National Research Higher School of Economics (HSE)

SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation

A Summary and Analysis of the Paper by Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla

Presentation by: **John Eshun**

Table of Content

1. INTRODUCTION

Introduction

- The paper presents a comprehensive study of SegNet, a novel deep fully convolutional neural network architecture developed for semantic pixel-wise image segmentation.

- SegNet utilises an encoder-decoder structure where the decoder efficiently upsamples feature maps using pooling indices transferred from the corresponding encoder, which is topologically identical to the VGG16 network's convolutional layers.

- The authors compare SegNet, motivated primarily by scene understanding applications like autonomous driving, with competing architectures such as FCN and DeepLab-LargeFOV, highlighting its efficiency in terms of memory and computational time during inference.

- The paper conducts a controlled benchmark analysis on both road and indoor scene datasets, exploring the trade-offs between memory, accuracy, and performance across various decoder designs.

2. The Challenge of Semantic Segmentation

The Challenge of Semantic Segmentation

- Semantic pixel-wise segmentation is the task of assigning a class label to every pixel in an image.

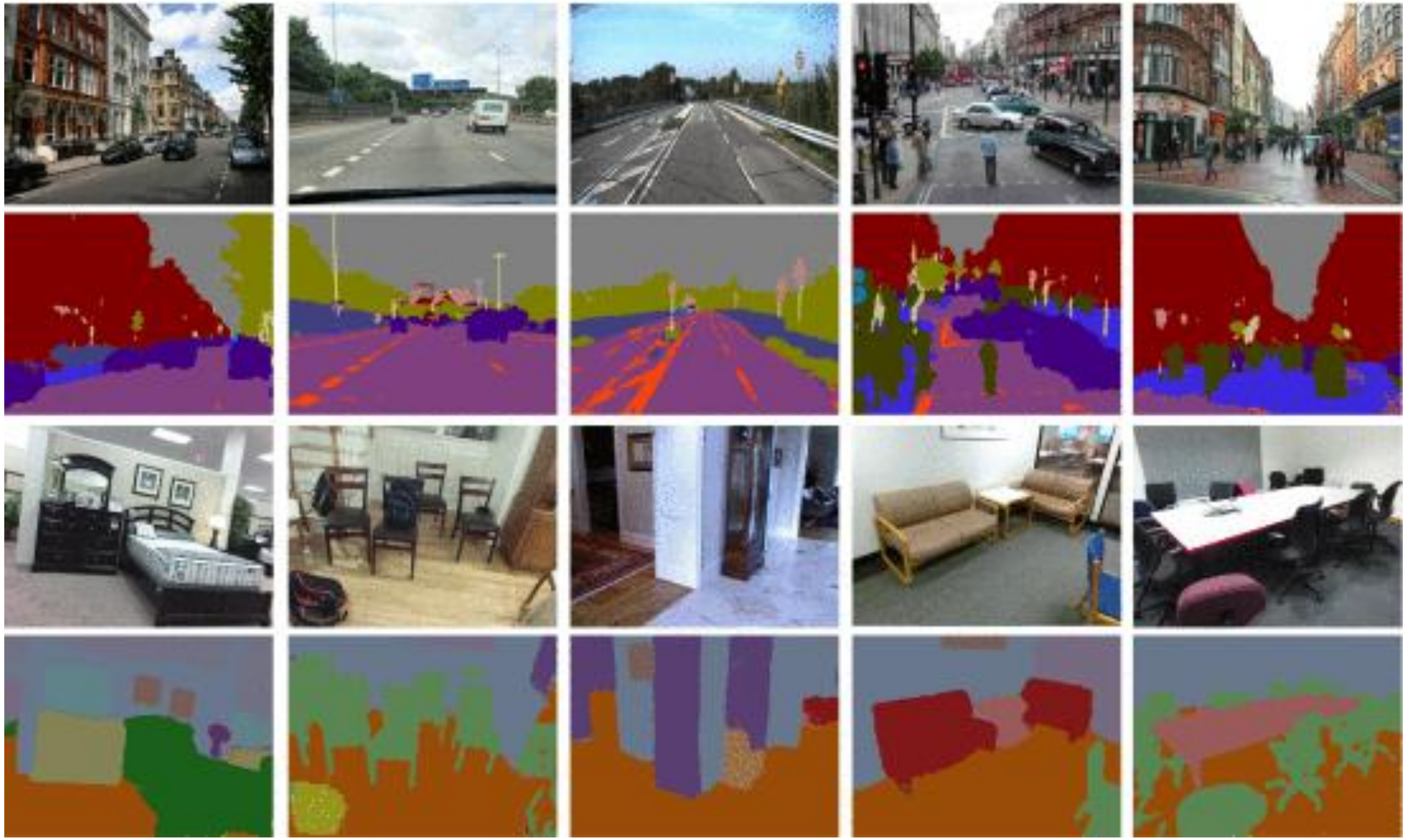
- Unlike object detection, which places bounding boxes around objects, segmentation creates a dense, detailed map that outlines the precise shape and location of each object class.

- This level of detail is crucial for a wide array of applications, including **autonomous driving, general scene understanding, and inferring support-relationships among objects in a scene**.

The Core Challenge:

- ➢ Standard classification networks use **max-pooling** and **sub-sampling**, which reduce feature map resolution.
- ➢ This process *loses* fine-grained spatial detail and boundary information, which is vital for accurate segmentation.

