Master Thesis

Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Cybernetics

Active Learning for Semantic Segmentation in 3D Point Cloud Sequences

Petr Šebek

Supervisor: prof. Ing. Tomáš Svoboda, Ph.D.
May 2023

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name:	**Šebek Petr**	Personal ID number:	**483508**
Faculty / Institute:	**Faculty of Electrical Engineering**		
Department / Institute:	**Department of Cybernetics**		
Study program:	**Cybernetics and Robotics**		

II. Master’s thesis details

Master’s thesis title in English:

Active Learning for Semantic Segmentation in 3D Point Cloud Sequences

Master’s thesis title in Czech:

Aktivní učení pro sémantickou segmentaci sekvence mračen 3D bodů

Guidelines:

Data annotation is needed to achieve state-of-the-art performance on detection tasks. This calls for an effective and scalable labeling pipeline. The nature of the data in robotic perception allows for exploiting numerous constraints to improve the annotation process, such as the spatial-temporal consistency of the objects in the recording sequence. The goal of the thesis is to explore current methods of active learning and preferably design own robust method to be used in the data engine framework with its user interface for further application. The main focus will be dynamic and sparse object categories in large-scale point clouds, as the objects are still out of the scope of the current approaches [1,2,3]. The steps can be summarized as follows:
1) Get familiar with the publicly available datasets for point clouds and the active learning branch of machine learning. Implement active learning baselines ([1] for example).
2) Discuss pitfalls of current methods and design extensions that deal with drawbacks on semantic categories.
3) Verify the method by training neural networks for perception with a low amount of data and human-in-the-loop annotation feedback.
4) Compare your results and discuss the suitability for specific classes, data, and sensors.
5) Program user interface for a fast annotation process connected with your active learning framework.

Bibliography / sources:

[1] Tsung-Han Wu, Yueh-Cheng Liu, Yu-Kai Huang, Hsin-Ying Lee, Hung-Ting Su, Ping-Chia Huang, Winston H. Hsu; ReDAL: Region-based and Diversity-aware Active Learning for Point Cloud Semantic Segmentation, Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 15510-15519
[2] Liu, M., Zhou, Y., Qi, C.R., Gong, B., Su, H., Anguelov, D. LESS: Label-Efficient Semantic Segmentation for LiDAR Point Clouds. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds) Computer Vision – ECCV 2022. ECCV 2022. Lecture Notes in Computer Science, vol 13699. Springer, Cham. https://doi.org/10.1007/978-3-031-19842-7_5
[3] Chaplot, D.S., Dalal, M., Gupta, S., Malik, J. and Salakhutdinov, R. SEAL: Self-supervised Embodied Active Learning. In NeurIPS 2021.

Name and workplace of master’s thesis supervisor:

prof. Ing. Tomáš Svoboda, Ph.D. Vision for Robotics and Autonomous Systems FEE

Name and workplace of second master’s thesis supervisor or consultant:

Date of master’s thesis assignment: **03.02.2023** Deadline for master's thesis submission: **26.05.2023**

Assignment valid until: **22.09.2024**

prof. Ing. Tomáš Svoboda, Ph.D.	prof. Ing. Tomáš Svoboda, Ph.D.	prof. Mgr. Petr Páta, Ph.D.
Supervisor’s signature	Head of department’s signature	Dean’s signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student's signature

Acknowledgements

I would like to give thanks to my supervisor, Tomáš Svoboda, for all his advice and for guidance of this thesis.

Declaration

Abstract

We propose an extension of the ReDAL[15] active learning method. Our extension makes annotations of dynamic objects in 3D LiDAR point cloud easier and more effective. It can create instances of objects in point cloud and link them through time, without any manual annotations. Therefore, the annotator is able to annotate an object in a whole sequence just by one action. Thanks to our pipeline, we can improve mIoU by 1.67 percentage points over ReDAL[15], i.e. from 92.1% of performance of fully supervised learning to 94.7%, with the same amount of manually annotated data. For our extension, we create a user interface for fast labeling that uses the output of our extension.

Keywords: active learning, LiDAR data, deep learning

Supervisor: prof. Ing. Tomáš Svoboda, Ph.D.
ČVUT,
Resslova 307/9,
Praha

Abstrakt

Vytvořili jsme vylepšení metody ReDAL[15], která se zabývá aktivním učením. Naše vylepšení zjednodušuje a zefektivňuje anotace dynamických objektů v 3D mračnech bodů. Dokáže vytvořit instance dynamických objektů a propojit je v čase a to bez potřeby jakékoli manuální anotace. Anotátor tedy může anotovat objekt ve všech časech jednou akcí. Pomocí naši metody jsme zlepšili výsledek učení o 1.67 procentního bodu, oproti ReDALu[15] a to z 92.1% výsledku plně supervizovaného učení na 94.7%, při stejném množství manuálně anotovaných dat. Pro naše rozšíření jsme také vytvořili uživatelské rozhraní pro rychlé anotování, které využívá výsledky naší metody.

Klíčová slova: Aktivní učení, LiDARová data, hluboké učení

Překlad názvu: Aktivní učení pro sémantickou segmentaci sekvence mračen 3D bodů

Contents

Figures

Tables

Chapter 1

Introduction

Training of neural networks requires a large number of training frames with a high variety. Today, it is not a problem to capture a large number of frames with a wide variety, but the bottleneck is creating annotations for these large datasets. Annotations are usually created manually, which is time-consuming and costly, especially for 3D point clouds. One way to handle this problem is to use active learning[10], which focuses on minimizing the time required for annotations while trying to maintain the high performance of the neural network. The main idea of active learning is to select for annotation only frames (or parts of frames) that have the highest probability of benefiting from training the most. The suitability for annotation is driven by a value called *information gain* or *information score*, and it is estimated by a wide range of criteria, which we will mention later. Active learning is a cyclic algorithm that consists of the following steps.

1. The training dataset is initially divided into two parts: labeled dataset (D_L) and unlabeled dataset (D_U). The unlabeled dataset is significantly larger (90 to 99% of the entire training dataset).
2. The neural network is trained on D_L.
3. The trained model is used to determine which frames from D_U (or parts of frames) should be labeled according to specific criteria for the computation of the information score.
4. The selected data are labeled by the annotator and moved from D_U to D_L. Annotations are usually manually made by humans.
5. The neural network is then fine-tuned on D_L.

Steps 2 to 5 are executed multiple times to ensure that only high-potential data are annotated. This cycle is shown in figure 1.1.

There are many different approaches to computing the information score. From simple ones like active learning with least confidence (AL-LC)[14], margin sampling (AL-MS)[14], and entropy (AL-Entropy)[14] to more complex ones such as LiDAL[6], LESS[8], and ReDAL[15].

Selection by least confidence[14] picks the frame with the smallest prediction confidence. Margin sampling[14] looking for the frame, which has the smallest

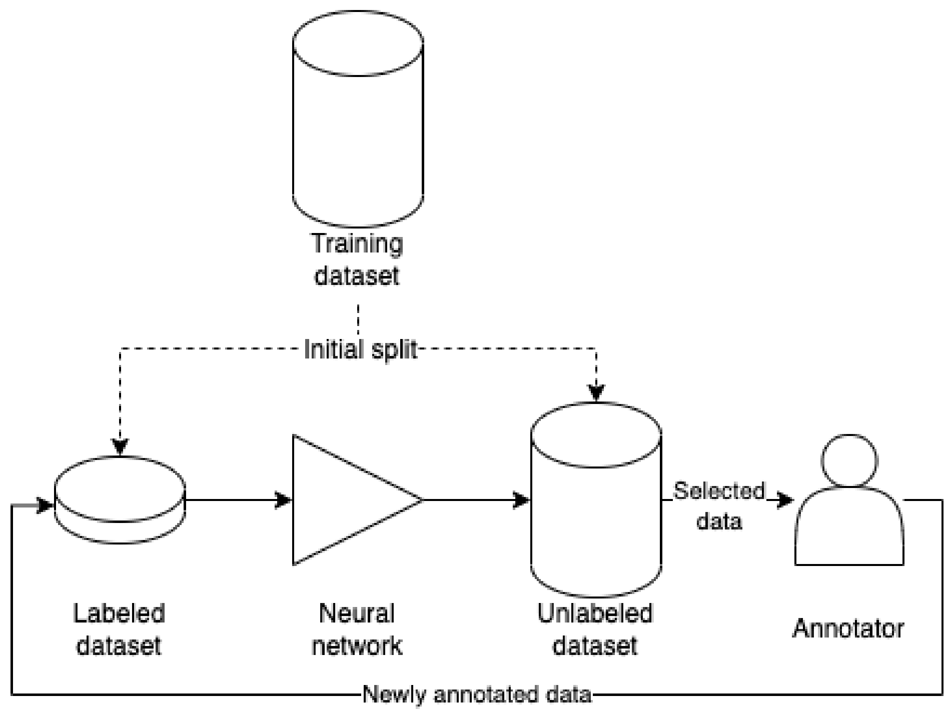


Figure 1.1: Main idea of active learning

gap between two most probable classes from a neural network perspective. Labeling using entropy[14] (sometimes called softmax entropy), which is a widely used criterion, means that frames, which have the highest entropy in frame prediction probabilities, are selected for annotation.

ReDAL [15] changes the "labeling unit" from the entire frame (point cloud) to regions (part of point cloud). The underlying idea is that not every part of the point cloud contributes equally to training. The possible information gain of each region is computed by multiple criteria. The first criterion is softmax entropy (similar to AL-entropy[14]), second is color discontinuity. This criterion is used only for datasets that include information on the color of points, e.g. S3DIS[1]. This criterion benefits regions that contain large color differences, which are usually semantically discontinuous. The last criterion is structural complexity, which has a higher response on edges and corners, therefore it also benefits semantically discontinuous regions. This method also penalized scores of similar regions.

LiDAL[6] proposes two novel criteria. First is inter-frame divergence and second is inter-frame entropy. Inter-frame divergence computes inconsistency of the object predicted from different viewpoints. This is evaluated on the augmented frame (by global translation, rotation, scaling, and jittering), but also on the neighboring frames. Matches between points in different time frames are obtained just by the odometry transformation, but there is no special method for points that belong to dynamic objects, where this method does not have to work properly. Inter-frame entropy works similarly to softmax entropy, but it works with mean of all prediction distributions that are used to compute inter-frame divergence (prediction on augmented point cloud and neighboring frames). Also, they implement the self-training method based on pseudo-labeling. To select which parts of the dataset use pseudo-labels, they use the same two criteria as are mentioned above, but they are looking for regions, which have the lowest values in this criteria.

LESS [8] aims to divide the point cloud into semantically pure components. For pure components annotator needs to annotate just one point and to the rest of the component is label propagated. However, for a semantically impure, the annotator is supposed to label one point for each semantic label that is located in the component. The presence of different semantic labels among the remaining points in the component, which do not match the annotated ones, is forbidden and results in penalization during training through the loss function.

We create an extension of the ReDAL[15] method. Our extension focuses on an easier annotation of dynamic objects, where reliable predictions are most crucial in the real application of autonomous driving, and any modern method does not provide any special treatment for these objects. Our pipeline creates semantically pure regions for each dynamic object in the point cloud and links regions of the same object through a point cloud sequence. The annotator can then annotate the object in a complete sequence with just one action.

Chapter 2
Method

Our method is an extension of the ReDAL method[15], which is oriented to make semantic annotations of dynamic objects in the 3D point cloud easier and more efficient. It gives a proposal of regions that contains only a single object in different time frames. The annotator can then annotate this object in a whole sequence by one action. An example of a semantically annotated LiDAR 3D point cloud is shown in figure 2.1.

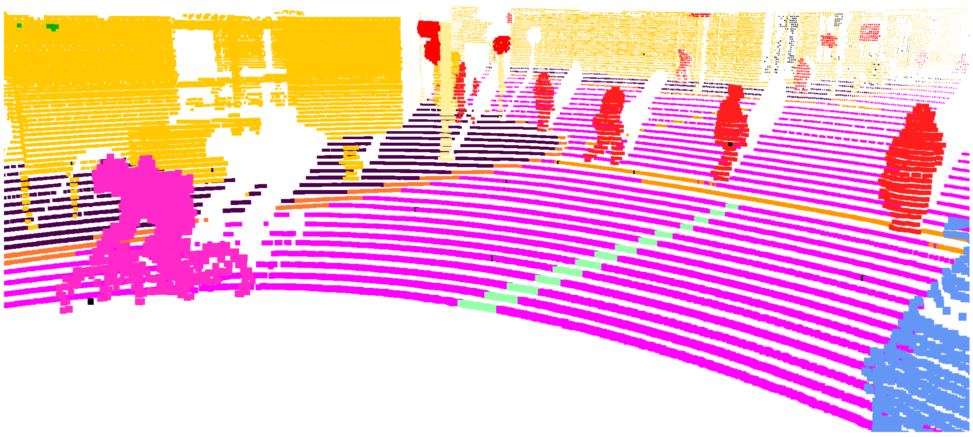


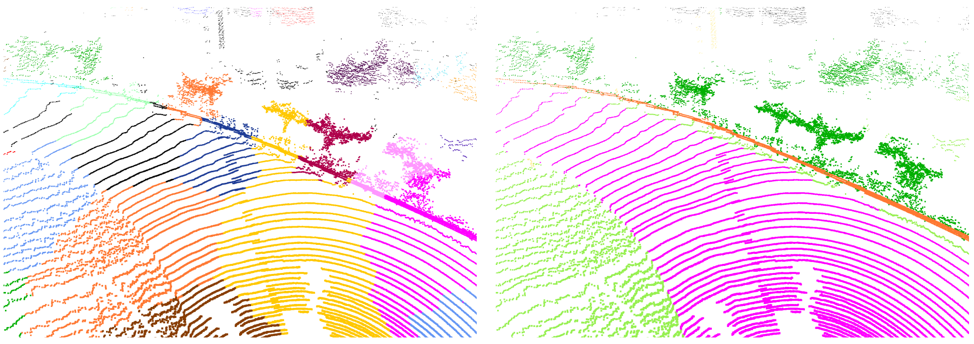
Figure 2.1: Semantically labeled LiDAR 3D point cloud

Our codes are available at:
`https://github.com/sebekpe1/ReDAL-extension-by-motion-flow/tree/main`.

2.1 ReDAL overview

ReDAL[15] divides the point cloud into regions using VCCS[9], which is an unsupervised over-segmentation method. However, this method struggles to create regions that are semantically pure (i.e., regions that contain only points with just a single semantic label) on complex outdoor point clouds, as shown in figure 2.2.

The final region information score, computed for each region separately, is



(a) : Regions created by VCCS[9] **(b) :** Semantic labels

Figure 2.2: Example of not ideal behaviour of VCCS[9] e.g. the yellow region in subfigure 2.2a contains three semantic labels (2.2b).

the weighted sum of three main components, as in equation (2.1).

$$\varphi_n = \alpha H_n + \beta C_n + \gamma S_n, \tag{2.1}$$

where H_n stands for softmax entropy (2.2), C_n is color discontinuity (2.3), and S_n means structural complexity (2.4). Parameters α, β, and γ are constants that the authors set. α and γ have the same value for all datasets in the ReDAL[15] paper (SemanticKITTI[3], S3DIS[1] and Scannet[4]). α being 1 and γ being 0.05. β is set on SemanticKITTI[3] to 0, because this dataset does not include information about the color of the points, but for the rest of the datasets is set to 0.1.

Softmax entropy measures how uncertain the model is in its prediction by following a formula (2.2):

$$H_n = \frac{1}{|R_n|} \sum_{i \in R_n} -P_i \log(P_i), \tag{2.2}$$

where R_n represents set of points in n^{th} region, $|R_n|$ is number of points in the region and P is softmax probabilities of region points.

Color discontinuity measures how rapidly the color changes in the region. It is computed by equation (2.3).

$$C_n = \frac{1}{k|R_n|} \sum_{i \in R_n} \sum_{j \in d_i} ||I_i - I_j||_1, \tag{2.3}$$

where I stands for color intensity, and k is number of k-nearest neighbor between which the color discontinuity is computed.

Structural complexity measures how much is the region geometrically diverse by equation (2.4).

$$S_n = \frac{1}{|R_n|} \sum_{i \in R_n} \sigma_i, \tag{2.4}$$

where σ is surface variation, which is computed by equation (2.5).

$$\sigma_k = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2}, \tag{2.5}$$

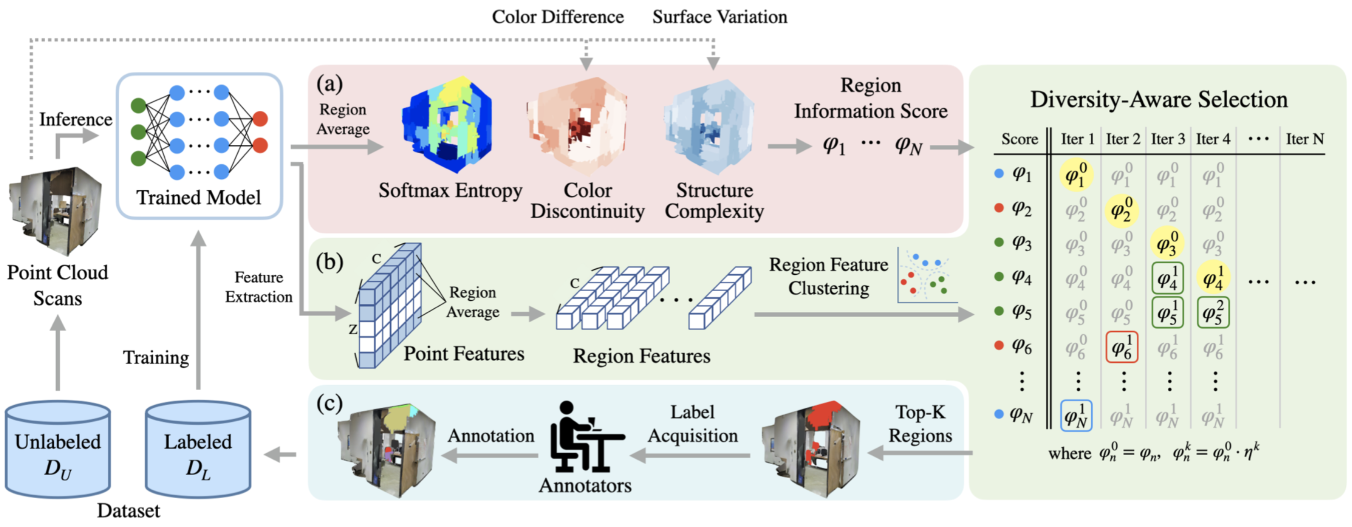


Figure 2.3: ReDAL[15] method overview

where λs are eigenvalues of 3×3 covariance matrix of 50 nearest neighbors of point k and $\lambda_0 \leq \lambda_1 \leq \lambda_2$.

