DS-GA 1008 Spring 2020 Deep Learning

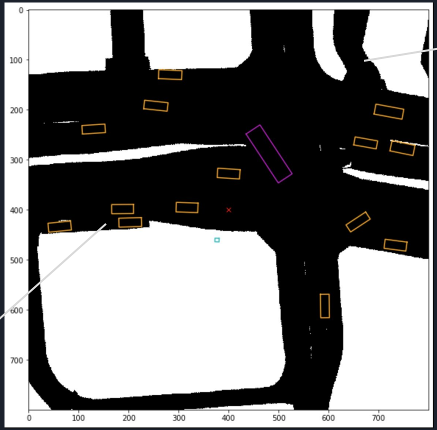
Final Project: Road Map + Object Detection

Team Mini-Batch

- Henry Steinitz (hjs410)
- Chady Raach (cr3144)
- Jatin Khilnani (jk6373)

Problem Statement + Solution Approach

Input Sample (6 images)



Task 1: Road Map Layout
Akin to Semantic Segmentation

Task 2: Bounding Box (Object Detection)

Semantic-aware Dense Projection Network (requires input frontal coordinates)

Deep Learning based Vehicle Position and Orientation Estimation via Inverse Perspective Mapping Image (Utilizes camera extrinsics)

Combine image features to generate BEV & predict boxes

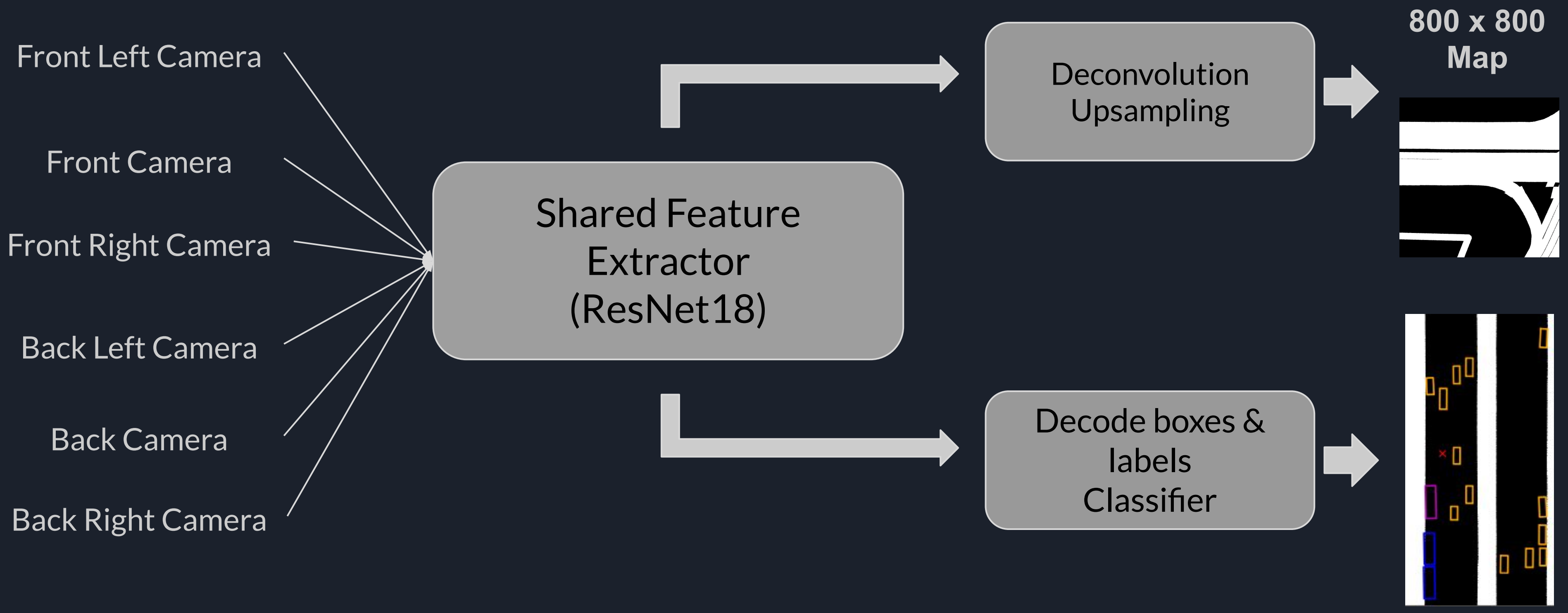
Combined Approach

Panoptic Segmentation (Semantic + Instance Segmentation)

