

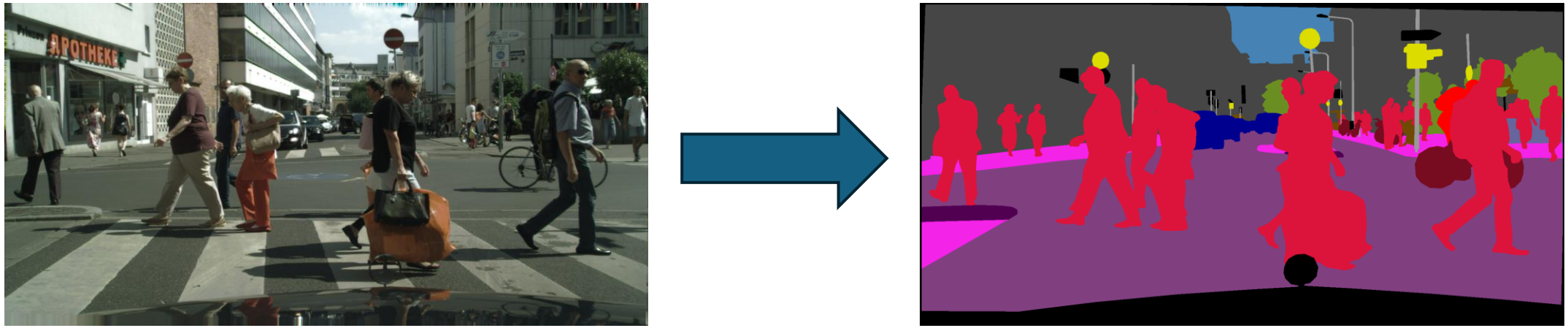

DIPARTIMENTO
MATEMATICA
DIPARTIMENTO DI MATEMATICA "TULLIO LEVI-CIVITA"

# Semantic Segmentation with DenseASPP

Ivan Tepliashin (mat. 2112455)
Nikita Paltov (mat. 2107498)
Ilia Smirnov (mat. 2106051)

# Semantic Image Segmentation

A computer vision task in which the goal is to categorize each pixel in an image into a class or object.




**Applications:** autonomous driving, medical imaging, robotics, …

# Dataset: Cityscapes

- A large-scale dataset of street-view images from 50 cities across Germany, Switzerland and Austria.
- 5000 finely annotated images divided into training (2975) and validation (500) set. (There is also a test set that we don't use.)
- Resolution of $2048 \times 1024$ pixels.
- 35 classes (only 20 usually used).

(a) Image (b) Ground Truth

# Dataset: Preprocessing

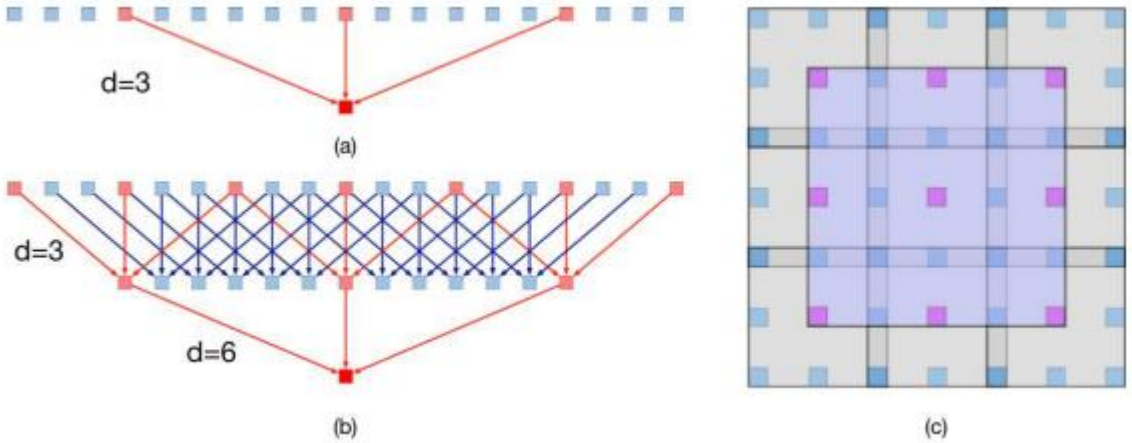
- Map 35 classes to 20.
- Resize from $2048 \times 1024$ to $1024 \times 512$.
- Normalize using ImageNet mean and standard deviation values to align with a pre-trained feature extractor.

# Atrous (Dilated) Convolutions

High-level feature representation in semantic segmentation is sensitive to scale changes.

