usually 0) and variance, while salt and pepper noise randomly replaces some pixels in an image with either black or white, simulating random disturbances. Also, there is ISO noise, often referred to as digital noise, appears as grainy or speckled patterns in images. The figure 3 below show an example of noise addition augmentation.

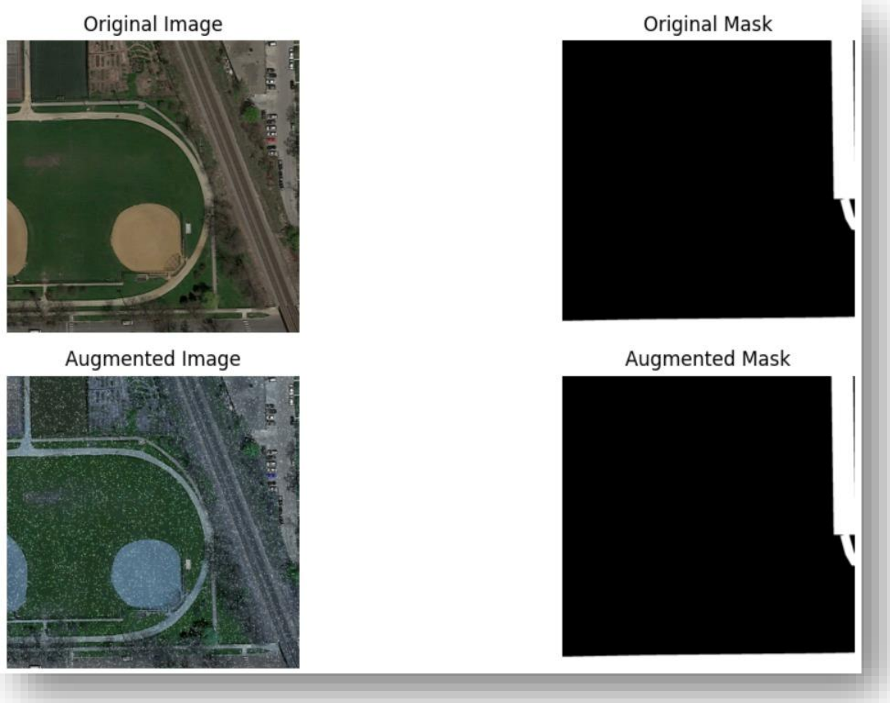

*Figure 3 : Noise Addition Augmentation*

**C. Synthetic Weather Augmentation**

These techniques are designed to simulate various adverse weather conditions that can affect the performance of machine learning models, particularly in computer vision tasks. Weather conditions such as **fog, snow** and **rain** [32]**.**methods in this case involves using models that account for light attenuation through particles in the air to simulate fog, overlaying snow effects on images, to simulate snow, and adding synthetic rain streaks to images, simulating the rain as shown in the figure 4 below.

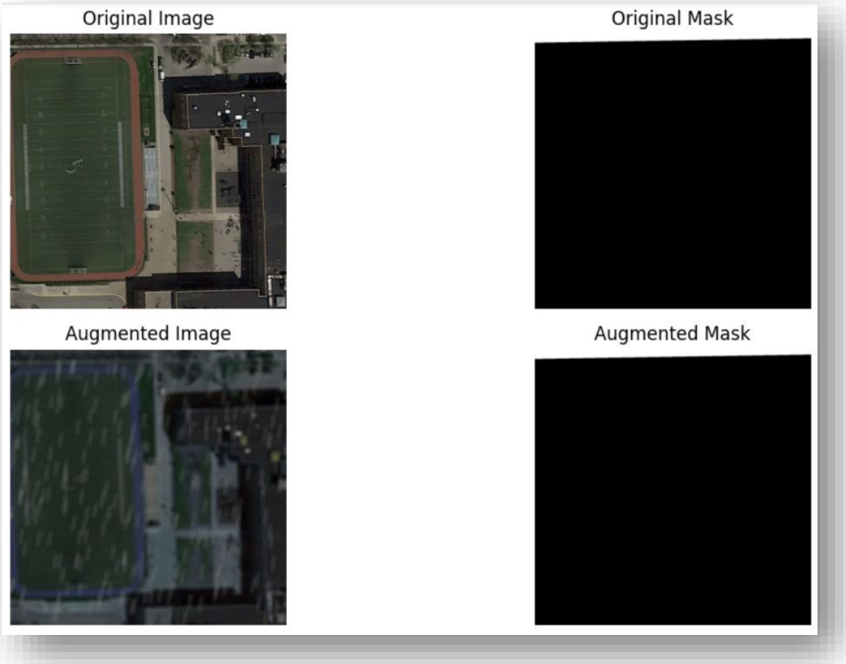

*Figure 4 : Weather Augmentation*

### D. Kernel filters

Kernel filters are a very popular technique in image processing to sharpen and blur images. These filters work by sliding an (n×n) matrix across an image with either a Gaussian blur filter, which will result in a blurrier image, or a high contrast vertical or horizontal edge filter which will result in a sharper image along edges. Intuitively, blurring images for Data Augmentation could lead to higher resistance to motion blur during testing [30] [33] . The figure 5 below show an example of kernel filters augmentation.

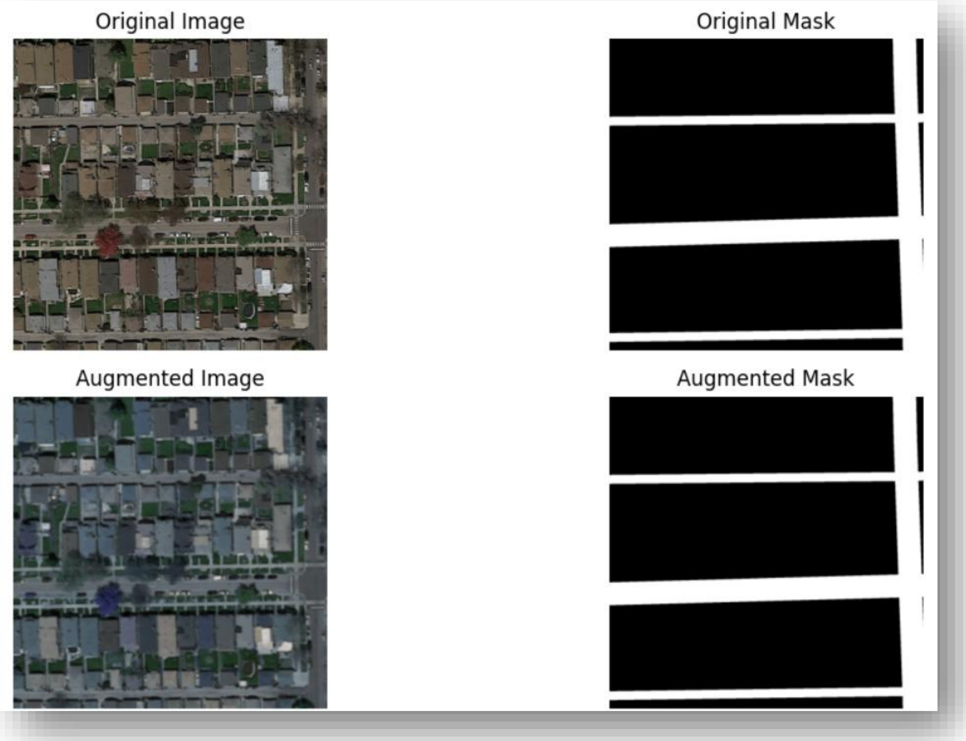

*Figure 5 : Kernel Filter augmentation*

### E. Erasing

Erasing as a data augmentation technique involves randomly removing parts of an image to create new training samples. This method helps improve the robustness of machine learning models by forcing them to focus on the most relevant features of the data, rather than relying on specific details that might not be present in real-world scenarios. By introducing variations in the training data, erasing can enhance the model's ability to generalize and perform better on unseen data [34]. The figure below shows an erasing augmentation example.