The Challenge of Semantic Segmentation

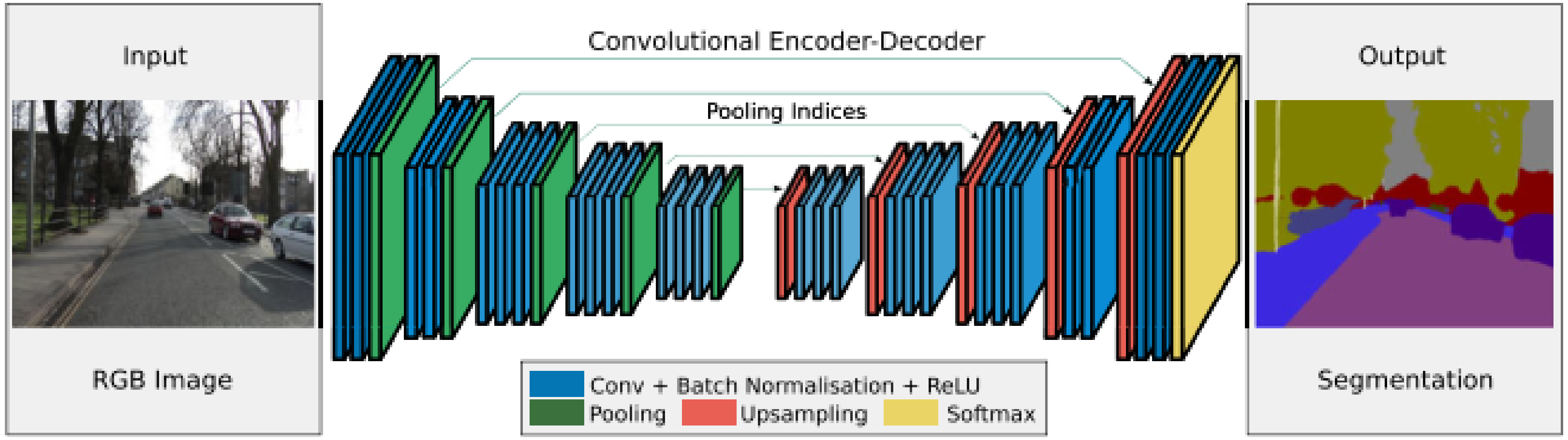


This figure perfectly illustrates the "problem" and the "goal." It shows five different input images (road and indoor scenes) and their corresponding "ground truth" segmentation maps. This shows what "pixel-wise segmentation" means in practice.

The Solution: SegNet

- **SegNet** is a deep, fully convolutional neural network designed to address the challenge of information loss, providing an efficient and accurate solution for semantic segmentation.

- Its architecture is composed of three main components:
  - An encoder network that captures high-level semantic information.
  - A corresponding decoder network that maps the low-resolution feature maps back to the original image resolution.
  - A final pixel-wise classification layer that produces the final segmentation map.

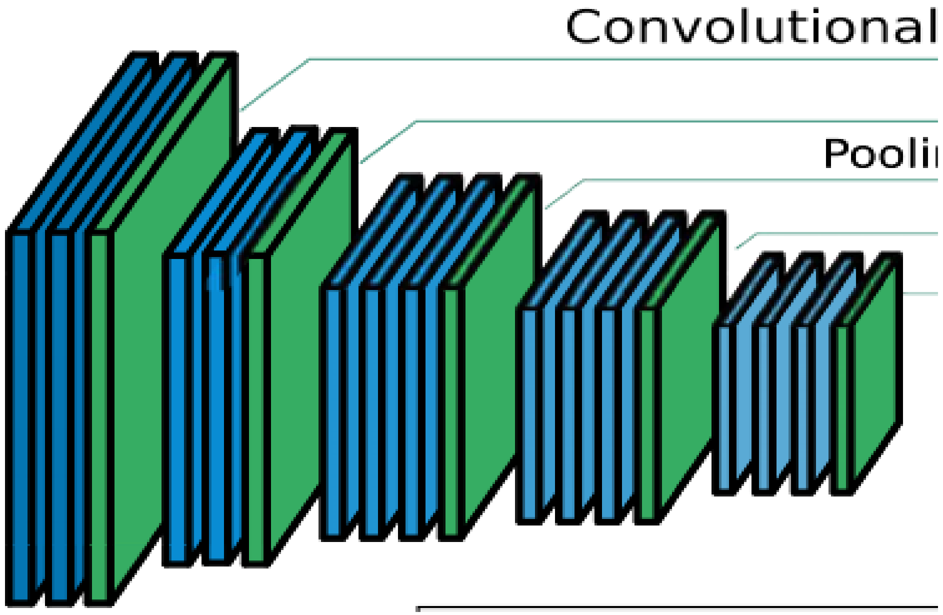


*This is the **main architecture diagram** for the entire paper. It visually explains the full end-to-end structure, showing the encoder layers, the decoder layers, and (most importantly) the "Pooling Indices" being transferred from the encoder to the decoder.*

3. SegNet's Core Idea and Architecture

Architecture: The Encoder Network

- The **SegNet** encoder is responsible for extracting a hierarchy of features from the input image.
- Its architecture is identical to the first 13 convolutional layers of the renowned **VGG16 network**.
- It reduces the number of trainable parameters from 134 million to just 14.7 million. This directly serves the primary design goal of efficiency by reducing parameter count and memory footprint.
- It retains higher-resolution feature maps at the deepest encoder output, preserving more spatial detail.

