Sparse Data Fusion and Class Imbalance Correction Techniques for Efficient Multi-Class Point Cloud Semantic Segmentation

Ryan Sander
Massachusetts Institute of Technology
77 Massachusetts Avenue, Cambridge MA 02139
rmsander@mit.edu

Abstract

Semantic scene segmentation is a widely-used computer vision technique that is most commonly applied in a 2D RGB imagery setting. However, due to imagery's dense data structure, this computer vision task can rapidly become computationally intractable, particularly for real-time scene segmentation applications such as autonomous driving. With RGB-augmented point cloud data, however, semantic scene segmentation can be applied in a more sparse setting, enabling for more tractable training and inference for time-sensitive applications. We demonstrate baseline binary and six-class road segmentation frameworks using sparse data fusion that achieve 80% and 32% IoU, respectively. Furthermore, semantic segmentation, which is fundamentally an element-wise multi-class classification problem, suffers performance losses from unbalanced datasets. We investigate several machine learning techniques for mitigating class imbalance, namely focal loss, weighted cross-entropy, and transfer learning.

1. Introduction

Recent advances in deep learning and cloud and hardware-accelerated computing have facilitated rapid improvements in semantic scene segmentation, a widely-applicable computer vision task. Namely, semantic scene segmentation is an element-wise multi-class classification problem; the goal of a semantic segmentation prediction system is to correctly predict a discrete class output for each spatial element (such as pixel or point) of the input spatial data it is given. Mathematically:

Given : Input $A \in \mathbb{R}^{N \times D}$, Ground Truth $Y \in \mathbb{R}^{N \times 1}$
Find : Mapping $f : \mathbb{R}^{N \times D} \rightarrow \mathbb{R}^{N \times 1}$ such that $f(A) = Y$

Semantic scene segmentation is a crucial computer vision task that is highly applicable to fields ranging from autonomous vehicles to aerial remote sensing systems. Semantic scene segmentation can provide these systems with perception capabilities that enable for a myriad of intelligent downstream applications, such as localization, tracking, path planning, and area/volume estimates.

One application we choose to focus on through this research is autonomous driving. Effective and efficient scene segmentation can facilitate safer decision-making through faster scene inference on more sparse feature inputs.

Though robust, generalizable, supervised point cloud models have been developed on broad, annotated datasets, quite often a model is needed that is specifically designed to detect, classify, or segment specific instances of potentially rare objects, in which perhaps only a single-digit number of annotated examples exist. Because there are so few annotated examples of these objects of interest, the model cannot only be trained on these examples - this would likely lead to over-fitting. On the contrary, if the model is trained on all the data points without some sort of re-weighting or transfer learning mechanism to learn less common examples, the model will simply learn to always predict only the most statistically-common classes.

With class re-weighting and transfer learning techniques, however, models can be designed that are both robust and highly tuned to detect specific objects of interest. In this work, we study the effect of leveraging transfer learning on an urban autonomous vehicle lidar and RGB imagery dataset to detect instances of "rare" objects by only training on a few instances of these images after training on the subset of data in the dataset that doesn't contain these specific "rare" objects of interest.

2. Related Work

This research aims to build off of previous insights from 3D point cloud scene segmentation and the class imbalance correction techniques described above. Each facet of research that this project builds off of is briefly discussed below:

2.1. PointNet

Qi et al. developed the landmark PointNet neural network framework in 2017, revolutionizing deep learning applications for lidar and other point cloud data types. PointNet's use of the symmetric max pooling function in this framework respects the permutation invariance of the point cloud data type, a constraint which had prevented other network architectures from being used in this domain. Our framework, which uses the PointNet-based PointNet++ architecture, also adheres to this permutation-invariant constraint.