Final adjustment of region score is done by penalizing score of similar regions in order to prevent manual annotation of multiple similar regions (e.g. the same place in different time frames), when possibly annotating one could be sufficient. To determine which reasons are similar to each other, regions are described by their features before the final classification layer in the neural network. These features of all regions are then clustered by the k-mean algorithm. Information scores of regions in the same cluster are then penalized (region with highest information score is not, the second one is penalized one time, third is penalized two times, etc.).

$$\varphi_n^* = \varphi_n \cdot \eta^{k-1}, \tag{2.6}$$

where φ_n^* is region final score, φ_n is region information score, η is decay rate (set to 0.95) and k is region information score position in k-mean cluster (decreasing).

Then the top-k numbers (based on the final score) of the regions are humanly annotated. Parameter k is in implementation determined by how many points you want to annotate, for example, k is the number of top regions, where the sum of points is equal to 1% of all points in the training part of the dataset.

The overview of the ReDAL[15] method is shown in figure 2.3.

2.2 Semantically cured regions linked in time for ReDAL

From our point of view, the biggest weakness of the ReDAL[15] method is that its regions are not semantically pure. Pure region contains points that belong to only one semantic class (as shown in figure 2.2). Due to this, the annotator still has a lot of work with annotating the region because the annotator needs to separate different objects in the region to be able to annotate them separately. We want to help with this problem by creating an extension that will be able to obtain semantically pure regions. We are focused on creating regions that would belong to dynamic objects in the point cloud sequence for two reasons. The first is that we could use one labeling action to annotate an object in all time frames. The second reason is that reliable detection of “dynamic classes” is, from our perspective, one of the most critical tasks in autonomous driving.

Our extension adjusts the VCCS[9] regions in such a way that it creates a new region for each dynamic object in the point cloud (visualized in figure 2.4) and links them through different time frames (visualized in figure 2.5). So when the annotator selects the label for the region in any time frame, which should be just one “click” due to the semantic purity of these new regions, the label will be propagated to all linked regions. In order to address this problem we devided it into tree steps:

- Create instances for dynamic objects in all time frames. Details for method that uses ground truth annotations in subsection 2.3.1 and with use of motion flow model in subsection 2.3.2

- Link these instances through time frames. Details for motion flow in subsection 2.4

- Adjust regions, which contain dynamic object, by separating dynamic object to new region. This enables the possibility to link regions in the same way as dynamic objects through time, which can provide an easy way for annotator to label dynamic object in a whole sequence in one action.

2.3 Creating instances for dynamic objects

Obtaining instances of dynamic objects is crucial for our method. Therefore, we conducted experiments where instances are based on ground truth (details in subsection 2.3.1). This will be the upper bound of the performance of our method. Then we run experiments that use instances that are created by our method and do not need any manual annotations (details in subsection 2.3).

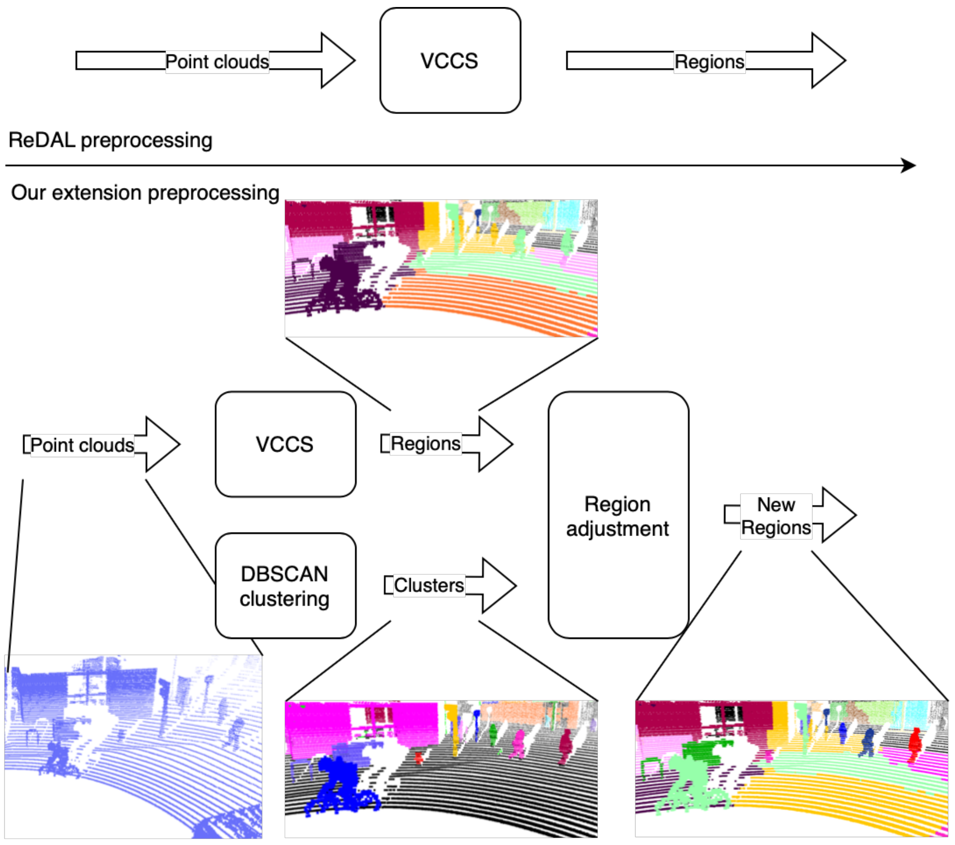


Figure 2.4: Difference between ReDAL[15] region creation and our extension

2.3.1 Instances based on ground truth

SemanticKITTI[3] contains per-point instance annotations, which are the same for a single object through the whole sequence. So we use all these annotations, which belong to the dynamic class (semantic labels with indexes 252 and higher, e.g., moving car, moving person, etc.).

Waymo Open Dataset[11] does not have per-point instance annotations. However, it has bounding boxes for vehicles, pedestrians, and cyclists. All bounding boxes also contain the object ID, which is fixed for each object in all time frames. We are able to decide if an object is dynamic or static based on its bounding box location in different time frames and odometry between frames (which is provided in the dataset). All objects that move at least by 0.5 m are denoted as dynamic. We obtain per-point instances of the dynamic object by cutting its bounding box from the frame (point cloud) and then filtering points based on their semantic label.

2.3.2 Instance based on motion flow

We used a neural network, which is trained to estimate motion flow to determine which points are dynamic and where they are in the next frame. We also needed an approach that is self-supervised, i.e., a neural network

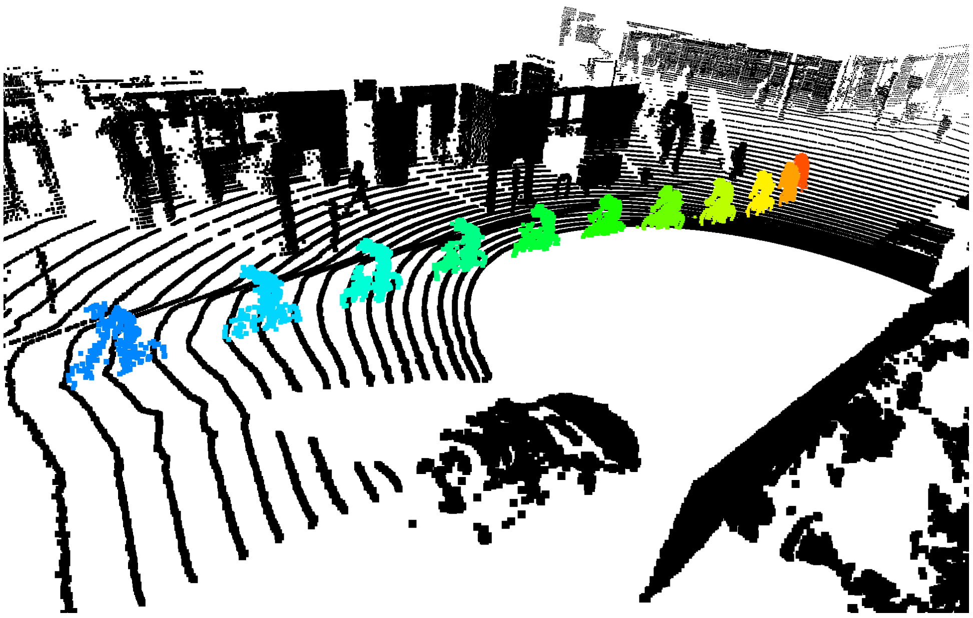


Figure 2.5: Visualization of linked regions of a dynamic object (cyclist). Background points (black), regions from different time frames(red to blue).

does not need any annotation for training. Therefore, we use SLIM[2]. For optimal run of SLIM[2], we needed to satisfy a few conditions for input point cloud:

- Point cloud should not contain any ground points. We address this problem in subsection 2.3.2.1
- Point cloud should be limited in the XY plane (horizontal) to square 70 m by 70 m, so any point which has a larger absolute value of x or y than 35 m is masked.

2.3.2.1 Ground masking

For the removal of ground, we consider two approaches. First is naive, it is just cropping point clouds by setting the minimal value of the points Z coordinate. We set the minimum value as the value that corresponds to the level, which is 20 cm above ground in the position of the ego (LiDAR). This method creates usable results on the SemanticKITTI dataset[3], however, the ground in the Waymo Open Dataset[11] is much more diverse, so a different approach needs to be used. The second approach is the use of Patchwork++[7]. Patchwork++ is a method that segments points into two classes: ground or non-ground. This method performs well on all types of surfaces, e.g. road, sidewalk, curbs, grassy hills, etc. The output of this method can be seen in figure 2.6.

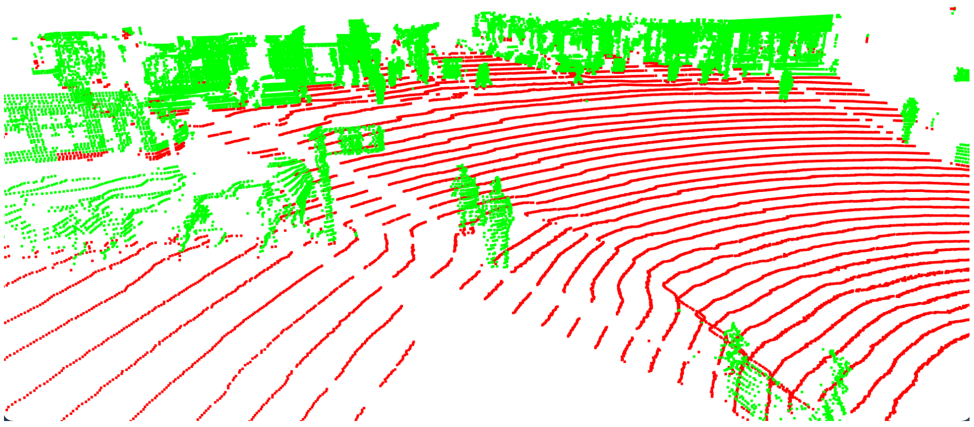


Figure 2.6: Segmentation of ground points by Patchwork++[7]. Ground points (red), non-ground points (green).

2.3.2.2 Motion flow computation

SLIM[2] is an unsupervised method that uses two sequential frames (point clouds) t_0 and t_1 and predicts the motion of each point in the frame t_0. This model simultaneously computes two flows (dynamic and static). For the computation of dynamic flow, the model divides the point cloud into pillars and for each pillar the flow is computed separately by RAFT[13]. Static flow is computed on the whole point cloud by using weighted Kabsch's algorithm. Finally, the neural network determines which flow to use on each pillar by predicting whether the points in the pillar are static or dynamic. An example of the output can be found in figure 2.7.

2.3.2.3 Instance creation

SLIM[2] gives us its prediction of flow for each point for the next frame (from t_0 to t_1) and a prediction of whether the point is static or dynamic. However, these static/dynamic predictions are noisy as can be seen in figure 2.8.

To determine whether the instances are static or dynamic, we used DBSCAN[5], which divides the masked point cloud into instances/clusters. Visualization of these clusters is shown in figure 2.9

For each DBSAN[5] cluster, we decide if it is dynamic based on SLIM[2]. To denote the cluster as dynamic, it must meet at least one of the two criteria. The first is that at least 50% of the points that create the cluster are predicted to be dynamic. The second one is that some dynamic cluster from the previous frame is fitted to this one. The second criterion is for cases where the object was not moving in all frames in a sequence, e.g. a pedestrian walks to crosswalk and waits for a green light. Also, to cover inverse cases where, e.g. at the start of the sequence, a pedestrian standing and then starts moving, we evaluated the second criterion later once again but in backward

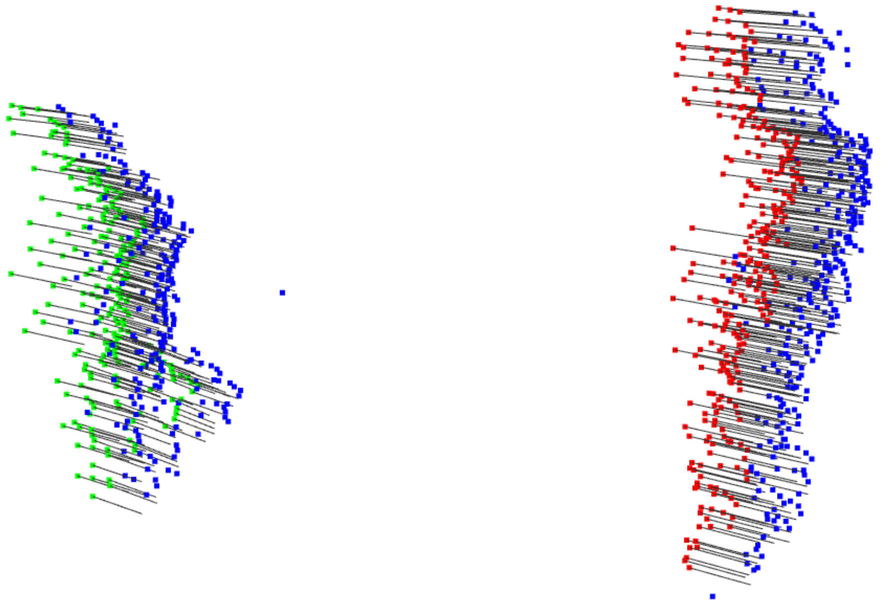


Figure 2.7: Prediction of SLIM[2]. Dynamic points of t_0 (red), static points of t_0 (green), points of t_1 (blue), flow (black lines).

order of the frames. The output after this step is visualized in figure 2.10.

2.4 Instance matching

Matching instances that are based on ground truth is simple because, as mentioned above, SemanticKITTI[3] contains instance indexes which are fixed for each object through the whole sequence. Waymo Open Dataset[11] contains bounding box ID, which can be used in the same way as indexes in SemanticKITTI[3].

Matching instances created in subsection above 2.3.2.3 are more complicated. We match the dynamic clusters with the clusters in the next frame (t_1). It is done by moving all the points in the t_0 cluster by their flow. Then we are looking for all points in t_1 that are maximally 10 cm from any moved point from t_0. Then we are looking for a cluster in t_1, that contains the most points in t_1, which satisfy the mentioned criterion. This cluster is then matched with the one from t_0 if their volumes (in terms of the number of points) are similar (we set the threshold as $\pm 40\%$ of the number of points in t_0 cluster). To prevent any false matching and isolate cases where e.g. a pedestrian comes to the traffic light and DBSCAN includes them in the same cluster, but it has one drawback, which is that clusters of one real object are not matched together if the occlusion of the object highly varies between frames.

The time difference between frames is 0.1 s, but not every frame in Waymo Open Dataset[11] contains semantic labels. Usually every 5^{th} frame (time difference 0.5 s) contains semantic labels. However, this gap could be even

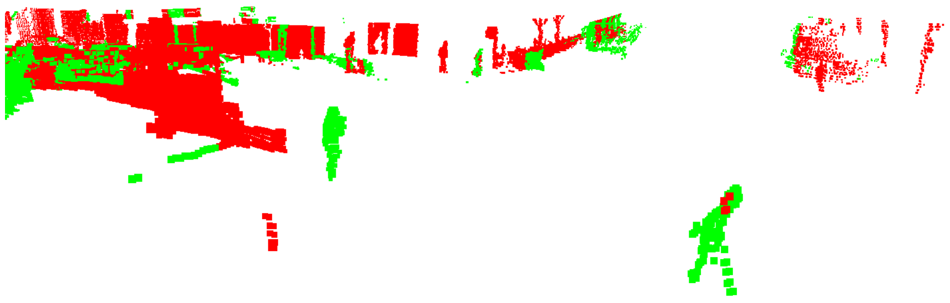


Figure 2.8: Dynamic (green) vs static (red) point SLIM[2] prediction

55 frames (5.5 s) which is too large for a motion flow model. Therefore, we applied this pipeline to all frames, but in order to have fair comparison, we used results only on originally labeled frames. Visualization of matched frames can be seen in figure 2.11

2.5 Annotator simulation

In order to conduct our experiments, we needed to simulate human labeling on our extension. For that we created the assumption that the annotator would label our new regions if they are at least 90% semantically pure and the region does not contain more than 99 points that have a different semantic label. If these two conditions are met, the annotator would assign the same semantic label to all points in the region. If not, the proposed region would not be used.