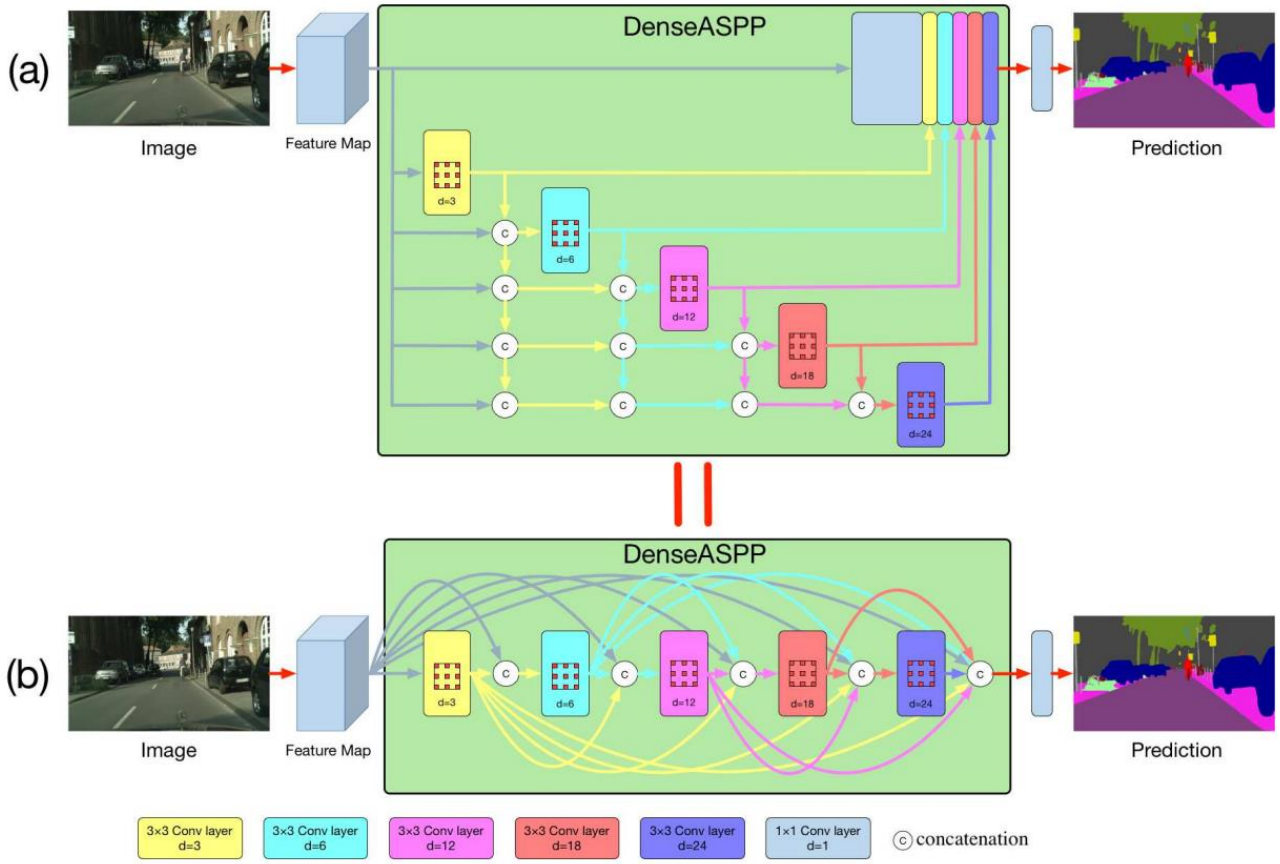
A large receptive field is required. One approach to balance between feature map size and receptive field size is to use atrous convolutions.

Each output neuron possesses a larger receptive field without increasing the number of kernel parameters.



# DenseASPP

- A cascade of atrous convolution layers with increasing dilation rate.
- The output of each layer is concatenated with the input feature map and fed into the following layer.
- Ensembling of layers due to dense connections.



# DenseASPP: Receptive Field Size

Receptive field size for an atrous convolution layer with dilation rate $d$ and kernel size $K$:

$$R = (d - 1) \times (K - 1) + K$$

Stacking two convolutional layers together:

$$K = K_1 + K_2 - 1$$

This approach provides us with a large receptive field allowing to capture global information in high resolution images.

