Week 9 Report

Doğaç Güzelkokar, Niklas Bubeck

August 2022

1 Week 9 Goals

We aimed for completing the testing of new architectures and the losses. Ultimately, starting the training after debugging our implementation of the architecture and the losses.

2 Implementation

2.1 Architectures

We currently have four variants of architectures. The original and three other shrunk versions of PointPWC[1].

2.2 Losses

We implemented the Chamfer distance loss with smoothing and laplace constraints from PointPWC. Additionally, we reimplemented the Cycle and Nearest Neighbour losses from Just Go With The Flow[2] in PyTorch. We observed that, PointPWC loss and JGWTF loss have different scales, and JGWTF loss require good initialization. Consequently, our current plan is to first train with PointPWC loss, and then fine tune with JGWTF loss.

3 Testing

3.1 Model size problem

We initially shrunk PointPWC from 7.8 million parameters to 3.3 million parameters. However, it still require 8 hours per epoch so we needed to shrunk the model further. Therefore we implemented a 3-level version of the original PointPWC, and a 3-level PointPWC with even less channels, which have around 2.7 million and 2 million parameters, respectively. However, we did not start any runs on the 3-level versions of PointPWC.

3.2 Overfitting

Before starting the training, we tried to overfit the original PointPWC and our implementations to see whether the model capability is sufficient for our task.

4 Training

4.1 Original PWC: Overfit to sequence 00 frames 0-10

As the first step we wanted to have a comparison between the original architecture and our adapted one with less parameters. Thus we trained both on the same dataset, with the same hyperparameters.

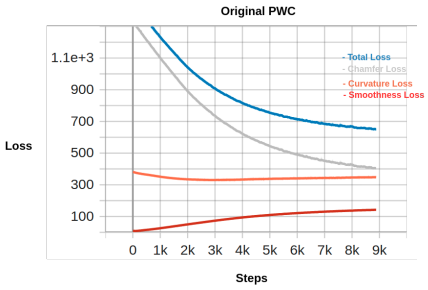

Figure 1: Loss functions for the original architecture

4.1.1 Findings

We found out, that the loss functions and thus also the training behaves very similar. Therefore, we concluded that our adaptions to the network seem to work out fine.

4.2 Linear Downsampling with no class removed: Overfit to sequence 00 frames 0-10, Loss: All PWC losses

With this first check, we wanted to see if we can overfit to a small set of data. Furthermore, we wanted to see the behavior of the loss functions and how it acts on the pointclouds.

A video of the network learning the flow can be seen Here (LRZ Sync+Share)

4.2.1 Findings

From the loss functions one can see that the curvature loss and the smoothing loss work as some kind of constraint in this case. Also we found that the network can estimate greater changes in position of the pointcloud but mostly struggles

