DS-GA 1008 Deep Learning: Final Project Report

Kawshik Kannan [1] **Hsin-Rung Chou** [2] **Dipika Rajesh** [2]

Abstract

In this project, we are given a task to generate a top-down view of the surrounding area from 6 different camera views attached to the same car. We tried several strategies including Object Detection, VAE and GAN. We found that using deterministic model (composed of a decoder and an encoder) produced best result in both road map generation and bounding box prediction int test set.

1. Introduction

Recent research achievements in the interaction of Computer Vision and Deep Learning have revolutionized the field of autonomous driving, thanks to several novel techniques proposed in Object Detection, Semantic Segmentation and Generative Adversarial Networks. For this final project, we are given a task to generate a top-down view of the surrounding area from 6 different cameras view attached to the same car. There are 2 tasks for the generation of the top-down view, one is generating the road map layout, while the other is predicting the bounding box of the objects.

According to (Palazzi et al., 2017), we can map front-view image directly to the bird-eye view image, therefore, we build our approach based on this assumption.

2. Previous Work

2.1. Objection Detection

Object detection is an important and challenging problem in computer vision. These days object detection is classified in to two categories: One Stage Detectors and Two-Stage Detectors. Two stage detection tried predict the bounding box proposals separately and then work on the corresponding feature maps using layers like ROIPool, ROI Align etc. One stage detectors (Retinanet(Lin et al., 2017b), YOLO(Redmon et al., 2016)) combine all of this intuitively and train end-to-end models.

2.2. Map generation

(Mani et al., 2020) predicts the bird's eye view layout of the road and other traffic participants from a single color images. They represent scene layout as a multi-channel semantic occupancy grid, and leverage adversarial feature learning to generate plausible completions for occluded parts in image. Simiarly Monoccupancy (Lu et al., 2019) predicts the top down view using variational auto encoders. (Pan et al., 2019) predicts the top down semantic occupancy grid from 6 different perspecptive images similar to the task given to us.

2.3. Self Supervised Learning

Self-Supervised Learning aims to take advantage of the huge amount of unlabelled data available by relying on the information present in the pixels. (SIMCLR(Chen et al., 2020), PIRL(Misra & van der Maaten, 2019), MOCO(He et al., 2019)) have gained exceptional results using the model of contrastive learning and sometimes even fares better than supervised models by finetuning on fewer labels in comparison.

3. Our Approach

3.1. Self Supervised Pretraining

We use PIRL for training a resnet18 image feature extractor using the unlabelled dataset given. We use the standard training procedure and settings as mentioned in the paper(Misra & van der Maaten, 2019). This method helps in learning a generalized image feature extractor which we then finetune for the downstream tasks.

3.2. Map Generation

We experimented with three different architectures here. All three of our models differ after the fusion layers as depicted in Figure 1. The encoder consists of a resnet18 network which is used to extract the representations of all 6 views separately and then fuse them using various methods to get the final feature maps for the top down view.

3.2.1. Dense Projection

A fully convolutional network assumes a one to one mapping between each part of the input to each corresponding part of the output. We tried to induce this in the simplest way by using fully connected layers in the network. This allows the model to have the capacity to associate every pixel

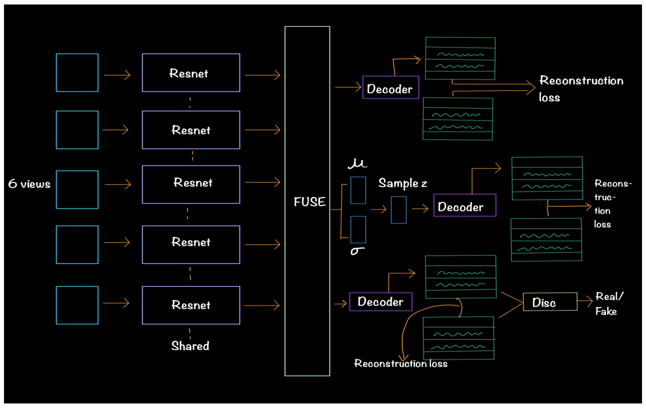

Figure 1. Architecture for Map generation tasks

in the top down view's feature map to every input pixel in the perspective views' feature map. We experimented with different combinations of projections to fully understand the requirements for fusion and view point transformation.

3.2.2. Fusion Mechanisms

We experiment with three different fusion techniques for comparison, concatenate and reduce, sum and mean. There are residual layers applied before and after the fusion to refine the features acquired. The fusion is then applied after refining the concatenated features from all six views.

3.2.3. The Deterministic Model

This is the simplest model we trained. It comprises of a standard decoder architecture comprising of upsampling layers and convolution blocks.

3.2.4. The Variational Auto Encoder

The next model we tried out was the variational auto encoder. This model projects the convolutional feature maps to linear layers for sampling a distribution which is regularized assuming a prior gaussian distribution by applying penalties on the relative entropy of the two distributions. The generative power in this model is necessary for modelling uncertainities (due to occlusions) while mapping the inputs to the given output.