Utilization of Temporal aspect of data in a recurrent network fashion

Self-supervised learning through pre-text task

Implemented Architecture Overview



Front Left Camera

Front Camera

Front Right Camera

Back Left Camera

Back Camera

Back Right Camera

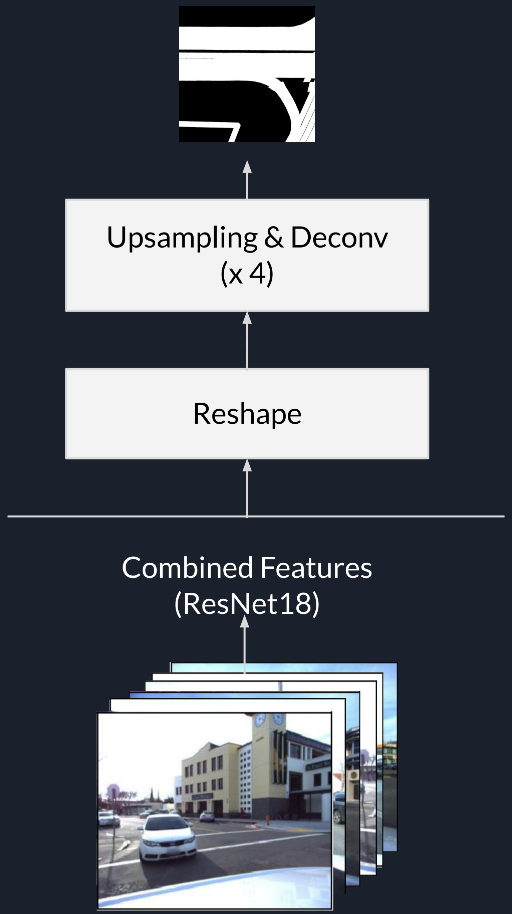
Shared Feature Extractor (ResNet18)