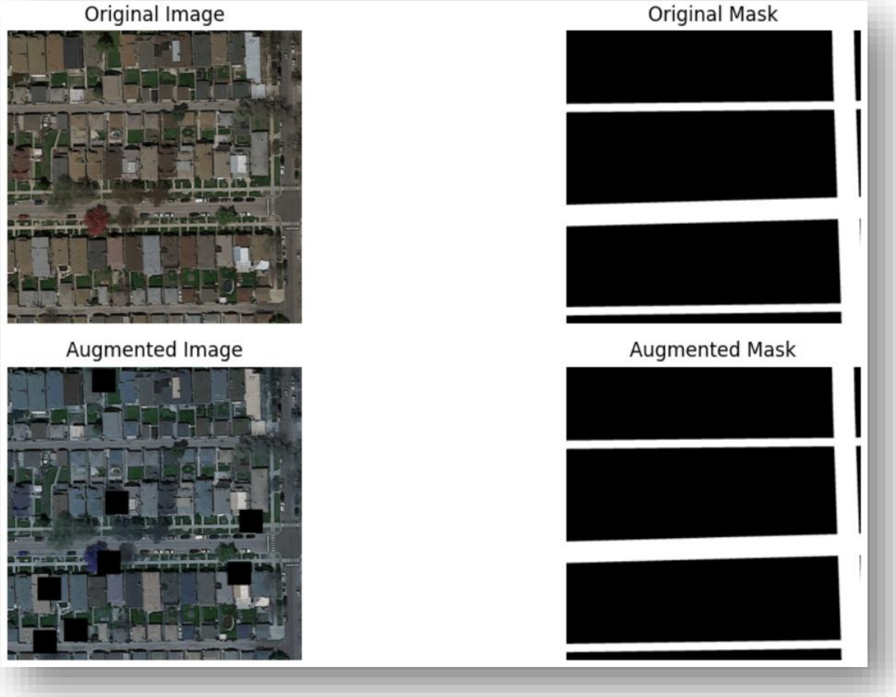

*Figure 6 : Erasing Augmentation*

### 4.3.2 Implementation and Augmented Dataset Details

For implementing data augmentation, the **Albumentations** library was used, which is an open-source Python library designed for fast and flexible image augmentations [35].Several augmentation techniques were applied to enhance the dataset, including geometric transformations, color manipulations, noise injection, and structural modifications. The probability of applying each transformation was set to 0.5 for most methods, except for certain techniques where the probability was specified differently (e.g., for Gaussian and ISO noise). The dataset size after augmentation was equal to 236 MB (248,054,537 bytes). Figure 7 below shows a combination of different type of augmentation applied at once.

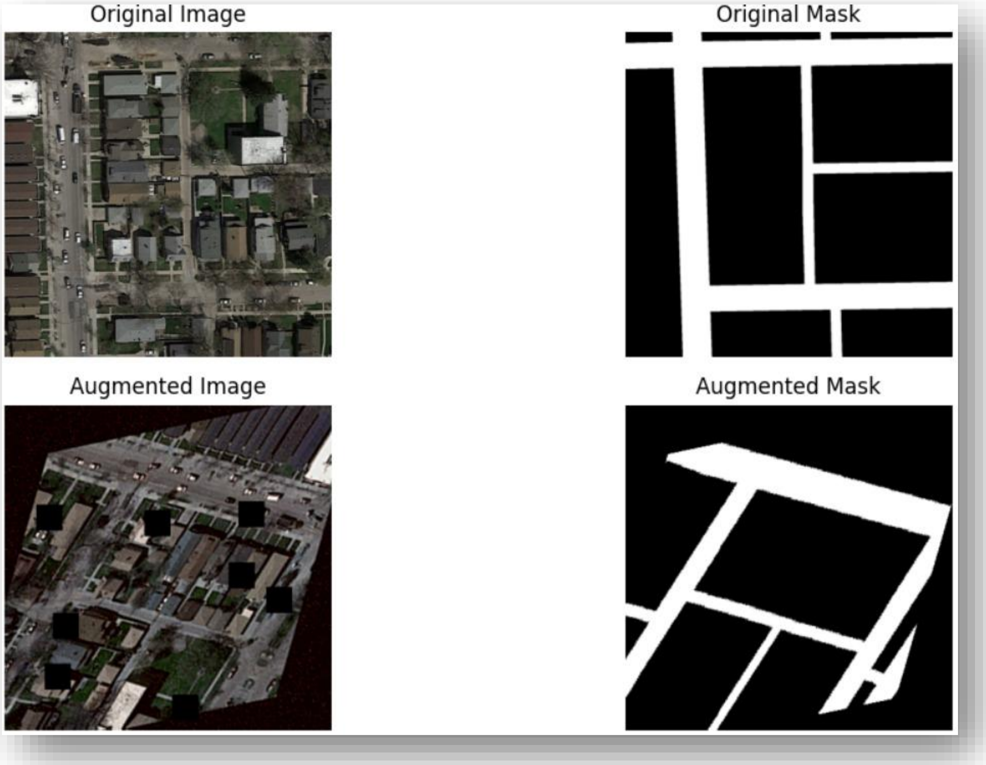

*Figure 7 : Geometric, noise and kernel applied at once.*

# 5 Models Architecture and Implementation

## 5.1 U-Net

### 5.1.1 U-Net Architecture

U-Net is a convolutional neural network architecture primarily designed for image segmentation tasks. It was first introduced in 2015 and has since become a standard model in the field. The U-Net architecture -as shown in Figure1- is marked by its U-shaped structure, which consists of a reducing path (encoder) and an expanding path (decoder). This design allows the network to capture both the context and the exact localization needed for accurate segmentation [36] [37].Figure 8 below shows the U-Net model architecture.

1) Encoder: The encoder is a typical convolutional network that reduces the spatial dimensions of the input while increasing the depth of the feature maps. This path consists of repeated applications of convolutions, each followed by a rectified linear unit (ReLU) and a max pooling operation.

2) Decoder: The decoder upsamples the feature maps and combines them with the corresponding feature maps from the encoder through skip connections. This helps in retaining spatial information that might be lost during downsampling. The expansive path consists of up-convolutions and concatenations with high-resolution features from the contracting path.

3) Skip Connections: These connections between the encoder and decoder ensure that the high-resolution features from the encoder are directly accessible to the decoder, which improves the segmentation accuracy.

4) Final Segmentation Layer: The final layer is a convolutional layer with a sigmoid activation function that outputs the segmentation map.

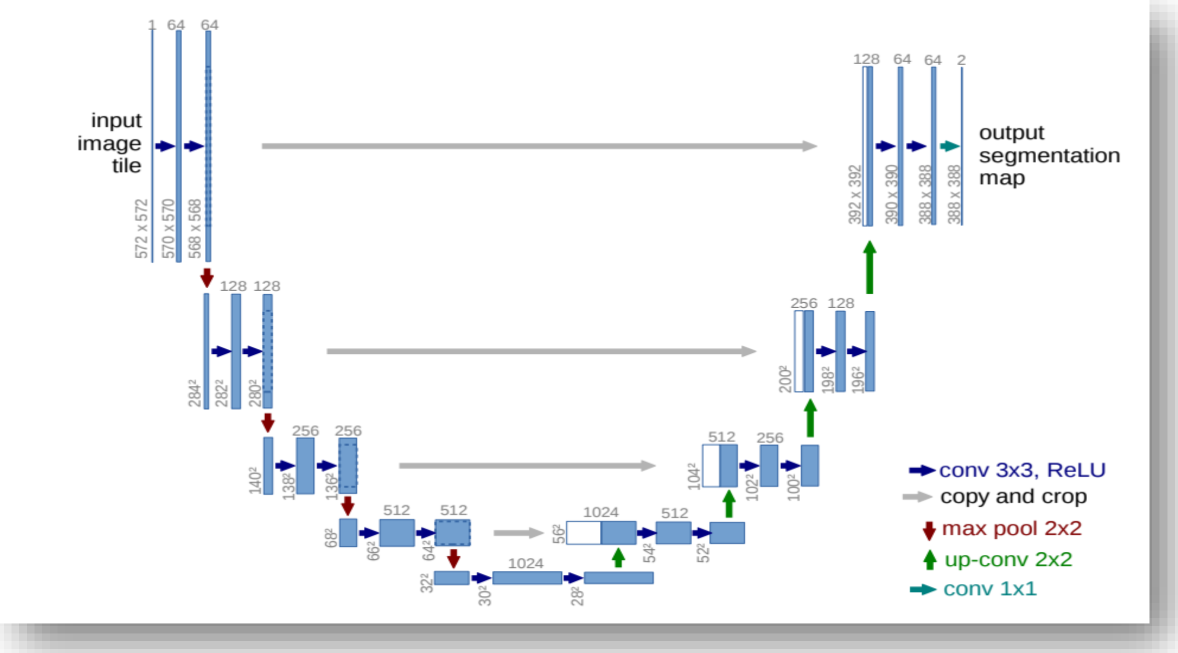

*Figure 8 : U-Net Model Architecture*

### 5.1.2 U-Net Implementation for the Road Segmentation Task

Different set of parameters were used to train the model for our task, and 700 images were used in exp1 and exp2 while only the 100 images were used in exp3 and the data were splitted into 80% for training and 20% for both validation and testing sets. The table below shows the experimental results of the U-Net model based on different parameters.