3.2.5. The adversarial model

The current state of the art in generative models involve a lot of models with GAN based objectives. We follow the approach of Monolayout(Mani et al., 2020) to include a adversarial objective for consistency regularization of the road map distributions. It consists of a decoder on top of the top down view feature maps to give generated road map layout. This works as the generator for our model and tries to fool the disciminator by producing road maps which

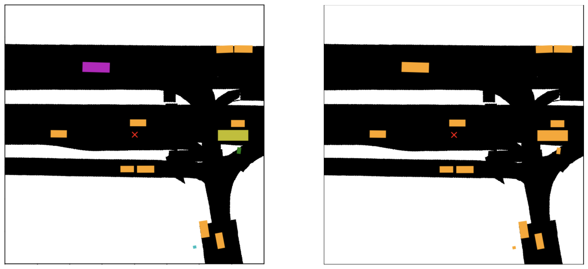

Figure 2. The semantic map(left) and object map(right)

are very similar to the given ground truth distribution. For the object detection, we then used OpenCV based contour detection and fitted a rectangle to get the final coordinates from the generated map layouts.

3.2.6. Map generation tasks

All of our final submission are a form of the above architecture and uses map layouts for either generating the output or deriving the output. Simply put we implement three different tasks, road map generation, semantic map generation (consists of 11 classes including the 9 given classes of objects in additional to the road and the background) and finally an object map (consists of 3 classes - obtained by clubbing all the 9 classes as a single class) Fig 4 to help make the semantic map task easier. We apply suitable final activations to get the final representations from the model. We experiment with different reconstruction loss objectives to understand the problem better.

3.3. Bounding Box Detection

We use RetinaNet(Lin et al., 2017b) as the base method for the bounding box detection. First we transformed the bounding box to image coordinates. The encoder is same as discussed above. Then we build a Feature Pyramid Network(FPN)(Lin et al., 2017a) on the top of the fusion architecture. Each level of pyramid is taking different scales of feature maps from ResNet and detecting objects at different scale. The detail of FPN generally follows(Lin et al., 2017a).

For the anchors, we change the scale and ratio of the anchors according to the distribution of the bounding box labels in dataset. We use anchors at three aspect ratios 0.2, 0.5, 1, 2 and scales 0.02, 0.05, 0.08. We also experiment on using the rotation information during training. Our initial angles ranges from -90° to +90° with an interval of 15°

We derive the angle of the box from the given box coordinates by the following steps with Figure 3:

$$\theta = tan^{-1}(\frac{Cy - bry}{Cx - brx}) \tag{1}$$

$$\vec{a} = \vec{fl} - C, \vec{b} = \vec{bl} - C \quad (2)$$

$$\gamma = cos^{-1}(\frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|}) \quad (3)$$

$$\beta = \frac{\pi - \gamma}{2}, \alpha + \beta = \theta \quad (4)$$

After we derive the angle from the bonding box label, we use angle-related IoU (ArIoU) (Liu et al., 2017) to compute the IoU and the loss.

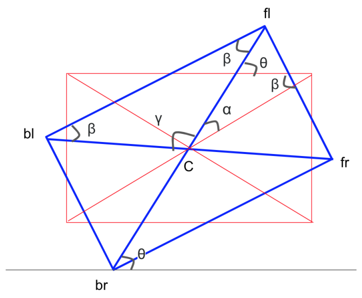

Figure 3. Rotation computation

4. Experiments

4.1. Implementation details

We randomly permute the dataset and split it with respect to scenes to get the val and the test set with four scenes each. We used the unlabelled set for self supervised pretraining. We used the Stochastic gradient descent optimizer with momentum for all models except the adversarial models where we used Adam. Dropouts were applied (p=0.5) after each dense layer if used. We stick with the resnet 18 architecture for all our experiments.

4.2. Bounding Box Detection

We conducted 4 experiments on the Bounding Box Detection. First we trained the model with the best settings for fusions and without rotation (which means we assume every bounding box is parallel to the coordinate). We got 0.0002 threat scores on the validation set. Then we transformed the bounding box label and added the rotation information during training, but we got 0 threat scores no matter what kind of fusion method we used (mean , sum, concatenate). We think we might have problems in the rotation part in our implementation.

4.3. Map Generation

4.3.1. Dense Projection

Applying dense projections before and for fusion gave decremental results. However dense projections after fusion gave slightly better results and training stability. This model did not tend to highly overfit in comparison to the fully convolutional networks. We believe this is because the fusion still requires the spatial information in the features to find correspondences between objects in different views. The view point transformation however seems to be more stable when we use dense projections which allows many to many mapping.