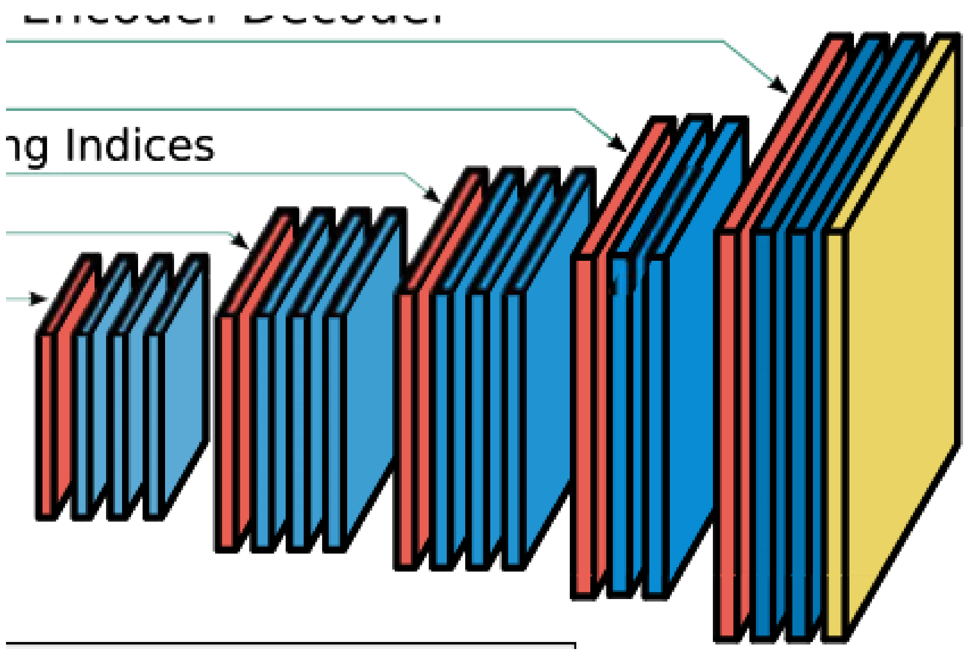


How it Works (for each block):

- Performs convolution, batch normalization, and ReLU activation
- Performs 2×2 max-pooling to downsample the feature map
- **CRUCIAL STEP:** It **memorizes and stores the *locations* (indices)** of the maximum value chosen in each 2×2 pooling window

Architecture: The Decoder Network

- The primary role of the decoder network is to **upsample** the low-resolution feature maps from the **encoder** back to the full input resolution, enabling dense, pixel-wise classification.
- The decoder mirrors the encoder, consisting of a corresponding hierarchy of 13 layers.



How it Works (The "SegNet Way"):

- The decoder receives an input feature map to be upsampled.
- It uses the memorized max-pooling indices from the corresponding encoder layer to perform non-linear upsampling.
- This upsampling places the feature map's values into a larger, sparse feature map at the exact locations the max values came from (all other pixels are zero).
- This sparse map is then convolved with a trainable decoder filter bank to produce a dense feature map.
- This process is repeated through all 13 decoder layers.

4. The Key Innovation: Upsampling with Pooling Indices

Advantages of the features of SegNet

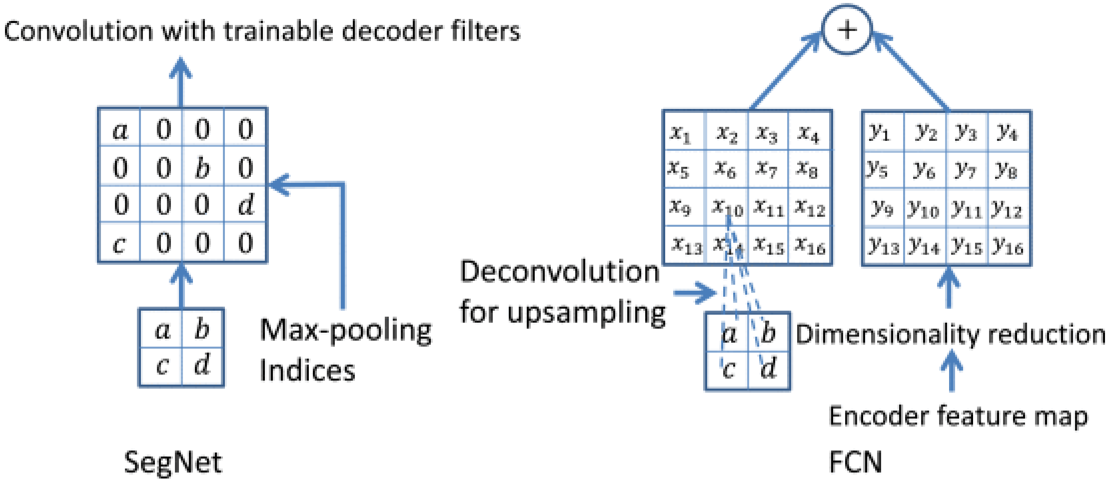
- **SegNet's** method of upsampling is the most novel contribution in the paper.
- It provides several practical advantages over learning-based upsampling techniques.

Feature	Advantage
Reusing max-pooling indices for non-linear upsampling.	Eliminates the need for the network to learn how to upsample.
Storing indices instead of full feature maps.	Drastically reduces memory requirements during inference.
Retaining max value locations.	Improves boundary delineation and localization.
Parameter Reduction.	Enables efficient, end-to-end training of the entire network.

Architectural Comparison: SegNet vs. FCN Decoders

The decoding strategy is the primary point of difference between **SegNet** and the popular Fully Convolutional Network (FCN) architecture.

SegNet Decoder	FCN Decoder
Upsamples using stored max-pooling indices (no learning required).	Learns to upsample using a trainable deconvolution layer.
Convolves the sparse, upsampled map with a trainable filter bank to create dense features.	Adds the corresponding feature map from the encoder to the upsampled output.
Stores only indices, making it highly memory-efficient.	Must store the full encoder feature maps, making it memory-intensive.