| | **Experiments** | | |
|---|---|---|---|
| | **exp1** | **exp2** | **exp3** |
| **# Of images** | 700 | 700 | 100 |
| **optimizer** | Adam | SGD | Adam |
| **batch size** | 8 | 16 | 8 |
| **learning rate** | 0.001 | 0.1 | 0.01 |
| **Loss metric** | binary cross entropy | binary cross entropy | binary cross entropy |
| **Epochs** | 25 | 35 | 100 |
| **F1-score training** | 0.8488 | 0.8946 | 0.7408 |
| **F1-score testing** | 0.8878 | **0.9081** | 0.7084 |

The Figure below shows the best model performance on predicting the mask of a testing sample.

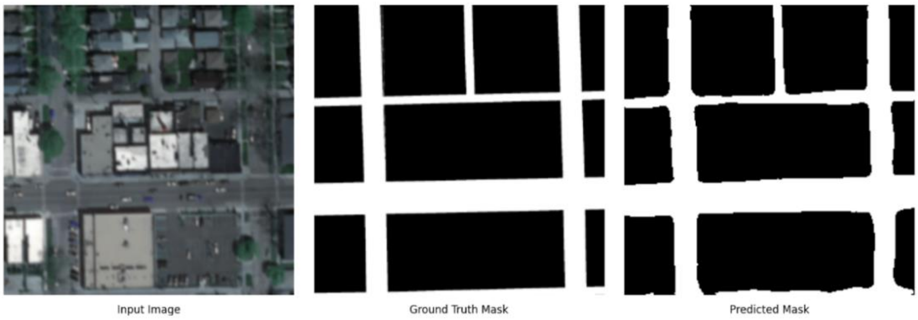

*Figure 9 :Test Sample Prediction Based on U-Net exp2*

### 5.1.3 U-Net with VGG-16 Encoder Implementation for the Road Segmentation Task

In this implementation, a pre-trained VGG16 network is used as the encoder. The initial layers of VGG16 are frozen to retain the learned features from ImageNet. The decoder then upsamples the feature maps and concatenates them with the corresponding encoder feature maps to reconstruct the segmented image.
VGG16 architecture consists of 16 layers, including 13 convolutional layers and 3 fully connected layers. The key characteristic of VGG16 is its use of small 3x3 convolution filters, which allows the network to learn more complex features while keeping the number of parameters manageable [38].

Different set of parameters were used to train the model for our task, and 700 images were used in exp1 and exp2 while only the 100 images were used in exp3 and the data were splitted into 80% for training and 20% for both validation and testing sets.

The table below shows the experimental results of the U-Net with VGG-16 Encoder based on different parameters.

| | Experiments | | | |
|---|---|---|---|---|
| | **exp1** | **exp2** | **exp3** | **exp4** |
| **# Of images** | 700 | 700 | 700 | 100 |
| **optimizer** | SGD | Adam | SGD | Adam |
| **batch size** | 16 | 16 | 16 | 16 |
| **learning rate** | 1e-3 | 5e-4 | 1e-3 | 1e-3 |
| **Loss metric** | dice loss | dice loss | dice loss | dice loss |
| **Epochs** | 250 | 250 | 100 | 100 |
| **F1-score training** | 0.8382 | 0.9319 | 0.8138 | 0.4983 |
| **F1-score testing** | 0.8150 | **0.9046** | 0.8072 | 0.226 |

The Figure 10 below shows the best U-Net with VGG-16 Encoder model performance on predicting the mask of a testing sample.

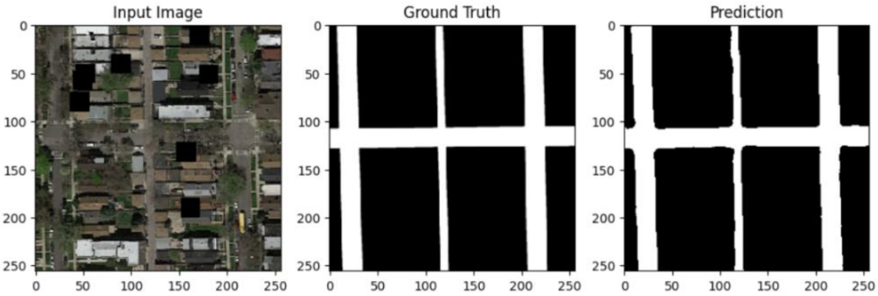

*Figure 10 : Results of Exp2 in U-Net with VGG-16 Encoder*

# 5.3 SegNet

## 5.3.1 SegNet Architecture

SegNet is a deep learning model that helps identify different parts of an image, like roads in satellite pictures. It works by first breaking down the image through a series of layers that extract important features, such as colors and shapes. This part is called the encoder. After that, it rebuilds the image using another set of layers called the decoder, which makes a detailed map showing where the roads are. One of the best things about SegNet is that it uses memory efficiently, so it can run well even on less powerful computers. This makes it great for real-time tasks, like tracking city changes or helping self-driving cars. Using SegNet for road segmentation allows you to accurately spot paved roads in satellite images, which is important for city planning and managing infrastructure. Its ability to deal with different looks and sizes of roads makes it very useful in real-life situations. Figure 11 below shows encoder decoder architecture in the SegNet model.

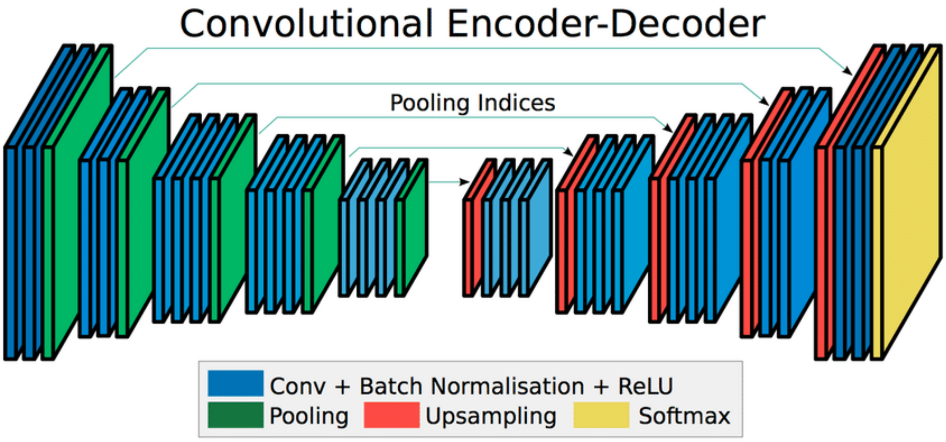
Convolutional Encoder-Decoder
Pooling Indices
Conv + Batch Normalisation + ReLU
Pooling
Upsampling
Softmax

*Figure 11 : SegNet Encoder Decoder*

## 5.3.2 SegNet Implementation for the Road Segmentation Task

### Baseline Experiments

The SegNet model was utilized for road image segmentation on a dataset of 100 paired road images and masks. Images were resized to a target size of (256, 256), and masks were normalized to a range of 0 to 1. The dataset was split into a training set (80 images), validation set (10 images), and test set (10 images). Training was conducted over 50 epochs with a batch size of 16, using the Adam optimizer and binary cross-entropy loss function. The model's architecture had 16,238,339 parameters, of which 16,207,747 were trainable.

During training, the SegNet model demonstrated progressive improvement in accuracy and loss. By epoch 22, it achieved a training accuracy of 75.31% with a loss of 0.4181 and a validation accuracy of 77.71% with a loss of 0.3882. Over subsequent epochs, the performance fluctuated slightly but showed a consistent trend of improvement. By epoch 50, the test accuracy reached 77.15%, with a test loss of 0.2902, confirming the model's ability to generalize effectively.