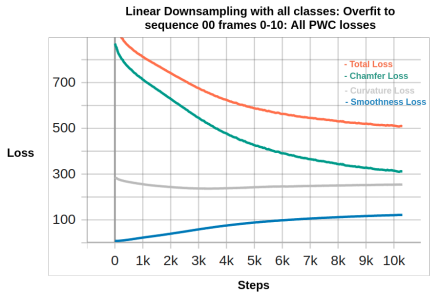

Figure 2: Loss functions for the adapted pwc architecture

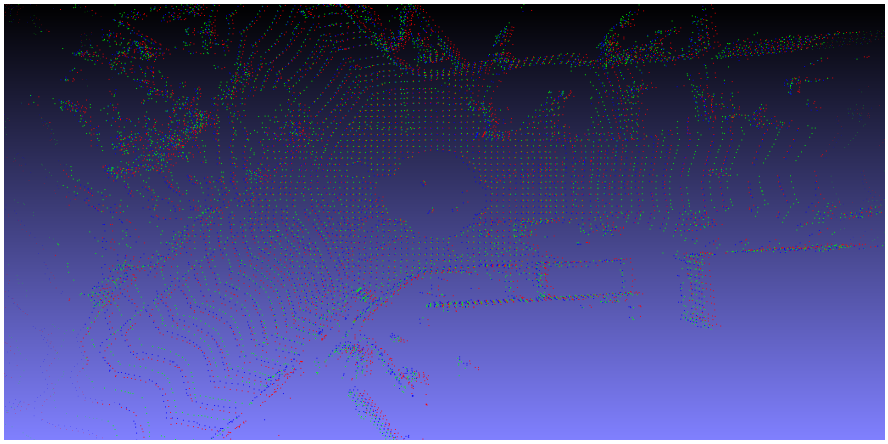

Figure 3: Endresult for the adapted pwc architecture

with points that classify as buildings, vegetation, or ground. As most of the points belong to one of those classes, they create a lot of noise.

4.3 Linear Downsampling with no class removed: Overfit to sequence 00 frames 0-10, Loss: Only Chamfer Loss

To see if we are right with our assumptions of the curvature and smoothing loss, we trained a network only using the chamfer loss.

4.3.1 Findings

From the image we can see, that the network became more flexible. The lines in the lower left that scanned the road are matching better and gained more flexibility to transform to the curve. But also we gained scenarios, in which the points start to clutter and do not represent the initial shape well. This is

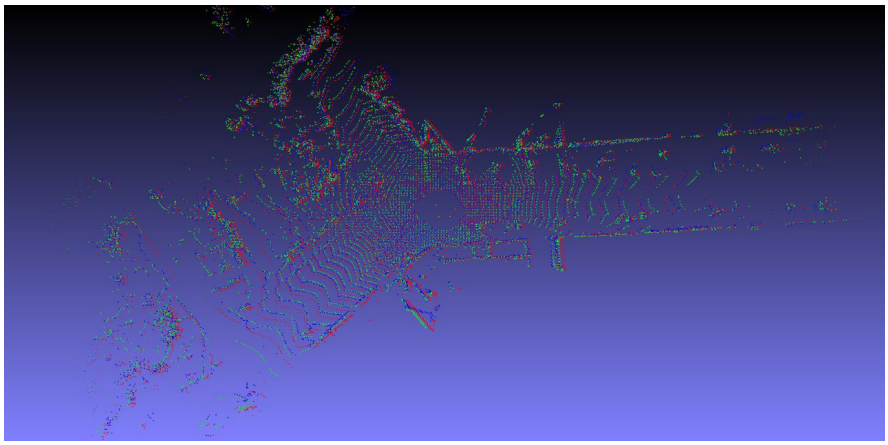

Figure 4: Endresult for the adapted pwc architecture using only the chamfer loss

the case for the wall in the lower mid, as well as for the building in the upper right corner. The Overall loss is a weighted sum of all other losses. So it can be controlled by seeing the weights as a hyperparameter.

4.4 Linear Downsampling with no class removed: Overfit to sequence 00 frames 0-10, Loss: JGWTF Cycle Loss

4.5 Adapted Downsampling with only dynamic classes (e.g. car, no matter if it moves or not), sequence 02, Loss: All PWC losses

To see how things behave when we remove unwanted classes/noise we set up a model that is only overfitted on dynamic classes. This gives us an additional problem which is size instability. Meaning depending on the scene we have a higher variance of points that we can actually use. As the pwc network can handle different sizes, but the creation of batches not (cant stack torches with different size). We evaluated two solutions for this problem after first trying to overfit. First we trained only with one batch and variable sizes. Second we used zero padding to scale all pointclouds to a size of 8192.

4.5.1 Overfit to frames 65-75

Overfitting on different sizes with batchsize 1, went great and resulted in almost perfect flow.
Looking especially at the curvature and smoothing loss, we can see that they are more stable and also contribute to a lower overall error. We conclude that non-dynamic classes are responsible for a lot of noise in the curvature and smoothing loss.

