Semantic Segmentation in Autonomous Systems

GitHub: https://github.com/hp2304/AI-Project

Team Members

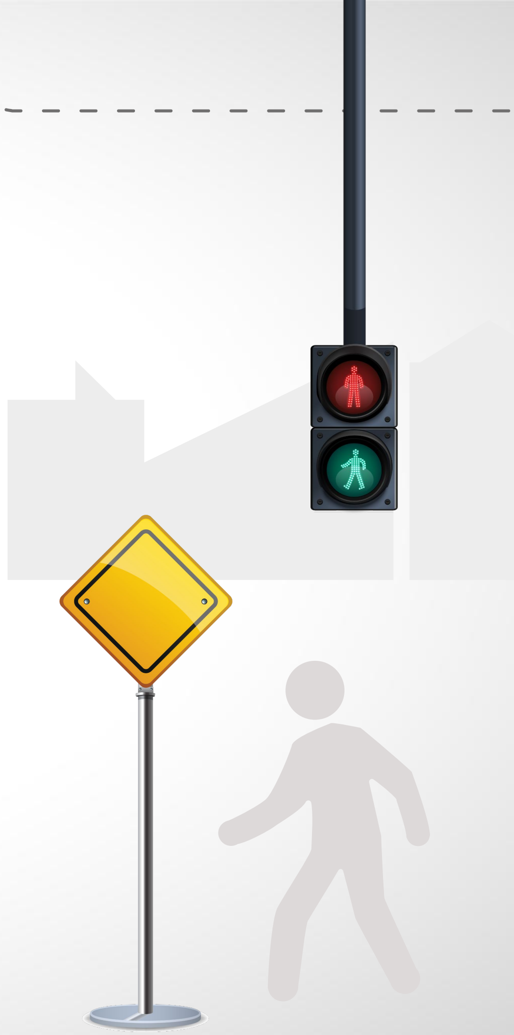
—Vaibhav Parikh (1217458)

—Hitarth Panchal (1214537)

—Saurav Patel (1220888)

—Jatin Patel (1229894)

Table of contents



01 Introduction

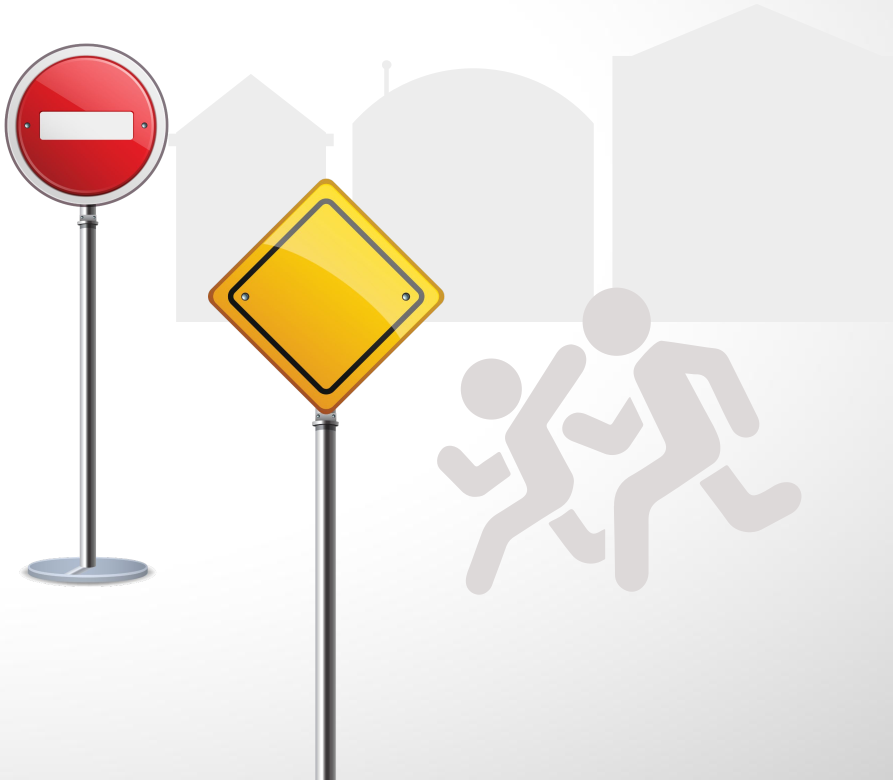
Introduction

Semantic segmentation is a critical task in computer vision, particularly for autonomous systems, as it involves classifying each pixel in an image into a predefined category.

This capability is essential for autonomous vehicles to understand and navigate their environment effectively.

By segmenting road scenes, autonomous systems can identify various objects, such as vehicles, pedestrians, and drivable regions, which is crucial for safe and efficient driving.