The SegNet model's effectiveness was demonstrated in segmenting road areas with high accuracy. The use of the Adam optimizer and binary cross-entropy loss ensured effective weight updates and precise handling of pixel-wise classification. While the initial performance was promising, future optimizations, such as experimenting with advanced architectures or fine-tuning hyperparameters, could further improve the segmentation quality for applications like autonomous driving and road analysis. The figure shows the predicted mask when the F1 score was 0.7039. Figure 12 below shows a sample of the best result prediction on test set.

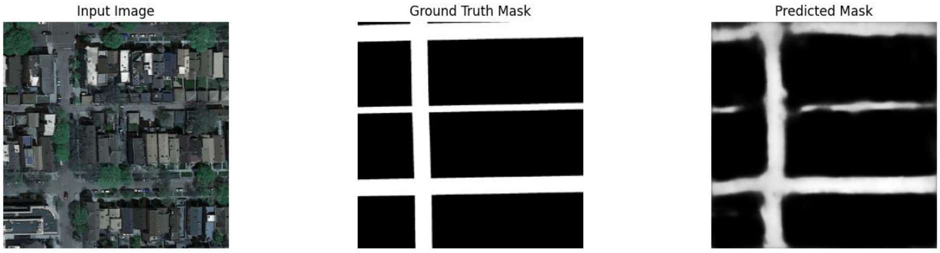

*Figure 12 : SegNet Mask Prediction on 100 images*

### Augmented data Experiments

The SegNet model was applied to road image segmentation using a dataset of 700 paired road images and masks, divided into training (80%), validation (10%), and test (10%) sets. The model, employing an encoder-decoder architecture with an input shape of (256, 256, 3), was trained over 50 epochs using the Adam optimizer and binary cross-entropy loss, achieving a training accuracy of 76.75% (loss: 0.0763), validation accuracy of 75.85% (loss: 0.1494), and test accuracy of 76.43% (loss: 0.136).

To enhance the model's performance, additional data augmentation techniques were applied during training. These included random rotations, shifts, flipping, zooming, and the addition of Gaussian noise. The augmentations significantly improved the results, increasing the test accuracy to 81.94% and the F1 score to 0.9387. The predicted masks at this stage closely resembled the ground truth, highlighting the effectiveness of these augmentations in improving the model's ability to segment road areas accurately.

The augmented images were generated dynamically on-the-fly, ensuring that the total dataset size remained at 700 while increasing the diversity and effective number of training examples. This method not only prevented overfitting but also helped the model generalize better. Evaluation metrics, including Precision-Recall curves, ROC curves, and the Dice Coefficient, further validated the model's strong ability to accurately capture road regions.

This project demonstrated SegNet's robustness and potential in road segmentation tasks. Future directions for improvement could include exploring alternative architectures like U-Net, experimenting with advanced loss functions such as focal loss, or leveraging pre-trained models. This work emphasizes the practicality of deep learning techniques in real-world applications, such as autonomous driving and road condition monitoring.

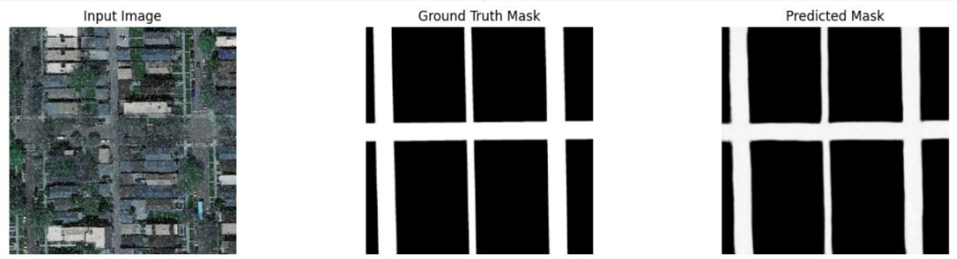

*Figure 13 : SegNet Prediction on 700 images*

The Table shows Segnet Summary of results.

| Metric | 100 Images | 700 Images |
|---|---|---|
| Training Accuracy | 75.31% | 76.75% |
| Validation Accuracy | 77.71% | 75.85% |
| Test Accuracy | 77.15% | 76.43% |
| F1 Score | 0.7039 | 0.9336 |
| Test Loss | 0.2902 | 0.136 |

## 5.4 DeepLabv3

### 5.4.1 DeepLabv3 Architecture

The Deeplabv3 model is designed to address the challenge of segmenting objects at multiple scales. Unlike traditional approaches, where the encoder relies on repeated max-pooling and striding operations to produce a compressed representation at a reduced resolution [39], Deeplabv3 introduces a unique feature: atrous convolution also known as dilated convolution. It is applied in cascade or in parallel using multiple atrous rates to capture multi-scale context effectively [40].

Atrous convolution allows for explicit control over the resolution of feature computed by deep convolutional neural networks. By spacing out filter parameters with inserted zeros, the convolution kernel increases its receptive field size without adding more learnable parameters. This adjustment enables the filter to capture multi-scale information and generalizes the standard convolution operation [41] . For instance, a 3*3 kernel, which represents the number of parameters, achieves 5*5 receptive field when an atrous rate of 1 is applied. This capability facilitates dense feature extraction and support the construction of deeper network without compromising dense prediction tasks, where detailed spatial information is crucial [40].

The architecture of the deeplabv3 model comprises two primary components: the backbone with atrous convolution and the Deeplabv3 head.

The down sampling backbone is responsible for extracting the shallow features from the input image, while atrous backbone encodes deep features at a desired resolution. Together, these two components form subparts of backbone with atrous convolution.

The Deeplabv3 head includes an Atrous Spatial Pyramid Pooling (ASPP) block, which applies four parallel atrous convolutions with varying atrous rates to the feature map. This enables the model to effectively capture multi-scale context. The project to output subblock, projects the feature maps to the desired number of segmentation classes two classes in our case: 0 for the background and 1 for the road.

Finally, the model employs bilinear upsampling as the final subblock, ensuring that the output image matches the resolution of the input [40]. In this task, the input images were resized to 256x256, resulting in output masks of the same size. DeepLabv3 architecture shown in Figure 14 below.

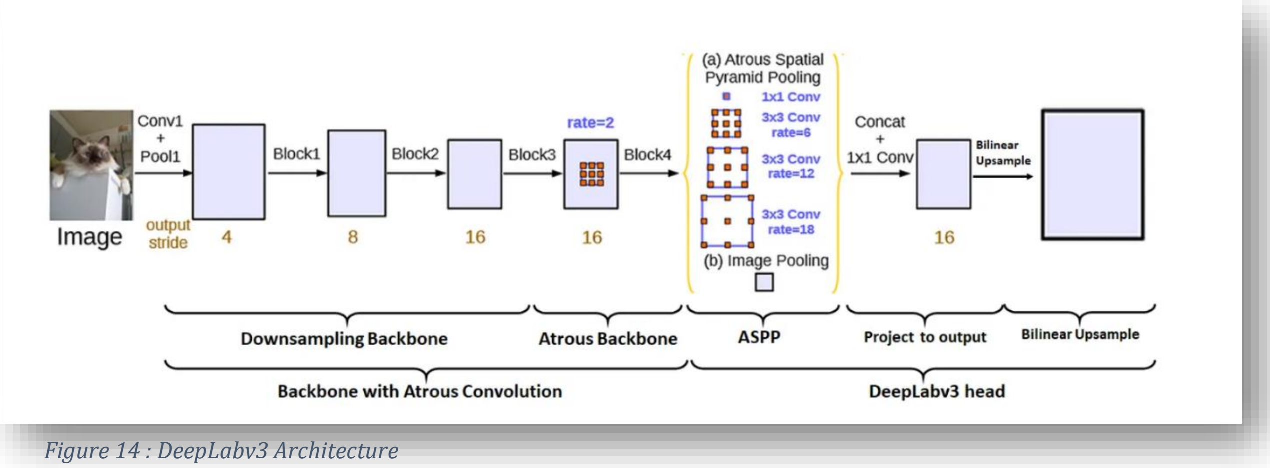

*Figure 14 : DeepLabv3 Architecture*

### 5.4.2 DeepLabV3 Implementation for the Road Segmentation Task

To apply the DeeplabV3 model to our road segmentation task, we used the pre trained version of Deeplabv3 with Resnet101 backbone. The 101 refers to 101 layers of the ResNet architecture. The Atrous Spatial Pyramid Pooling (ASPP) module has 4 layers, each with different dilation rates. The output layer was modified for binary classification, converting the features from the backbone and ASPP into pixel-wise segmentation output.