2.6 User interface

For real application, we design the user interface, which shows the annotator all linked clusters from different time frames in a single visualization and a visualization of the cluster with the most points (for easier recognition of the object). The annotator then has the option to exclude clusters from time frames, where they are not semantically pure, and then select semantic label to correct clusters. Visualization of this interface can be seen in figure 2.12.

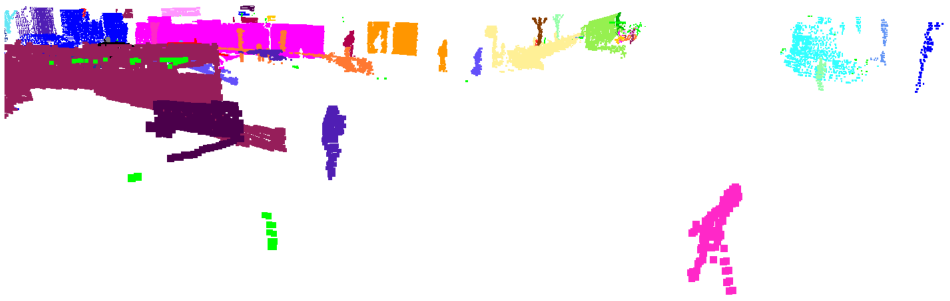


Figure 2.9: DBSCAN[5] clusters. Points in the cluster have the same color.

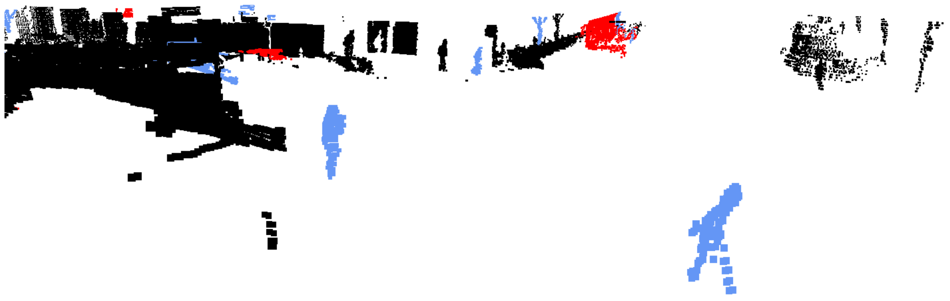
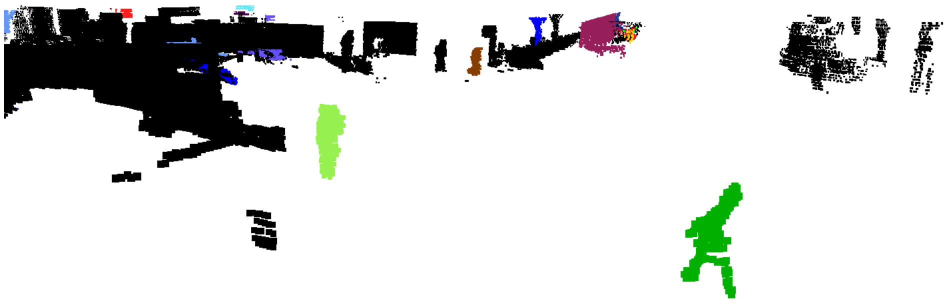
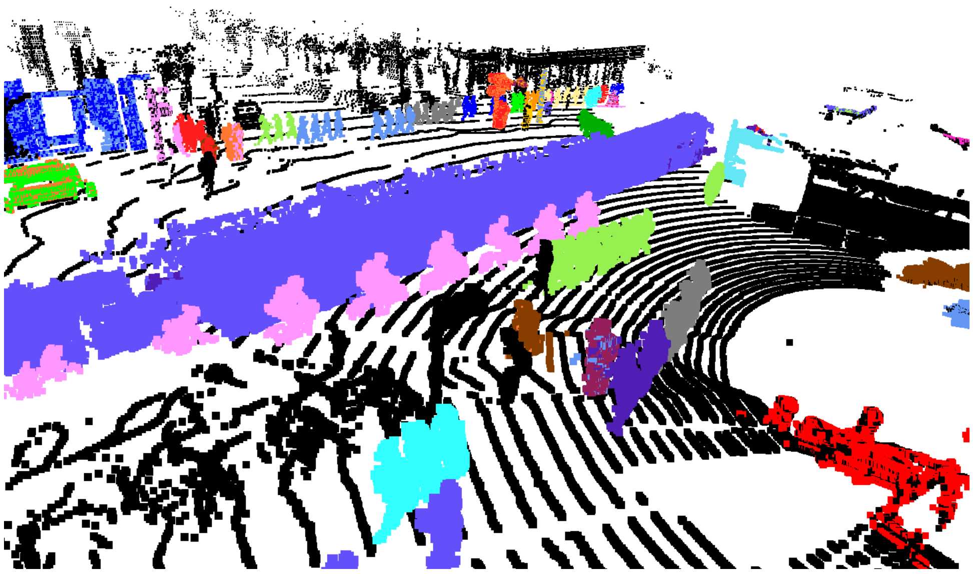


Figure 2.10: Dynamic points prediction adjusted by DBSCAN clusters. Dynamic instance which is "new" (red), dynamic instance that is also in the previous frame (blue), and static points (black).

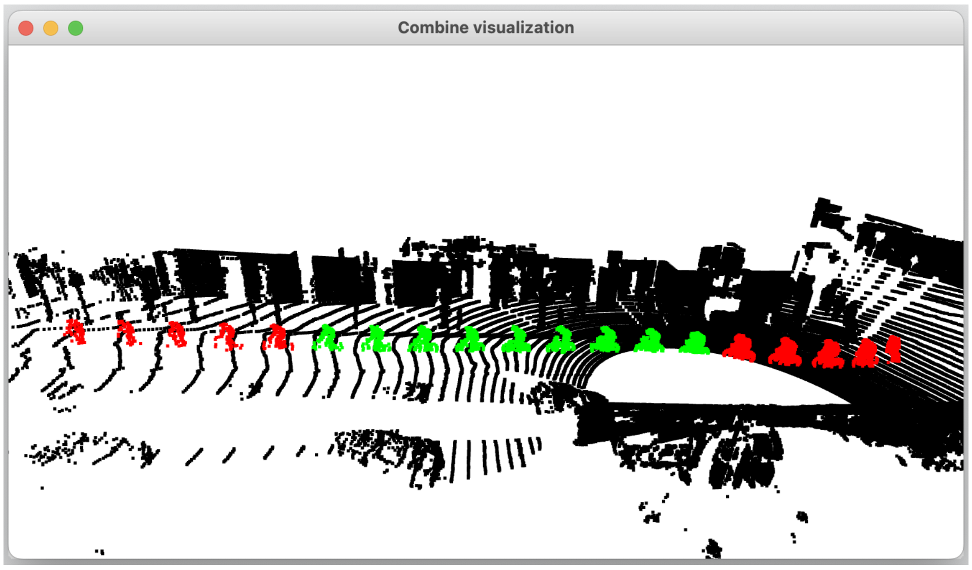


(a) : Matched instances between two frames which have a 0.1 s time difference. Static points (black), dynamic instances (coded by color).

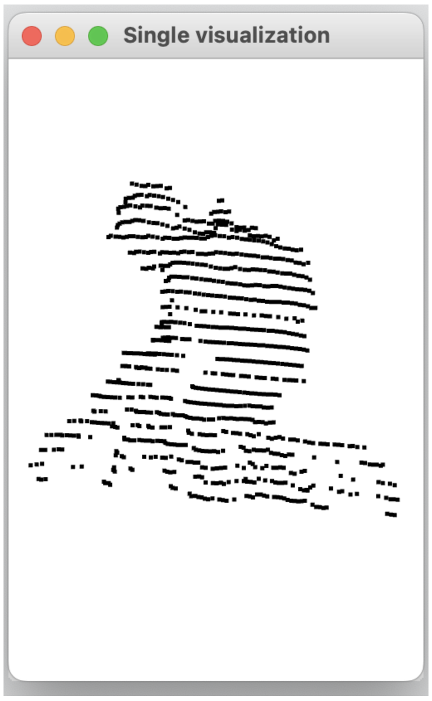


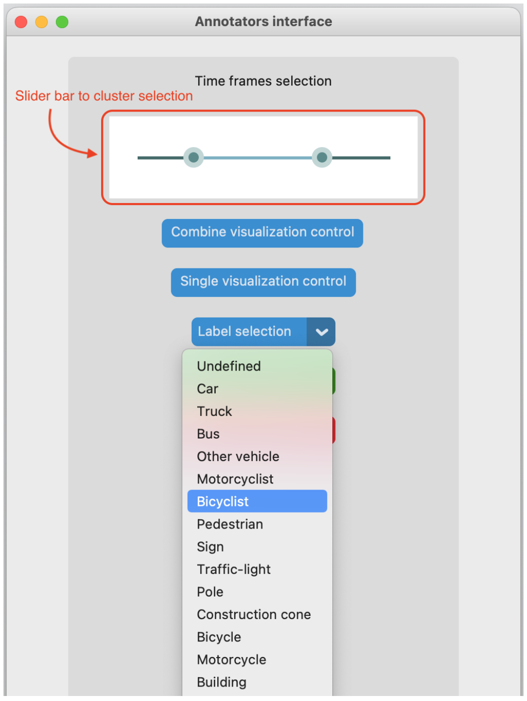
(b) : Accumulated dynamic instances throw sequence. Static points (black), dynamic instances (coded by color).

Figure 2.11: Dynamic instances created by motion flow



(a) : Visualization of all linked clusters on single background. Excluded clusters (red), selected clusters (green), background (black).





(b) : Visualization of single cluster

(c) : Labeling window of user interface

Figure 2.12: Visualization of user interface

Chapter 3

Training neural network

One of the main challenges in our experiments is to solve training on an unbalanced dataset. Unbalanced datasets are common in terms of autonomous driving, and neural networks can handle this imbalance to a certain level. However, our extension added a large number of points to the training, which belongs to just a few classes and imbalances the dataset even more. We think that the main problem is that large number of automatically annotated points, which belong to "dynamic classes" (e.g. vehicles, pedestrians, etc.) unbalance beyond some threshold, where it worsens training. To address this problem we implemented a few different training approaches (beside automatically annotation all points of annotated dynamic object):

- **Weighing Crossentropy loss:** The original implementation does not use class weights into Crossentropy loss. This is a pretty common approach to dealing with an imbalanced dataset. We add class weights into the loss function and update them every time when new data is added to the labeled part of the dataset. The weight of each class is calculated by equation (3.1)

  $$w_{class} = \frac{|C|}{P_{class} \sum_{c \in C} \frac{1}{P_c}}, \tag{3.1}$$

  where w_{class} is the weight of a certain class, P_n is the percentage of n^{th} class in the labeled dataset. and C represents all segmentation classes. In experiments, this method is denoted by subfix -W (as weighted).

- **Starting from ReDAL[15] 1^{st} checkpoint:** In this approach we just delay the start of our extension. So, the model is trained only on initially annotated points at the beginning, and all points, which can be annotated automatically, are enabled after selecting new data for manual annotation. We hoped that newly annotated data will help to balance the dataset without any additional action. In experiments, this method is denoted by subfix -DS (as delay start).

- **Loss scaling by number of labeled points:** Our method adds many frames to the labeled dataset, where only a few points belonging to some dynamic class are labeled, and the rest of the frame is not. However, each frame has the same impact on the final value of the loss function and therefore on the direction of the gradients. Therefore, these additional data can push the model into prediction "dynamic classes" much more frequently. In experiments, this method is denoted by subfix -WB (as weighted batch).

We address this problem by scaling the loss of each frame by the percentage of points, which are labeled in the whole point cloud. The final loss value is calculated by equation (3.2)

$$\mathcal{L} = \frac{1}{|B|} \sum_{b \in B} \mathcal{L}_b \frac{|p_b^l|}{|p_b|}, \tag{3.2}$$

where $\mathcal{L}$ is the value of the loss function. B is training batch, p represents points and ^{l} means labeled. In experiments, this method is denoted by subfix -WB (as wei).

- **Pseudolabeling of unlabeled points:** This approach addresses the same problem as is described in the point above, but instead of scaling loss, we used pseudolabels to achieve that every point in the frame has a semantic label. Training started from 1^{st} ReDAL[15] checkpoint, and pseudolabels are created/updated with each expansion of the labeled dataset, by the best performing model on the validation part of the dataset, which was trained on old labeled dataset. But with this approach, there is danger of some kind of label drift, because pseudolabels are at the beginning of the training created by a model with relatively low performance. In experiments, this method is denoted by subfix -PL (as pseudolabels).

- **Partial pseudolabeling of unlabeled points with loss scaling:** This approach combines two previous approaches together. Pseudolabels are used for every 40^{th} unlabeled region, which is selected randomly. This should lower the risk of label drift that is mentioned in the previous point and provide equal impact of each point on the final loss value. In experiments, this method is denoted by subfix -PL-WB (as a combination of pseudolabels and weighted batch).

Chapter 4

Experiments

In our experiments, we want to improve the performance of the model as much as possible. Our extension adds more annotations of "dynamic classes", therefore, the model should have a similar or even better performance of these classes with fewer manual annotations. These spared annotations could then be used to improve performance of other classes.

4.1 Datasets

For our experiments, we need to use datasets, which are focused on the semantic segmentation task and are captured in sequences. Therefore, we choose SemanticKITTI[3] and Waymo Open Dataset[11].

4.1.1 SemanticKITTI

SemanticKITTI[3] dataset was captured in rural parts of Karlsruhe (Germany) and on the highway, resulting in a small number of dynamic objects. The dataset was captured by a 64-beam LiDAR with a maximum range of 120 m and a vertical field of view 26.9°, which spins at a frequency of 10 Hz. The average point cloud contains 100,000 points.

The dataset contains 10 training sequences (19,130 frames) and 1 validation sequence (4,071 frames). From the dataset, we use point clouds, per-point semantic labels, and instances. The dataset divides points to 32 semantic classes (without "Unlabeled" and "Outlier") which are then combined to 19 classes (by commonly used learning map) in training, e.g. Lane marking is combined with Road and Moving cars are merged with Car.

4.1.2 Waymo Open Dataset

Waymo Open Dataset[11] was captured in San Francisco, Mountain View, Los Angeles, Detroit, Seattle, and Phoenix (all in the USA). The dataset was captured by 5 LiDARs, however we are using only points from the "TOP" LiDAR, which has 64 beams, maximum restricted range of 75 m and a vertical field of view 20.0°. The dataset also provides two returns for each laser pulse, but we decided to use only the first return.

Semantic labels were not included in the first version of Waymo Open Dataset[11] and were added in version 1.3.0 and improved in version 1.3.2, which is the version we are using in our experiments. But even in this version only usually every 5^{th} frame contains semantic labels, which slightly complicate our application of motion flow, as discussed in subsection 2.4.

This dataset is one of the largest publicly available LiDAR datasets, therefore we decide to use only the subset, which will have a number of frames similar to the SemanticKITTI dataset[3]. We used all frames with semantic labels from the training part of the official dataset. In total it is 23,691 frames, these frames we split into a training and a validation part. The training part contains 19,526 frames and the validation part 4,165. For creating a training and validation part of a dataset, it is important to ensure that both parts have a similar class distribution, without tearing sequences apart. For solving this problem, we need to determine which sequences to put into the training/validation part. We did it by this steps:

1. We randomly added sequences to validation part until validation dataset contains at least 4,150 frames. The rest of the sequences would create a training dataset.

2. We compute a per-point class distribution for validation and training dataset.

3. We divide these distributions (element vise) to compute the ratio between class percentages in different parts of datasets.

4. These values we put in logarithm to scale the ratio "correctly" (ratio 0.5 and 2 represent the same error) and then to the absolute value.

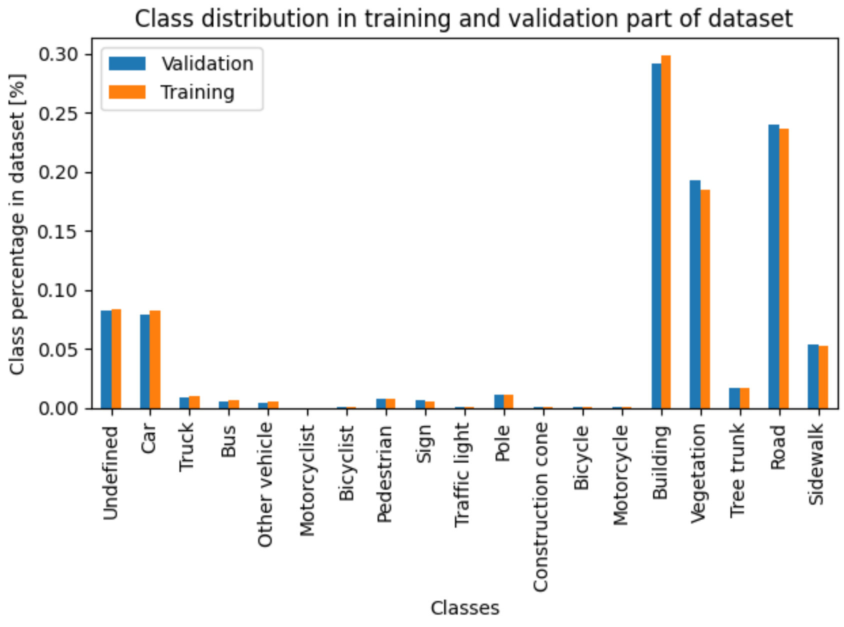
5. The score of these splits is determined by the sum of values from the previous step. We want to minimize this value. The optimal value is 0, which would mean that the class distribution is identical for the training and validation part of the dataset.

