Final Project Report - Deep Learning DS GA 1008

Shreyas Chandrakaladharan [*1] **Marina Zavalina** [*1] **Philip Ekfeldt** [*1]

Abstract

With the recent advances and burgeoning interest in self-driving cars, processing and understanding the environment around the car has become an important problem. In this project we focus on Bird Eye View (BEV) prediction based on monocular photos taken by the cameras on top of the car. We present a Maximum Mean Discrepancy Variational Auto Encoder (VAE) to predict the BEV road layout. We also contribute an approach combining Image Warping, U-Net and Post-processing to predict the bounding boxes (BB) on the BEV layout. We achieved 0.81 test threat score on the road layout prediction task and 0.07 test threat score on the BB prediction task. Figure 1 visualizes the predictions of our final models.

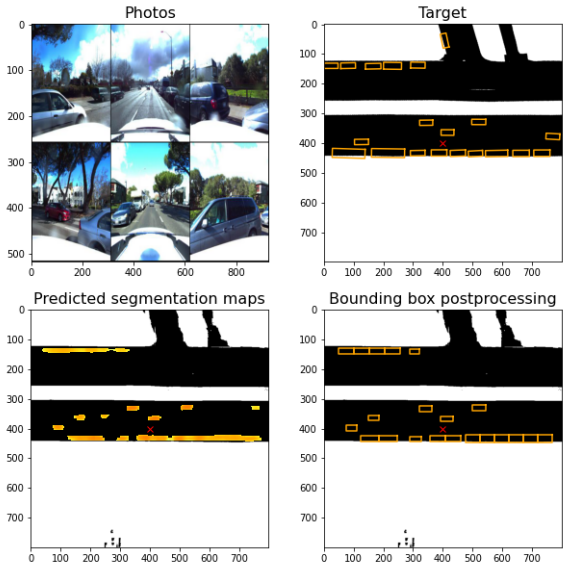


Figure 1. Example from validation set: input photos, target and prediction of our best model on validation example (segmentation maps and final prediction).

[*]Equal contribution [1]Center for Data Science, New York University, New York, New York.

1. Related Work

One of the early works trying to predict BEV road layout from monocular photos was done by (Schulter et al., 2018). They train a CNN to predict the semantic segmentation map and depth map of a given photo. They combine these predictions and warp them into the BEV. They use supervisory signals from segmentation maps of the monocular photos and a depth sensor to train their CNN. Furthermore, they refine their outputs using supervisory signals from OpenStreetMap (OSM) data.

A related approach was pursued by (Lu et al., 2018) wherein they trained a Variational Auto Encoder to do the same task. However, in addition to predicting the BEV road and sidewalk layouts like in (Schulter et al., 2018), they predict a semantic map BEV with different colors for road, terrain, sidewalk, etc. Interestingly, they obtained their supervisory signal by generating weak ground truth using the semantic and depth maps of the input photo.

Improving on these works, in (Mani et al., 2020) the authors trained an end-end deep learning model that predicts BEV road layout and BEV BB segments of objects in the scene. They used two separate decoders and a shared encoder to predict static (road) pixels and dynamic (object) pixels in their output. Additionally, like in (Schulter et al., 2018) they use OSM data for adversarial training to refine their predictions.

2. Data Description

Given 6 monocular RGB photos of size 256×306 that capture the 360° scene around the ego car, the task of the project is to predict the BEV of the scene with the ego car in the center, specifically the goal is to predict the road and the cars. The target is an 800×800 binary map with 1 indicating presence of road. The target also contains the BBs coordinates for all objects found in the scene.

We had a total of 134 scenes in the given dataset, each scene spans 25 seconds of the ego car's journey divided into 126 samples (snapshots at given timestamp), each sample containing 6 monocular photos. Out of these 134 scenes, 28 scenes were labelled scenes with the road maps and corresponding BBs available as labels. We use first 25 labelled scenes as the training set and remaining 3 scenes

as validation set. Also for the road layout task, we did data augmentation during training using torch transforms. We used grayscale, horizontal flip, color distortion and Gaussian blurring.

To better understand the data, we visualized the distribution of roads and cars in the training set as a probability map, shown in Figure 2.