**Baseline Experiments**

The experiment began with training this model on 100 satellite images provided by the "Road segmentation" competition hosted on AIcrowd. The dataset was split into training, validation and test sets in a 70:15:15 ratio. We started by training the model using Adam optimizer, an adaptive learning rate optimization algorithm that combines momentum and scaling. Adam is suited for non-stationary problems with noisy and/or sparse gradients, providing advantages over standard SGD [42]. The learning rate was set to 0.0001, which is a standard practice for initial experiments. We used a batch size of 16 and trained for 10 epochs.

After training, the model achieved a loss of 0.2903 on the validation dataset and F1 score of 0.9026 on the test dataset. Observing the loss decreases over the epochs, we decided to increase the number of epochs to 35. This improved the loss to 0.1420 on the validation dataset and boosted the F1 to 0.9136 on the test dataset. Finally, before increasing the dataset to 700 images through augmentation, we trained for 50 epochs, achieving a validation loss of 0.0994 and an F1 score of 0.9225 on the test dataset.

**Augmented data Experiments**

Different Data augmentation techniques were applied on the original dataset, including geometric transformations, noise addition, synthetic weather effects, and blur. This process expanded the dataset from the original 100 images to a total of 700 images, with 600 new augmented samples.

For this experiment, we introduced cosine annealing as the learning schedule. Cosine annealing starts with a large learning rate that gradually decreases to a minimum value before increasing rapidly again, mimicking a simulated restart of the training process. This approach, combined with warm restarts, reuses effective weights as the starting point, in contrast to cold restart that initialize the weights randomly [43].

We increased the number of epochs to 80 while keeping the optimizer, Adam, and batch size, 16, unchanged. This setup achieved a validation loss of 0.1349 and an F1 score of 0.9113 on the test dataset.

The table below shows the summary of results of DeepLabv3 model.

| | **Experiments** | | | |
|---|---|---|---|---|
| | **Exp1** | **Exp2** | **Exp3** | **Exp4** |
| **optimizer** | Adam | Adam | Adam | Adam |
| **batch size** | 16 | 16 | 16 | 16 |
| **learning rate** | 0.0001 | 0.0001 | 0.0001 | 0.001 |
| **Loss metric** | Binary Cross entropy with logits loss | Binary Cross entropy with logits loss | Binary Cross entropy with logits loss | Binary Cross entropy with logits loss |
| **Epochs** | 10 | 35 | 50 | 80 |
| **F1-score on test** | 0.9026 | 0.9136 | 0.9225 | 0.9113 |
| **Data augmentation** | None | None | None | 600 new instances |

The figure below shows a one sample mask prediction by the best DeepLabv3 model that achieve 0.9225 as F1 on test set, trained on 50 epochs with no applied augmentation.

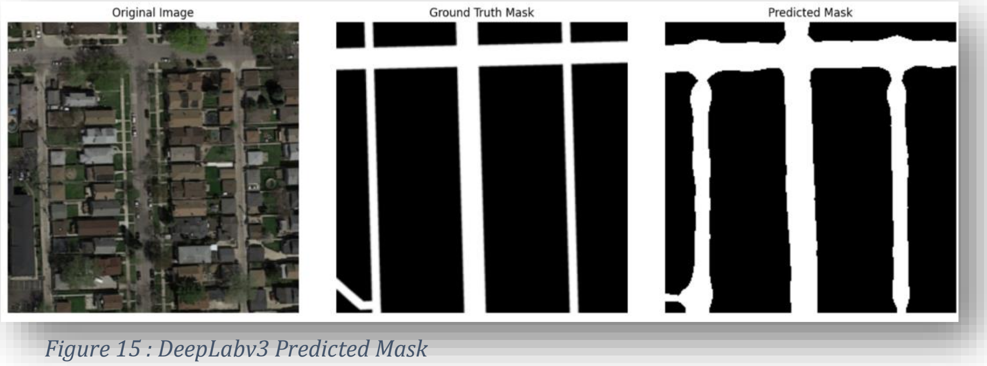

*Figure 15 : DeepLabv3 Predicted Mask*

## 5.5 PSPNet

### 5.5.1 Pyramid Scene Parsing Network (PSPNet) Architecture

PSPNet stands for Pyramid Scene Parsing Network. It is a semantic segmentation model that uses a pyramid parsing module, which leverages global context information through different-region based context aggregation. These complementary local and global clues further present the final prediction in a much more reliable way [15].

**Pyramid Pooling Module**

Pyramid Pooling Module or PPM is the key concept of PSPNet. PPM address the challenge of incorporating global context into semantic segmentation models. While theoretically, deep networks like ResNet have a large receptive field the empirical receptive field—the actual area of influence during feature extraction—is often much smaller, especially in higher layers [15]. This limited receptive field reduces the model capacity to effectively leverage global information about the scene, which is essential for parsing complex scenes.

The PMM resolves this by employing a multi-scale pooling strategy to aggregate context from diverse spatial scales where the Feature maps from the last convolutional layer of the network are pooled at four scales which are Global pooling (1*1) to capture the overall context of the image, half pooling (2*2) that capture medium scale spatial relationships. Then Fine-grained pooling (3×3) and Smaller sub-region pooling (6×6) capture local details. This pyramid structure enables the model to capture subtle variations in boundaries or textures of objects. Experiments in research papers shows that incorporating all four levels significantly outperforms fewer scales or single-level pooling, especially in datasets that have a high diversity of scene layouts [15] [44] .

Each pooled feature map undergoes a 1×1 convolution that reduces the dimensionality of each scale to 1/N of the original feature size, where N represents the number of pyramid levels. These downsampled features are further upsampled to match the dimensions of the original feature map with bilinear interpolation. Last, features from all levels are concatenated with the original feature map after upsampling, providing the final fused representation effectively combining the global and local hierarchical context [15] [44] .

The Pyramid Pooling Module is designed to adapt to different sizes of the input image or feature map resolution by employing its four pyramid levels. The figure below shows the procedures of PPM.

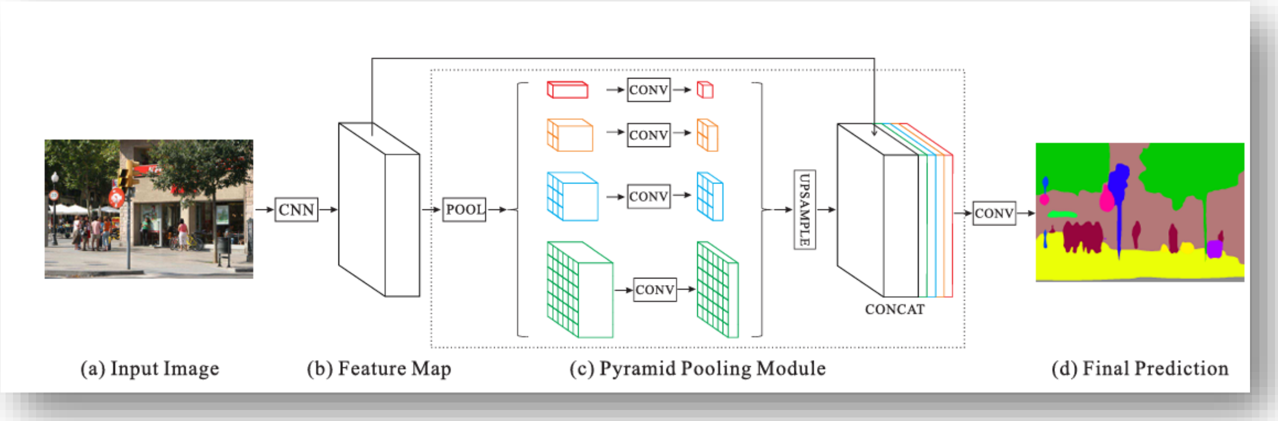

*Figure 16 : PPM Architecture*

**Pyramid Scene Parsing Network**

The Design structure of PSPNet integrates the PPM with a ResNet-based encoder in order to create a robust framework for semantic segmentation. The pipeline begins with the ResNet encoder, which extracts hierarchical features from the input image. Using a dilated convolution strategy, the encoder generates a feature map that is 1/8 the size of the input image, preserving fine-grained spatial details while expanding the receptive field.