Whole formula for score computation is shown in equation (4.1)

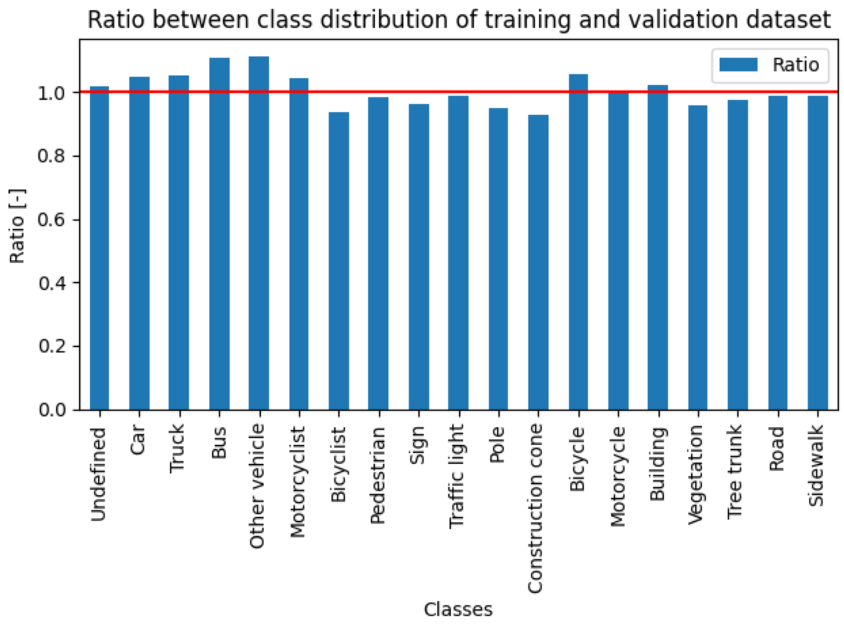
$$\text{score} = \sum_{class} \left| ln\left(\frac{\varphi^{\text{trn}}_{\text{class}}}{\varphi^{\text{val}}_{\text{class}}}\right) \right|, \tag{4.1}$$

where φ^{trn} stands for class distribution of training dataset and φ^{val} represents class distribution of validation dataset.

We run the algorithm above multiple times (in millions) and obtain a split, which has a score of 0.8. Its class distribution in the training and validation part and their ratio is shown in figure 4.1.



(a) : Class distribution of training and validation part of dataset



(b) : Ratio of class distribution of training and validation part of dataset

Figure 4.1: Class distribution and their ration of training and validation part of dataset

ReDAL[15] method is designed so that the labeled part of the training dataset contains 1% of fully annotated frames from the whole training dataset at the beginning. These frames are fixed for all experiments, and their selection on SemanticKITTI[3] appears to be random. Therefore, we randomly selected

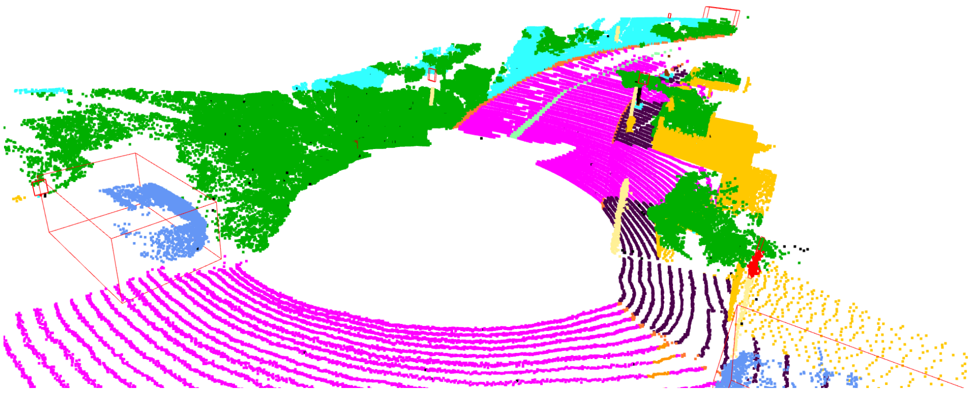


Figure 4.2: Example of walkable annotation (light blue) in Waymo Open Dataset[11]

1% of frames from the training part of the dataset, but we aim for frames that are in the middle of the sequence and one frame from the sequence is selected. From this selection of initially annotated frames our method should benefit the most, because in the middle of the sequence, from our point of view, is the highest probability that the frame contains the highest number of dynamic objects. Our extension can propagate these objects to other frames from the beginning. This selection should not negatively influence ReDAL[15].

This dataset contains 22 semantic labels (without "Undefined"). But similarly as in SemanticKITTI[3] we merged some classes together. However, we cannot find a commonly used learning map, so we created our own. We merged classes Curb, Road, Lane marking and Other ground (bumps, cat's eyes, railtracks) to Road. Also we moved Walkable to Undefined, because it includes, e.g. grassy hill, pedestrian walkway stairs, which is somewhere between classes Vegetation and Sidewalk and even for humans it would be very challenging to determine between these classes. Example of Walkable annotation is shown in figure 4.2

4.2 Neural network

LiDAR data creates a challenge for the architecture of neural networks. The data are sparse and without any structure, therefore, using some kind of voxelization is one of the commonly used approaches. For our experiments, we used the SPVCNN[12] neural network, which came up with the type of convolution on point clouds, called *Sparse Point-Voxel Convolution (SPVConv)*, which is much more efficient on sparse point clouds in terms of memory and computation efficiency. Also, it is one of the neural networks that the authors of ReDAL[15] used to evaluate their method.

4.3 Training results

We performed experiments on two datasets, and the results shown in the following tables are the performance on the validation part of each dataset. We trained three models with each method, and recorded average scores to create a more meaningful and robust comparison between methods (results of each run can be found in the appendix B). The tables contain results of the original implementation of ReDAL[15], results of fully supervised training (i.e. training on the whole training dataset) compared to our methods, where instances are based on the ground truth (GT) annotation or based on the prediction of the motion flow model (MF).

We use the same ReDAL[15] training parameters as the authors mention in the article, i.e. the initially labeled dataset contains 1% of training data, training contains 4 rounds of additional annotations, where in each round 1% of the training data would be humanly annotated and added to the training.

Our metric for evaluating the performance of the model is Intersection over Union (IoU). It is a commonly used metric for object detection and semantic segmentation, which penalizes false positive prediction along side with false negative prediction. The highest value of this metric is 1 and the minimum value is 0. However, it is usually presented as a percentage, which we do as well. It means that the value we want to achieve is 100. This metric is computed by equation (4.2).

$$\text{IoU} = \frac{\text{True Positives}}{\text{False Positives} + \text{True Positives} + \text{False Negatives}} \tag{4.2}$$

In same cases, we also use Recall and Precision. It is usually when we want to know whether the model tends to over-predict some class. This can be deduced by a high value of Recall and low value of Precision, because Recall means: how many points are positively predicted from all the points from a certain class. Precision means: how “reliable” is the positive prediction on certain class. The ranges of these two matrices are the same as for IoU and are computed by equation (4.3).

$$\begin{aligned} \text{Recall} &= \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \\ \text{Precision} &= \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \end{aligned} \tag{4.3}$$

Our extension of the ReDAL[15] method does not have to improve performance on “dynamic classes”, which could be intuitive. This is due to the fact that labeling occurs during training multiple times. So the neural network could be confident with prediction on these classes sooner and the ReDAL[15] method would start selecting different regions, which does not include “dynamic classes”, for manual annotation.

4.3.1 SemanticKITTI

On SemanticKITTI[3] dataset we are able to slightly outperform the ReDAL[15] method as you can see in figure 4.1 and from training on the dataset with 3% of humanly annotated data we achieve a higher mIoU than ReDAL[15] on the same volume of annotation.

	mIoU	Car *	Bicycle	Motorcycle	Truck *	Other vehicle *	Person *	Bicyclist *	Motorcyclist *	Road	Parking	Sidewalk	Other ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic sign
Supervised	63.43	96.12	36.20	64.06	81.51	55.47	68.78	84.35	0.01	93.14	45.68	80.39	1.12	90.93	63.04	87.90	68.14	74.09	63.80	50.45
ReDAL[15]	55.67	**94.76**	11.36	**46.76**	55.17	**39.52**	57.74	71.53	**1.54**	**89.44**	**31.20**	**75.54**	**0.84**	88.94	**53.71**	**87.34**	**66.20**	**74.44**	**62.13**	**49.49**
Our GT	**56.01**	93.73	**20.08**	44.44	**64.51**	38.87	**59.98**	**73.71**	0.45	89.38	26.78	75.33	0.83	**88.98**	53.64	86.92	64.29	72.51	60.63	49.07

Table 4.1: Final results of experiments on 5% of the manually annotated SemanticKITTI[3] dataset. Supervised model is trained on the whole dataset. * means that the class contains dynamic objects. Other-vehicle includes: Other-vehicle, Bus, On-rails. Road also include Line-marking.

As you can see in table 4.2 additional annotations of dynamic objects worsen the overall performance of the neural network. The number of automatically annotated points can be seen in figure 4.3. It helped in all "dynamic classes" by separating them (compared to ReDAL[15]), as shown in figure 4.4. The only "dynamic class" where extension does not improve performance is Car. Additional annotation of cars probably unbalance the dataset in favor of this class and the model starts to predict this class much more, which worsens the precision on this class, as you can see in table 4.3. This claim, that the model trained on our extension is overconfident on "dynamic classes", is even supported by the fact that regions which are selected to annotation contain much fewer points which belong to these classes, as is visualized in figure 4.5.

	mIoU	Car *	Bicycle	Motorcycle	Truck *	Other vehicle *	Person *	Bicyclist *	Motorcyclist *	Road	Parking	Sidewalk	Other ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic sign
ReDAL[15]	**32.30**	**87.47**	0.00	0.00	4.85	2.07	0.00	0.00	0.00	**79.01**	**10.49**	**62.61**	0.00	**79.71**	**33.95**	**84.54**	**46.08**	**72.29**	**46.11**	4.65
Our GT	28.39	71.18	0.00	0.00	**7.79**	**9.23**	**6.62**	**12.92**	**0.26**	69.05	1.38	47.85	0.00	71.56	27.70	78.45	24.43	67.52	35.95	**7.39**

Table 4.2: Models performance on 1% of the manually annotated SemanticKITTI[3]. * means that the class contains dynamic objects. Other-vehicle includes: Other-vehicle, Bus, On-rails. Road also include Line-marking.

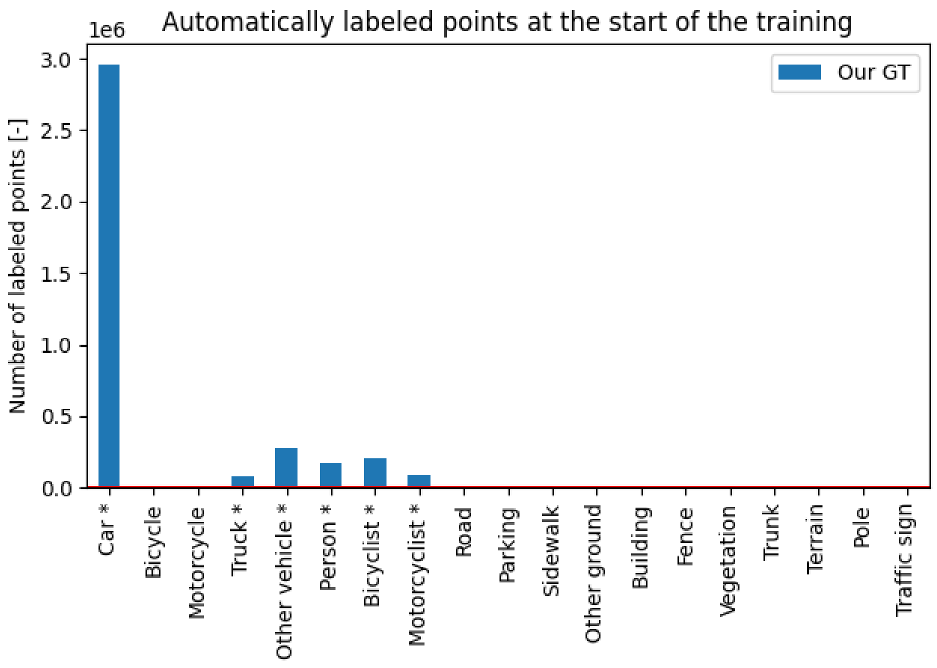
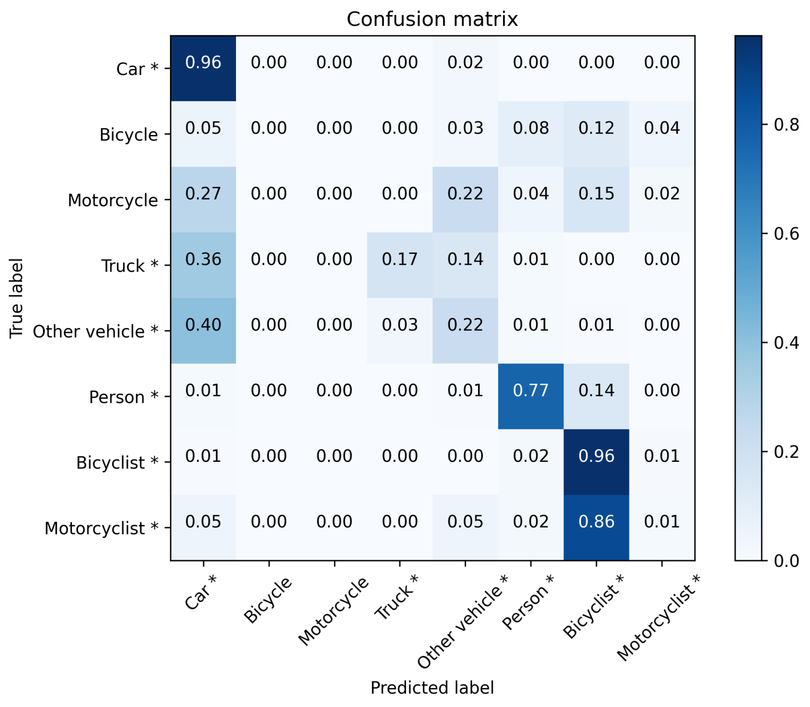
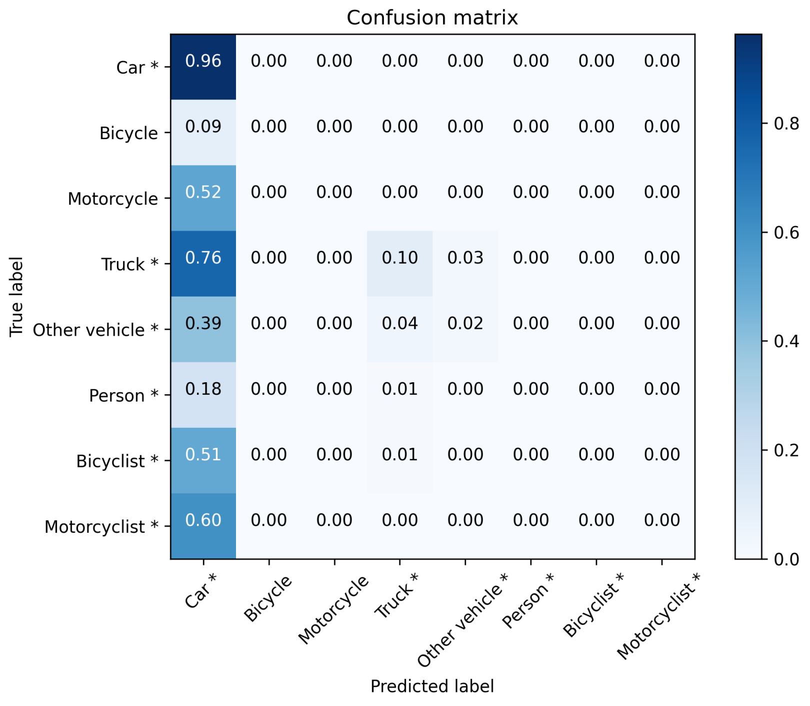


Figure 4.3: Automatically labeled points at the start of the training



(a) : Cropped confusion matrix of model trained with Our GT



(b) : Cropped confusion matrix of model trained with ReDAL[15]

Figure 4.4: Cropped confusion matrix of models trained on SemanticKITTI[3] with 1% of manual annotation.

Classes	Our GT	ReDAL[15]
Car *	0.73	0.90
Truck *	0.13	0.21
Other vehicle *	0.13	0.30
Person *	0.07	1.00
Bicyclist *	0.13	0.00
Motorcyclist *	0.00	0.00

(a) : Models precision

Classes	Our GT	ReDAL[15]
Car *	0.96	0.96
Truck *	0.17	0.10
Other vehicle *	0.22	0.02
Person *	0.77	0.00
Bicyclist *	0.96	0.00
Motorcyclist *	0.01	0.00

(b) : Models recall

Table 4.3: Precision and recall of models trained in 1% of manually annotated SemanticKITTI[3].