02 Cityscapes Dataset

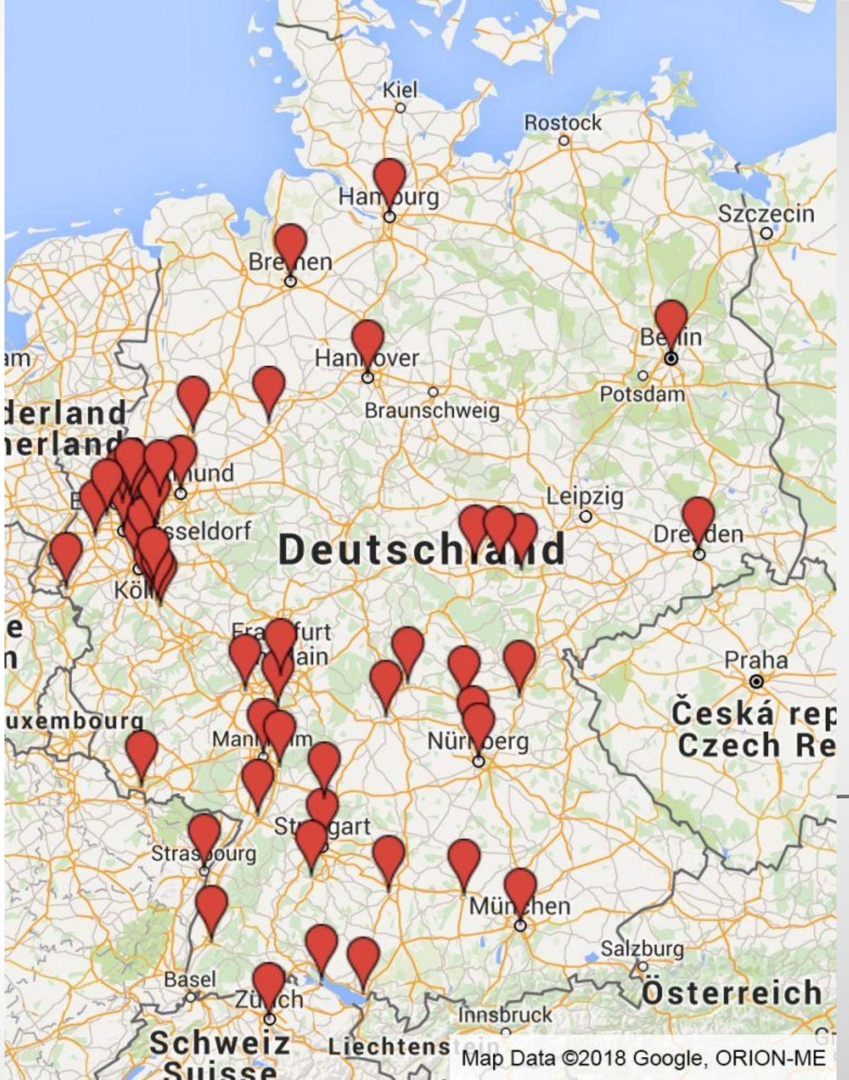


CityScapes Dataset

The CityScapes dataset was chosen for this project due to its comprehensive and high-quality pixel-level annotations of urban scenes.

- 50 cities of Germany
- Different Seasons
- Large number of dynamic objects
- Varying scene layout
- Varying background

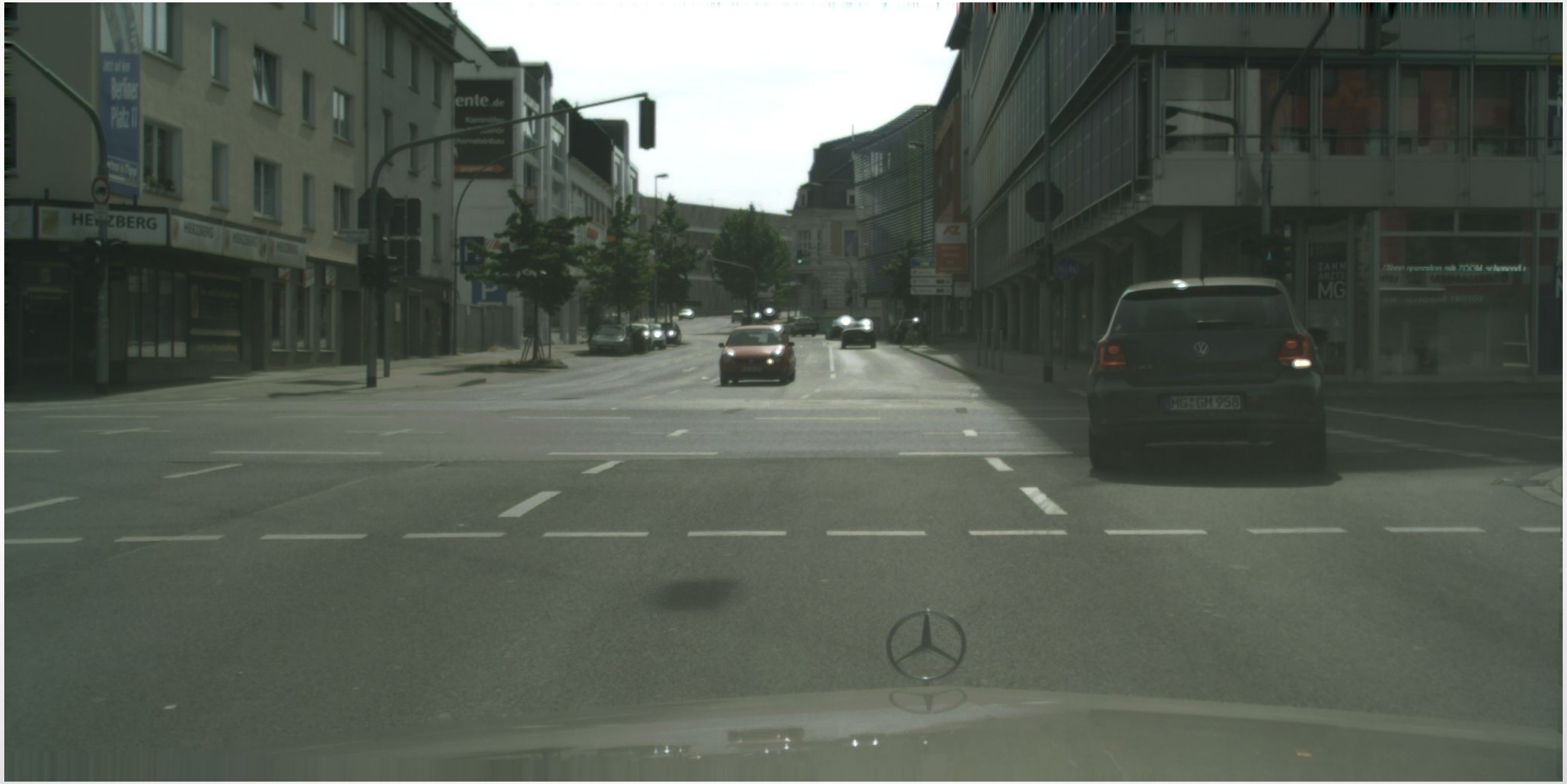
Training Images - 2975
Test Images - 500



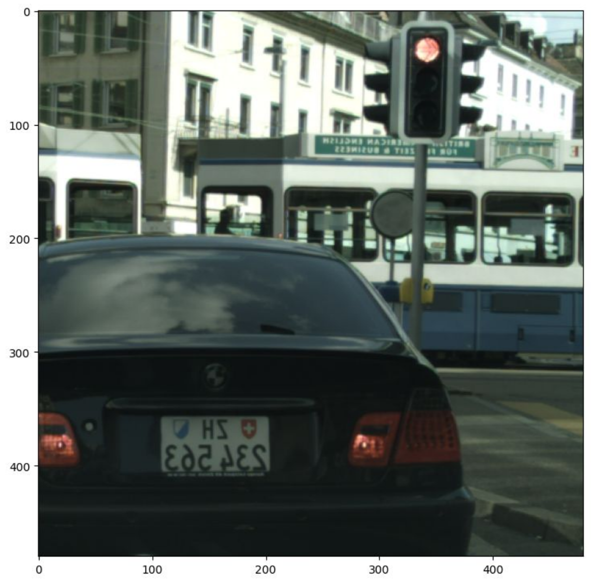
CityScapes Dataset

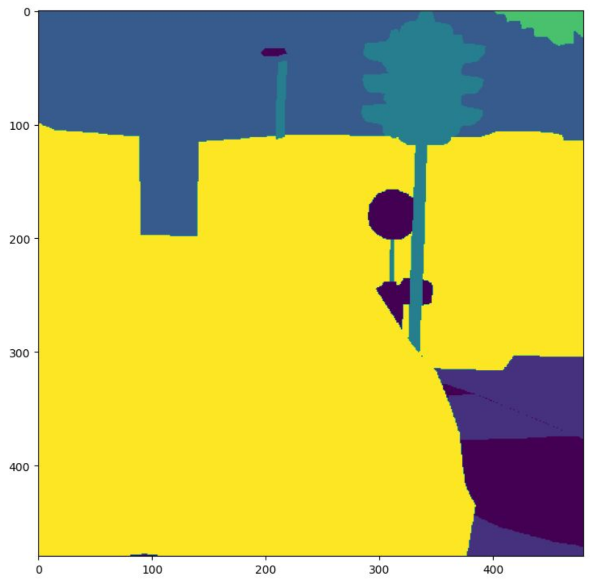
THE CAMBR UTE
BRITISH AMERICAN ENGLISH FÜR FREIZEIT & BUSINESS
chilloutparadiesTRAUMferienmacher
ZH 234 563