After that, the feature map is run via the Pyramid Pooling Module, which, as previously mentioned in this section, collects local and global context. To guarantee that the network maintains both hierarchical global context and specific local features, the PPM's output is then concatenated with the encoder's original feature map. This combined representation is processed by a convolutional layer, which generates the final pixel-wise predictions. A segmentation map that matches the dimensions of the input image is then created by bilinearly upsampling the predictions to the original input resolution [15].

### 5.5.2 PSPNet Implementation for the Road Segmentation Task

We used the **segmentation_models_pytorch** library's PSPNet architecture in this implementation, which makes it easier to deploy cutting-edge semantic segmentation models. The encoder in the model is a ResNet-101 backbone that has been initialized with ImageNet pretrained weights. Especially for jobs requiring complex images, these pretrained weights offer a considerable advantage by giving the model a solid baseline for feature extraction. Hierarchical characteristics that form the basis of segmentation are extracted by the encoder.

The PSPNet architecture has three major components: the encoder, the Pyramid Pooling Module, and the segmentation head. The encoder, based on a ResNet-101 backbone, starts with a 7x7 convolution, batch normalization, ReLU activation, and a max pooling layer to reduce spatial resolution. It has four residual layers, each with multiple bottlenecks: Layer 1 contains 3 bottlenecks, Layer 2 has 4, Layer 3 has 23, and Layer 4 has 3. Each of the bottlenecks comprises three convolutional layers with

1x1, 3x3, and 1x1 configurations, respectively, together with batch normalization, ReLU activation, and residual connections. Downsampling is performed in the first block of every layer, reducing dimensions. PPM applies adaptive average pooling at different scales, including 1 × 1, 2 × 2, 3 × 3, and 6 × 6, followed by a 1 × 1 convolution, batch normalization, ReLU activation, and upsampling. These are then concatenated with the encoder output for enhancing spatial and global context. Finally, it applies a 3 × 3 convolution, bilinear upsampling, and sigmoid activation in the segmentation head to produce the final pixel-wise segmentation mask, allowing PSPNet to efficiently harvest both local and global features for accurate segmentation.

To secure standardized model input dimensions, the original size of the data we have was 400×400 for each image and mask. These images were resized to 256×256 before being fed into the model. This, therefore, made the model yield segmentation masks of size 256 × 256, matching the input that has been resized. However, in cases where masks need to be in their actual resolution, for example, 400 × 400, the output can be resized back into original dimensions as a post-processing using nearestneighbor interpolation. The resizing approach here enable the model to have a standard input size while the model remains compatible with downstream tasks that require the original resolution.

When pretrained weights were used, PSPNet has 2,237,889 trainable parameters. These weights allow fine-tuning while retaining the base ImageNet features. If one does not use pretrained weights, all the parameters are initialized randomly; this means the number of trainable parameters will be the same but it will require a lot more training data and time to have similar performances.

## Baseline Experiments

The model is first trained on the original dataset that contain 100 RGB image with their 100 gray scale corresponding masks. each have the size of 400*400 pixels then resized into 256*256. The model training employs a 5-fold cross-validation strategy, with each fold trained for 50 epochs using a learning rate of 0.02. Stochastic Gradient Descent (SGD) with a momentum of 0.9 is used as the optimizer. The table below show the test set F1 and loss results with different batch sizes (BZ).

| | **8 BZ** | **16 BZ** | **32 BZ** |
|---|---|---|---|
| **F1** | 0.9577 | 0.9314 | 0.9145 |
| **Loss** | 0.0451 | 0.0727 | 0.0908 |

## Augmented data Experiments

In the second phase, the data was augmented, 700 RGB image with their 700 gray scale corresponding each having the size of 400*400 pixels. The dataset was then split into 70% for training, 15% for validation, and 15% for testing. Stochastic Gradient Descent (SGD) with a momentum of 0.9 is used as the optimizer. The number of epochs, learning rate (LR), and batch size are the variables that have been manipulated. The table below show the test set F1 and loss results for each different variable.

- Test results of test set each with different batch sizes, fixed LR equal to 0.02 and 200 epochs.

| | **8 BZ** | **16 BZ** | **32 BZ** |
|---|---|---|---|
| **F1** | 0.9668 | 0.9585 | 0.9465 |
| **Loss** | 0.0345 | 0.0433 | 0.0559 |

- Test results of test set when batch size equal to 8 and LR equal to 0.01

| | **50 epochs** | **100 epochs** | **200 epochs** |
|---|---|---|---|
| **F1** | 0.9233 | 0.9470 | 0.9590 |
| **Loss** | 0.0800 | 0.0555 | 0.0430 |

- Test results of test set when batch size equal to 8 and LR equal to 0.02 with different number of epochs.

| | **200 epochs** | **350 epochs** | **500 epochs** |
|---|---|---|---|
| **F1** | 0.9668 | 0.9710 | 0.9723 |
| **Loss** | 0.0345 | 0.0301 | 0.0286 |

The combination of small batch size, 8, and learning rate of 0.02, in this task, served the PSPNet model very well, as can be judged from the training and validation curves showing steady convergence and very minimal loss with an F1 score nearing 1. Even though a small batch size normally introduces stochasticity in gradient updates, this may serve as implicit regularization and prove to be very useful for pixel-wise segmentation tasks such as road segmentation, where fine-grained details are so crucial. The architecture of the PSPNet model, based on the ResNet-101 backbone with hierarchical feature extraction, amplifies this advantage since it effectively leveraged detailed updates from smaller batch sizes. Additionally, in the case of PSPNet, its Pyramid Pooling Module (PPM) more effectively integrates local and global features, thus being more resistant to noisy updates that a smaller batch size can yield. The relatively high learning rate of 0.02 speeds up convergence by allowing the model to achieve a near-optimal solution in fewer epochs. Studies have suggested that small batch sizes allow the models to generalize better because they explore a wider region of the loss landscape and avoid sharp minima, which may cause overfitting, particularly in deep architectures such as PSPNet. Moreover, the introduction of momentum in the optimizer (SGD with momentum 0.9) smoothens noisy updates, thus making training stable with this higher learning rate. The small batch size, high learning rate, and architectural robustness balance out in such a way that this parameter setting was particularly effective [45] [15]. The relatively high learning rate accelerates the convergence, allowing the model to reach near-optimal solutions in fewer epochs. This rapid convergence is of essence in training deep learning models efficiently while maintaining performance. Some studies suggest that higher learning rates can help models generalize better by facilitating exploration of the loss landscape [46].

The learning curves for 500 epochs show a continued improvement in performance, with the F1 score reaching 0.9723 and the loss dropping to 0.0286. These results show that extending the training to 500 epochs allows the model to further refine its predictions, achieving marginal but significant improvements over 200 and 350 epochs. The convergence of both training and validation curves indicates strong generalization and no signs of overfitting, even at 500 epochs. This balance highlights the effectiveness of the PSPNet architecture and the chosen parameters in terms of batch size and learning rate to leverage extended training for pixel-precise segmentation. The figure below shows the learning curves of the loss and F1 score between training and validation sets.

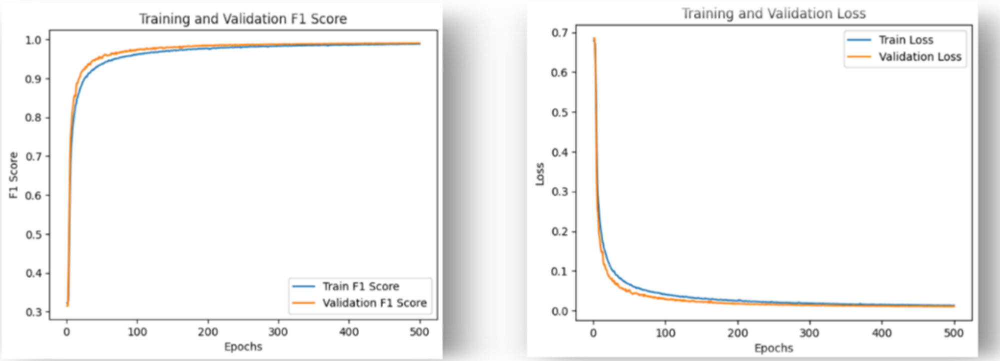

*Figure 17 : F1 and loss Learning curves*

The decision to stop at 500 epochs instead of at 350 epochs was based on the observation from the experiments of incremental improvements in performance and diminishing returns. Comparing the F1 score and loss, the difference between 350 and 500 epochs is small: the F1 score improved from 0.9710 to 0.9723, while the loss decreased from 0.0301 to 0.0286. But this minor enhancement can be crucial in tasks like road segmentation, where even marginal gains in precision can have huge implications for downstream applications. Still, 350 epochs provide a great result and would be chosen over the choice of 500 epochs in return of resources and time complexity.