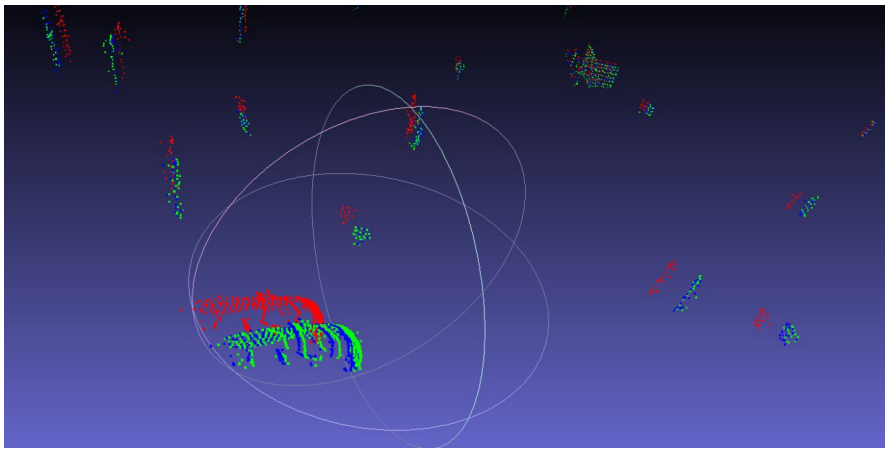

Figure 5: Endresult for the adapted pwc architecture using only the dynamic classes

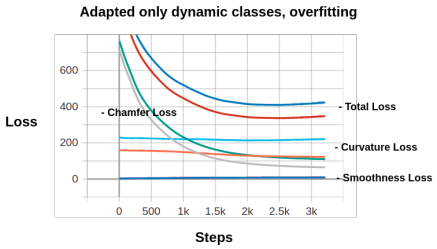

Figure 6: Loss functions for overfitting dynamic classes

4.5.2 Whole sequence 2, with no padding

When applying the idea from the overfit to the whole sequence, we had a lot of issues to generalize with the validation set.

Just using one batch is too instable for the training process of alot of data.

4.5.3 Whole sequence 2, with padding

With using padding, using the rtx2080 we can scale at least to a batch size of three. This gave some stability in the training process, until we also greatly overfitted to this case.

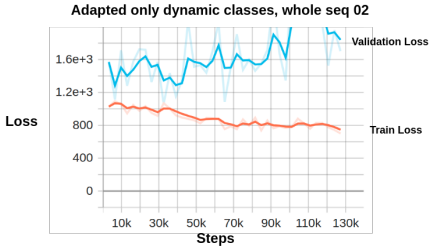

Figure 7: Loss functions for training on seq 02 and dynamic classes

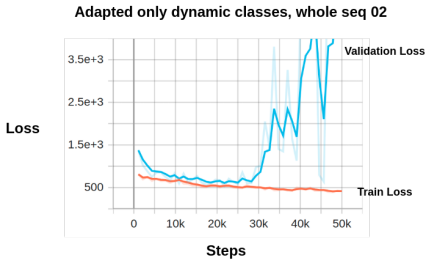

Figure 8: Loss functions for training on seq 02 and dynamic classes with using padding

5 Discussion

We figured out that we definitely need to remove non-dynamic classes in order to have good results. Furthermore, we make more sense of the influence of our data. Removing non-dynamic classes we are left with a sparse matrix that doesnt represent the scanned space well anymore. Leading to fast overfitting. Therefore, we need to train on a lot of data covering all the varying cases.

6 Next Steps

We want to find a valid preprocessing which gives us enough points to always sample 8192 points. Without introducing too much noise. We then start training on even more data to capture more scenarios and counteract the overfitting.

We then stack everything together with the DSNet.

References

[1] W. Wu, Z. Wang, Z. Li, W. Liu, and L. Fuxin, "Pointpwc-net: A coarse-to-fine network for supervised and self-supervised scene flow estimation on 3d

point clouds," *arXiv preprint arXiv:1911.12408*, 2019.

[2] H. Mittal, B. Okorn, and D. Held, "Just go with the flow: Self-supervised scene flow estimation," 2019. [Online]. Available: https://arxiv.org/abs/1912.00497