CityScapes Dataset

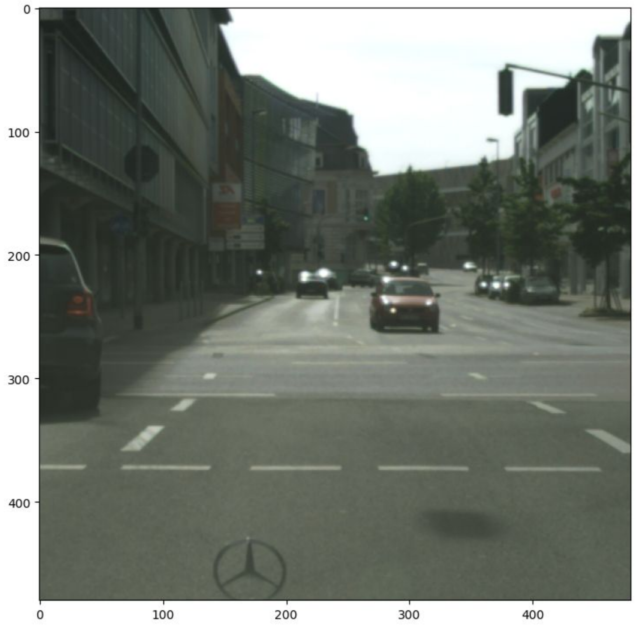


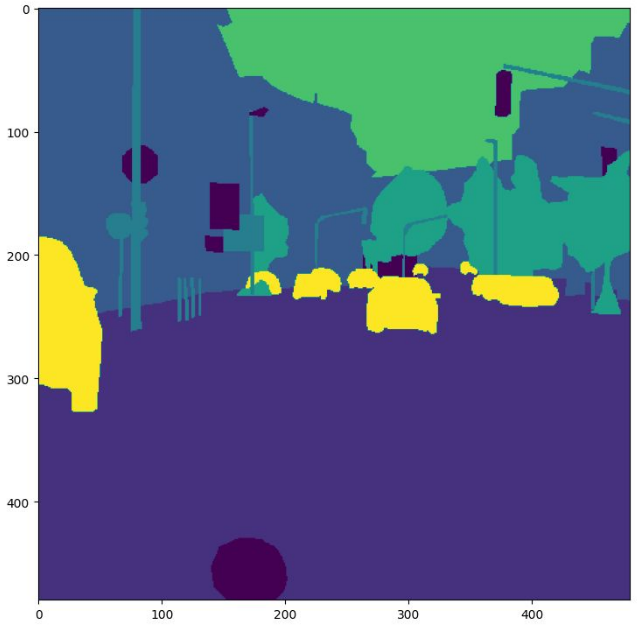
CityScapes Dataset

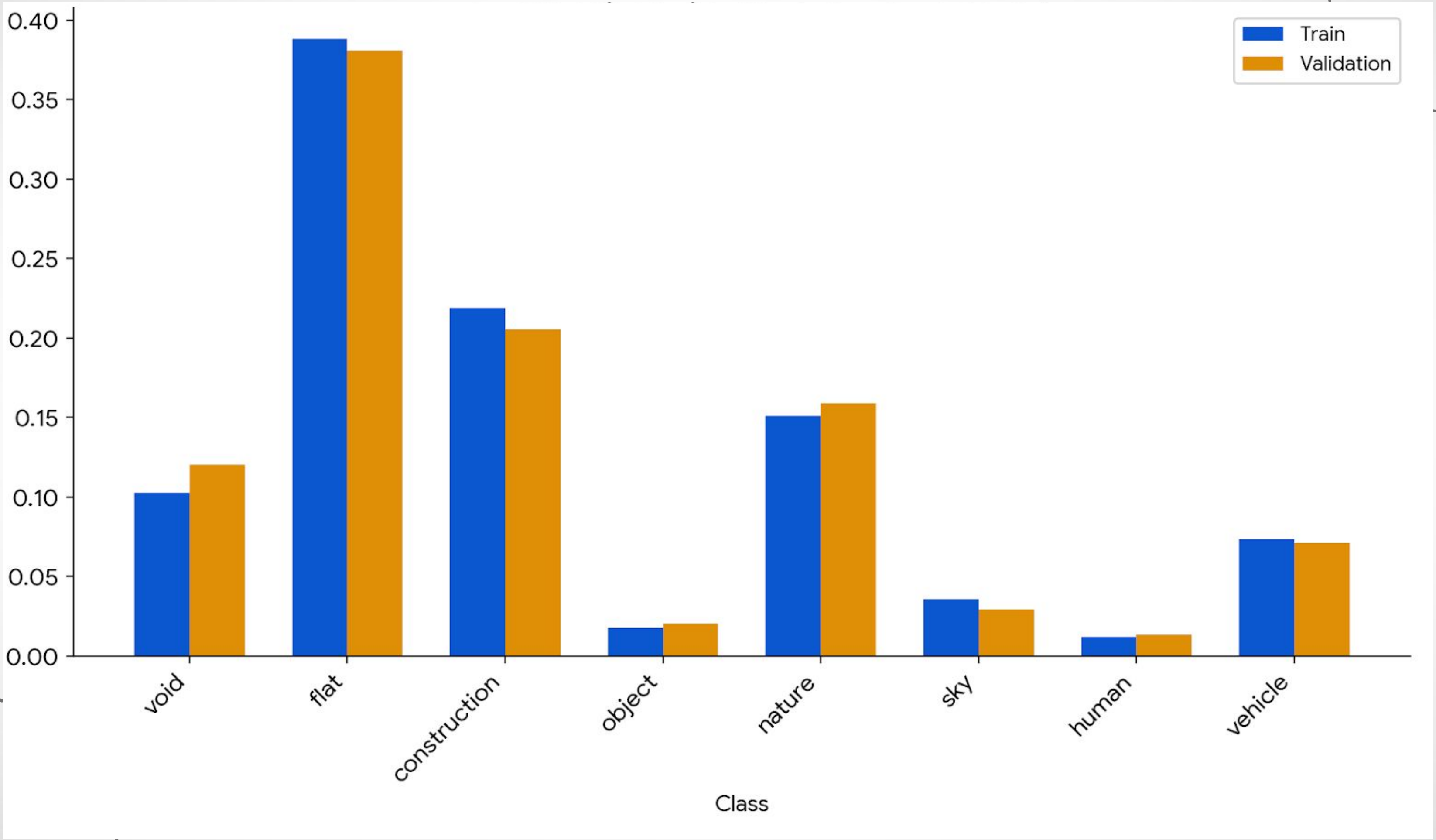




CityScapes Dataset







Train
Validation
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
void
flat
construction
object
nature
sky
human
vehicle
Class