Looking at earlier results, the improvement from 200 epochs, F1 = 0.9668, Loss = 0.0345, to 350 epochs, F1 = 0.9710, Loss = 0.0301, is more pronounced than the improvement from 350 to 500 epochs. This indicates that by 350 epochs, the model is close to convergence but still benefits from the extra training to fine-tune its predictions.

This is justified at 500 epochs if the improvement is as desired by the project, because further training beyond 500 epochs would likely result in diminishing returns while increasing computational cost and training time. Results have shown that the combination of batch size 8 and learning rate 0.02 enables the model to generalize well, leveraging longer training durations without overfitting. Below best PSPNet model predictions.

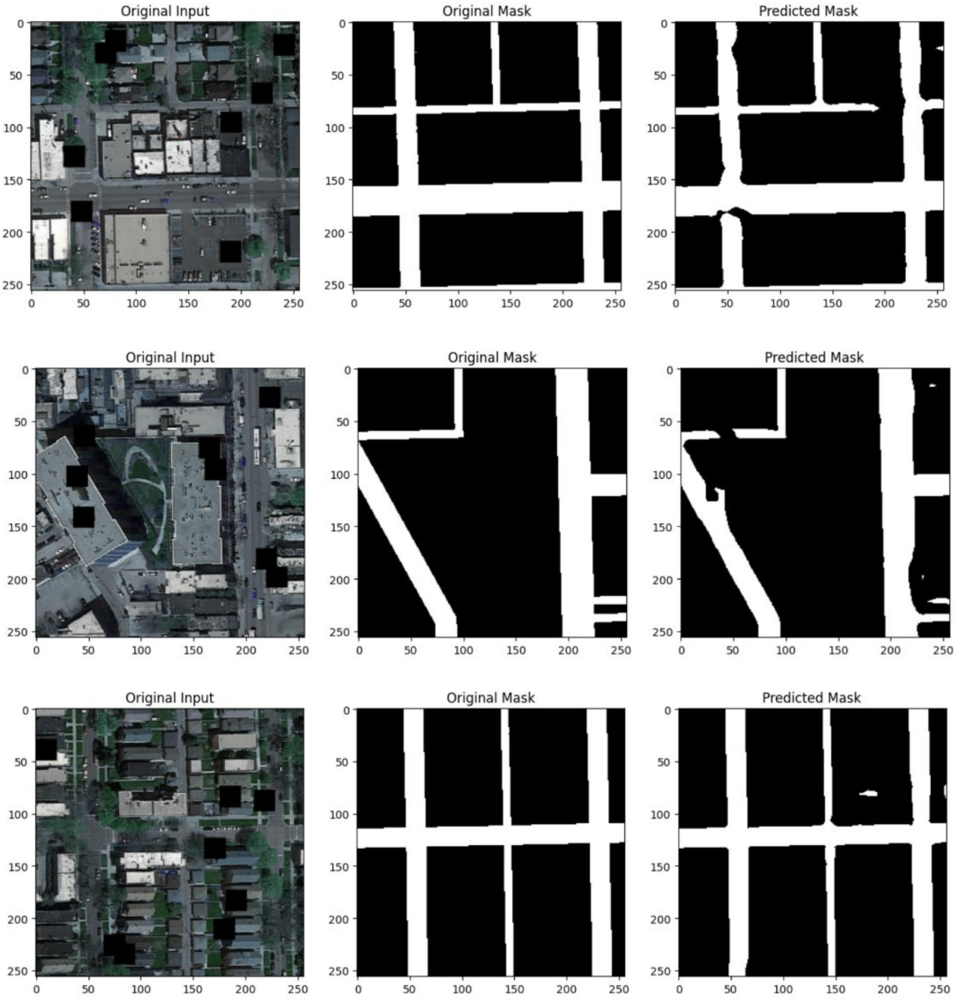

*Figure 18 : PSPNet predictions*

# 6 Model Comparison and Selection

The best model for the segmentation of roads was PSPNet since it showed the highest performance: F1-score with 0.9723 and test loss at 0.0286, significantly outperforming the other models. While U-Net and DeepLabv3 report competitive performances, the PSPNet architecture includes a Pyramid Pooling Module (PPM) that effectively integrates both local and global contextual information. This allows PSPNet to capture finer details in images, especially for road segmentation. Also, the ResNet-101 backbone of PSPNet can go deeper in feature extraction with spatial information preserved, and its hierarchical combination of features is one of the reasons behind its high segmentation accuracy.

Compared to SegNet, with an F1-score of 0.9336, PSPNet enjoys advanced context aggregation with multi-scale pooling, so it can generalize well from variations in road textures and boundaries. Additionally, with a small batch size of 8 and a high learning rate of 0.02, the PSPNet generalizes well to avoid overfitting at maximum performance.

While there are other models, such as U-Net with VGG-16, that are similarly based on pre-trained encoders, these seemed to underperform the PSPNet, possibly because these did not have mechanisms to embed global context information into pixel classification, like those that are in PPM. DeepLabv3 recorded an F1-score of 0.9225; though performing well, multi-scale context through atrous convolution seems less effective compared with the multi-level pyramid of PSPNet.

In the end, it is the ability of PSPNet to integrate global and local features, together with efficient handling of hierarchical data, that makes it the best choice for the road segmentation task.

**The table below shows the comparisons between models.**

| Model | Dataset Size | Optimizer | Batch Size | Learning Rate | Loss Metric | Epochs | F1-Score (Test) |
|---|---|---|---|---|---|---|---|
| **U-Net** | 700 images | Adam | 16 | 0.001 | Binary Cross-Entropy | 35 | 0.9081 |
| **U-Net with VGG-16** | 700 images | Adam | 16 | 0.0005 | Dice Loss | 250 | 0.9046 |
| **SegNet** | 700 images | Adam | 16 | 0.001 | Binary Cross-Entropy | 50 | 0.9336 |
| **DeepLabv3** | 700 images | Adam | 16 | 0.0001 | Binary Cross-Entropy with Logits | 50 | 0.9225 |
| **PSPNet** | 700 images | SGD | 8 | 0.02 | Binary Cross-Entropy | 500 | 0.9723 |

# 7 Conclusion

In this report, we explored the challenges and advancements in detecting roads from satellite images using deep learning techniques. After evaluating several models, we found that the Pyramid Scene Parsing Network (PSPNet) emerged as the best performer for road segmentation tasks. This model achieved an impressive F1 score of 0.9723 and a loss of 0.0286 after 500 epochs, with a batch size of 8 and a learning rate of 0.02. This performance demonstrates its ability to accurately identify road areas even in challenging conditions like shadows and occlusions.

The success of the PSPNet model can be attributed to its advanced architecture, which effectively captures both local and global contextual information. This capability is crucial for accurately delineating roads, especially in complex environments where road boundaries may not be clearly defined.

Our findings highlight the importance of accurate road detection for applications in urban planning, traffic management, and emergency response. The improvements we achieved with PSPNet over previous models underscore the potential of deep learning in enhancing road segmentation.

Looking ahead, future work could focus on optimizing the PSPNet model for even greater efficiency, exploring additional data augmentation techniques, and testing the model in real-world scenarios. Additionally, incorporating temporal data from satellite imagery could further enhance the model's adaptability to changing environments.

In summary, this project not only showcases the effectiveness of the PSPNet model in road segmentation but also paves the way for future innovations in the field of satellite imagery and urban development.

# References

[1] N. J. Yuan, Y. Zheng and X. Xie, SegmentationofUrbanAreasUsingRoadNetworks, Beijing,China, 2012.

[2] R. R. S. J. J. Daniel Joseph Cook, "Effect of Road Segmentation on Highway Safety Analysis," 2011.

[3] K. Z. S. J. Yao Wei, "Road Network Extraction from Satellite Images Using CNN Based Segmentation and Tracing," 2019.

[4] H.-P. C. J.-M. W. S.-W. C. Chiung-Yao Fang, "Automatic Road Segmentation of Traffic Images," 2015.