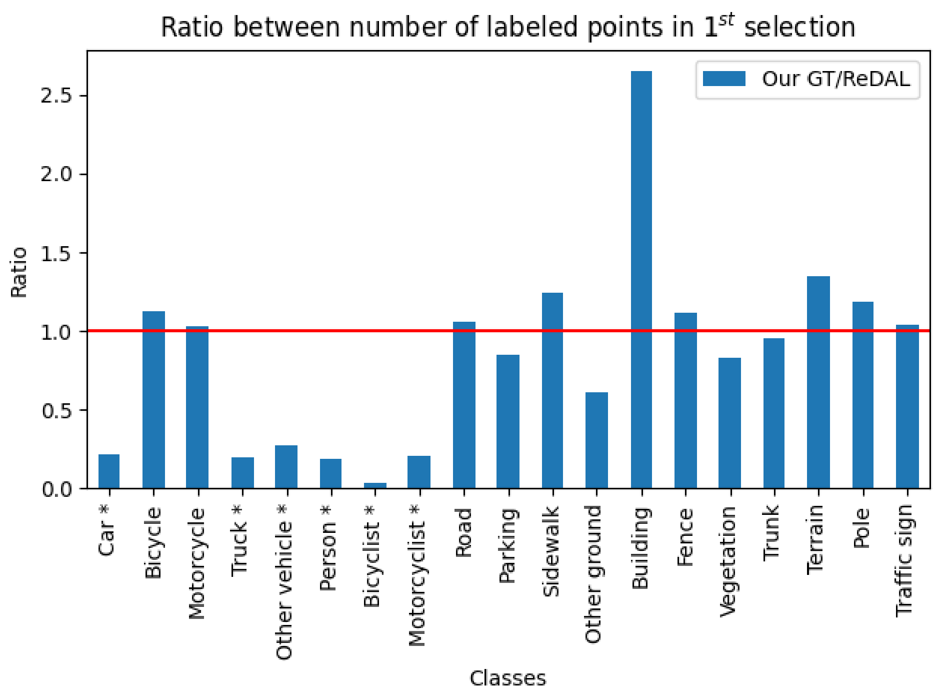


Figure 4.5: Ratio between number of labeled points in 1st selection.

	mIoU	Car *	Bicycle	Motorcycle	Truck *	Other vehicle *	Person *	Bicyclist *	Motorcyclist *	Road	Parking	Sidewalk	Other ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic sign
ReDAL[15]	**44.58**	**90.86**	**5.00**	**12.06**	34.11	**23.76**	**44.52**	**54.44**	0.00	**78.58**	**9.47**	60.97	0.25	84.48	**40.84**	83.02	**60.13**	65.09	**56.01**	**43.48**
Our GT	43.87	88.74	0.45	11.24	**44.45**	20.84	41.73	53.66	**0.16**	78.29	5.55	**61.67**	**0.30**	**84.83**	39.32	**84.98**	55.83	**69.68**	52.55	39.18

Table 4.4: Models performance on 2% of the manually annotated SemanticKITTI[3]. * means that the class contains dynamic objects. Other-vehicle includes: Other-vehicle, Bus, On-rails. Road also include Line-marking.

	mIoU	Car *	Bicycle	Motorcycle	Truck *	Other vehicle *	Person *	Bicyclist *	Motorcyclist *	Road	Parking	Sidewalk	Other ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic sign
ReDAL[15]	51.00	**93.66**	9.45	27.72	59.80	**34.71**	48.78	64.39	0.00	**83.69**	13.05	67.64	**0.67**	87.02	**52.17**	**87.17**	**63.56**	**72.84**	57.73	**44.92**
Our GT	**51.46**	91.61	**13.28**	**31.67**	**67.21**	34.09	**50.32**	**66.21**	**0.36**	83.31	**15.93**	**68.02**	0.26	**87.44**	49.04	86.09	61.65	70.64	**58.30**	42.39

Table 4.5: Models performance on 3% of the manually annotated SemanticKITTI[3]. * means that the class contains dynamic objects. Other-vehicle includes: Other-vehicle, Bus, On-rails. Road also include Line-marking.

	mIoU	Car *	Bicycle	Motorcycle	Truck *	Other vehicle *	Person *	Bicyclist *	Motorcyclist *	Road	Parking	Sidewalk	Other ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic sign
ReDAL[15]	53.98	**94.52**	5.67	**44.96**	62.32	**42.19**	**54.64**	64.07	**0.22**	**87.21**	**26.25**	**73.05**	**1.25**	**89.08**	**55.74**	85.98	**63.13**	69.12	59.77	**46.38**
Our GT	**54.21**	93.09	**16.81**	37.77	**62.62**	41.62	54.35	**74.47**	0.19	86.10	20.79	72.24	1.00	87.41	52.67	**87.43**	62.36	**75.06**	**59.94**	44.12

Table 4.6: Models performance on 4% of the manually annotated SemanticKITTI[3]. * means that the class contains dynamic objects. Other-vehicle includes: Other-vehicle, Bus, On-rails. Road also include Line-marking.

We analyze how many dynamic objects are added throughout the experiments, and we find that above 90% of the dynamic objects have been used at the end of the experiments (specific numbers can be found in table 4.7). Therefore, we conducted future experiments on Waymo Open Dataset[11], which contains much more dynamic objects.

	run_1	run_2	run_3
beginning	83	83	83
selection 1	216	175	215
selection 2	289	316	304
selection 3	310	319	314
selection 4	314	321	315

Table 4.7: Number of dynamic objects, which were are in labeled part of training dataset. Total number of dynamic objects in SemanticKITTI[3] is 336.

With our extension model achieves, similar results on "dynamic classes" (table 4.8) with a lower amount of manual annotation (figure 4.6) of these classes.

	Dynamic classes mIoU	Static classes mIoU
ReDAL[15]	53.38	56.72
Our GT	55.21	56.38

Table 4.8: Comparison of our extension with ReDAL[15] on 5% of manually annotated SemanticKITTI[3]

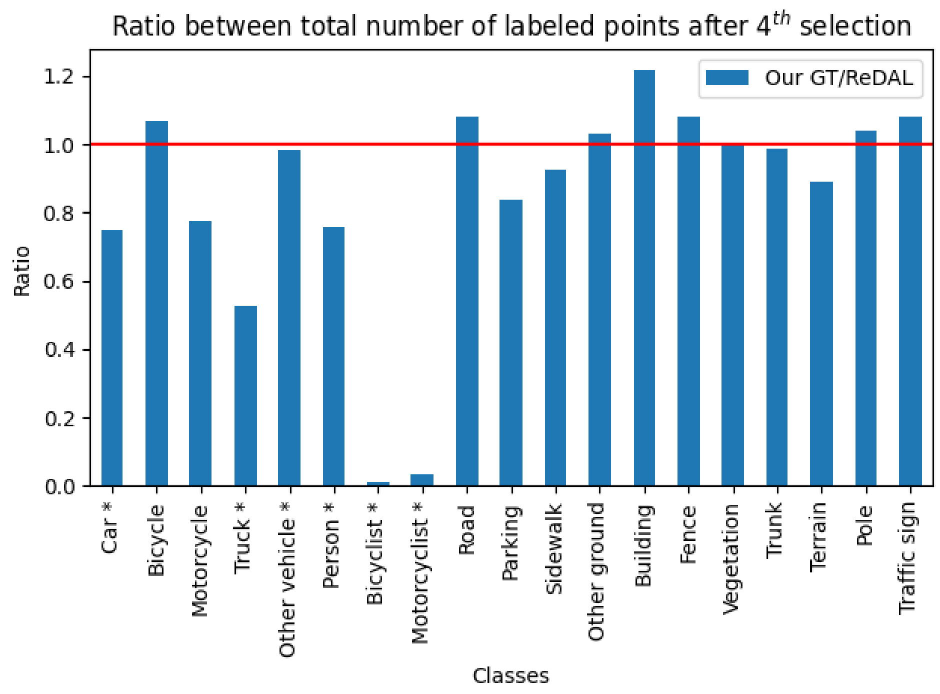


Figure 4.6: Ratio between total number of labeled points after 4th selection

4.3.2 Waymo Open Dataset

Waymo Open Dataset[11] is a much more interesting dataset for our extension, because it includes much more dynamic objects. On the other hand, the large number of dynamic objects enlarges the main disadvantage of our method, which is that it increases imbalance of the dataset towards "dynamic classes". We are not able to adjust training and fine-tuning of the neural network to use the benefit, which our method can provide, when we use ground truth annotations for adjusting regions (table 4.9). However, when we adjust regions with use of motion flow and DBSCANs, pipeline creates even semantically pure instances of static objects. It was not our intention, and it could only happen due to wrong SLIM static vs. dynamic segmentation. These instances are used in training and added more automatically annotated points, but what is more important, they helped to balance the dataset. Therefore, the model that is trained with our extension, which uses motion flow model and our pipeline for region adjustment and matching, outperform ReDAL[15] in all checkpoints in training (tables 4.9, 4.10, 4.12, 4.13, 4.14).

	mIoU	Car	Truck	Bus	Other vehicle	Motorcyclist	Bicyclist	Pedestrian	Sign	Traffic light	Pole	Construction cone	Bicycle	Motorcycle	Building	Vegetation	Tree trunk	Road	Sidewalk
Supervised	64.30	96.00	65.04	68.09	27.94	7.22	69.36	87.26	69.11	30.29	69.83	45.15	43.18	54.65	93.26	91.08	63.88	95.39	80.67
ReDAL[15]	59.24	95.07	61.97	**68.11**	26.23	0.00	54.03	81.06	**67.13**	31.95	**66.29**	42.59	26.62	45.61	91.48	88.91	59.84	91.74	67.75
Our GT	58.90	93.16	63.33	60.23	21.16	0.14	54.89	76.64	65.78	31.11	65.76	44.41	32.77	48.08	**92.03**	**89.32**	59.92	92.05	69.50
Our GT-DS	58.01	92.09	61.81	54.62	28.71	0.09	50.91	73.47	66.05	**32.47**	66.27	39.03	30.21	50.51	91.42	89.00	**60.42**	90.26	66.86
Our GT-W	46.25	90.16	49.25	49.28	25.43	0.02	33.98	58.80	53.48	22.00	56.41	18.04	14.50	10.43	86.32	83.79	52.06	80.01	48.56
Our GT-WB	54.51	91.05	58.59	54.63	25.86	0.00	40.79	71.56	62.62	23.90	59.53	39.34	25.53	36.66	90.07	87.27	54.01	91.87	67.82
Our GT-PL	51.01	92.47	**63.56**	61.69	**33.16**	0.00	37.21	75.40	58.06	22.46	56.59	29.44	2.22	22.72	89.57	85.79	49.78	88.08	49.95
Our GT-PL-WB	56.42	92.26	63.43	57.57	23.95	0.00	39.79	72.27	63.29	27.17	63.27	43.30	26.15	43.30	91.10	88.42	58.61	92.25	69.54
Our MF	**60.91**	**95.33**	63.10	59.14	19.62	**18.46**	**63.13**	**81.35**	65.88	29.40	64.95	**45.00**	**34.80**	**55.56**	91.56	89.03	57.72	**92.37**	**69.98**

Table 4.9: Final results of experiments on 5% of the manually annotated Waymo Open Dataset[11]. Supervised model is trained on the whole dataset. GT stands for creating dynamic objects based on ground truth. GT-DS is the same but training starts by using the 1st checkpoint from original ReDAL[15]. GT-W uses weighted Cross entropy loss, GT-WB uses loss scaling through batch frames, GT-PL uses pseudolabels for all unlabeled points, GT-PL-WB uses pseudolabels on some regions and loss scaling through batch. MF stands for creating dynamic objects based on motion flow.

In table 4.10 you can see similar repercussions of our extension, which is based on ground truth data, as on SemanticKITTI[3] dataset, but here the difference is even larger (here it is 6.16 vs 3.91 points of mIoU on SemanticKITTI[3]). The model is same as on SemanticKITTI[3], overconfident on "dynamic classes", as can be seen in figure 4.11. Number of automatically labeled points can be seen in figure 4.7. However, training with weighted batch (GT-WB and GT-PL-WB) can at this point outperform ReDAL[15] by 2.75 and 2.41 points of IoU, along with models trained with our extension based on the motion flow model (MF), which outperform ReDAL[15] by 2.69 points.

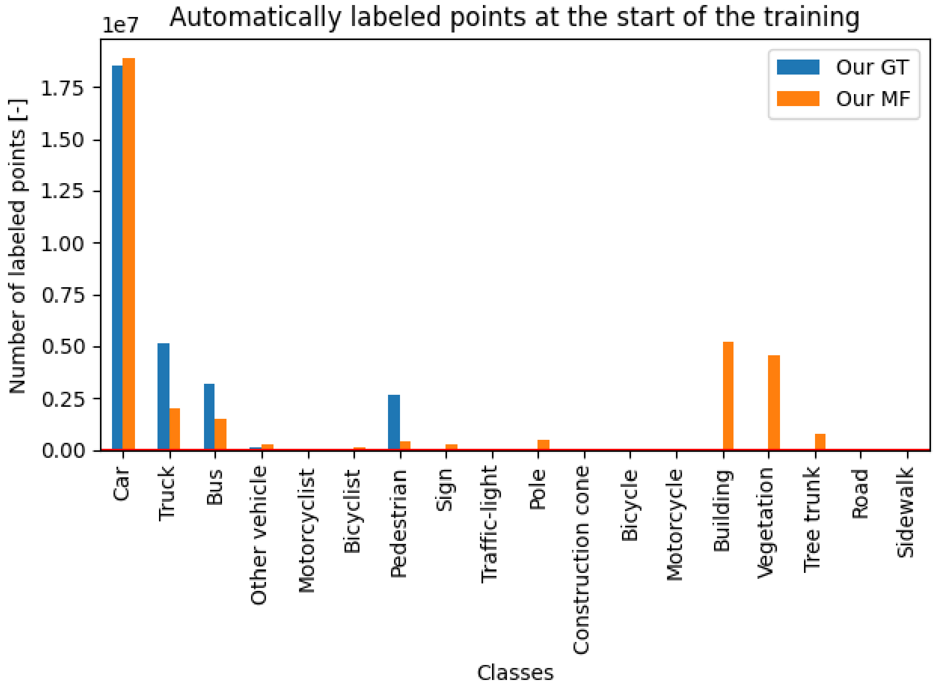


Figure 4.7: Automatically labeled points at the start of the training

	mIoU	Car	Truck	Bus	Other vehicle	Motorcyclist	Bicyclist	Pedestrian	Sign	Traffic light	Pole	Construction cone	Bicycle	Motorcycle	Building	Vegetation	Tree trunk	Road	Sidewalk
ReDAL[15]	35.22	85.68	26.74	23.50	0.32	0.00	5.37	49.03	40.63	0.00	46.90	**21.33**	0.00	0.00	**83.76**	80.11	39.25	86.14	45.17
Our GT	29.06	68.16	18.46	20.32	0.18	0.00	0.02	27.35	34.17	0.03	38.64	9.88	0.00	0.00	73.87	76.33	31.26	83.04	41.37
Our GT-DS	35.22	85.68	26.74	23.50	0.32	0.00	5.37	49.03	40.63	0.00	46.90	**21.33**	0.00	0.00	**83.76**	80.11	39.25	86.14	45.17
Our GT-W	28.34	69.34	11.38	13.92	0.80	0.00	12.65	28.05	35.32	6.57	41.25	19.71	0.77	2.17	55.58	69.03	30.95	76.55	36.04
Our GT-WB	**37.97**	**87.84**	**35.22**	34.57	0.00	0.00	8.45	55.51	**41.06**	6.82	**47.60**	**21.33**	0.04	0.01	84.17	81.70	39.31	**87.40**	**52.38**
Our GT-PL	35.22	85.68	26.74	23.50	0.32	0.00	5.37	49.03	40.63	0.00	46.90	**21.33**	0.00	0.00	**83.76**	80.11	39.25	86.14	45.17
Our GT-PL-WB	37.63	86.09	34.28	35.32	0.00	0.00	9.24	**56.83**	40.65	**7.43**	46.32	18.35	0.02	0.00	83.98	**82.06**	**40.30**	87.06	49.45
Our MF	37.91	86.32	31.11	**39.06**	**4.84**	0.00	**32.21**	54.20	29.37	0.03	40.67	18.88	**1.53**	**16.04**	82.38	80.47	37.12	84.17	43.91

Table 4.10: Models performance on 1% of the manually annotated Waymo Open Dataset[11]. Our GT-DS and Our GT-PL is not trained at this point, but they start from this ReDAL[15] checkpoint.