Training Data Pipeline

Training Data Pipeline

- A crucial component in model development.
- Ensures data is in the correct format and augmented for optimal performance.
- Key stages: data loading, preprocessing, augmentation, and batching.

Data Loading:

- Dataset organization: CityScapes into train and test sets.
- Efficient data loading with batching for memory management.

Data Preprocessing and Augmentation

Data Preprocessing and Augmentation

- **Data Augmentation:**
  - Random cropping for consistent input size and variability.
  - Horizontal flipping for robustness.
- **Normalization:** Pixel normalization based on ImageNet statistics for stability.

Batch Processing:

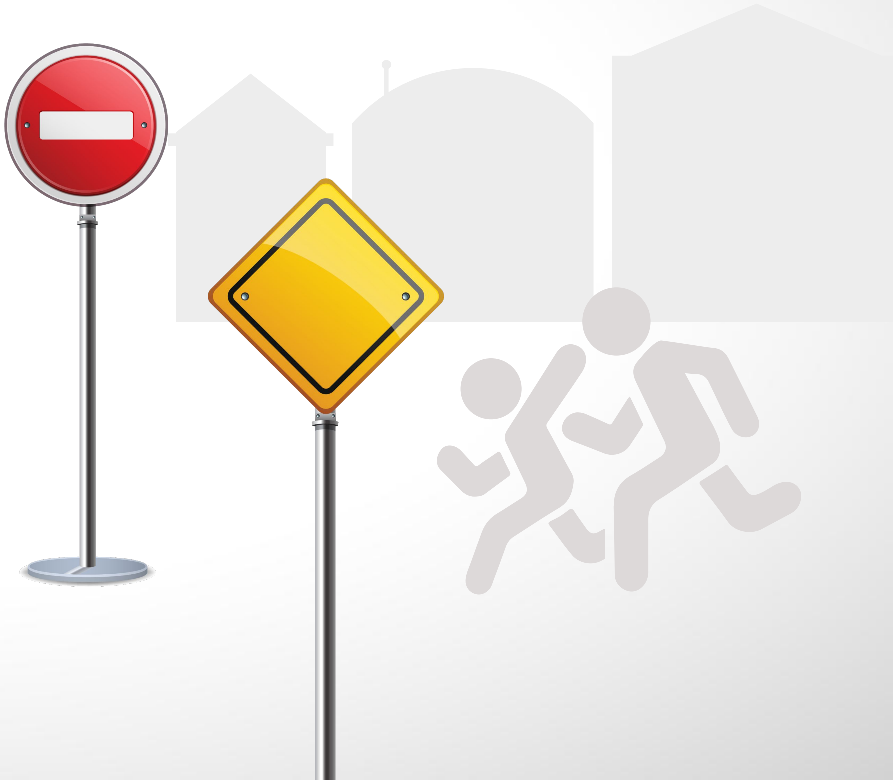
- Batch size of 48 for balance between memory and efficiency.
- Data shuffling for diverse training examples.

Workflow:

1. Load images and annotations.
2. Apply augmentations.
3. Normalize pixel values.
4. Form batches and shuffle.

03

FCN Architecture



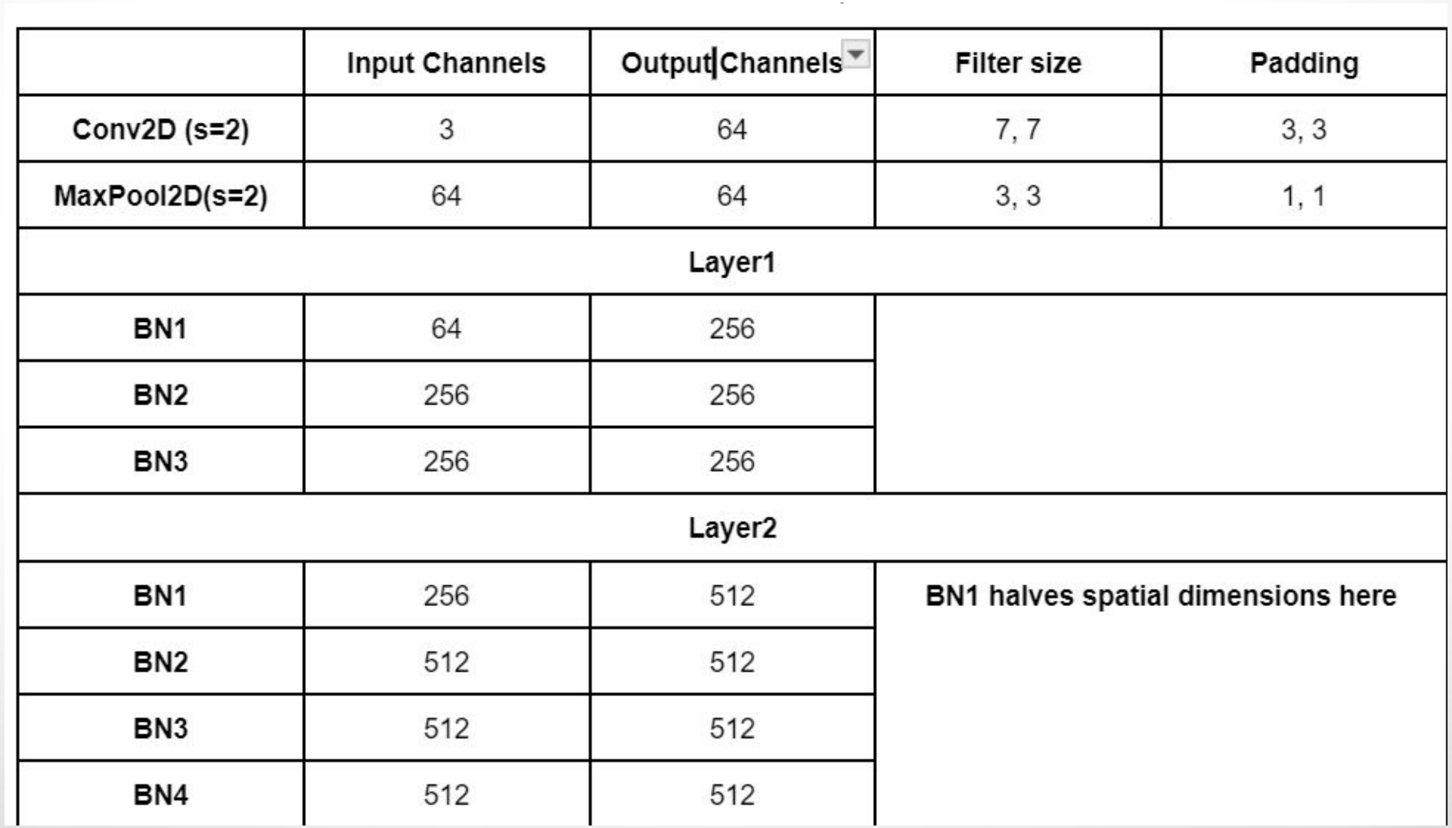
FCN Architecture

Fully Convolutional Networks (FCNs) are a powerful architecture for semantic segmentation tasks. They excel in end-to-end learning and producing dense predictions for pixel-wise classification. A ResNet50 backbone is used for feature extraction.

Key components:

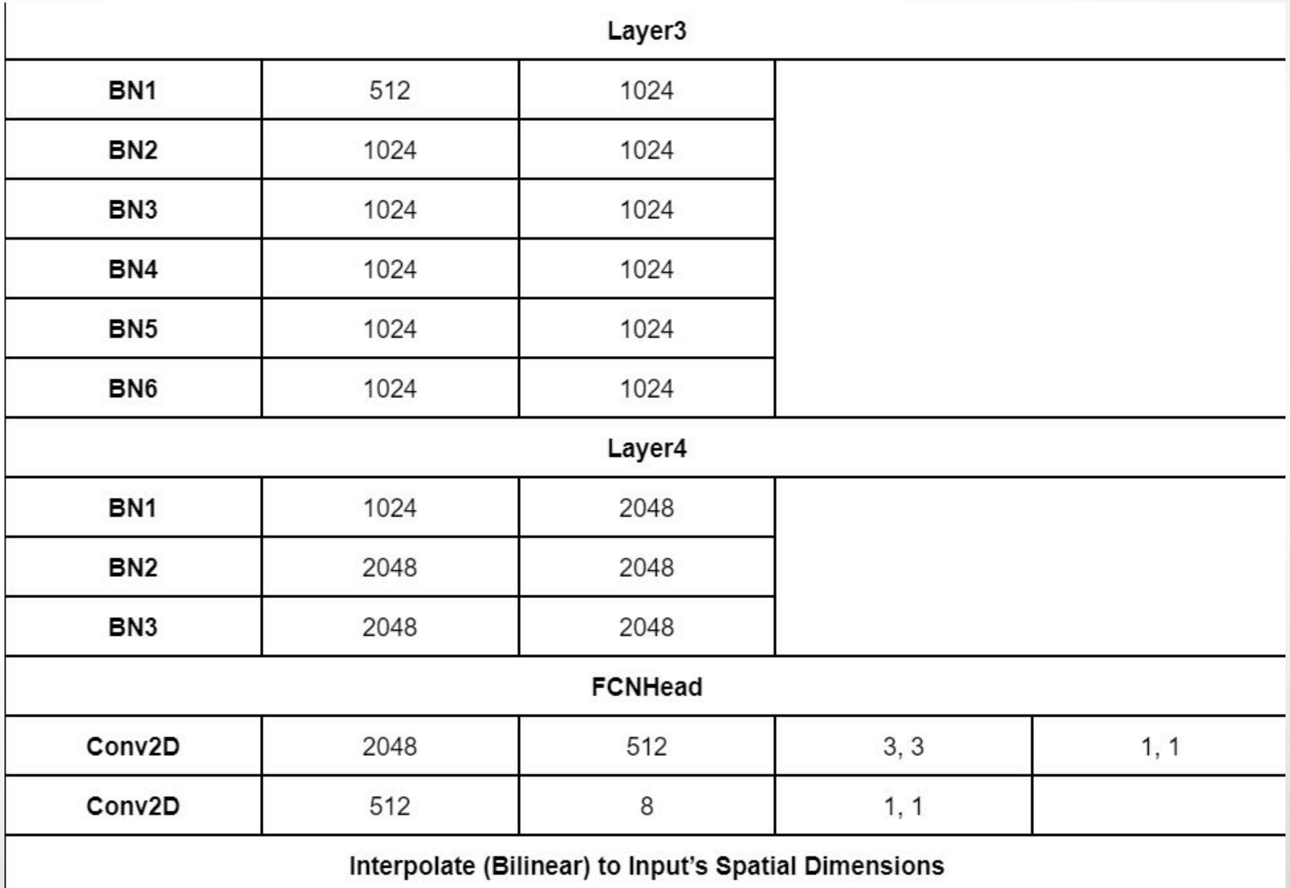
Conversion to Fully Convolutional Layers: Enables pixel-wise prediction by converting fully connected layers to convolutional layers.
Upsampling Layers: Restore spatial dimensions using transposed convolutions (deconvolutions).
Output Layer: Softmax activation produces probability distribution for each pixel class.

FCN Architecture



	Input Channels	Output Channels	Filter size	Padding
Conv2D (s=2)	3	64	7, 7	3, 3
MaxPool2D(s=2)	64	64	3, 3	1, 1
Layer1				
BN1	64	256		
BN2	256	256		
BN3	256	256		
Layer2				
BN1	256	512	BN1 halves spatial dimensions here	
BN2	512	512		
BN3	512	512		
BN4	512	512		

FCN Architecture



		Layer3		
BN1	512	1024		
BN2	1024	1024		
BN3	1024	1024		
BN4	1024	1024		
BN5	1024	1024		
BN6	1024	1024		
		Layer4		
BN1	1024	2048		
BN2	2048	2048		
BN3	2048	2048		
		FCNHead		
Conv2D	2048	512	3, 3	1, 1
Conv2D	512	8	1, 1	
		Interpolate (Bilinear) to Input's Spatial Dimensions		