# Best Model Architecture

3 modules:

- **Feature extractor:** a pre-trained **DenseNet-121**. Two last pooling layers are removed and the dilation rates of the convolution layers after the removed pooling layers are set to 2 and 4 respectively.
- **DenseASPP module:** 5 atrous convolution blocks with dilation rates [3, 6, 12, 18, 24]. Each block consists of a $1 \times 1$ convolutional layer and a main atrous convolutional layer.
- **Classifier head:** $1 \times 1$ convolutional layer with 20 filters. The output is upsampled by a factor of 8.

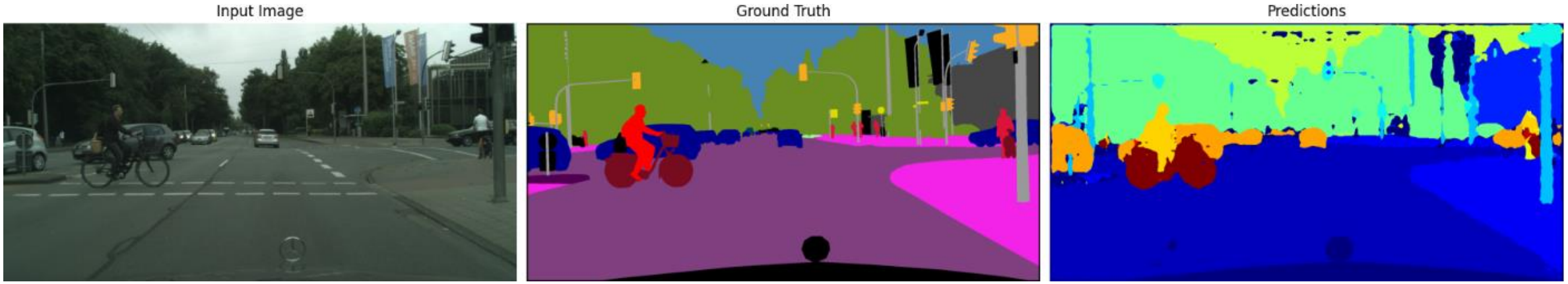
# Best Model Parameters

- Output feature map of DenseNet-121 – 1024 channels.
- Reduction of channels in the feature map before each atrous layer into 256.
- Dilation rates in the DenseASPP module: [3, 6, 12, 18, 24].
- Each atrous layer outputs 64 channels (growth rate).

Trained for 60 epochs using Adam optimizer with learning rate of 0.0003 and weight decay of 0.00001.

Reached mIoU score of 54.0% on validation set.