This fundamental architectural difference leads to a clear and important trade-off between model accuracy and computational efficiency the ***memory-accuracy trade-off*** *as we will see in the results.*

5. Performance Benchmarking and Results

Datasets & Competitors

The performance of **SegNet** was evaluated in a controlled benchmark using the following setup:

- Datasets:
  - **CamVid**: A road scene dataset with 11 classes, including cars, pedestrians, roads, and buildings.
  - **SUN RGB-D:** A highly challenging indoor scene dataset with 37 classes.
- Competitors:
  - FCN (Fully Convolutional Network)
  - DeepLab-LargeFOV
  - DeconvNet

Benchmark 1: Road Scene Segmentation (CamVid)

On the **CamVid** road scene dataset, SegNet demonstrated :

- **Superior Performance:** In a controlled benchmark, SegNet achieved competitive or superior results compared to well-known architectures like FCN, DeepLab-LargeFOV, and DeconvNet.
- **Boundary Delineation:** Qualitative results confirm SegNet's ability to accurately segment smaller, thinner classes (e.g., pedestrians, poles) and produce sharp, well-defined boundaries.
- **Efficiency**: Compared to larger networks like DeconvNet, SegNet learns to perform better in a shorter amount of training time.
- **Large Datasets**: When trained on a larger dataset of 3,433 images, SegNet's performance improved significantly, achieving a 60.10 mIoU score.

TABLE 2
Quantitative Comparisons of SegNet with Traditional Methods on the CamVid 11 Road Class Segmentation Problem [22]

Method	Building	Tree	Sky	Car	Sign-Symbol	Road	Pedestrian	Fence	Column-Pole	Side-walk	Bicyclist	Class avg.	Global avg.	mIoU	BF
SfM+Appearance [28]	46.2	61.9	89.7	68.6	42.9	89.5	53.6	46.6	0.7	60.5	22.5	53.0	69.1	n/a*	
Boosting [29]	61.9	67.3	91.1	71.1	58.5	92.9	49.5	37.6	25.8	77.8	24.7	59.8	76.4	n/a*	
Dense Depth Maps [32]	85.3	57.3	95.4	69.2	46.5	**98.5**	23.8	44.3	22.0	38.1	28.7	55.4	82.1	n/a*	
Structured Random Forests [31]	n/a											51.4	72.5	n/a*	
Neural Decision Forests [64]	n/a											56.1	82.1	n/a*	
Local Label Descriptors [65]	80.7	61.5	88.8	16.4	n/a	98.0	1.09	0.05	4.13	12.4	0.07	36.3	73.6	n/a*	
Super Parsing [33]	87.0	67.1	96.9	62.7	30.1	95.9	14.7	17.9	1.7	70.0	19.4	51.2	83.3	n/a*	
SegNet (3.5K dataset training - 140K)	**89.6**	**83.4**	96.1	**87.7**	52.7	96.4	**62.2**	**53.45**	**32.1**	**93.3**	**36.5**	**71.20**	**90.40**	60.10	46.84
CRF based approaches															
Boosting + pairwise CRF [29]	70.7	70.8	94.7	74.4	55.9	94.1	45.7	37.2	13.0	79.3	23.1	59.9	79.8	n/a*	
Boosting+Higher order [29]	84.5	72.6	**97.5**	72.7	34.1	95.3	34.2	45.7	8.1	77.6	28.5	59.2	83.8	n/a*	
Boosting+Detectors+CRF [30]	81.5	76.6	96.2	78.7	40.2	93.9	43.0	47.6	14.3	81.5	33.9	62.5	83.8	n/a*	

TABLE 3
Quantitative Comparison of Deep Networks for Semantic Segmentation on the CamVid Test Set When Trained on a Corpus of 3,433 Road Scenes *Without Class Balancing*

Network/Iterations	40 K				80 K				> 80 K				Max iter
	G	C	mIoU	BF	G	C	mIoU	BF	G	C	mIoU	BF	
SegNet	88.81	59.93	50.02	35.78	89.68	69.82	57.18	42.08	90.40	71.20	60.10	46.84	140 K
DeepLab-LargeFOV[3]	85.95	60.41	50.18	26.25	87.76	62.57	53.34	32.04	88.20	62.53	53.88	32.77	140 K
DeepLab-LargeFOV-denseCRF[3]	not computed								89.71	60.67	54.74	40.79	140 K
FCN	81.97	54.38	46.59	22.86	82.71	56.22	47.95	24.76	83.27	59.56	49.83	27.99	200 K
FCN (learnt deconv) [2]	83.21	56.05	48.68	27.40	83.71	59.64	50.80	31.01	83.14	64.21	51.96	33.18	160 K
DeconvNet [4]	85.26	46.40	39.69	27.36	85.19	54.08	43.74	29.33	89.58	70.24	59.77	52.23	260 K

Benchmark 2: Indoor Scene Segmentation (SUN RGB-D)

- Performance: While all architectures performed poorly in absolute terms on this task, SegNet outperformed the other methods in Global Accuracy (G), Class Average (C), and the Boundary F1-measure (BF).

- Qualitative Insights: Qualitatively, SegNet produces reasonable predictions for large, common classes like "wall" and "floor." However, the segmentation quality becomes noisy and less reliable in highly cluttered scenes.