3. Uniqueness of the Task

In our task, we have ground truth labels available for BEV layout and no labels for segmentation / depth estimation for monocular photos. This is in contrast to the the works described in section 1 and most other works in the domain of BEV prediction where semantic / depth maps are usually used in training.

Moreover, for object detection in particular, successful approaches like Faster RCNN, YOLO, Mask RCNN, RetinaNet use Region Proposal Networks / Feature Pyramid Networks that are trained using high resolution pixel-level segmentation maps of images. These need lots of high quality data to train irrespective of whether they are single stage or two-stage methods. This enables these architectures to even regress BB coordinates precisely. In our task, we do not have access to such high quality segmentation maps for our monocular photos. The best way we can use such architectures is to generate weak ground truth by using homography or other mapping techniques to transfer BBs from our BEV targets to our monocular photos. We do not use RPN/FPN architectures like in Mask RCNN/RetinaNet because such architectures would need lots of data, high quality segmentation maps as labels and consequently lots of compute power as well.

4. Methodology

4.1. Baseline

While analyzing the data using the probability maps shown in Figure 2, we create simple baselines for both tasks, again shown in Figure 2. For road layout prediction, our baseline always predicts two roads and obtains a validation threat score of 0.72. For BBs prediction, our baseline always predicts a line of parked cars to the right of the ego car and obtains a validation threat score of 0.011.

4.2. General Approach

Inspired by (Mani et al., 2020), we wanted to build an end-to-end deep learning model for our task.

Naturally, we split our task into two: road layout prediction and BB prediction. We treat both tasks as semantic segmentation problems and train models for each task separately.

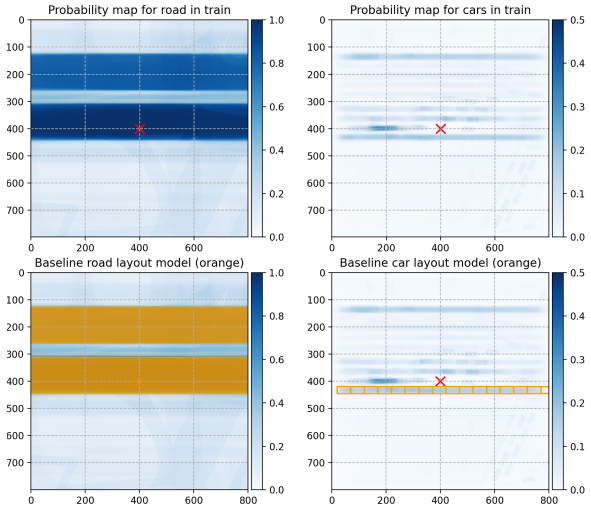


Figure 2. Above: Probability distribution of target maps for roads and cars in train set. Note that the color scale for cars is [0, 0.5]. *Below*: Baseline models for both tasks shown in orange.

This way we can test the same architecture for both tasks. Though each scene captures continuous journey of the ego car, we do not use temporal information. We use 6 monocular photos at a given timestamp as input. Since we are predicting binary maps as targets, we use cross entropy loss.

4.3. Road Layout Prediction

For the Road Layout prediction, we use the 6 raw photos as input and predict 800×800 tensor of pixel-wise probabilities and use a fixed threshold to convert it to a binary map. We experimented with several architectures such as Autoencoder (AE), VAE and MMD VAE described in detail below.

Autoencoder The first architecture we tested was a vanilla AE. We use a shared encoder to get representations from each of the six photos, concatenate the six tensors along the channel dimension and pass the resultant tensors into our decoder which outputs a 800×800 binary map. Our shared encoder is a ResNet18 backbone with the final linear layer removed. We also change the output shape of the penultimate Average Pooling layer of ResNet18 from (1,1) to (8,8). Our decoder is a sequential stack of transpose convolutional layers with batch norm and Leaky ReLU layers inbetween. We have a final sigmoid activation layer in our decoder.