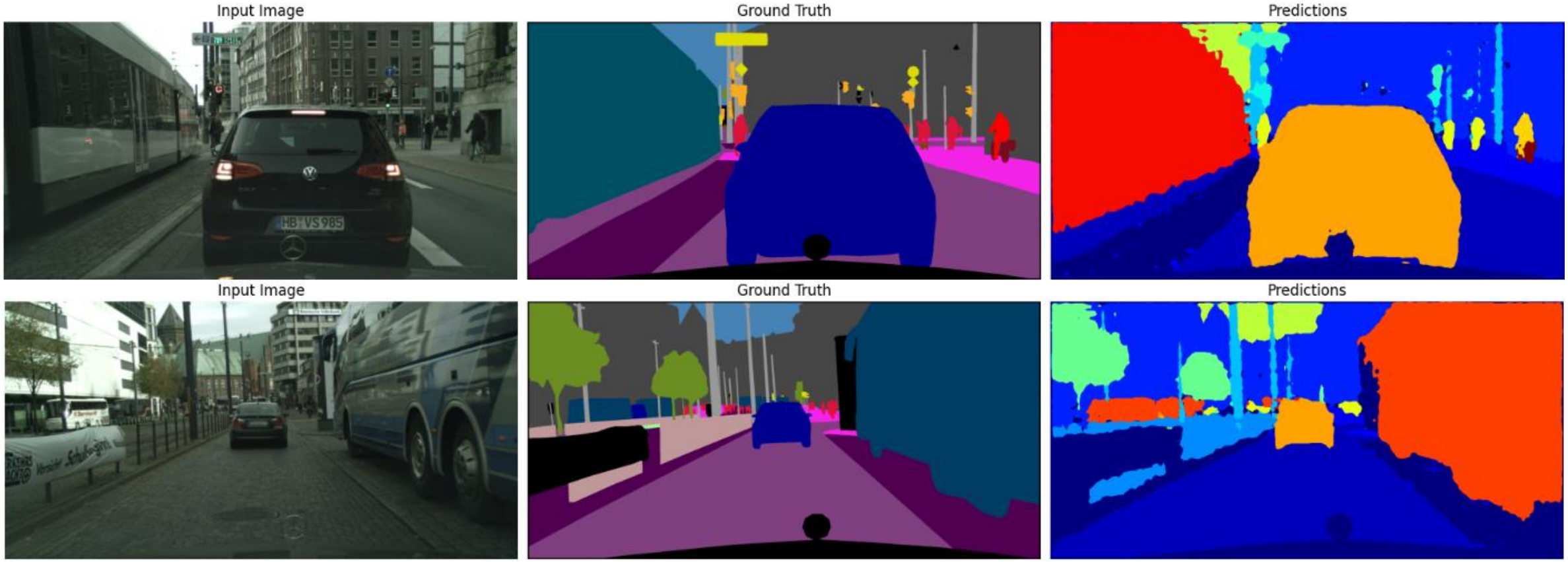
# Results of the Best Model




- Large objects are segmented correctly
- Cars are segmented best of all.
- Small objects (traffic signs, traffic lights, etc.) are segmented poorly

# Results of the Best Model

Another couple of examples:



# Experiments: Search for the Best Model

| Model | DenseASPP | Growth rate | # parameters | Image size | # epochs | mIoU train | mIoU val |
|---|---|---|---|---|---|---|---|
| #1 | (3, 6, 9, 12) | 32 | 662420 | $224 \times 224$ | 50 | 51.4% | 31.6% |
| #2 | (3, 6, 9, 12) | 32 | 1266580 | $224 \times 224$ | 60 | 54.9% | 29.9% |
| #3 | (6, 12, 18, 24) | 32 | 1266580 | $512 \times 512$ | 80 | 63.3% | 41.4% |
| #4 | (3, 6, 12, 18, 24) | 64 | 2252820 | $512 \times 1024$ | 60 | **73.5%** | **54.0%** |
| #5 | (3, 6, 9, 12, 18, 24) | 64 | 2748179 | $512 \times 1024$ | 60 | 55.0% | 39.5% |

- In models #1 and #2 we removed last dense blocks in the extractor.
- The best model is #4.
- We tried training #5 on 19 classes instead of 20.

# Experiments: Feature Extractor Variation

| Feature extractor | # epochs | mIoU on training set | mIoU on validation set |
|---|---|---|---|
| DenseNet-121 | 60 | 73.5% | 54.0% |
| DenseNet-201 | 60 | **73.9%** | **54.2%** |
| MobileNet V3 | 40 | 67.7% | 41.9% |
| ConvNeXt | 30 | 65.9% | 53.0% |

- **DenseNet-201:** DenseNet family, heavier than DenseNet-121.
- **MobileNet V3:** lightweight CNN optimized for mobile and edge devices, introduced in 2019.
- **ConvNeXt:** inspired by Transformer-based models, introduced in 2022.