TABLE 4

Quantitative Comparison of Deep Architectures on the SUNRGB-D Dataset When Trained on a Corpus of 5,250 Indoor Scenes

Network/Iterations	80 K				140 K				> 140 K				Max iter
	G	C	mIoU	BF	G	C	mIoU	BF	G	C	mIoU	BF	
SegNet	70.73	30.82	22.52	9.16	71.66	37.60	27.46	11.33	72.63	44.76	31.84	12.66	240 K
DeepLab-LargeFOV [3]	70.70	41.75	30.67	7.28	71.16	42.71	31.29	7.57	71.90	42.21	32.08	8.26	240 K
DeepLab-LargeFOV-denseCRF [3]	not computed								66.96	33.06	24.13	9.41	240 K
FCN (learnt deconv) [2]	67.31	34.32	24.05	7.88	68.04	37.2	26.33	9.0	68.18	38.41	27.39	9.68	200 K
DeconvNet [4]	59.62	12.93	8.35	6.50	63.28	22.53	15.14	7.86	66.13	32.28	22.57	10.47	380 K

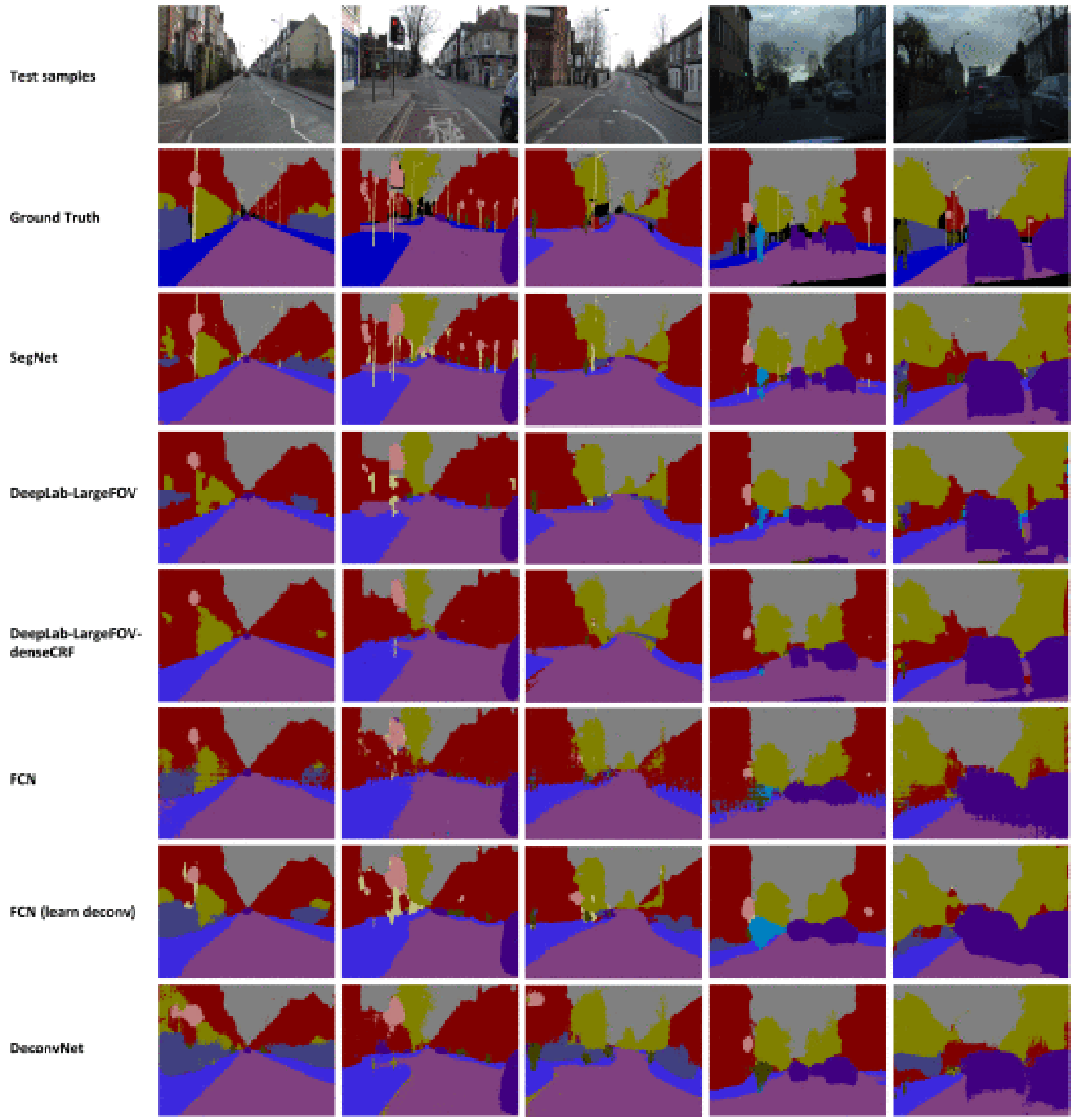
TABLE 5

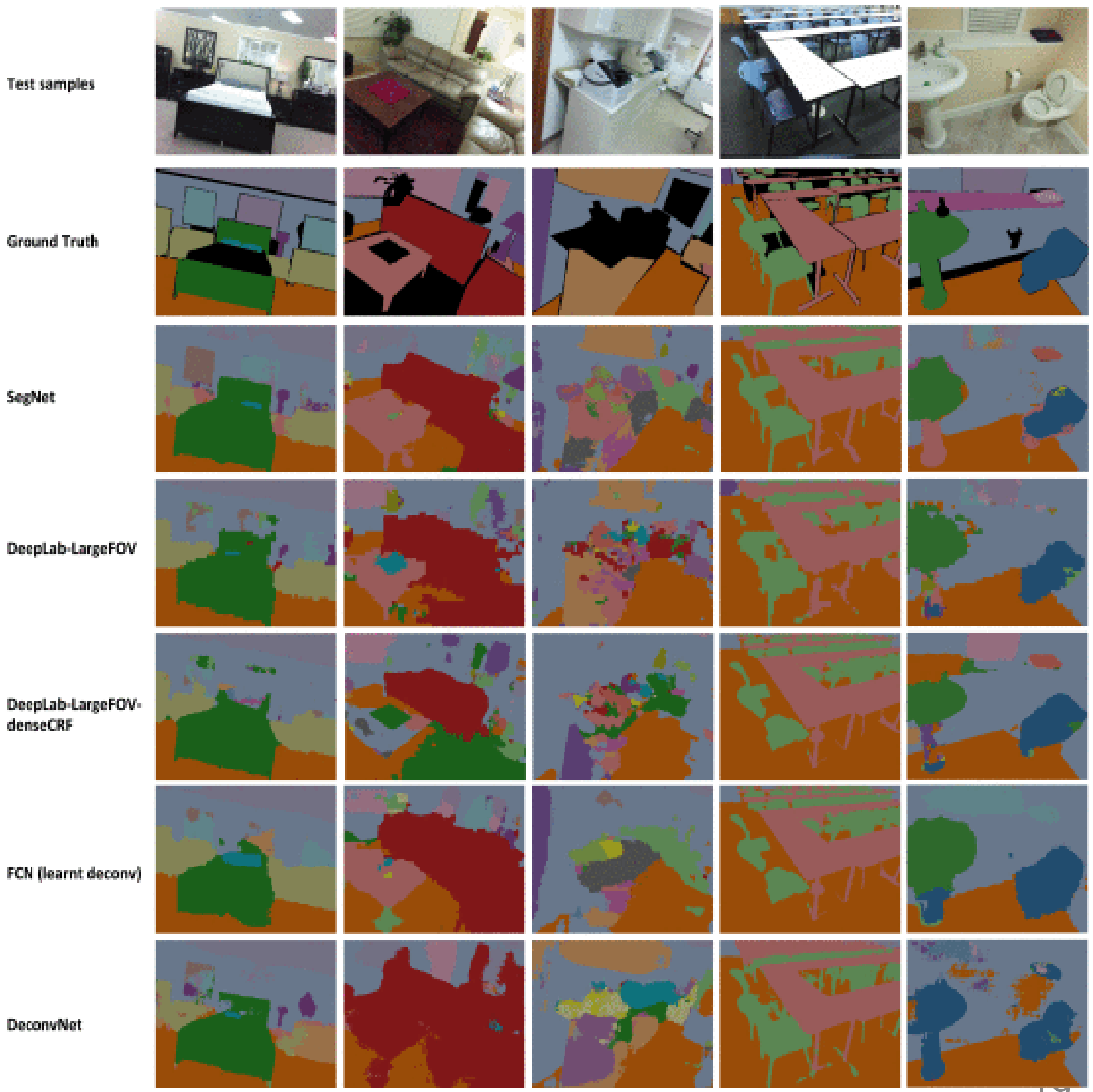
Class Average Accuracies of SegNet Predictions for the 37 Indoor Scene Classes in the SUN RGB-D Benchmark Dataset

Wall	Floor	Cabinet	Bed	Chair	Sofa	Table	Door	Window	Bookshelf	Picture	Counter	Blinds
83.42	93.43	63.37	73.18	75.92	59.57	64.18	52.50	57.51	42.05	56.17	37.66	40.29
Desk	Shelves	Curtain	Dresser	Pillow	Mirror	Floor mat	Clothes	Ceiling	Books	Fridge	TV	Paper
11.92	11.45	66.56	52.73	43.80	26.30	0.00	34.31	74.11	53.77	29.85	33.76	22.73
Towel	Shower curtain	Box	Whiteboard	Person	Night stand	Toilet	Sink	Lamp	Bathtub	Bag		
19.83	0.03	23.14	60.25	27.27	29.88	76.00	58.10	35.27	48.86	16.76		

The performance correlates well with size of the classes in indoor scenes. Note that class average accuracy has a strong correlation with mIoU metric.

Road Scene Segmentation and Indoor Scene Segmentation





6. Efficiency Analysis: Memory vs. Accuracy

Performance Analysis: Efficiency & Resource Usage

Network	Inference Memory (GPU MB)	Model Size (Disk MB)	Key Insight
SegNet	**1,052**	**117**	**Most memory-efficient during inference.**
DeepLab-LargeFOV	1,993	83	Smallest model on disk and fastest training and inference time.
FCN	1,806	539	Large model with high memory requirements.
DeconvNet	1,872	877	Largest model, very inefficient to train.