Deconvolution Upsampling

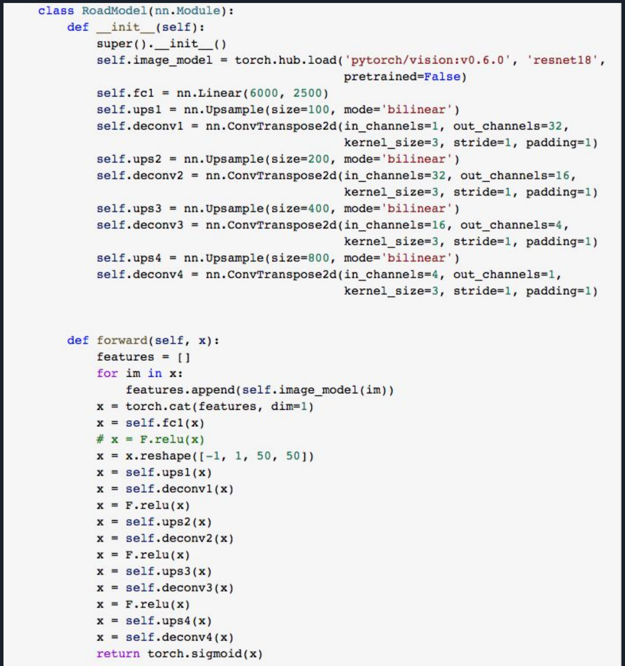
Decode boxes & labels Classifier

800 x 800 Map

Road Map Model



Model Code

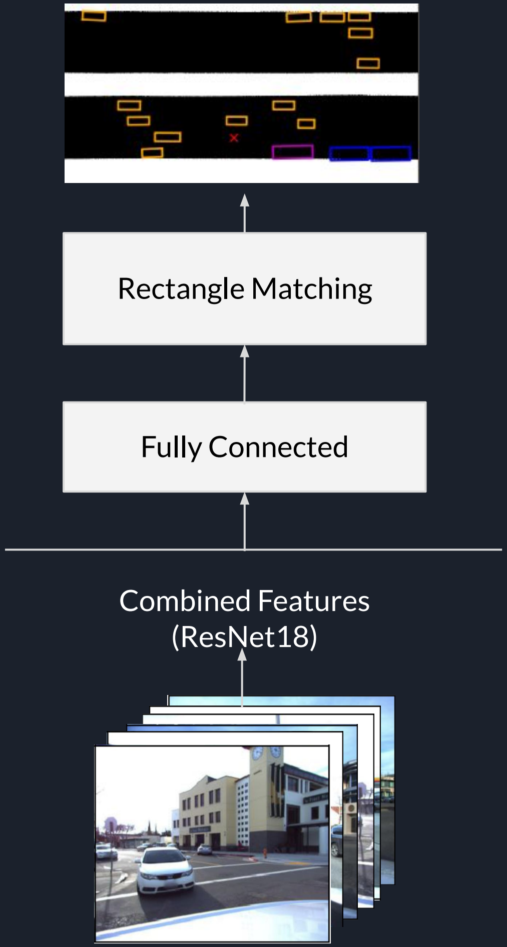


```python
class RoadModel(nn.Module):
    def __init__(self):
        super().__init__()
        self.image_model = torch.hub.load('pytorch/vision:v0.6.0', 'resnet18',
                                          pretrained=False)
        self.fc1 = nn.Linear(6000, 2500)
        self.ups1 = nn.Upsample(size=100, mode='bilinear')
        self.deconv1 = nn.ConvTranspose2d(in_channels=1, out_channels=32,
                                          kernel_size=3, stride=1, padding=1)
        self.ups2 = nn.Upsample(size=200, mode='bilinear')
        self.deconv2 = nn.ConvTranspose2d(in_channels=32, out_channels=16,
                                          kernel_size=3, stride=1, padding=1)
        self.ups3 = nn.Upsample(size=400, mode='bilinear')
        self.deconv3 = nn.ConvTranspose2d(in_channels=16, out_channels=4,
                                          kernel_size=3, stride=1, padding=1)
        self.ups4 = nn.Upsample(size=800, mode='bilinear')
        self.deconv4 = nn.ConvTranspose2d(in_channels=4, out_channels=1,
                                          kernel_size=3, stride=1, padding=1)


    def forward(self, x):
        features = []
        for im in x:
            features.append(self.image_model(im))
        x = torch.cat(features, dim=1)
        x = self.fc1(x)
        # x = F.relu(x)
        x = x.reshape([-1, 1, 50, 50])
        x = self.ups1(x)
        x = self.deconv1(x)
        x = F.relu(x)
        x = self.ups2(x)
        x = self.deconv2(x)
        x = F.relu(x)
        x = self.ups3(x)
        x = self.deconv3(x)
        x = F.relu(x)
        x = self.ups4(x)
        x = self.deconv4(x)
        return torch.sigmoid(x)
```