Classes	Our GT	ReDAL[15]
Car	0.70	0.92
Truck	0.19	0.43
Bus	0.24	0.36
Pedestrian	0.28	0.82

(a) : Models precision

Classes	Our GT	ReDAL[15]
Car	0.97	0.92
Truck	0.78	0.42
Bus	0.54	0.41
Pedestrian	0.93	0.55

(b) : Models recall

Table 4.11: Precision and recall of models trained in 1% of manually annotated Waymo Open Dataset[11].

	mIoU	Car	Truck	Bus	Other vehicle	Motorcyclist	Bicyclist	Pedestrian	Sign	Traffic light	Pole	Construction cone	Bicycle	Motorcycle	Building	Vegetation	Tree trunk	Road	Sidewalk
ReDAL[15]	43.10	88.20	32.39	34.33	9.46	0.00	22.72	69.99	**61.18**	25.91	55.99	35.68	2.95	1.24	84.28	82.13	49.69	84.65	34.97
Our GT	40.37	86.94	31.45	30.47	1.60	0.00	0.02	54.43	58.27	**27.07**	56.25	23.60	5.12	0.68	84.52	**84.93**	**53.81**	84.79	42.64
Our GT-DS	40.66	83.06	36.26	28.97	5.71	0.00	10.37	56.04	59.49	25.75	**57.26**	30.56	4.89	1.75	82.88	81.99	46.70	83.43	36.84
Our GT-W	35.19	84.09	16.15	25.61	5.01	0.07	20.01	35.78	48.40	15.76	49.35	15.18	3.53	2.32	70.96	75.67	45.07	79.69	40.78
Our GT-WB	41.71	84.88	41.58	45.13	5.09	0.00	10.91	54.08	52.67	13.44	53.48	16.79	3.85	4.54	86.15	82.80	48.15	89.17	**58.07**
Our GT-PL	41.18	89.67	**48.90**	**47.57**	0.71	0.00	3.78	68.76	50.05	5.50	51.49	25.23	0.00	0.00	**87.15**	83.31	43.98	87.34	47.74
Our GT-PL-WB	43.08	85.05	39.15	45.40	12.60	0.00	12.08	56.69	52.83	16.19	53.51	32.67	0.55	7.56	85.03	83.09	48.19	**88.96**	55.96
Our MF	**50.27**	**92.54**	47.12	43.59	**14.04**	**0.24**	**50.60**	**72.69**	59.94	20.54	57.73	**36.25**	**16.14**	**36.30**	87.08	84.19	51.65	86.97	47.22

Table 4.12: Models performance on 2% of the manually annotated Waymo Open Dataset[11].

	mIoU	Car	Truck	Bus	Other vehicle	Motorcyclist	Bicyclist	Pedestrian	Sign	Traffic light	Pole	Construction cone	Bicycle	Motorcycle	Building	Vegetation	Tree trunk	Road	Sidewalk
ReDAL[15]	52.06	93.01	53.46	54.20	23.62	0.00	41.55	73.86	63.32	**28.29**	61.32	37.94	16.41	28.51	89.24	86.09	54.16	86.48	45.57
Our GT	51.54	89.79	56.39	54.28	11.37	0.01	18.70	69.54	62.45	**28.29**	**62.10**	40.06	18.11	34.98	**90.52**	**88.05**	56.62	88.37	58.05
Our GT-DS	50.68	89.35	53.12	**55.88**	**16.11**	0.00	21.52	64.75	**63.72**	26.96	61.37	42.59	11.10	30.60	89.39	86.68	**57.49**	86.62	55.06
Our GT-W	40.75	88.48	37.24	41.88	7.96	0.03	23.42	51.47	48.61	18.12	49.12	15.24	8.46	7.80	82.39	82.59	47.95	79.85	42.89
Our GT-WB	48.36	88.40	50.14	52.66	12.11	0.00	25.93	61.58	58.32	22.30	56.95	33.00	15.01	19.72	88.35	85.54	51.36	89.41	59.66
Our GT-PL	44.46	90.75	**58.55**	54.04	14.50	0.00	9.46	72.94	53.27	11.25	53.08	27.39	0.04	0.01	88.27	84.48	45.79	87.68	48.88
Our GT-PL-WB	49.46	88.76	52.87	50.67	15.72	0.00	23.62	68.21	57.35	22.64	55.47	33.20	12.65	26.27	88.73	85.92	51.71	**91.00**	**65.43**
Our MF	**54.55**	**93.27**	53.78	52.66	13.39	**7.96**	**53.85**	**75.63**	63.02	23.82	60.61	**43.84**	**27.84**	**38.80**	88.75	85.79	52.76	88.65	57.52

Table 4.13: Models performance on 3% of the manually annotated Waymo Open Dataset[11].

All our different training approaches (from chapter 3) perform below our expectation on our extension that uses ground truth data (GT). Only training with the weighted batch (-WB and -PL-WB) can improve training at the 1st checkpoint, as mentioned above. However, with a larger dataset all our training approaches perform below ReDAL[15] and the original training with our extension (Our GT).

	mIoU	Car	Truck	Bus	Other vehicle	Motorcyclist	Bicyclist	Pedestrian	Sign	Traffic light	Pole	Construction cone	Bicycle	Motorcycle	Building	Vegetation	Tree trunk	Road	Sidewalk
ReDAL[15]	56.22	94.37	57.29	**63.82**	23.72	0.00	50.26	78.15	**65.37**	**28.42**	63.57	41.42	22.79	35.61	90.80	88.31	59.81	89.28	59.04
Our GT	56.73	92.17	58.95	56.34	26.94	0.00	46.71	73.24	65.33	28.30	**64.86**	43.06	26.37	43.62	**91.38**	**88.67**	58.60	90.77	65.84
Our GT-DS	55.79	91.53	59.83	57.44	25.00	0.00	42.73	69.00	65.18	27.80	62.94	41.90	23.19	43.88	90.84	88.35	**60.04**	90.37	64.21
Our GT-W	44.93	91.30	52.29	47.92	20.03	0.02	24.71	58.81	51.55	18.64	52.94	20.53	11.19	6.22	86.40	83.98	49.75	82.98	49.39
Our GT-WB	51.94	90.51	55.52	57.55	16.77	0.00	31.09	65.53	60.30	23.19	59.94	37.25	20.27	31.65	88.83	85.46	54.33	90.97	65.70
Our GT-PL	48.25	91.50	**61.80**	58.99	**27.92**	0.00	29.17	74.84	55.24	20.62	54.68	28.48	0.34	6.93	88.88	85.06	47.22	87.82	49.00
Our GT-PL-WB	53.68	90.51	56.99	55.61	19.08	0.00	34.76	69.10	62.16	24.40	61.49	40.67	19.96	36.70	90.10	87.48	57.11	**91.72**	**68.45**
Our MF	**58.35**	**94.53**	58.44	59.85	18.87	**6.59**	**61.20**	**79.40**	64.43	26.88	62.54	**44.91**	**32.68**	**47.83**	90.63	88.08	58.32	90.41	64.77

Table 4.14: Models performance on 4% of the manually annotated Waymo Open Dataset[11].

The ratio between the total number of manually annotated points at the end of the experiments can be found in figure 4.8, from this visualization we can see that with our extension the model needs much less manual annotation of “dynamic classes”.

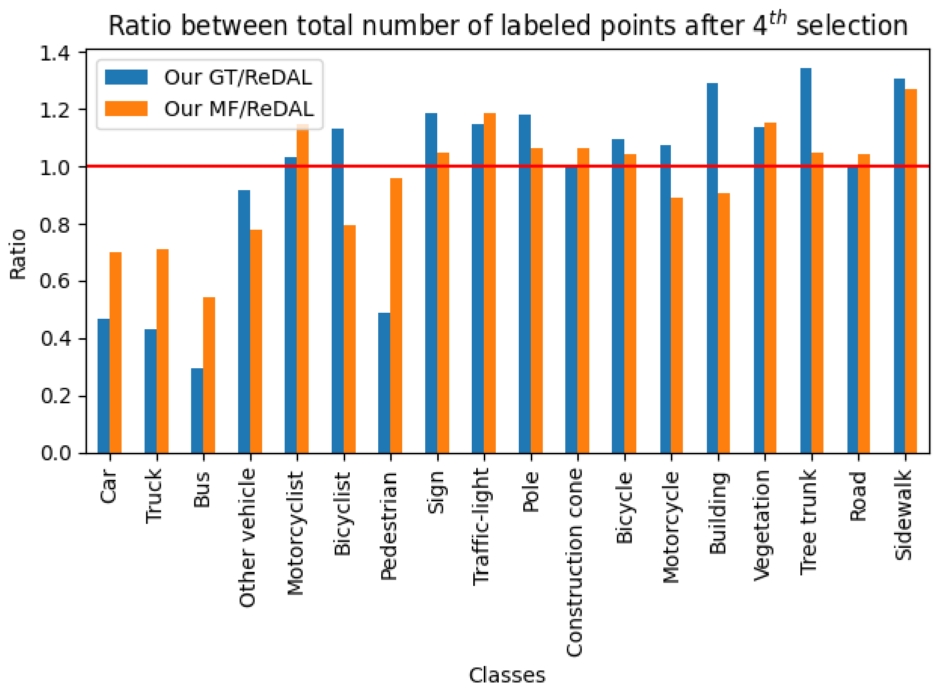


Figure 4.8: Ratio between total number of manually labeled points after 4th selection

Chapter 5
Conclusion

In this thesis, we created an extension for the ReDAL[15] method. This extension can create semantically pure regions that contain dynamic objects and other well separated objects in 3D point cloud and link them through a sequence of point clouds with the same objects without any manual annotation. Therefore, the annotator will be able to annotate an object in a whole sequence by one action. This is much more efficient than with original regions which could contain many semantic classes, which the annotator needs to manually separate and label.

For this extension, we create a user interface, where the annotator can see an object from all time frames at ones, along with the visualization from one time frame (where the object contains the most points) for a more detailed look at the object. Then the annotator can adjust time frames selection in case of mistakes of the new regions prediction and select the semantic label.

From experiments we find out the main disadvantage of using our extension is unbalancing of the dataset, which occurs with usage of our method based on ground truth data. However, when we used our whole pipeline, the method is able to automatically propagate labels even to static objects, which reduces the imbalance of the dataset (this was not our intention and it could be suppressed by usage of dataset odometry). By using the whole pipeline, we are able to improve mIoU by 1.67 percentage points, that is, from a value equal to 92.1% of performance of the fully supervised model, which ReDAL[15] achieves, to 94.7% with usage of our extension.

Our extension can be used on all sensors that output a sequence of point clouds. Therefore, it can be used even for stereo cameras. In future work, it would be interesting to add a criterion which would focus on predictions of a neural network on linked regions.

Appendix A

Bibliography

[1] Iro Armeni, Ozan Sener, Amir R. Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3D Semantic Parsing of Large-Scale Indoor Spaces. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[2] Stefan Baur, David Emmerichs, Frank Moosmann, Peter Pinggera, Bjorn Ommer, and Andreas Geiger. SLIM: Self-Supervised LiDAR Scene Flow and Motion Segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[3] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[4] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[5] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Knowledge Discovery and Data Mining*, 1996.

[6] Zeyu Hu, Xuyang Bai, Runze Zhang, Xin Wang, Guangyuan Sun, Hongbo Fu, and Chiew-Lan Tai. LiDAL: Inter-frame Uncertainty Based Active Learning for 3D LiDAR Semantic Segmentation. In *European Conference on Computer Vision (ECCV)*, 2022.

[7] Seungjae Lee, Hyungtae Lim, and Hyun Myung. Patchwork++: Fast and Robust Ground Segmentation Solving Partial Under-Segmentation Using 3D Point Cloud. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[8] Minghua Liu, Yin Zhou, Charles R Qi, Boqing Gong, Hao Su, and Dragomir Anguelov. LESS: Label-Efficient Semantic Segmentation for LiDAR Point Clouds. In *European Conference on Computer Vision (ECCV)*, 2022.

[9] Jeremie Papon, Alexey Abramov, Markus Schoeler, and Florentin Worgotter. Voxel Cloud Connectivity Segmentation - Supervoxels for Point Clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[10] Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[11] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in Perception for Autonomous Driving: Waymo Open Dataset. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[12] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching Efficient 3D Architectures with Sparse Point-Voxel Convolution. In *European Conference on Computer Vision (ECCV)*, 2020.

[13] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision (ECCV)*, 2020.

[14] Dan Wang and Yi Shang. A New Active Labeling Method for Deep Learning. In *International Joint Conference on Neural Networks (IJCNN)*, 2014.

[15] Tsung-Han Wu, Yueh-Cheng Liu, Yu-Kai Huang, Hsin-Ying Lee, Hung-Ting Su, Ping-Chia Huang, and Winston H. Hsu. ReDAL: Region-Based and Diversity-Aware Active Learning for Point Cloud Semantic Segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Appendix B

Results of each training run

B.1 SemanticKITTI

	mIoU	Car *	Bicycle	Motorcycle	Truck *	Other vehicle *	Person *	Bicyclist *	Motorcyclist *	Road	Parking	Sidewalk	Other ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic sign
Supervised$_{\text{run 1}}$	63.24	95.94	38.69	60.72	82.05	53.54	70.49	86.44	0.01	93.01	44.61	80.26	1.36	90.91	64.17	86.78	66.52	71.36	63.26	51.52
Supervised$_{\text{run 2}}$	63.23	95.81	36.47	65.65	79.85	49.81	69.15	83.67	0.00	92.81	46.43	79.77	1.08	90.49	61.14	89.05	69.81	76.65	63.92	49.79
Supervised$_{\text{run 3}}$	63.82	96.62	33.45	65.81	82.63	63.05	66.70	82.95	0.03	93.59	46.00	81.15	0.93	91.39	63.81	87.88	68.08	74.26	64.23	50.04
ReDAL$_{\text{run 1}}$	54.64	94.11	9.42	44.52	57.03	26.68	52.83	71.81	0.04	88.91	29.58	75.00	1.10	89.76	55.52	87.60	66.81	76.11	62.06	49.19
ReDAL$_{\text{run 2}}$	56.08	95.08	15.95	41.72	54.55	47.66	63.68	68.73	0.22	89.71	31.56	75.59	0.63	87.76	51.36	88.96	64.76	77.52	60.74	49.42
ReDAL$_{\text{run 3}}$	56.28	95.10	8.70	54.04	53.93	44.21	56.72	74.06	4.36	89.70	32.45	76.03	0.80	89.31	54.26	85.45	67.02	69.68	63.59	49.85
Our GT$_{\text{run 1}}$	56.28	93.97	23.12	43.86	67.25	43.14	64.45	77.12	0.85	88.33	21.21	73.99	0.25	89.10	54.13	86.46	62.42	71.42	60.43	47.77
Our GT$_{\text{run 2}}$	55.75	93.47	21.14	37.57	69.87	39.12	55.28	69.51	0.27	90.39	33.49	76.31	1.41	88.14	53.18	86.45	63.94	70.10	59.72	49.92
Our GT$_{\text{run 3}}$	55.99	93.74	15.97	51.89	56.40	34.35	60.22	74.50	0.24	89.42	25.63	75.69	0.82	89.69	53.60	87.85	66.52	76.01	61.73	49.52

Table B.1: Final results of experiments on 5% of the manually annotated SemanticKITTI[3] dataset. Supervised model is trained on the whole dataset. * means that the class contains dynamic objects. Other-vehicle includes: Other-vehicle, Bus, On-rails. Road also include Line-marking.

	mIoU	Car *	Bicycle	Motorcycle	Truck *	Other vehicle *	Person *	Bicyclist *	Motorcyclist *	Road	Parking	Sidewalk	Other ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic sign
ReDAL$_{\text{run 1}}$	32.18	86.16	0.00	0.00	0.01	0.52	0.00	0.00	0.00	79.11	11.49	62.86	0.00	80.58	35.49	84.92	48.56	72.94	45.51	3.36
ReDAL$_{\text{run 2}}$	32.22	87.41	0.00	0.00	0.59	0.50	0.00	0.00	0.00	78.68	11.22	62.33	0.00	79.95	36.07	84.95	48.65	73.17	45.26	3.40
ReDAL$_{\text{run 3}}$	32.51	88.85	0.00	0.00	13.94	5.18	0.00	0.00	0.00	79.23	8.75	62.65	0.00	78.60	30.28	83.74	41.03	70.75	47.55	7.19
Our GT$_{\text{run 1}}$	29.20	71.75	0.00	0.00	11.49	8.21	7.59	11.74	0.06	69.07	1.71	53.93	0.00	73.20	27.05	79.10	27.01	68.46	36.89	7.52
Our GT$_{\text{run 2}}$	28.83	67.72	0.00	0.00	5.95	10.40	5.63	13.25	0.08	69.56	0.53	48.04	0.00	75.20	34.45	79.24	26.46	65.55	39.65	5.96
Our GT$_{\text{run 3}}$	27.13	74.06	0.00	0.00	5.94	9.08	6.63	13.78	0.64	68.53	1.89	41.59	0.00	66.27	21.61	77.02	19.83	68.55	31.30	8.70

Table B.2: Models performance on 1% of the manually annotated SemanticKITTI[3]. * means that the class contains dynamic objects. Other-vehicle includes: Other-vehicle, Bus, On-rails. Road also include Line-marking.

	mIoU	Car *	Bicycle	Motorcycle	Truck *	Other vehicle *	Person *	Bicyclist *	Motorcyclist *	Road	Parking	Sidewalk	Other ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic sign
ReDAL$_{\text{run 1}}$	44.52	91.00	9.57	8.47	29.83	27.20	45.17	52.64	0.00	78.16	11.00	61.75	0.06	84.07	41.00	82.24	59.76	63.33	57.45	43.22
ReDAL$_{\text{run 2}}$	45.17	91.48	3.52	17.86	47.87	20.91	47.74	59.43	0.00	79.54	0.60	61.94	0.38	84.15	38.20	82.62	59.08	63.93	56.08	42.94
ReDAL$_{\text{run 3}}$	44.05	90.09	1.90	9.85	24.64	23.17	40.65	51.25	0.00	78.04	16.81	59.21	0.31	85.21	43.31	84.19	61.55	68.00	54.49	44.27
Our GT$_{\text{run 1}}$	43.87	88.67	0.07	10.82	47.78	20.07	37.47	55.45	0.21	80.22	5.93	63.67	0.32	82.98	31.11	85.18	58.21	68.17	54.97	42.14
Our GT$_{\text{run 2}}$	41.83	88.22	0.56	1.62	34.27	17.78	43.80	45.36	0.15	77.99	4.35	61.07	0.25	85.00	37.21	84.33	54.09	72.24	49.22	37.34
Our GT$_{\text{run 3}}$	45.90	89.32	0.73	21.28	51.30	24.66	43.91	60.18	0.12	76.67	6.38	60.27	0.34	86.50	49.64	85.44	55.19	68.62	53.47	38.06