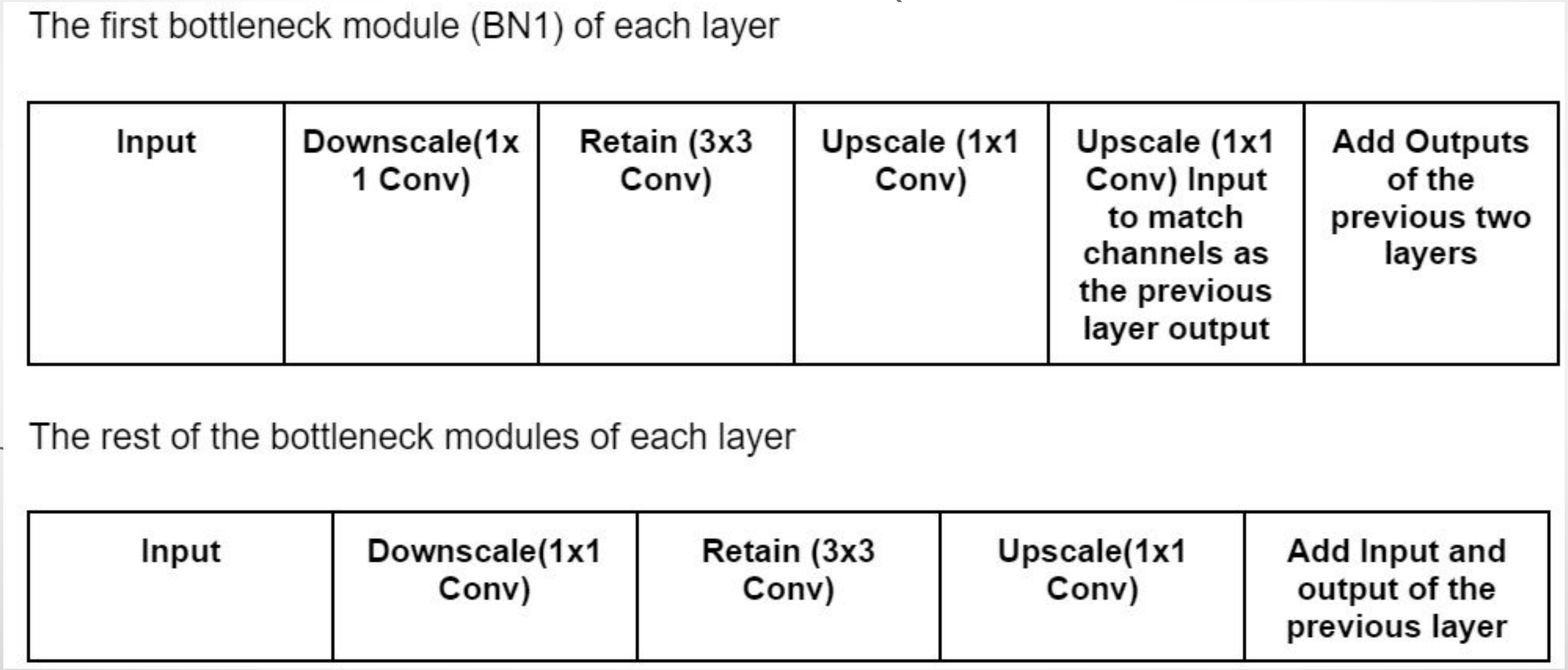
FCN Architecture

Detail of Bottleneck Module:

Input: Original feature map.
Downscale (1x1 Conv): Reduce feature map channels.
Retain (3x3 Conv): Extract spatial features.
Upscale (1x1 Conv): Increase feature map channels to match previous layer's output.

The module's output is either added to the input or the previous bottleneck module's output.

FCN Architecture



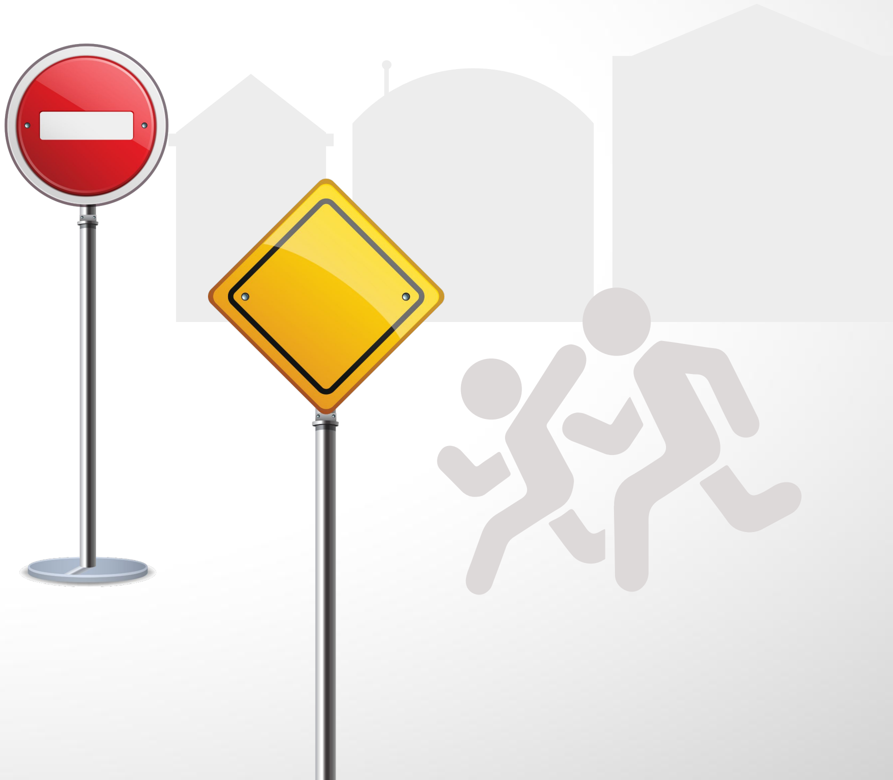
The first bottleneck module (BN1) of each layer

Input	Downscale(1x 1 Conv)	Retain (3x3 Conv)	Upscale (1x1 Conv)	Upscale (1x1 Conv) Input to match channels as the previous layer output	Add Outputs of the previous two layers

The rest of the bottleneck modules of each layer

Input	Downscale(1x1 Conv)	Retain (3x3 Conv)	Upscale(1x1 Conv)	Add Input and output of the previous layer

04 Training Process



Training Process

Model Preparation: Assemble the FCN architecture with ResNet50 backbone.

Loss Function: Cross-entropy loss for multi-class pixel-wise classification.

Optimizer: SGD with weight decay and momentum for efficient optimization.

Learning Rate Scheduler: Polynomial learning rate decay for stable convergence.