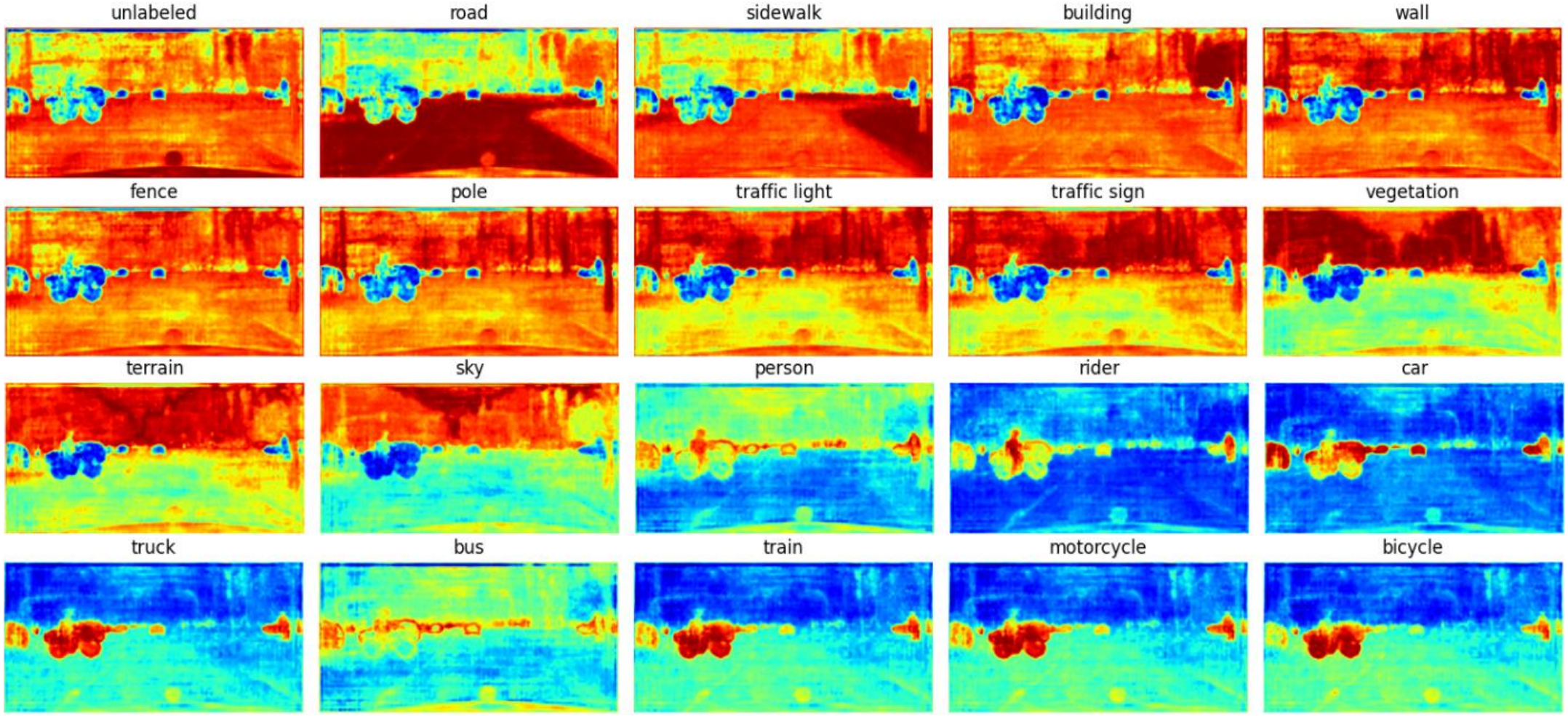
# Experiments: Feature Similarity

1. For each unique label in ground truth a mean feature vector is calculated.
2. Cosine similarity is calculated between each pixel's feature vector and this mean feature vector.
3. The similarity scores are stored as a heatmap.

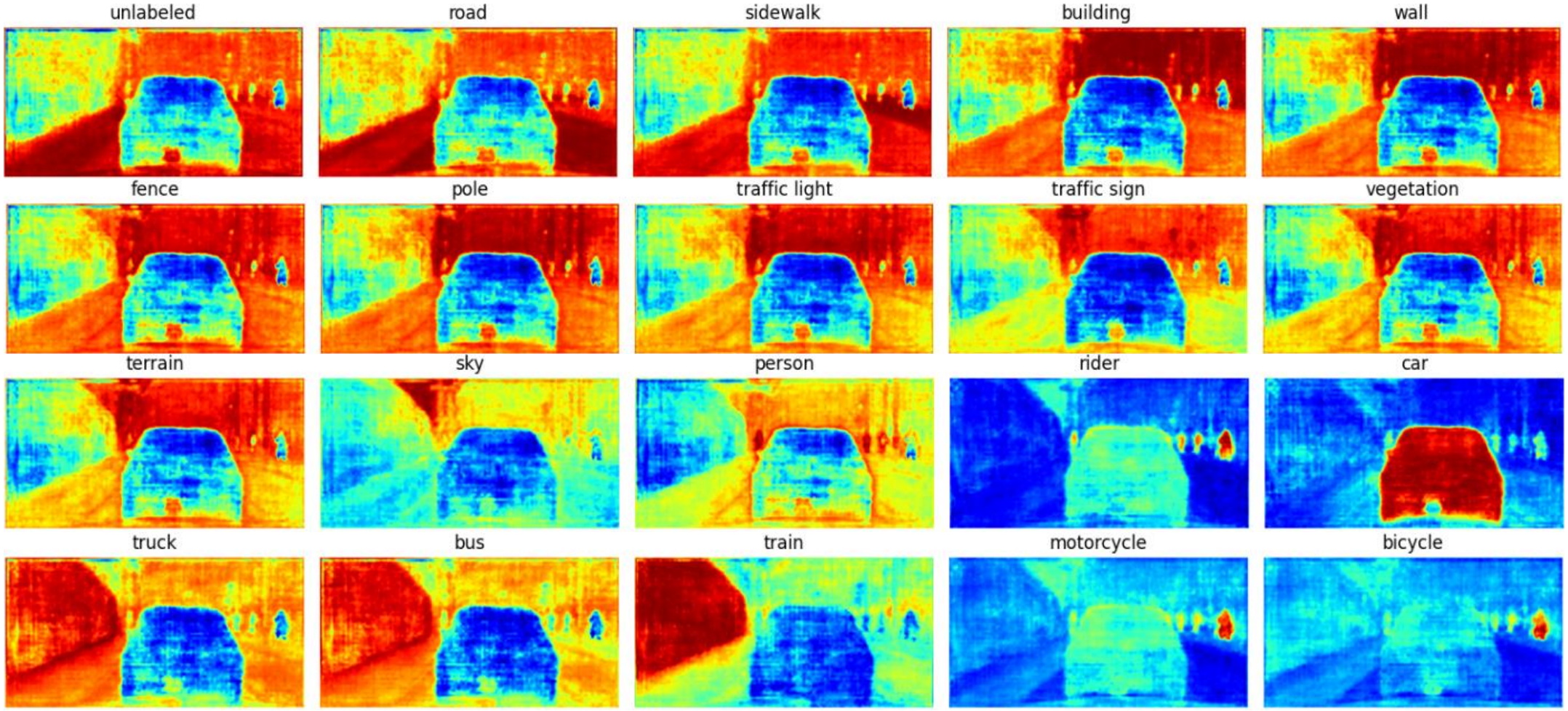
Some classes often have high feature similarity:

- "Wall" and "Building"
- "Bicycle" and "Motorcycle"

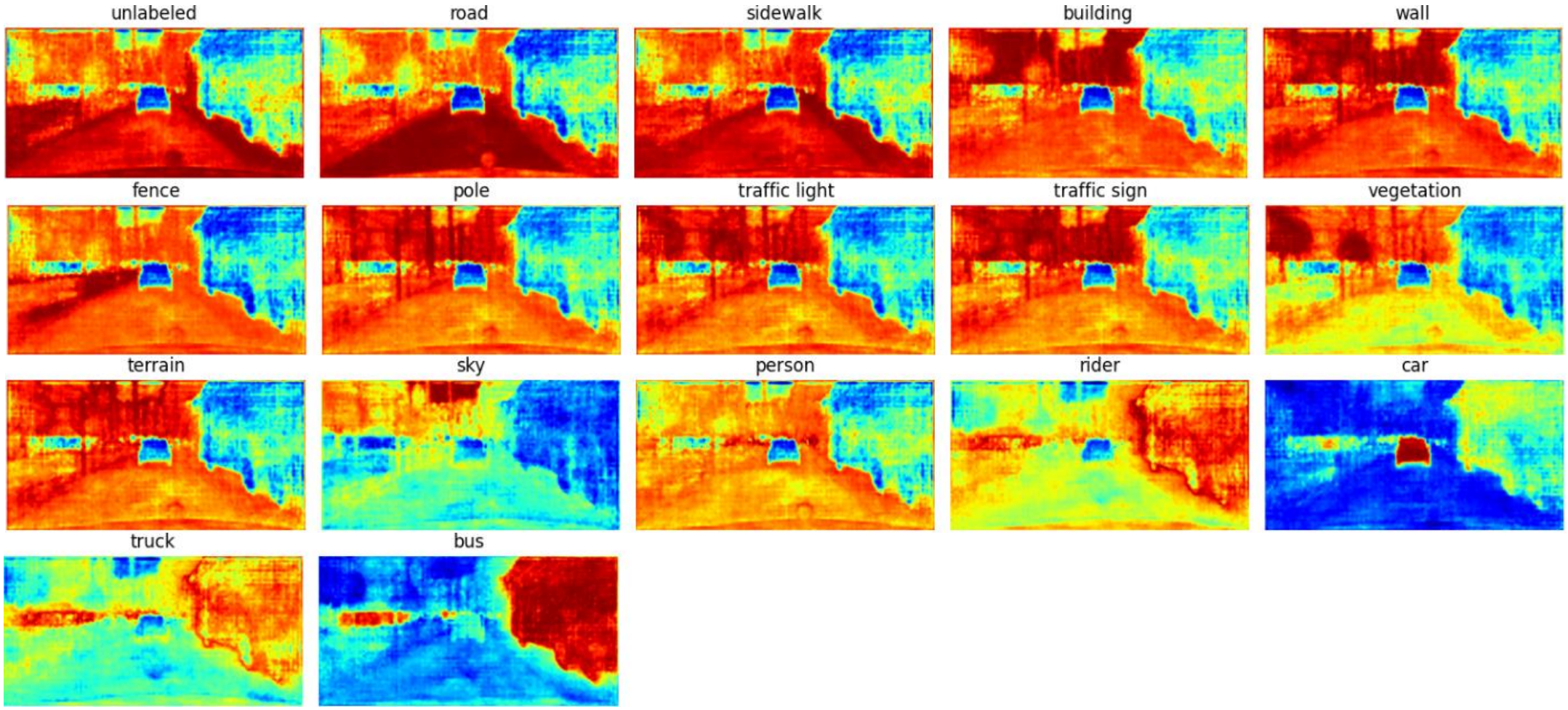
# Experiments: Feature Similarity – Example 1