2.2. Multi-modal Semantic Segmentation for Autonomous Vehicles

Data fusion, which is also leveraged in our model, has been used for semantic scene segmentation as well. We aim to expand off of this work, but in the less traditional sense. In [4], and in most data fusion frameworks, the spatial point-wise data is typically discretized into a 2D or 3D voxel grid, and then concatenated with the RGB image to form a four-dimensional RGB-D dataset. Our approach, as we will enumerate below, aims to use data fusion in a more sparse, efficient setting. Though we do not directly compare the results of our sparse data fusion approach to traditional data fusion results, this would be an ultimate goal for further studies.

3. Approach

3.1. Overview

To solve our semantic scene segmentation class imbalance inference problem, as well as to generate a semantic scene segmentation framework that is efficient and amenable for fast inference, we leverage the PointNet++ neural network architecture data for inference on sparsely-fused lidar point cloud and camera RGB imagery. We chose to apply our combined neural network and sparse data fusion framework to Audi's AEV dataset (A2D2).

3.2. A2D2 Dataset

As mentioned above, we chose the A2D2 dataset, an open-source autonomous vehicles dataset from the Audi Electronics Venture Group, for this project. This consists of co-registered and ground truth annotated lidar and RGB imagery, gathered from outdoor driving scenes in several German cities. This dataset consists of more than 40000 ground truth annotated observations, where each observation may consists of an RGB image, a lidar point cloud, and semantic labels that identify the 55 discrete ground-truth classes of each of the points and/or pixels.

3.3. Sparse Data Fusion

Furthermore, we leverage the A2D2 dataset's mapping from point cloud space (x, y, z) to 2D image space (i, j) for sparse data fusion. In some data fusion settings between point clouds and RGB imagery, a point cloud is converted into a 2D or 3D voxel via interpolation and discretization algorithms to create digital terrain, digital elevation, and digital surface models (DTMs, DEMs, and DSMs, respectively), and the data is concatenated into four-dimensional RGB-D data. These types of data fusion approaches requires substantially more computational power to compute and store than ours: rather than interpolating the point cloud to the image size, we sample the RGB image at the N pairs of indices that the point cloud's points map to in (i, j) image space:

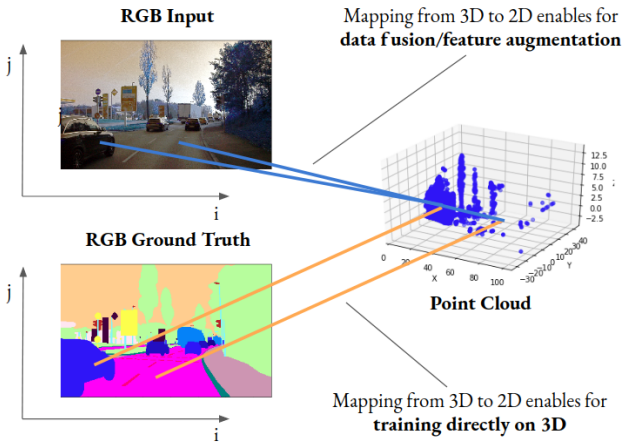

Figure 1. Diagram of our sparse data fusion procedure for augmenting point cloud data with features and ground truth labels.

3.4. Pre-processing Steps

Though the A2D2 dataset was highly curated and well-organized, it still required some processing for our specific application. Due to computational constraints, we chose to analyze only 10000 samples of this dataset from the original 40000 ground truth annotated driving frames. Our pre-processing steps are outlined below:

3.4.1 Point Cloud and RGB Image Normalization

In order to make our sparse fusion segmentation framework generalizable to any point cloud input or coordinate system, we normalized our point cloud inputs to the unit sphere.

Another motivation for this pre-processing step is that the predictions made by our neural network depend largely on the geometric relations between different points in the point cloud. Normalization was accomplished via the following mean-center normalization algorithm:

$$\bar{X}, X \in \mathbb{R}^{N\times 3}; \bar{X} = \frac{X - \mu}{\max_i(||X_i||_2)} \quad (1)$$

Where $\bar{X}$, X, and μ represent the normalized and un-normalized point clouds, and μ represents the mean of all points, respectively.