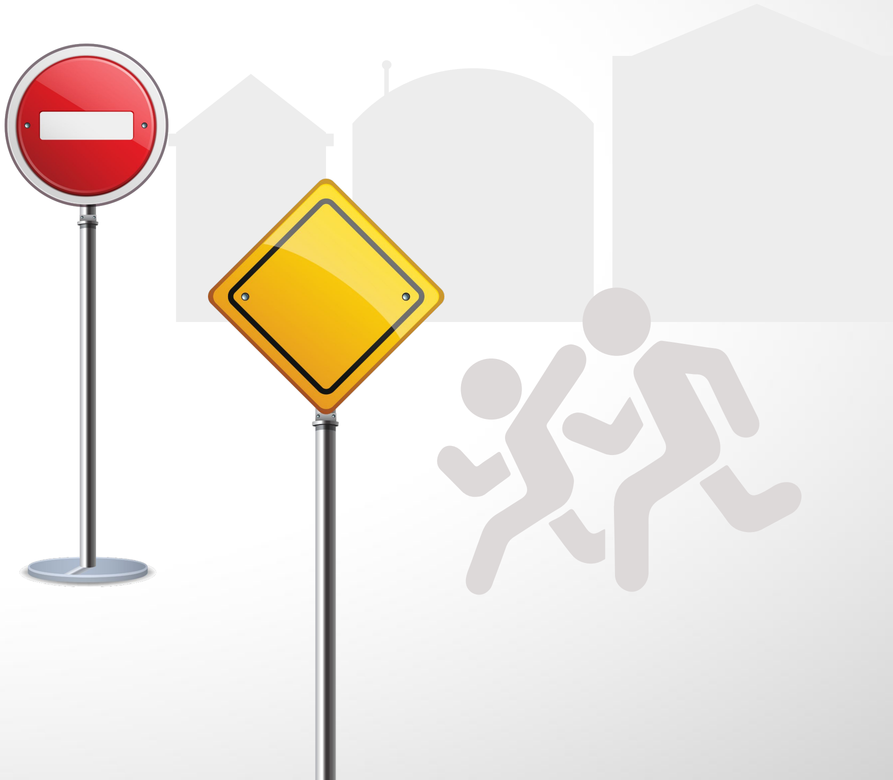
Training Parameters: 10 epochs, batch size of 48, GPU acceleration.

Training Workflow

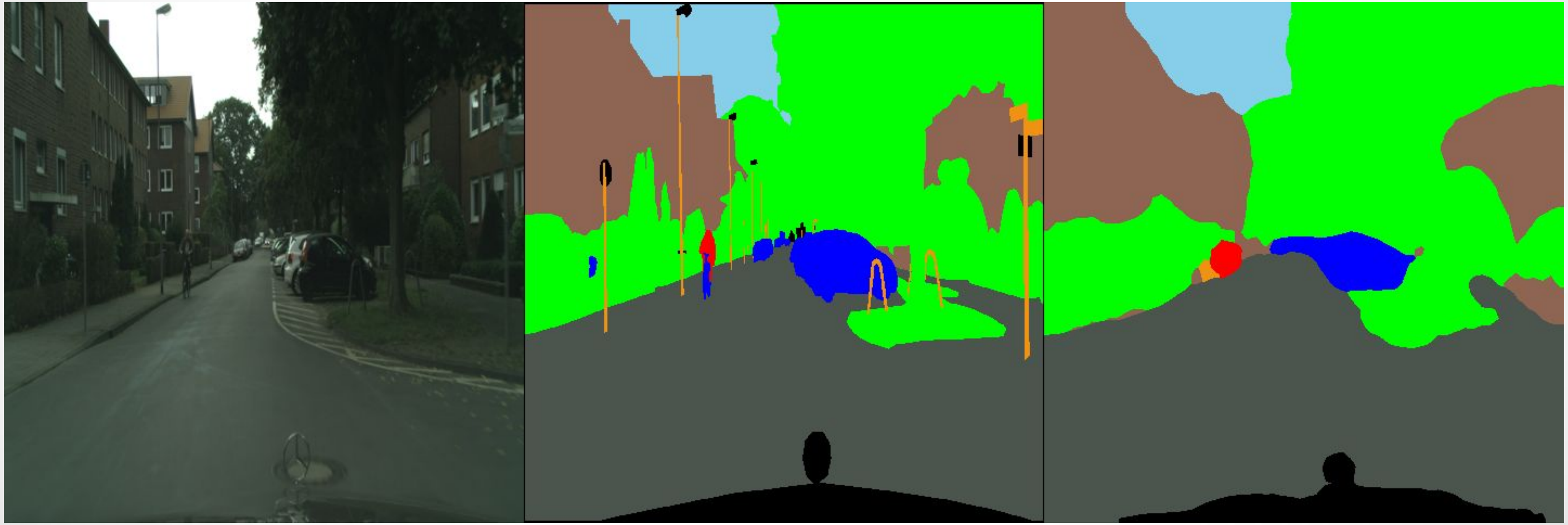
Initialization: Set up model, loss, optimizer, and learning rate scheduler.
Epoch Loop:

- Divide dataset into batches of 48 images.
- Forward pass: Compute predicted segmentation maps.
- Loss calculation: Calculate cross-entropy loss.
- Backward pass: Compute gradients.
- Weight update: Update weights using SGD with momentum and weight decay.
- Learning rate adjustment: Apply polynomial decay.
- Evaluation: Assess model performance on validation set.