4.3.2. Fusion

In general mean worked better and gave 0.02 increase in the ts scores for validation. We feel that the majority of the computation required is fulfilled by the refinement layer applied prior to fusion and hence the different fusion tactics in turn did not give a serious difference in the final validation scores.

4.3.3. Deterministic Models

These set of models worked the best among the lot and was an integral part in all of our submissions. Simplicity played its part and this model generalized much better than every other model on the test set.

4.3.4. VAE

We found that the projection to linear layers for sampling proved to be decremental to the performance. The authors in the monolayout paper(Mani et al., 2020) draw a comparison with Mono-Occupancy(Lu et al., 2019) and conclude that it does not output sharp estimates of road boundaries by virtue of being a variational autoencoder (VAE), as mean-squared error objectives and Gaussian prior assumptions often result in blurry generation of samples.

4.3.5. Adversarial Model

Interestingly enough this model gave unbelievable performance in the validation scores (0.9 threat scores) but did not fare well when validated against the custom test set that we created. We know that Adversarial models are notoriously hard to train and our implementation might need further inspection to find out what's up.

4.3.6. Loss Functions

We experiment with three different reconstruction loss objectives DICE loss, Cross Entropy Loss and Mean Squared Error loss.

$$L_{mse}(y_c, \hat{y_c}) = \frac{1}{C}\sum_{c=1}^{C}(y_c - \hat{y_c})^2 \quad (5)$$

$$L_{CE}(y_c, \hat{y_c}) = -\frac{1}{C}\sum_{c=1}^{C}\hat{y}\log y \quad (6)$$

$$L_{Dice}(y_c, \hat{y_c}) = \sum_{c=1}^{C} 1 - \frac{2|y_c \cap \hat{y_c}|}{|y_c| + |\hat{y_c}|} \quad (7)$$

Mean Squared error did not perform well in any case. However the performances of DICE loss and Cross entropy loss were interesting. Although DICE loss turned out to be better than the other two loss functions, in the road map task, the difference was not significant however, without dice loss object map and semantic map generations did not work well at all. This arises due to a low number of pixels which are actually objects in the object detection task.

5. Results

The tables below are the results for experiments on validation set.

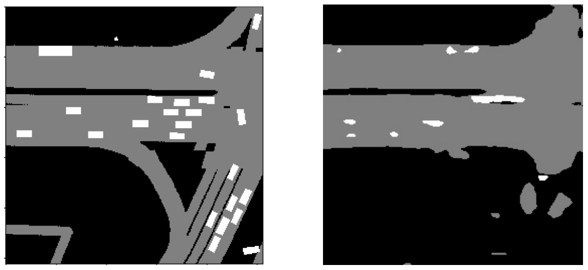

Figure 4. The ground truth(left) and predicted object map(right)

Table 1. Result of experimenting on different losses

Model	Loss Function	Threat Score
Deterministic Model	BCE	0.8226
	DICE Loss	0.8442
	MSE Loss	0.756

Table 2. Threat scores for road map generation

Experiment	Method	Threat Score
Model	Variational Model	0.482
	Deterministic Model	0.8442
	Adversarial Model	0.904
Fuse Type	Dense Fuse	0.6108
	Conv. Fuse	0.8442
Finetuning Method	PIRL - Frozen	0.790
	PIRL - FineTune	0.827
	None	0.812

Table 3. Threat scores for object detection models

Model	Map Type	Threat Score
Adversarial Model	Semantic Map	0.0034
Deterministic Model	Semantic Map	0.0038
Deterministic Model	Object Map	0.0044
RetinaNet w/o Rotation	-	0.0002

References

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125, 2017a.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017b.

Liu, L., Pan, Z., and Lei, B. Learning a rotation invariant detector with rotatable bounding box. *arXiv preprint arXiv:1711.09405*, 2017.

Lu, C., van de Molengraft, M. J. G., and Dubbelman, G. Monocular semantic occupancy grid mapping with convolutional variational encoder–decoder networks. *IEEE Robotics and Automation Letters*, 4(2):445–452, 2019.

Mani, K., Daga, S., Garg, S., Narasimhan, S. S., Krishna, M., and Jatavallabhula, K. M. Monolayout: Amodal scene layout from a single image. In *The IEEE Winter Conference on Applications of Computer Vision*, pp. 1689–1697, 2020.

Misra, I. and van der Maaten, L. Self-supervised learning of pretext-invariant representations. *arXiv preprint arXiv:1912.01991*, 2019.

Palazzi, A., Borghi, G., Abati, D., Calderara, S., and Cucchiara, R. Learning to map vehicles into bird's eye view. In *International Conference on Image Analysis and Processing*, pp. 233–243. Springer, 2017.

Pan, B., Sun, J., Andonian, A., Oliva, A., and Zhou, B. Cross-view semantic segmentation for sensing surroundings. *arXiv preprint arXiv:1906.03560*, 2019.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.