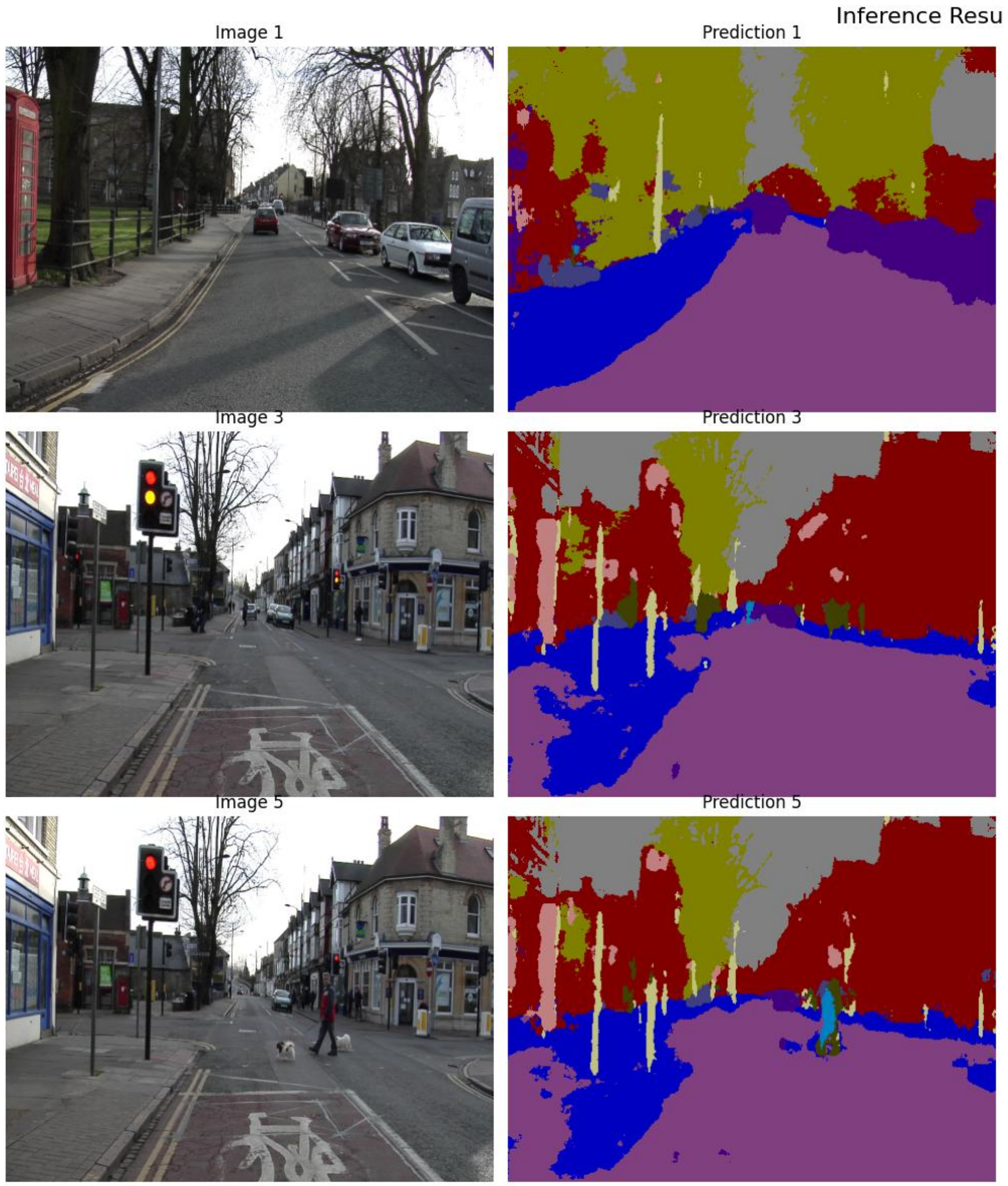
SegNet provides a compelling balance of strong performance and best-in-class inference memory usage, making it a practical choice for resource-constrained applications.

7. My Implementation and Improvement

Implementation Details: Reproducing SegNet

1. **Core Architecture**
   - Framework: PyTorch (Custom *nn.Module*).
   - Encoder: VGG16-style with 13 convolutional layers.
   - Mechanism: Implemented *return_indices=True* in *MaxPool* layers to capture spatial location of features for the decoder.

2. **Experimental Setup**
   - Dataset: CamVid (Road Scenes), 11 Classes.
   - Scope: Trained on the standard subset (367 train images) for 100 Epochs due to GPU resource limits (**vs. 40k iterations in paper**).
   - Training: 100 Epochs, Batch Size 4, SGD Optimizer (LR=0.001).
   - Loss Function: Cross-Entropy Loss with **Median Frequency Balancing** to handle the class imbalance.



Improvement model

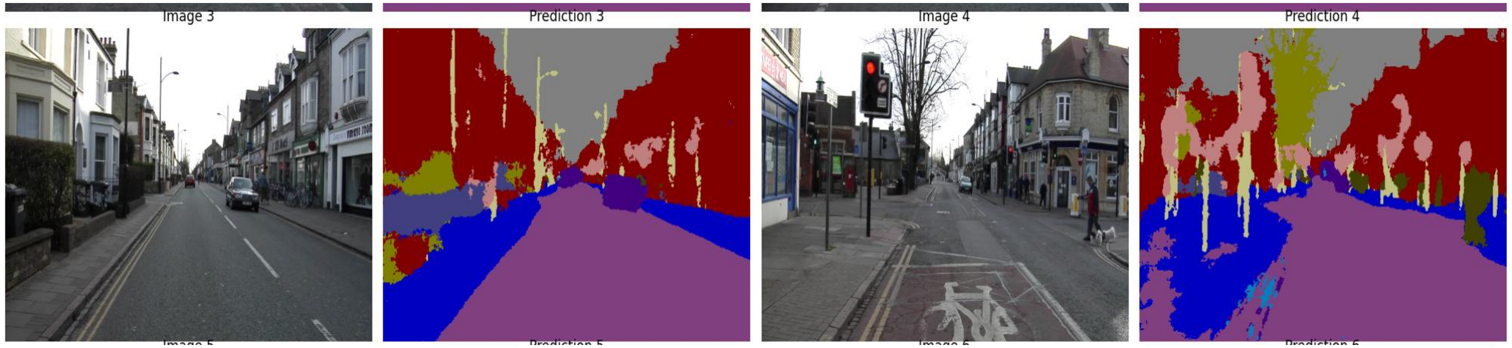
Because the CamVid dataset is small (367 images).

I hypothesized that the model was overfitting to specific lighting conditions and road orientations.