# Experiments: Feature Similarity – Example 2



# Experiments: Feature Similarity – Example 3

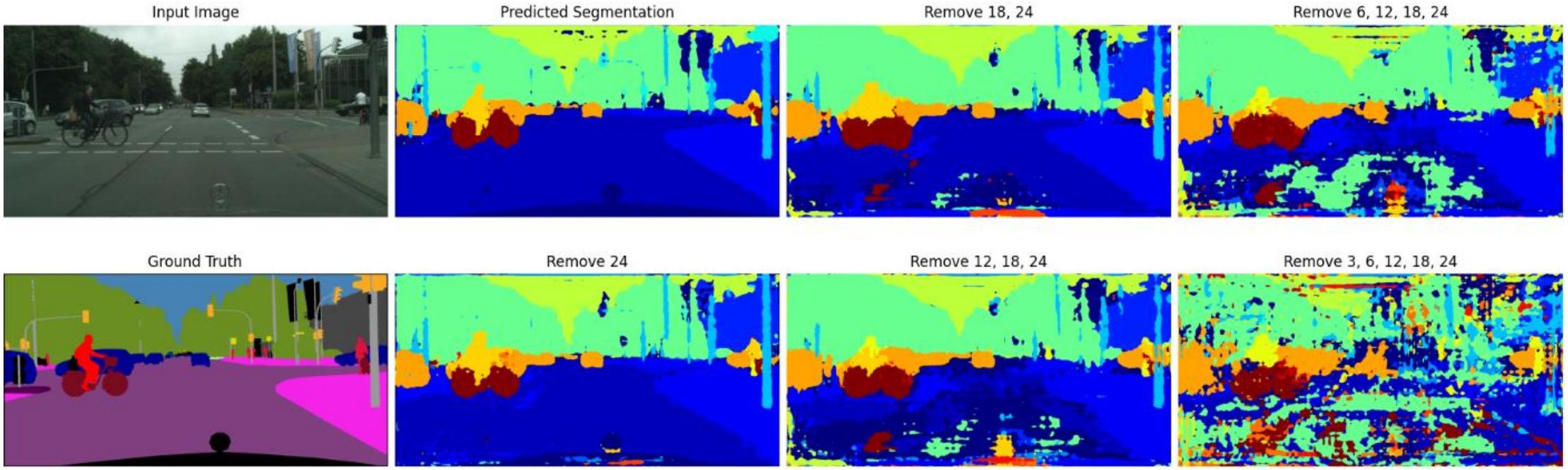


# Experiments: Layers Removal

We tried removing top DenseASPP blocks from the best model.

- Larger objects are still recognizable with 1-2 top blocks removed, further they suffer significantly.
- Smaller objects are still recognizable with even 3 top blocks removed.
- When all the DenseASPP blocks are removed, the result makes almost no sense at all.