Table B.3: Models performance on 2% of the manually annotated SemanticKITTI[3]. * means that the class contains dynamic objects. Other-vehicle includes: Other-vehicle, Bus, On-rails. Road also include Line-marking.

	mIoU	Car *	Bicycle	Motorcycle	Truck *	Other vehicle *	Person *	Bicyclist *	Motorcyclist *	Road	Parking	Sidewalk	Other ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic sign
ReDAL$_{\text{run 1}}$	51.48	93.98	4.06	33.31	70.25	35.75	44.68	60.20	0.01	84.70	11.55	68.68	0.52	86.99	53.73	88.24	61.20	74.53	59.45	46.33
ReDAL$_{\text{run 2}}$	50.88	93.93	10.00	16.13	66.56	35.40	55.17	74.60	0.00	82.48	8.53	65.67	0.41	87.48	49.53	85.71	63.40	69.68	56.77	45.31
ReDAL$_{\text{run 3}}$	50.63	93.08	14.30	33.72	42.60	32.97	46.48	58.36	0.00	83.88	19.06	68.56	1.08	86.58	53.24	87.57	66.07	74.31	56.97	43.11
Our GT$_{\text{run 1}}$	50.91	91.25	24.25	18.80	62.82	37.45	45.80	61.68	0.96	84.32	17.65	68.76	0.38	87.65	47.14	85.90	62.86	69.18	58.15	42.38
Our GT$_{\text{run 2}}$	51.77	91.86	3.69	39.29	69.81	36.35	50.31	70.82	0.12	82.35	17.15	66.85	0.23	87.49	50.73	86.03	62.51	71.23	57.06	39.74
Our GT$_{\text{run 3}}$	51.71	91.72	11.91	36.91	69.00	28.47	54.86	66.13	0.00	83.26	13.00	68.44	0.17	87.18	49.24	86.33	59.59	71.51	59.70	45.04

Table B.4: Models performance on 3% of the manually annotated SemanticKITTI[3]. * means that the class contains dynamic objects. Other-vehicle includes: Other-vehicle, Bus, On-rails. Road also include Line-marking.

	mIoU	Car *	Bicycle	Motorcycle	Truck *	Other vehicle *	Person *	Bicyclist *	Motorcyclist *	Road	Parking	Sidewalk	Other ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic sign
ReDAL$_{\text{run 1}}$	53.99	94.30	3.73	44.39	66.20	45.90	50.06	68.02	0.10	86.82	25.35	72.33	0.49	89.77	58.25	84.23	65.09	63.06	61.63	46.15
ReDAL$_{\text{run 2}}$	54.70	94.93	7.51	45.29	67.79	45.38	58.59	55.63	0.35	87.63	26.35	73.47	0.69	89.44	55.94	86.86	60.53	71.42	60.99	50.50
ReDAL$_{\text{run 3}}$	53.24	94.34	5.76	45.20	52.96	35.29	55.26	68.56	0.20	87.18	27.04	73.35	2.58	88.04	53.04	86.84	63.77	72.87	56.70	42.49
Our GT$_{\text{run 1}}$	55.00	92.90	14.11	41.42	74.95	40.12	56.79	69.73	0.34	88.12	24.79	73.91	0.19	86.67	51.46	87.61	63.05	75.93	60.60	42.27
Our GT$_{\text{run 2}}$	53.47	93.33	12.23	29.92	53.01	43.08	55.25	76.82	0.23	84.56	23.87	71.97	2.60	87.20	53.69	87.65	63.43	75.35	58.79	42.87
Our GT$_{\text{run 3}}$	54.17	93.04	24.09	41.96	59.91	41.66	51.01	76.86	0.00	85.63	13.70	70.84	0.21	88.36	52.85	87.03	60.59	73.91	60.42	47.21

Table B.5: Models performance on 4% of the manually annotated SemanticKITTI[3]. * means that the class contains dynamic objects. Other-vehicle includes: Other-vehicle, Bus, On-rails. Road also include Line-marking.

B.2 Waymo Open Dataset

	mIoU	Car	Truck	Bus	Other vehicle	Motorcyclist	Bicyclist	Pedestrian	Sign	Traffic light	Pole	Construction cone	Bicycle	Motorcycle	Building	Vegetation	Tree trunk	Road	Sidewalk
Supervised$_{\text{run 1}}$	62.62	95.20	67.16	67.95	19.69	2.91	67.04	86.71	69.37	32.36	69.07	47.27	38.38	43.13	93.07	90.51	62.40	95.19	79.74
Supervised$_{\text{run 2}}$	65.56	96.33	63.51	71.52	31.39	8.78	71.15	87.56	68.60	28.84	70.29	44.85	47.04	63.77	93.33	91.38	64.95	95.49	81.25
Supervised$_{\text{run 3}}$	64.72	96.46	64.45	64.79	32.75	9.97	69.90	87.50	69.36	29.67	70.14	43.33	44.13	57.06	93.38	91.36	64.28	95.48	81.02
ReDAL$_{\text{run 1}}$	59.20	95.44	60.30	69.23	25.76	0.00	49.45	81.59	67.25	31.80	66.93	41.85	30.12	46.29	91.80	89.37	60.19	91.46	66.81
ReDAL$_{\text{run 2}}$	59.53	94.85	62.64	70.83	23.81	0.00	60.77	80.70	66.30	32.49	64.90	44.65	23.10	45.89	91.08	88.22	60.51	92.16	68.71
ReDAL$_{\text{run 3}}$	58.99	94.93	62.98	64.28	29.11	0.00	51.88	80.88	67.83	31.55	67.04	41.28	26.64	44.65	91.56	89.14	58.82	91.61	67.73
Our GT$_{\text{run 1}}$	59.65	92.86	61.79	61.46	29.43	0.00	59.40	76.42	66.83	30.25	66.30	45.68	30.10	50.01	91.83	89.08	60.79	92.11	69.33
Our GT$_{\text{run 2}}$	58.39	92.40	65.33	62.93	13.80	0.02	57.14	76.63	64.35	30.53	64.53	40.57	35.27	47.64	92.11	89.16	58.61	91.75	68.34
Our GT$_{\text{run 3}}$	58.67	94.23	62.86	56.30	20.24	0.41	48.12	76.86	66.15	32.55	66.44	46.98	32.93	46.58	92.16	89.72	60.37	92.29	70.84
Our GT-DS$_{\text{run 1}}$	58.41	91.52	63.68	56.31	31.99	0.26	47.15	72.20	66.44	33.08	66.59	39.73	30.81	49.17	91.85	89.13	59.83	91.42	70.15
Our GT-DS$_{\text{run 2}}$	58.22	93.28	61.06	59.48	29.91	0.00	58.13	72.58	66.21	32.79	66.01	31.00	33.39	52.78	91.30	89.30	61.36	88.03	61.38
Our GT-DS$_{\text{run 3}}$	57.41	91.48	60.68	48.06	24.23	0.00	47.45	75.64	65.50	31.55	66.20	46.36	26.43	49.57	91.12	88.56	60.08	91.34	69.06
Our GT-W$_{\text{run 1}}$	46.92	91.82	47.12	43.55	20.98	0.05	33.80	61.43	53.64	20.50	56.99	22.32	8.36	14.16	86.51	85.30	54.66	87.44	55.96
Our GT-W$_{\text{run 2}}$	44.11	87.49	48.48	46.97	29.65	0.00	33.05	56.96	55.85	23.17	56.52	13.05	13.18	8.16	86.95	82.42	48.89	67.31	35.87
Our GT-W$_{\text{run 3}}$	47.72	91.18	52.14	57.31	25.66	0.01	35.08	58.01	50.95	22.34	55.73	18.75	21.97	8.97	85.50	83.64	52.64	85.27	53.84
Our GT-WB$_{\text{run 1}}$	53.34	92.95	52.89	48.76	24.09	0.00	35.33	67.21	62.43	20.95	58.14	39.10	26.96	36.00	89.79	87.06	57.07	91.96	69.34
Our GT-WB$_{\text{run 2}}$	53.89	89.05	60.78	54.59	19.53	0.00	41.67	76.19	62.22	24.36	58.47	37.13	29.94	31.75	89.63	87.21	49.60	91.58	66.39
Our GT-WB$_{\text{run 3}}$	56.29	91.15	62.11	60.53	33.97	0.00	45.36	71.29	63.22	26.38	61.99	41.78	19.68	42.24	90.78	87.55	55.36	92.07	67.72
Our GT-PL$_{\text{run 1}}$	51.74	92.29	63.38	58.09	31.65	0.00	42.15	75.16	59.67	24.73	57.43	33.25	2.27	26.04	89.20	85.38	49.94	88.53	52.21
Our GT-PL$_{\text{run 2}}$	50.68	92.81	63.23	63.80	34.73	0.00	35.55	76.11	58.63	17.99	56.14	33.95	3.69	15.36	89.95	86.01	49.94	87.63	46.77
Our GT-PL$_{\text{run 3}}$	50.60	92.30	64.08	63.19	33.10	0.00	33.94	74.92	55.89	24.66	56.20	21.11	0.71	26.75	89.57	85.98	49.47	88.08	50.88
Our GT-PL-WB$_{\text{run 1}}$	57.72	92.56	65.08	64.91	25.63	0.00	43.18	72.98	63.68	25.99	63.65	44.48	27.02	46.02	91.16	88.53	60.57	92.40	71.21
Our GT-PL-WB$_{\text{run 2}}$	57.43	93.18	64.70	59.58	21.71	0.00	44.26	77.15	63.75	28.39	63.85	43.00	29.46	44.69	91.36	88.82	59.38	92.18	68.41
Our GT-PL-WB$_{\text{run 3}}$	54.11	91.04	60.52	48.21	24.51	0.00	31.92	66.67	62.43	27.13	62.32	42.41	21.96	39.18	90.79	87.90	55.88	92.16	69.00
Our MF$_{\text{run 1}}$	59.97	94.38	59.41	56.21	18.21	12.44	63.40	81.26	66.18	30.23	65.09	43.65	34.02	56.20	91.78	89.34	57.77	91.72	68.21
Our MF$_{\text{run 2}}$	61.31	95.59	65.28	58.63	28.71	17.63	61.88	81.00	65.33	29.49	64.40	48.73	35.80	51.99	91.28	88.78	55.09	92.83	71.17
Our MF$_{\text{run 3}}$	61.45	96.02	64.62	62.59	11.95	25.32	64.12	81.80	66.14	28.49	65.37	42.62	34.57	58.49	91.61	88.98	60.31	92.57	70.56

Table B.6: Final results of experiments on 5% of the manually annotated Waymo Open Dataset[11]. Supervised model is trained on the whole dataset. GT stands for creating dynamic objects based on ground truth. GT-DS is the same but training starts by using the 1st checkpoint from original ReDAL[15]. GT-W uses weighted Cross entropy loss, GT-WB uses loss scaling through batch frames, GT-PL uses pseudolabels for all unlabeled points, GT-PL-WB uses pseudolabels on some regions and loss scaling through batch. MF stands for creating dynamic objects based on motion flow.

	mIoU	Car	Truck	Bus	Other vehicle	Motorcyclist	Bicyclist	Pedestrian	Sign	Traffic light	Pole	Construction cone	Bicycle	Motorcycle	Building	Vegetation	Tree trunk	Road	Sidewalk
ReDAL$_{\text{run 1}}$	35.38	86.62	28.56	22.11	0.11	0.00	4.77	49.64	40.87	0.00	47.63	21.76	0.00	0.00	83.67	79.65	38.64	86.38	46.43
ReDAL$_{\text{run 2}}$	35.38	85.50	27.46	23.17	0.46	0.00	6.97	49.97	40.74	0.00	46.75	21.33	0.00	0.00	84.13	80.68	40.07	86.03	43.58
ReDAL$_{\text{run 3}}$	34.89	84.92	24.21	25.22	0.39	0.00	4.36	47.49	40.29	0.00	46.31	20.89	0.00	0.00	83.47	80.00	39.05	86.01	45.49
Our GT$_{\text{run 1}}$	28.89	66.24	20.48	21.26	0.13	0.00	0.07	25.98	37.82	0.02	34.44	6.20	0.00	0.00	73.75	76.10	32.32	83.24	41.90
Our GT$_{\text{run 2}}$	29.57	68.07	18.35	20.82	0.28	0.00	0.00	31.37	31.56	0.06	43.87	12.86	0.00	0.00	76.11	76.96	29.02	82.37	40.48
Our GT$_{\text{run 3}}$	28.73	70.16	16.56	18.89	0.14	0.00	0.00	24.71	33.12	0.00	37.62	10.59	0.00	0.00	71.74	75.94	32.44	83.52	41.72
Our GT-DS$_{\text{run 1}}$	35.38	86.62	28.56	22.11	0.11	0.00	4.77	49.64	40.87	0.00	47.63	21.76	0.00	0.00	83.67	79.65	38.64	86.38	46.43
Our GT-DS$_{\text{run 2}}$	35.38	85.50	27.46	23.17	0.46	0.00	6.97	49.97	40.74	0.00	46.75	21.33	0.00	0.00	84.13	80.68	40.07	86.03	43.58
Our GT-DS$_{\text{run 3}}$	34.89	84.92	24.21	25.22	0.39	0.00	4.36	47.49	40.29	0.00	46.31	20.89	0.00	0.00	83.47	80.00	39.05	86.01	45.49
Our GT-W$_{\text{run 1}}$	28.52	66.05	11.39	14.35	0.85	0.00	15.72	28.96	34.50	6.20	41.98	18.35	0.67	3.14	56.59	69.20	29.80	81.71	33.88
Our GT-W$_{\text{run 2}}$	28.62	72.50	11.72	13.76	0.66	0.00	11.02	27.78	36.58	6.96	41.72	20.78	0.93	1.61	56.39	69.70	31.97	73.93	37.17
Our GT-W$_{\text{run 3}}$	27.87	69.48	11.02	13.64	0.90	0.00	11.22	27.41	34.88	6.54	40.06	19.99	0.71	1.75	53.75	68.20	31.07	74.02	37.08
Our GT-WB$_{\text{run 1}}$	38.12	87.95	32.41	33.02	0.00	0.00	7.65	55.82	43.31	10.06	48.99	21.48	0.00	0.00	83.84	81.99	40.47	87.52	51.70
Our GT-WB$_{\text{run 2}}$	37.69	87.81	35.75	34.09	0.00	0.00	9.75	54.36	38.53	4.91	47.17	22.09	0.02	0.00	84.14	81.48	39.48	87.24	51.61
Our GT-WB$_{\text{run 3}}$	38.09	87.77	37.51	36.59	0.00	0.00	7.94	56.35	41.34	5.48	46.63	20.41	0.09	0.04	84.54	81.64	37.98	87.44	53.84
Our GT-PL$_{\text{run 1}}$	35.38	86.62	28.56	22.11	0.11	0.00	4.77	49.64	40.87	0.00	47.63	21.76	0.00	0.00	83.67	79.65	38.64	86.38	46.43
Our GT-PL$_{\text{run 2}}$	35.38	85.50	27.46	23.17	0.46	0.00	6.97	49.97	40.74	0.00	46.75	21.33	0.00	0.00	84.13	80.68	40.07	86.03	43.58
Our GT-PL$_{\text{run 3}}$	34.89	84.92	24.21	25.22	0.39	0.00	4.36	47.49	40.29	0.00	46.31	20.89	0.00	0.00	83.47	80.00	39.05	86.01	45.49
Our GT-PL-WB$_{\text{run 1}}$	37.98	87.06	35.03	35.60	0.01	0.00	7.48	56.83	40.43	9.12	48.70	19.40	0.00	0.00	84.45	82.38	38.40	87.53	51.26
Our GT-PL-WB$_{\text{run 2}}$	37.10	85.51	31.25	34.62	0.00	0.00	9.32	58.59	39.44	7.97	44.14	15.78	0.04	0.00	83.31	81.83	41.69	86.64	47.64
Our GT-PL-WB$_{\text{run 3}}$	37.82	85.71	36.56	35.73	0.00	0.00	10.91	55.07	42.07	5.21	46.13	19.87	0.02	0.01	84.18	81.96	40.80	87.01	49.46
Our MF$_{\text{run 1}}$	38.43	86.99	32.78	42.74	4.91	0.00	32.88	53.54	29.98	0.02	41.43	21.70	1.22	15.50	82.34	80.49	37.34	84.32	43.58
Our MF$_{\text{run 2}}$	37.52	85.87	29.34	32.80	3.78	0.00	29.73	54.43	31.06	0.08	40.64	19.92	2.06	17.78	82.51	80.30	37.37	83.93	43.68
Our MF$_{\text{run 3}}$	37.77	86.09	31.20	41.64	5.84	0.00	34.01	54.64	27.07	0.00	39.93	15.01	1.31	14.83	82.30	80.63	36.64	84.27	44.46

Table B.7: Models performance on 1% of the manually annotated Waymo Open Dataset[11]. Our GT-DS and Our GT-PL is not trained at this point, but they start from this ReDAL[15] checkpoint.