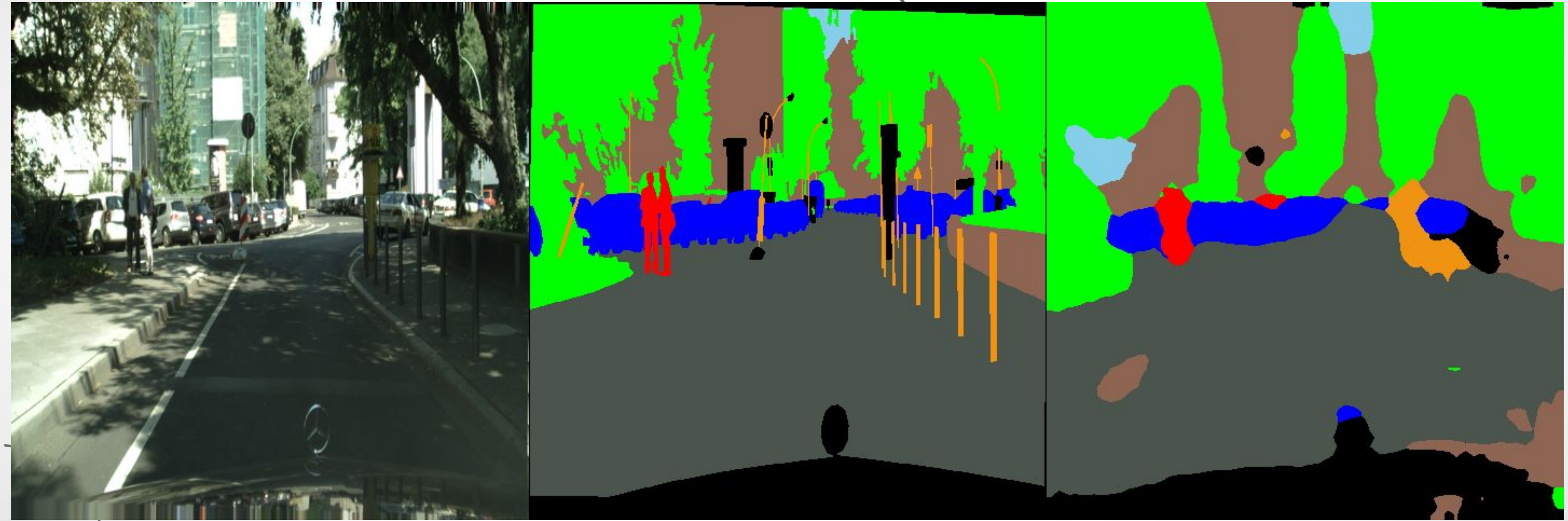
05
Result



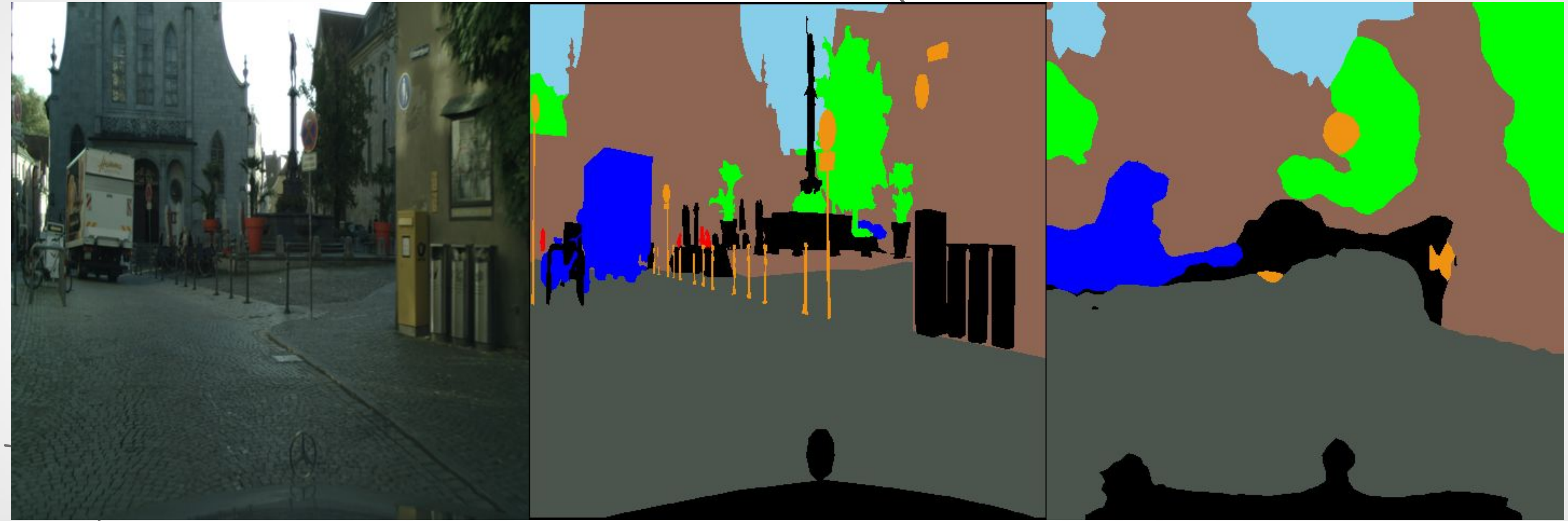
Result



Result



Result



Result

Pixel Accuracy

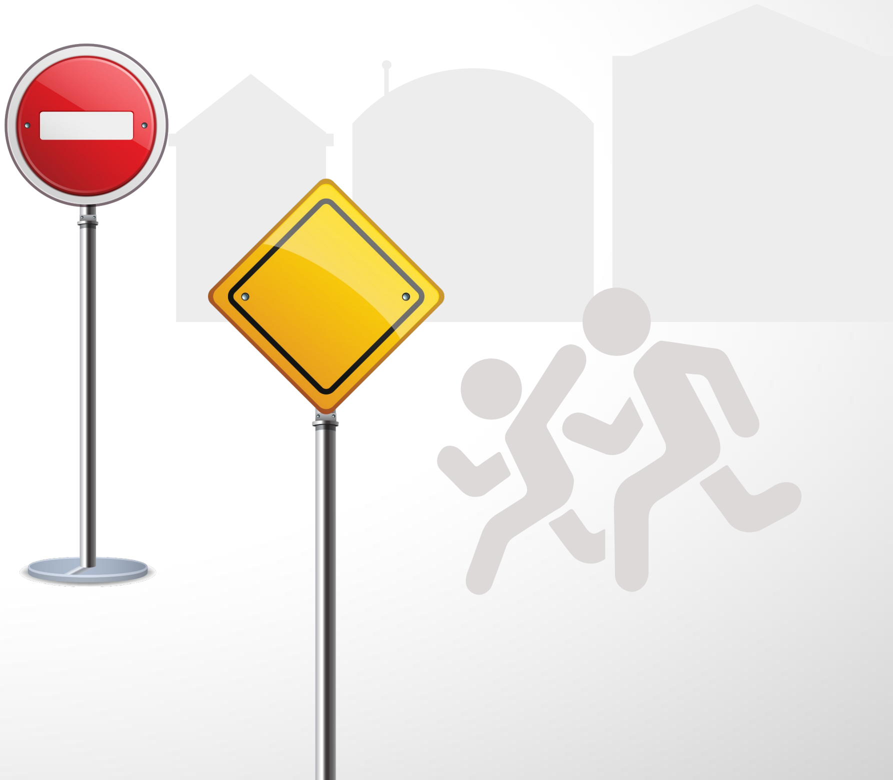
- **Definition:** Ratio of correctly classified pixels to total pixels.
  - Formula: Pixel Accuracy = Number of Correctly Classified Pixels / Total Number of Pixels
- **Purpose:** Overall accuracy measure.

Results on Cityscapes Validation Set

- **Pixel Accuracy:** 85%
  - The model correctly classified 85% of pixels.

06

Conclusion



Conclusion

The project demonstrated the effectiveness of using a Fully Convolutional Network with a ResNet50 backbone for semantic segmentation of the road environments.

With an 85% pixel accuracy, the model shows promise for the real-world applications.

However, there are several avenues for of the improvement, particularly in the terms of validation, handling class imbalance, and exploring more of the advanced model architectures.

Future work focusing on these areas could lead to even more accurate and robust segmentation models, contributing significantly to the development of autonomous systems.

Thank You