[5] Y. B. F. P. A. P. N. K. D. T. Shervin Minaee, "Image Segmentation Using Deep Learning: A Survey," 2021.

[6] H. B. W. M. S. K. A. A. F. a. A. J. Ghandorh, "Semantic Segmentation and Edge Detection—Approach to Road Detection in Very High Resolution Satellite Images," 2022.

[7] T.-W. K. S.-P. T. Y. C. S. C. J.-F. W. Y. G. a. T. C. Xiaodong Yu, "EnRDeA U-Net Deep Learning of Semantic Segmentation on Intricate Noise Roads," 2023.

[8] Y. T. Z. Y. Y. Z. a. T. Z. Xinyu Cao, "Semantic Segmentation Network for Unstructured Rural Roads Based on Improved SPPM and Fused Multiscale Features," 2024.

[9] G. S. T. K. B. K. ,. H. K. Jeongho Hyeon, "Challenges in Road Crack Segmentation Due to Coarse Annotation".

[10] Z. C. Z. S. H. G. B. L. Z. Y. Y. W. Z. H. X. L. a. J. Y. Jingjing Tao, "Seg-Road: A Segmentation Network for Road Extraction Based on Transformer and CNN with Connectivity Structures," 2023.

[11] C. M. ,. G. Y. ,. Y. S. ,. S. L. ,. J. F. Shun Xiong, "Semantic segmentation of remote sensing imagery for road extraction via joint angle prediction: comparisons to deep learning," 2023.

[12] W. G. D. Z. Cunge Guo, " Research on Road Surface Crack Detection Based on SegNet Network," *Journal of Engineering and Applied Science, Volume 71, Article 54,* 2024.

[13] H. M. C. d. A. João Batista Pacheco Junior, "Performance Analysis in the Segmentation of Urban Asphalted Roads in RGB Satellite Images Using K-Means++ and SegNet: Case Study in São Luís-MA," *Inteligencia Artificial Revista Iberoamericana de Inteligencia Artificial, Volume 24, Issue 68,* 2021.

[14] N. G. Y. K. Y. G. A. R. O. A. S. S. Vlad Taran, " Performance Evaluation of Deep Learning Networks for Semantic Segmentation of Traffic Stereo-Pair Images," 2017.

[15] J. S. X. Q. X. W. J. J. Hengshuang Zhao, "Pyramid Scene Parsing Network," *The Chinese University of Hong Kong, SenseTime Group Limited,* 2017.

[16] Y. L. M. J. Z. C. J. Z. Jiangwei Ge, "PFANet: A Network Improved by PSPNet for Semantic Segmentation of Street Scenes," 2023.

[17] S. M. A. N. M. Corentin Henry, "Road Segmentation in SAR Satellite Images With Deep Fully Convolutional Neural Networks," 2018.

[18] M. C. Z. X. Z. Z. Rui Wang, "Remote Sensing Image Road Segmentation Method Integrating CNN-Transformer and UNet," 2023.

[19] J.-A. Sarmiento, "Pavement Distress Detection and Segmentation using YOLOv4 and DeepLabv3 on Pavements in the Philippines," 2021.

[20] S.-Y. L. H.-M. H. S.-W. C. J.-J. L. Ping-Rong Chen, "Efficient Road Lane Marking Detection with Deep Learning," 2018.

[21] https://www.aicrowd.com/challenges/epfl-ml-road-segmentation.

[22] M. S. I. ,. K. ,. Z. S. Ozan Ozturk, "Improving Road Segmentation by Combining Satellite Images," p. 9, 2023.

[23] B. P. G. S. K. N. A. M. a. A. A. Abolfazl Abdollahi, " Improving Road Semantic Segmentation Using GAN," 2021.

[24] D. C. &. G. Jurman, "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation," 2020.

[25] D. V. K. K.M. Sharavana Raju, "Improved Satellite Image Preprocessing and Segmentation Using Wavelets and Enhanced Watershed Algorithms," 2012.

[26] C. R. R. D. J. M. L. F. P. N. Isaac Corley, "Revisiting Pre-trained Remote Sensing Model Benchmarks: Resizing and Normalization Matters," 2023.

[27] **P. M. Hossein Talebi, "Learning to Resize Images for Computer Vision Tasks," 2021.**

[28] **M. G. L. H. H. S. L. Marcelo Romero Aquino, "The Effect of Data Augmentation on the Performance of Convolutional Neural Networks," 2017.**

[29] **https://scikit-image.org/docs/stable/auto_examples/transform/plot_geometric.html.**

[30] **T. M. K. Connor Shorten, "A Survey on Image Data Augmentation for Deep Learning," 2019.**

[31] **https://docs.ultralytics.com/integrations/albumentations/.**

[32] **O. K. A. J. L. H. A. Himanshu Gupta, "Robust Object Detection in Challenging Weather Conditions".**

[33] **M. M. A. K. H. A. Abdulghani M. Abdulghani, "Data Augmentation with Noise and Blur to Enhance the Performance of YOLO7 Object Detection Algorithm," 2023.**

[34] **L. Z. G. K. S. L. Y. Y. Zhun Zhong, "Random Erasing Data Augmentation," 2017.**

[35] **V. I. I. E. K. A. P. M. D. A. A. K. Alexander Buslaev, "Albumentations: Fast and Flexible Image Augmentations," 2020.**

[36] **P. F. T. B. Olaf Ronneberger, "U-Net: Convolutional Networks for Biomedical Image Segmentation," 2015.**

[37] **F. F. G. D. C. C. H. A. D. S. S. Christopher Williams, "A Unified Framework for U-Net Design and Analysis," 2023.**

[38] **A. Z. Karen Simonyan, "Very Deep Convolutional Networks for Large-Scale Image Recognition," 2014.**

[39] **H. K. Hossein Gholamalinezhad, "Pooling Methods in Deep Neural Networks, a Review," 2020.**

[40] **G. P. F. S. H. A. Liang-Chieh Chen, "Rethinking Atrous Convolution for Semantic Image Segmentation," 2017.**

[41] **Y. Z. G. P. F. S. H. A. Liang-Chieh Chen, "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation," 2018.**

[42] **J. B. Diederik P. Kingma, "Adam: A Method for Stochastic Optimization," 2015.**

[43] **F. H. Ilya Loshchilov, "SGDR: Stochastic Gradient Descent with Warm Restarts," 2017.**

[44] **X. Z. Z. H. F. Z. Shan Zhao, "BMSeNet: Multiscale Context Pyramid Pooling and Spatial Detail Enhancement Network for Real-Time Semantic Segmentation,"** ***School of Software, Henan Polytechnic University, Jiaozuo, China,*** **2024.**

[45] **R. F. iang Jie, "A Road Extraction Method Based on Improved PSPNet,"** ***Proceedings Volume 12710, International Conference on Remote Sensing, Surveying, and Mapping (RSSM 2023),*** **2023.**

[46] **M. B. Solà, "Deep Multimodal Learning for Egocentric Storytelling and Food Analysis," 2020.**

[47] **W. Boulila, "A top-down approach for semantic segmentation of big remote sensing images," 2019.**

[48] **Y. Y. Y. K. T. A. T. H. K. a. R. Z. Wenmiao Hu, "GeoPalette: Road Segmentation with Limited Satellite Imagery," 2021.**

[49] **S. S. V. I. A. S. Alexander Buslaev, "Fully Convolutional Network for Automatic Road Extraction from Satellite Imagery," 2018.**

[50] **Y. &. A. H. Nachmany, "Detecting Roads from Satellite Imagery in the Developing World," 2018.**

[51] **J. Dai, T. Zhu, Y. Zhang, R. Ma and W. Li, "Lane-Level Road Extraction from High-Resolution Optical Satellite Images," 2019.**

[52] **H. I. A. ,. X. C. Khaled Alomar, "Data Augmentation in Classification and Segmentation: A Survey and New Strategies," 2023.**

[53] **X. Y. ,. Y. ,. Q. ,. M. ,. K. Y. Dongping Zhang, "Data-Augmented Deep Learning Models for Abnormal Road Manhole Cover Detection," 2023.**

[54] **K. M. M. A. E. A. H. M. &. M. S. D. Retaj Yousri, "A novel data augmentation approach for ego-lane detection enhancement," 2024.**

TABLE OF FIGURES