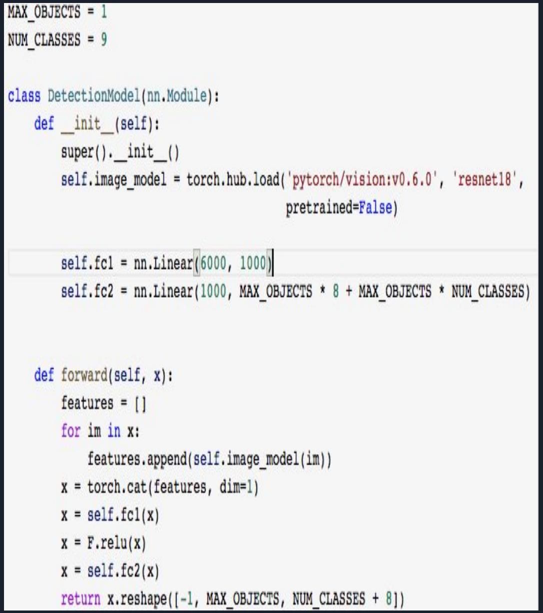
Parameters/Config

Batch Size	8
Learning Rate	0.001
Momentum	0.9, 0.99
Loss	MSE

Bounding Box Model



Model Code

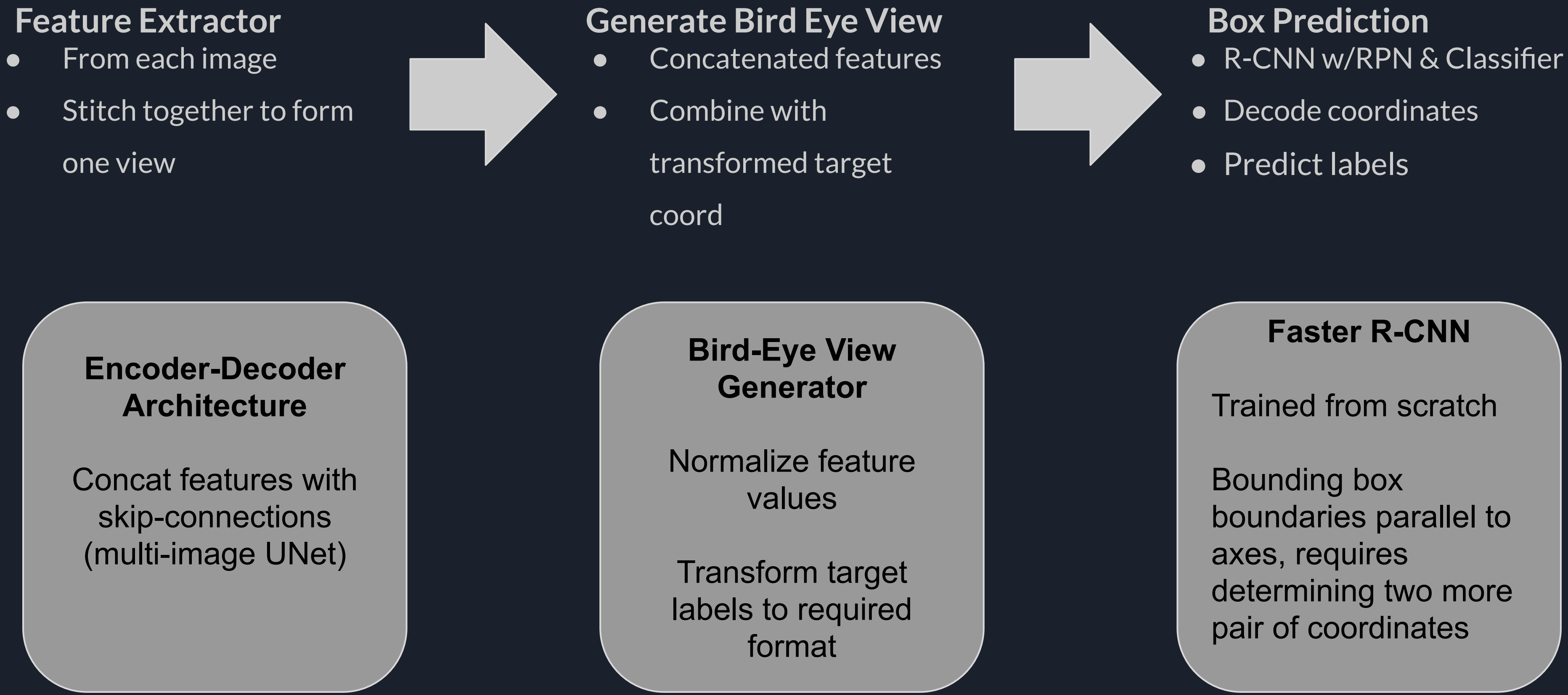


```
MAX_OBJECTS = 1
NUM_CLASSES = 9

class DetectionModel(nn.Module):
    def __init__(self):
        super().__init__()
        self.image_model = torch.hub.load('pytorch/vision:v0.6.0', 'resnet18',
                                          pretrained=False)

        self.fc1 = nn.Linear(6000, 1000)
        self.fc2 = nn.Linear(1000, MAX_OBJECTS * 8 + MAX_OBJECTS * NUM_CLASSES)


    def forward(self, x):
        features = []
        for im in x:
            features.append(self.image_model(im))
        x = torch.cat(features, dim=1)
        x = self.fc1(x)
        x = F.relu(x)
        x = self.fc2(x)
        return x.reshape([-1, MAX_OBJECTS, NUM_CLASSES + 8])
```