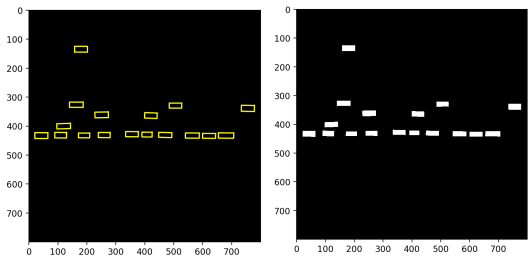
Variational Autoencoder A natural extension to improving the AE was introducing regularization in the latent dimension by using variational sampling and inference. So, the second architecture we tried was a VAE. To implement

the VAE, we modified the AE architecture by introducing a separate variational module. The module takes the concatenated representation from the shared encoder, passes them through intermediate convolutional layers and outputs the mean and variance tensors (4096-dim tensors) for the VAE. The decoder uses these tensors to generate a sample from the normal distribution and deconvolves this sample to the output 800×800 map. We also add a KL divergence term to our Cross Entropy loss. During inference, the mean vector is directly used as the sample.

Information Maximizing Variational Autoencoder Though we obtained good performance with our VAE, to further improve on our results on the Road Layout task, we implemented infoVAE (Zhao et al., 2017). The authors of the paper introduce the Information Maximizing VAE which improves VAE by using a Maximum Mean Discrepancy (MMD) divergence instead of the traditional KL Divergence. The MMD divergence tries to match distributions on all their moments instead of the ELBO approach. As shown in the paper, the KL divergence term can encourage uninformative latent codes and over-estimate the variance in feature space. MMD overcomes those pitfalls of KL divergence. Practically, implementing MMD involved replacing the variational module and the KL divergence loss term from our VAE to accommodate the MMD variational sampling and inference.

4.4. Bounding Boxes Prediction

For the BBs prediction, due to the uniqueness of our task as elaborated in section 3, we approach it as a semantic segmentation task. We convert target BB maps to segmentation maps using scipy.spatial library as shown in Figure 3.



(a) Bounding Boxes (b) Segmentation target

Figure 3. Target transformation for BBs prediction task

Though we tried all the architectures described above, we did not get great performance for predicting BBs. Our winning model uses a combination of Image Warping, U-Net and Post-processing which are described in detail below.

Image Warping Following convention in prior works, to combine the six input photos we first pass them through an encoder and concatenate the six representations. This approach works well for the Road Layout task but not the BB task.

Thus, for BBs prediction we use a different approach. We project each camera view to a BEV using Kornia (Riba et al., 2020) and combine them into one image as shown in Figure 4. For projecting, we use fixed projection matrices found by mapping corners of each camera view to corners of the corresponding BEV segment. We project only the lower part of the photo (below horizon), so we lose some information, e.g. traffic lights, top parts of the cars. We use this projected image, representing the combined BEV of the six input photos, as the input to our U-Net architecture.

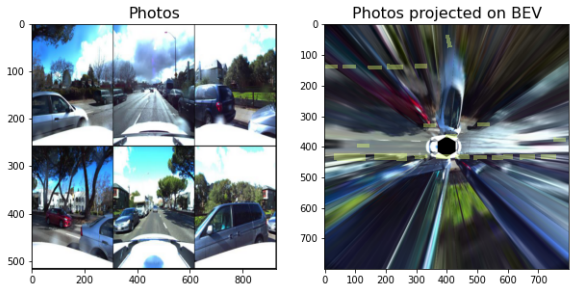


Figure 4. Left: raw camera views. *Right*: warped and glued camera views and target cars map (in light yellow)

U-Net U-Net (Ronneberger et al., 2015) has shown great performance on image segmentation tasks. U-Net consists of a contracting network that gradually reduces the size while increasing the number of channels and an upsampling module (with transpose convolutions) that uses the outputs of contracting network at different levels. For this task we use U-Net with a ResNet 34 encoder. We could not use U-Net directly as we did not have labels for photos, so we project raw photos to BEV as described above. To speed up the training, we downsample the combined image to a size of 200×200 before feeding into the model and upsample the output image back to 800×800. This architecture was tested only for BB task.

Post-processing After predicting the probability map, we use a threshold to get a binary map, then we convert the predicted blobs of pixels back to BB coordinates using OpenCV library. Additionally, we do post-processing of the predicted BBs: removing boxes that are too small and splitting up boxes that are unnaturally long compared to a car's size (results can be seen in Figure 1).