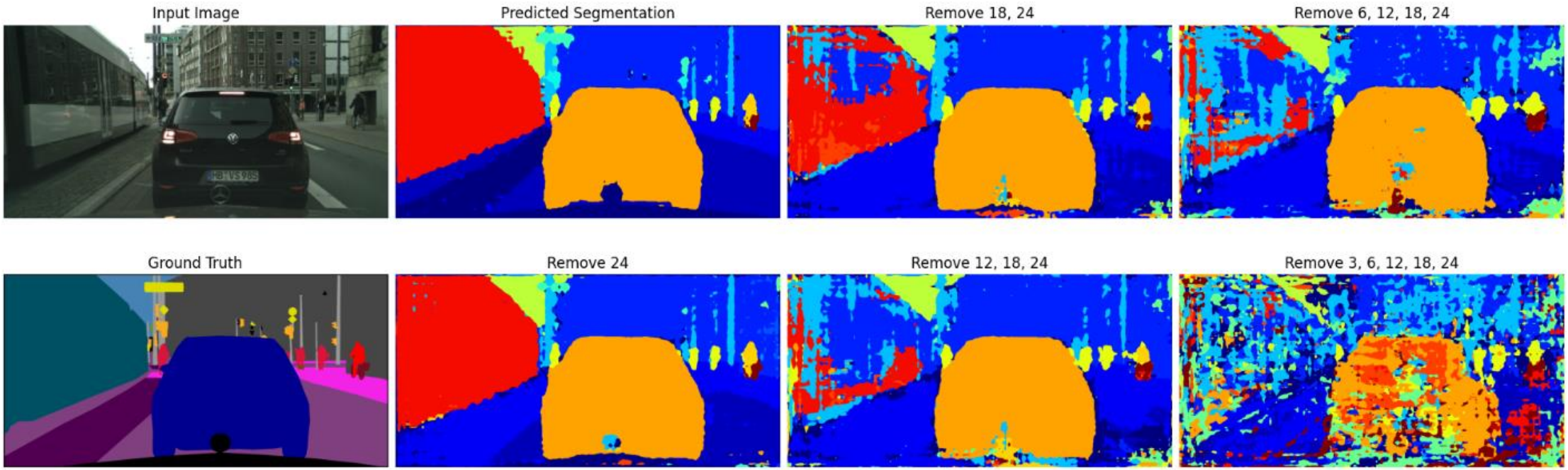
# Experiments: Layers Removal – Example 1




Bicycle, cars are recognized even with almost all layers removed.

Road suffers even with 2 layers removed.

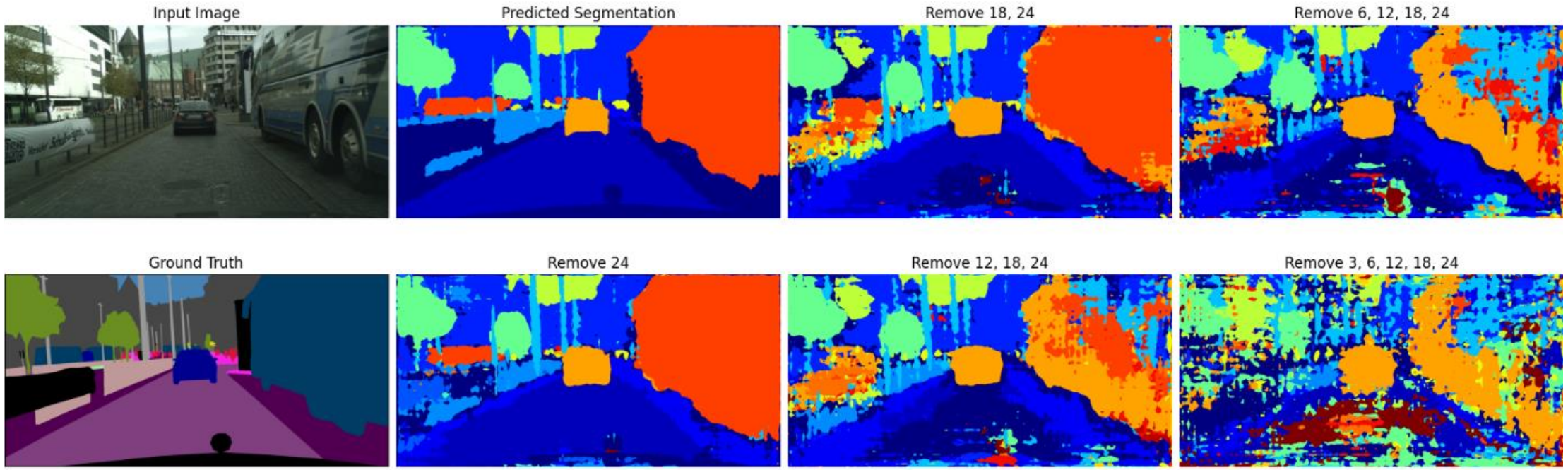
# Experiments: Layers Removal – Example 2




Car and people are recognized even with almost all layers removed.

Train suffers significantly even with 2 layers removed.

# Experiments: Layers Removal – Example 3




Car and trees are recognized even with almost all layers removed.

Bus is still recognized after 2 layers removed.

# Conclusion & Discussion

Overall, we have achieved a decent performance with 54.0% mIoU.

Possible further improvements (given enough computational capacity):

- Dataset augmentation (random crops, flips, scaling, etc.)
- Longer training (up to 80 epochs)
- Higher resolution inputs to capture finer details
- A better feature extractor (potentially ConvNeXt)

# Thank you!