Parameters/Config

Batch Size	8
Learning Rate	0.001
Momentum	0.9, 0.99
Loss	MSE (box coordinates) Cross-Entropy (box labels)

Box Prediction Alternative



Feature Extractor

- From each image
- Stitch together to form one view

Generate Bird Eye View

- Concatenated features
- Combine with transformed target coord

Box Prediction

- R-CNN w/RPN & Classifier
- Decode coordinates
- Predict labels

Encoder-Decoder Architecture

Concat features with skip-connections (multi-image UNet)

Bird-Eye View Generator

Normalize feature values

Transform target labels to required format

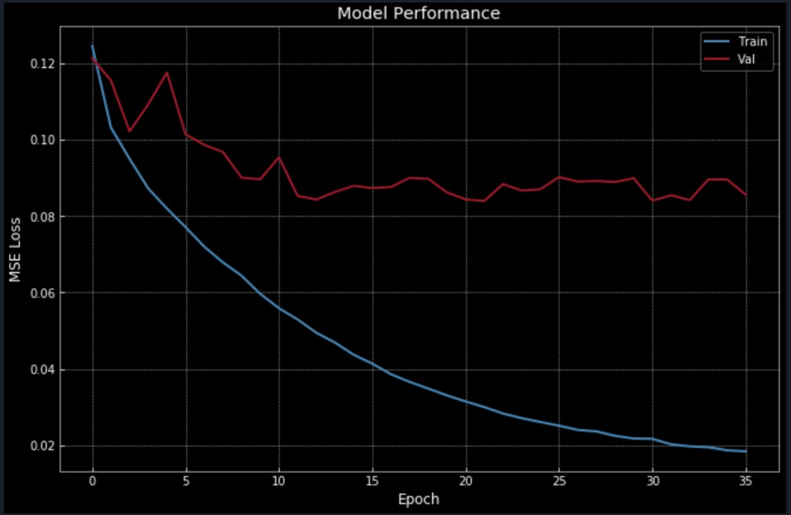
Faster R-CNN

Trained from scratch

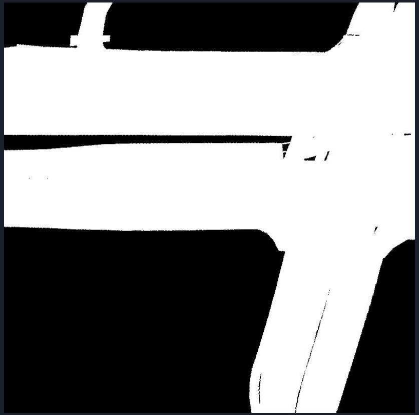
Bounding box boundaries parallel to axes, requires determining two more pair of coordinates

Results

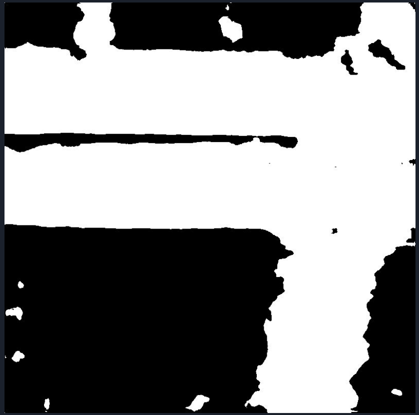
RoadMap Model



Loss curves



Target



Prediction

- Best model chosen from Epoch 21 for least validation loss
- Still not saturated, training loss continues to decrease while validation fluctuates around 0.84

Thank you