I introduced a dynamic augmentation techniques:

- ➢ **Random Horizontal Flipping (p=0.5):** Doubles the effective variety of pose orientations for "Pedestrians" and "Cars.“
- ➢ **Photometric Distortion:** Randomly jitters Brightness and Contrast (20%). Forces the model to learn structural shapes rather than relying on specific color values (e.g., recognizing a dark fence vs. a bright fence).

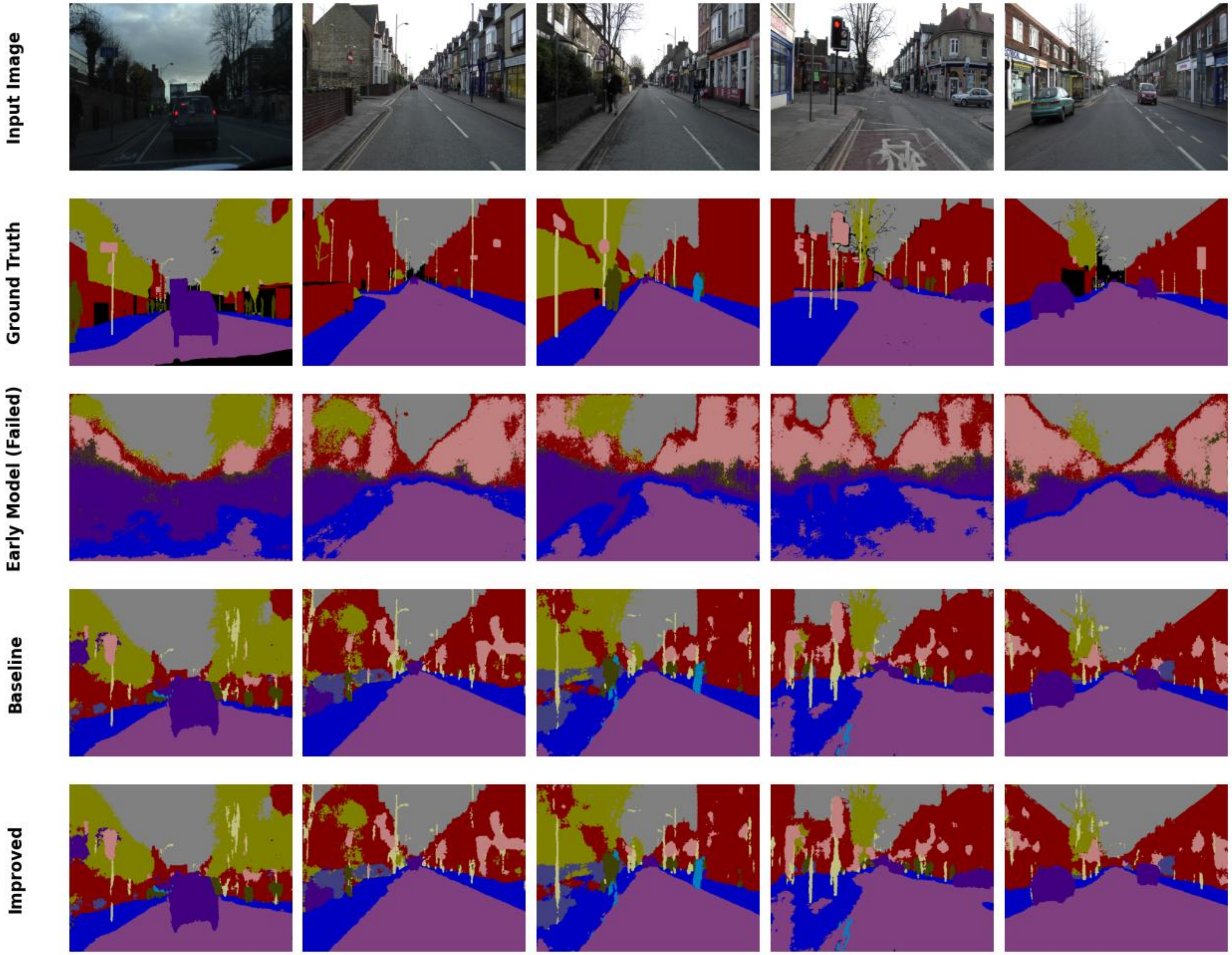
```python
# Augmentation for the Training Only
if self.train:
    if random.random() > 0.5:
        image = TF.hflip(image)
        mask = TF.hflip(mask)

    if random.random() > 0.5:
        image = TF.adjust_brightness(image, random.uniform(0.8, 1.2))
        image = TF.adjust_contrast(image, random.uniform(0.8, 1.2))

# Finalize
image = TF.to_tensor(image)
image = self.normalize(image)
if mask.mode != 'L': mask = mask.convert('RGB')
```



Performance Comparison

Metric	Paper Benchmark (SegNet-Basic)	My Baseline (No Aug)	My Improved (With Aug)
Global Accuracy	82.8%	83.76%	**83.61%**
Class Average	62.0%	59.33%	**67.52% (+8.2%)**
Mean IoU	46.3%	47.91%	**50.69% (+2.8%)**

Model Comparison



Challenges Faced

1. For the Challenges
    - Initial Failure: During early training, the model predicted "Sky" (Class 0) for every single pixel.
    - Cause: Extreme class imbalance. "Sky" and "Road" pixels account for >90% of the image. The model minimized loss by ignoring everything else.

2. The Solution
    - Median Frequency Balancing: I implemented the weighting formula from Badrinarayanan et al. paper. Classes like SignSymbol were assigned a weight of 9.64, while Road got 0.14.
    - Outcome: This forced the gradient descent to prioritize small objects, resolving the "Grey Screen" error.

REFERENCE:

SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation
Vijay Badrinarayanan, Alex Kendall , and Roberto Cipolla, Senior Member, IEEE

THANK YOU