	mIoU	Car	Truck	Bus	Other vehicle	Motorcyclist	Bicyclist	Pedestrian	Sign	Traffic light	Pole	Construction cone	Bicycle	Motorcycle	Building	Vegetation	Tree trunk	Road	Sidewalk
ReDAL$_{\text{run 1}}$	42.72	89.73	28.58	34.98	9.77	0.00	18.45	69.06	61.38	26.94	57.29	34.63	0.47	1.25	82.75	81.76	47.00	85.07	39.81
ReDAL$_{\text{run 2}}$	42.56	85.67	35.76	26.51	13.91	0.00	21.10	70.40	60.18	25.94	55.78	32.27	2.80	2.48	84.58	82.19	48.66	84.42	33.49
ReDAL$_{\text{run 3}}$	44.01	89.19	32.83	41.49	4.71	0.00	28.60	70.51	61.97	24.85	54.89	40.13	5.58	0.00	85.52	82.43	53.42	84.46	31.62
Our GT$_{\text{run 1}}$	41.17	86.34	35.78	33.75	0.96	0.00	0.06	53.20	57.08	29.50	54.75	24.99	4.18	0.00	85.45	85.22	55.58	84.45	49.78
Our GT$_{\text{run 2}}$	40.42	87.57	26.98	30.54	1.47	0.00	0.00	51.03	59.25	25.86	58.11	28.56	7.24	2.04	83.63	84.77	51.41	85.66	43.45
Our GT$_{\text{run 3}}$	39.51	86.91	31.59	27.13	2.36	0.00	0.00	59.05	58.49	25.84	55.90	17.26	3.94	0.00	84.48	84.81	54.45	84.25	34.68
Our GT-DS$_{\text{run 1}}$	42.39	84.30	43.13	30.32	6.56	0.00	6.61	56.01	60.00	28.05	56.68	39.84	8.11	4.86	83.66	81.31	44.16	84.39	45.03
Our GT-DS$_{\text{run 2}}$	40.10	84.65	32.99	26.47	3.57	0.00	10.32	53.04	59.40	27.01	58.71	27.56	0.36	0.25	82.62	82.80	52.79	83.78	35.51
Our GT-DS$_{\text{run 3}}$	39.50	80.22	32.67	30.13	7.00	0.00	14.17	59.08	59.06	22.20	56.39	24.28	6.19	0.15	82.35	81.85	43.15	82.13	29.98
Our GT-W$_{\text{run 1}}$	35.69	85.79	16.35	28.74	4.51	0.00	23.65	33.42	51.01	19.07	50.11	18.24	2.89	2.36	67.10	78.21	45.14	77.00	38.79
Our GT-W$_{\text{run 2}}$	34.44	82.56	12.05	19.65	6.86	0.18	17.71	38.13	47.83	12.39	50.02	17.08	1.94	1.57	71.22	71.95	45.64	80.89	42.32
Our GT-W$_{\text{run 3}}$	35.44	83.92	20.04	28.44	3.65	0.04	18.68	35.79	46.37	15.81	47.92	10.23	5.75	3.03	74.55	76.86	44.43	81.18	41.24
Our GT-WB$_{\text{run 1}}$	42.73	84.35	39.88	45.98	1.43	0.00	9.74	48.24	55.82	13.63	54.76	29.72	10.05	6.79	85.93	81.95	48.81	89.76	62.36
Our GT-WB$_{\text{run 2}}$	41.98	83.82	47.34	46.68	8.07	0.00	12.48	58.36	50.80	13.17	52.92	13.27	1.38	4.60	86.21	82.84	46.67	89.24	57.87
Our GT-WB$_{\text{run 3}}$	40.41	86.46	37.53	42.74	5.76	0.00	10.52	55.64	51.39	13.52	52.77	7.39	0.12	2.22	86.32	83.60	48.98	88.52	53.98
Our GT-PL$_{\text{run 1}}$	41.66	90.75	51.87	48.20	0.93	0.00	4.56	69.05	50.49	2.40	52.44	27.29	0.00	0.00	87.22	82.99	43.82	87.80	50.02
Our GT-PL$_{\text{run 2}}$	41.04	89.48	50.59	45.96	0.20	0.00	3.69	68.40	51.06	5.61	50.60	25.81	0.00	0.00	87.46	83.35	44.10	87.04	45.37
Our GT-PL$_{\text{run 3}}$	40.84	88.77	44.25	48.56	1.01	0.00	3.09	68.82	48.61	8.50	51.42	22.60	0.00	0.00	86.77	83.59	44.02	87.19	47.84
Our GT-PL-WB$_{\text{run 1}}$	45.36	86.62	45.93	47.70	18.85	0.00	12.88	56.40	52.68	18.54	54.29	35.90	0.34	15.87	87.29	84.16	50.45	89.62	58.99
Our GT-PL-WB$_{\text{run 2}}$	42.09	82.74	36.51	39.32	12.78	0.00	11.83	53.11	54.87	14.41	52.78	35.50	0.42	4.09	84.48	83.67	48.76	88.48	53.95
Our GT-PL-WB$_{\text{run 3}}$	41.80	85.79	35.02	49.19	6.16	0.00	11.54	60.56	50.94	15.63	53.45	26.61	0.90	2.72	83.33	81.44	45.37	88.79	54.94
Our MF$_{\text{run 1}}$	49.64	92.82	41.26	32.88	9.31	0.66	52.96	73.36	60.57	19.11	58.85	40.43	17.85	37.24	86.76	84.95	51.83	86.70	45.93
Our MF$_{\text{run 2}}$	50.82	92.71	50.66	53.06	13.59	0.06	47.31	72.00	59.09	24.90	55.99	34.34	14.90	34.45	88.44	85.42	51.44	87.30	49.10
Our MF$_{\text{run 3}}$	50.35	92.09	49.45	44.83	19.23	0.00	51.52	72.70	60.17	17.61	58.36	33.97	15.68	37.22	86.04	82.20	51.68	86.92	46.62

Table B.8: Models performance on 2% of the manually annotated Waymo Open Dataset[11].

	mIoU	Car	Truck	Bus	Other vehicle	Motorcyclist	Bicyclist	Pedestrian	Sign	Traffic light	Pole	Construction cone	Bicycle	Motorcycle	Building	Vegetation	Tree trunk	Road	Sidewalk
ReDAL$_{\text{run 1}}$	51.64	92.23	50.10	51.51	22.18	0.00	43.00	69.28	64.37	28.83	59.74	43.97	17.32	29.51	88.39	84.73	53.07	86.41	44.85
ReDAL$_{\text{run 2}}$	52.93	93.25	56.44	54.23	25.47	0.00	44.74	76.47	62.13	29.55	61.87	37.10	16.33	32.00	89.75	86.50	54.89	86.68	45.37
ReDAL$_{\text{run 3}}$	51.60	93.56	53.84	56.86	23.22	0.00	36.91	75.84	63.45	26.50	62.36	32.75	15.59	24.01	89.57	87.05	54.52	86.34	46.50
Our GT$_{\text{run 1}}$	52.05	89.82	56.45	55.52	10.46	0.04	22.63	70.58	60.74	27.83	62.82	40.09	22.11	38.29	89.96	87.57	55.87	88.60	57.45
Our GT$_{\text{run 2}}$	50.70	88.92	60.44	56.12	9.36	0.00	5.42	68.51	62.66	26.58	61.58	40.79	16.06	34.00	90.74	88.19	56.70	88.36	58.12
Our GT$_{\text{run 3}}$	51.87	90.64	52.28	51.19	14.28	0.00	28.04	69.54	63.94	30.47	61.90	39.31	16.15	32.65	90.87	88.38	57.30	88.15	58.58
Our GT-DS$_{\text{run 1}}$	51.79	89.48	45.74	59.64	13.00	0.00	30.61	63.44	63.91	27.85	62.74	44.76	18.09	35.81	88.93	86.42	56.82	87.07	57.83
Our GT-DS$_{\text{run 2}}$	50.23	88.69	56.21	52.02	15.62	0.00	18.95	64.94	63.79	26.50	61.17	43.38	8.44	26.37	90.18	87.47	58.71	86.39	55.38
Our GT-DS$_{\text{run 3}}$	50.03	89.87	57.41	55.98	19.70	0.00	14.99	65.86	63.46	26.52	60.20	39.63	6.77	29.63	89.07	86.15	56.95	86.41	51.96
Our GT-W$_{\text{run 1}}$	42.08	88.87	37.24	42.86	9.74	0.07	28.97	56.44	50.99	18.68	49.18	18.35	6.33	5.57	82.58	82.77	49.55	83.25	45.93
Our GT-W$_{\text{run 2}}$	39.15	87.42	34.51	44.40	5.67	0.01	18.89	45.88	47.34	19.85	49.04	8.69	11.24	11.99	82.25	82.42	42.61	72.97	39.54
Our GT-W$_{\text{run 3}}$	41.02	89.16	39.97	38.38	8.47	0.01	22.39	52.09	47.50	15.83	49.15	18.68	7.82	5.84	82.34	82.59	51.68	83.32	43.21
Our GT-WB$_{\text{run 1}}$	49.21	90.13	51.91	48.87	11.39	0.00	19.94	66.72	60.23	21.62	57.91	29.94	18.37	27.41	88.94	86.52	54.10	88.98	62.74
Our GT-WB$_{\text{run 2}}$	48.54	86.85	52.04	55.91	13.24	0.00	29.74	56.32	58.35	21.83	56.53	35.81	19.32	14.24	88.29	85.19	52.73	89.45	57.85
Our GT-WB$_{\text{run 3}}$	47.33	88.21	46.48	53.19	11.69	0.00	28.10	61.71	56.37	23.44	56.40	33.26	7.35	17.50	87.81	84.91	47.24	89.80	58.40
Our GT-PL$_{\text{run 1}}$	45.49	91.40	61.86	58.59	23.18	0.00	7.00	72.86	53.61	11.58	53.19	28.79	0.02	0.04	88.09	84.14	45.58	88.05	50.92
Our GT-PL$_{\text{run 2}}$	43.95	90.66	58.45	50.32	6.46	0.00	13.28	72.90	54.31	11.51	53.21	28.25	0.01	0.00	88.58	84.67	45.41	87.27	45.80
Our GT-PL$_{\text{run 3}}$	43.95	90.20	55.35	53.21	13.85	0.00	8.09	73.05	51.90	10.67	52.84	25.12	0.08	0.00	88.13	84.63	46.37	87.71	49.92
Our GT-PL-WB$_{\text{run 1}}$	50.58	89.63	57.08	55.11	15.97	0.00	22.22	64.40	57.32	20.77	57.54	37.00	11.49	34.49	89.35	86.36	53.56	91.49	66.70
Our GT-PL-WB$_{\text{run 2}}$	49.70	87.90	50.41	61.06	13.49	0.00	24.45	69.41	59.51	23.53	53.25	37.97	18.15	17.65	87.94	85.81	48.52	90.82	64.70
Our GT-PL-WB$_{\text{run 3}}$	48.09	88.75	51.13	35.85	17.70	0.00	24.18	70.81	55.23	23.63	55.62	24.62	8.30	26.68	88.91	85.59	53.04	90.70	64.89
Our MF$_{\text{run 1}}$	52.19	91.74	52.88	53.51	12.46	3.67	46.94	72.56	62.35	21.30	60.15	42.65	21.41	35.38	88.55	84.71	51.32	86.91	50.86
Our MF$_{\text{run 2}}$	54.94	94.05	51.97	48.97	14.16	3.77	55.50	76.60	62.82	28.08	59.99	43.47	32.30	39.63	88.53	86.15	53.34	89.29	60.34
Our MF$_{\text{run 3}}$	56.53	94.01	56.49	55.50	13.56	16.45	59.10	77.73	63.90	22.07	61.70	45.39	29.82	41.40	89.17	86.52	53.63	89.74	61.35

Table B.9: Models performance on 3% of the manually annotated Waymo Open Dataset[11].

	mIoU	Car	Truck	Bus	Other vehicle	Motorcyclist	Bicyclist	Pedestrian	Sign	Traffic light	Pole	Construction cone	Bicycle	Motorcycle	Building	Vegetation	Tree trunk	Road	Sidewalk
ReDAL$_{\text{run 1}}$	55.82	94.64	59.24	62.93	21.59	0.00	50.55	76.17	65.07	30.57	63.72	39.97	25.09	35.18	90.78	88.42	59.66	87.83	53.41
ReDAL$_{\text{run 2}}$	57.15	94.46	56.42	66.05	27.04	0.00	51.08	78.00	64.80	28.51	63.13	43.35	23.79	40.81	90.52	87.90	59.42	90.02	63.36
ReDAL$_{\text{run 3}}$	55.70	94.01	56.22	62.47	22.52	0.00	49.16	80.28	66.24	26.18	63.87	40.95	19.50	30.85	91.11	88.62	60.36	89.98	60.35
Our GT$_{\text{run 1}}$	57.76	92.87	62.57	56.58	29.71	0.00	51.44	74.57	65.15	29.08	65.40	44.09	24.79	46.96	91.30	88.55	58.31	91.45	66.92
Our GT$_{\text{run 2}}$	56.32	90.94	57.84	57.98	25.15	0.00	43.60	73.48	64.56	29.52	64.43	42.87	25.59	42.89	91.41	88.62	58.29	90.83	65.84
Our GT$_{\text{run 3}}$	56.11	92.70	56.45	54.46	25.97	0.00	45.10	71.66	66.29	26.30	64.75	42.22	28.72	41.01	91.44	88.84	59.20	90.03	64.76
Our GT-DS$_{\text{run 1}}$	55.18	91.22	56.41	56.42	24.34	0.00	41.96	69.40	65.64	25.23	63.11	41.73	18.62	44.81	90.36	88.16	59.50	90.78	65.58
Our GT-DS$_{\text{run 2}}$	56.49	93.20	61.45	62.75	24.99	0.00	42.74	68.45	64.82	30.46	63.27	40.36	23.65	45.99	91.07	88.57	60.70	90.48	63.85
Our GT-DS$_{\text{run 3}}$	55.70	90.16	61.64	53.15	25.68	0.00	43.49	69.14	65.08	27.72	62.44	43.62	27.31	40.83	91.09	88.31	59.93	89.85	63.21
Our GT-W$_{\text{run 1}}$	46.26	91.51	56.92	55.20	24.29	0.03	29.80	59.23	47.46	18.07	50.52	26.40	14.09	8.00	86.62	84.58	46.81	82.67	50.42
Our GT-W$_{\text{run 2}}$	43.79	91.79	49.06	43.01	18.94	0.04	12.86	59.50	53.10	16.58	54.22	20.54	8.83	5.81	86.68	83.27	50.20	83.62	50.11
Our GT-W$_{\text{run 3}}$	44.73	90.61	50.89	45.54	16.86	0.00	31.46	57.69	54.09	21.27	54.08	14.65	10.64	4.84	85.91	84.09	52.24	82.66	47.63
Our GT-WB$_{\text{run 1}}$	50.66	92.60	53.66	47.79	9.99	0.00	26.77	66.81	60.06	23.36	60.03	28.19	24.15	33.01	88.85	86.64	54.41	90.40	65.16
Our GT-WB$_{\text{run 2}}$	52.49	89.47	61.81	61.31	22.11	0.00	30.86	64.31	61.41	21.82	60.70	41.27	15.96	29.96	88.95	83.71	54.66	91.10	65.38
Our GT-WB$_{\text{run 3}}$	52.66	89.46	51.08	63.54	18.21	0.00	35.64	65.47	59.42	24.39	59.08	42.28	20.70	31.97	88.70	86.02	53.91	91.40	66.56
Our GT-PL$_{\text{run 1}}$	49.33	91.74	61.70	61.04	29.53	0.00	33.88	73.72	56.68	20.66	55.33	29.46	0.42	13.78	88.60	84.73	47.43	88.23	51.02
Our GT-PL$_{\text{run 2}}$	48.07	91.53	62.60	56.11	33.92	0.00	27.70	75.77	55.06	20.99	54.76	28.64	0.27	2.18	89.23	85.25	47.56	87.45	46.32
Our GT-PL$_{\text{run 3}}$	47.34	91.22	61.10	59.82	20.30	0.00	25.94	75.03	53.99	20.21	53.95	27.35	0.32	4.84	88.81	85.19	46.67	87.79	49.66
Our GT-PL-WB$_{\text{run 1}}$	54.68	91.70	63.25	60.28	16.71	0.00	31.20	71.74	63.40	23.57	62.09	42.65	16.58	42.27	90.89	88.07	57.98	92.16	69.75
Our GT-PL-WB$_{\text{run 2}}$	54.75	90.10	57.01	64.21	20.49	0.00	39.13	68.76	63.17	23.59	62.70	37.30	30.16	34.89	89.76	87.34	57.34	91.61	67.95
Our GT-PL-WB$_{\text{run 3}}$	51.62	89.74	50.70	42.33	20.05	0.00	33.95	66.81	59.90	26.03	59.69	42.07	13.13	32.94	89.65	87.04	56.00	91.39	67.66
Our MF$_{\text{run 1}}$	56.48	94.20	57.31	60.36	13.51	0.61	59.87	80.60	64.64	26.21	62.99	41.35	25.66	41.47	90.34	87.88	58.12	89.06	62.38
Our MF$_{\text{run 2}}$	59.12	94.54	58.92	56.65	22.01	15.39	60.36	77.47	64.09	28.03	62.94	48.15	36.20	46.81	90.73	88.42	58.12	90.76	64.53
Our MF$_{\text{run 3}}$	59.46	94.84	59.10	62.53	21.08	3.77	63.37	80.12	64.55	26.41	61.68	45.24	36.18	55.21	90.83	87.93	58.72	91.41	67.41

Table B.10: Models performance on 4% of the manually annotated Waymo Open Dataset[11].