5. Results

Threat score is used for model evaluation on both tasks. Test TS corresponds to the scores received on our submissions.

MODEL	VAL TS	TEST TS
BASELINE - TWO ROADS	0.71	-
AE	0.79	-
VAE	0.84	0.76
MMD VAE	0.90	0.81

Table 1. Validation threat scores for our architectures on Road Layout Task

From Table 1, we can see that, as expected, MMD VAE outperforms the other models in the Road Layout prediction task and is our winning model with a test threat score of 0.81. Figure 1 shows prediction on a validation example. The model learns to predict two straight roads that appear in most of the train examples. It also memorizes crossroads, intersections, and parking lots, even when they are not visible in photos, because scenes are similar.

MODEL	VAL TS	TEST TS
BASELINE - LINE OF PARKED CARS	0.011	-
VAE	0.020	0.020
VAE + POST-PROCESSING (PP)	0.033	-
IMAGE WARPING + UNET + PP	0.080	0.072

Table 2. Validation threat scores for our architectures on Bounding Box prediction task

From Table 2, we can see that, the combination of image warping, U-Net and postprocessing outperforms the other approaches in the BB prediction task and is our winning model with a test threat score of 0.072. Adding postprocessing significantly improves the performance of VAE model from 0.020 to 0.033 on val. Example of predicted probability map and corresponding BBs map is shown in Figure 1. The model tends to predict long blobs, which are later split to separate BBs at postprocessing step.

6. Unsupervised Approaches

As we had 106 unlabeled scenes, we experimented on using self-supervised methods to pretrain the shared encoder we use in our networks.

First, we tried the pretext task called 'Shuffle and Learn' (Misra et al., 2016), the goal of which is to predict if given sequence of frames is in correct temporal order or not.

Second, we tried the SimCLR (Chen et al., 2020) contrastive learning approach, where we try to learn representations that bring similar samples close to each other in the latent space and push dissimilar samples far from each other.

From our experiments, we observe that using the pretrained weights from these tasks did not improve our performance.

7. Future Work

Adding rotation and rescaling as data augmentations should help reduce overfitting.

To improve the quality of predicted probability maps on both tasks, we can apply a Conditional Random Field (CRF) as post-processing as cited in (Lu et al., 2018). Using CRFs has proved to make predicted segmentation maps less noisy.

The temporal structure of the data can be exploited to make the predictions smooth across different frames. Also it can be utilized for unsupervised learning by modifying SimCLR to do Temporal SimCLR.

Since the model is performing poorly while the car is turning, we can try oversampling strategies such as SMOTE to make the model learn turns.

References

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. E. A simple framework for contrastive learning of visual representations. *ArXiv*, abs/2002.05709, 2020.

Lu, C., Dubbelman, G., and van de Molengraft, M. J. G. Monocular semantic occupancy grid mapping with convolutional variational auto-encoders. *CoRR*, abs/1804.02176, 2018. URL http://arxiv.org/abs/1804.02176.

Mani, K., Daga, S., Garg, S., Shankar, S., Krishna Murthy, J., and Madhava Krishna, K. Monolayout: Amodal layout estimation from a single image. *Winter Applications of Computer Vision (WACV)*, 2020.

Misra, I., Zitnick, C. L., and Hebert, M. Unsupervised learning using sequential verification for action recognition. *CoRR*, abs/1603.08561, 2016. URL http://arxiv.org/abs/1603.08561.

Riba, E., Mishkin, D., Ponsa, D., Rublee, E., and Bradski, G. Kornia: an open source differentiable computer vision library for pytorch, 2020. URL https://arxiv.org/pdf/1910.02190.pdf.

Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015. URL http://arxiv.org/abs/1505.04597.

Schulter, S., Zhai, M., Jacobs, N., and Chandraker, M. Learning to look around objects for top-view representations of outdoor scenes. *CoRR*, abs/1803.10870, 2018. URL http://arxiv.org/abs/1803.10870.

Zhao, S., Song, J., and Ermon, S. Infovae: Information maximizing variational autoencoders. *ArXiv*, abs/1706.02262, 2017.