Additionally, we normalized our RGB values from the $\{0, \ldots, 255\}$ domain to the [0,1] domain in order to make our RGB features on the same scale as our point cloud features. This was accomplished by simply dividing each RGB channel of each pixel by 255. This pre-processing step was used to ensure our data inputs were over the same scale ($[0, 1]$) to prevent the neural network from becoming more responsive to RGB or point cloud data.

3.4.2 Padding via Resampling for Batch Normalization and Data Augmentation

To implement this framework in PyTorch, we needed all of our point cloud inputs for a given batch to have the same dimensionality, which in this case necessitated they have the same number of points. Because our PointNet++ neural network architecture uses batch normalization layers, it was critical that we have a batch size larger than one. Since all of our point clouds contained a different number of points, we needed to use data augmentation. We choose a "padding via resampling" data augmentation technique, in which we re-sample each point cloud from its already-existent points with a uniform distribution until the point cloud has the same number of points as the largest point cloud in its batch. Our motivation behind this data augmentation procedure, as opposed to decimating all point clouds in a batch to the number of points of the smallest point cloud, was to preserve as much information as possible. The aforementioned decimation technique unnecessarily discards valuable, labeled training data, and constrains each batch to train on the smallest point cloud, which in some cases was no more than 55 points (whereas the largest point clouds each contained 15000-20000 points). Our "padding via resampling" method, on the other hand, discards no information, and adds noise to our training procedure (since the sampling is non-deterministic), which in turn makes our semantic segmentation model more generalizable and robust to perturbations in future out-of-sample inputs.

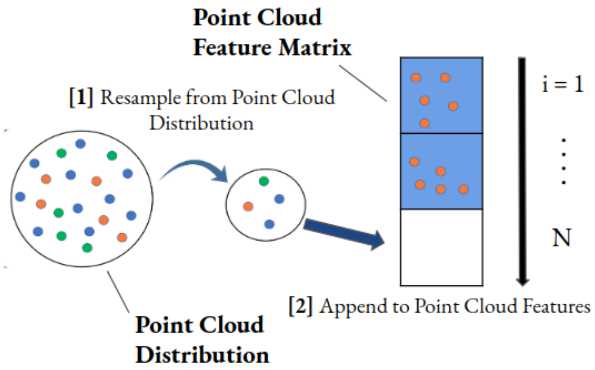

Figure 2. Our *padding via resampling* data augmentation approach uniformly resamples each point cloud from its existing points until the point cloud has the same number of points as the largest point cloud in a training batch. This data augmentation technique was used as a collation function for larger batch sizes in PyTorch.

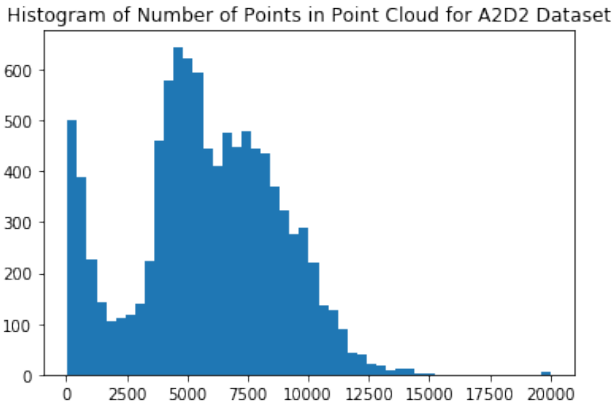

Figure 3. Histogram of number of points per point cloud. Because of the substantial spread in number of points across the A2D2 dataset, decimating our point clouds for batched training (since all point clouds have to be the same size) could result in large losses of training and validation data, which could lead our network to overfit to a small subset of our in-sample data.

3.5. Neural Network Architecture

For our sparse data fusion scene segmentation framework, we leverage an implementation of PointNet++ that is capable of ingesting xyz point cloud data augmented with RGB imagery data.

3.5.1 Motivation Behind Network

As mentioned above, we leverage PointNet++, an extension of the original PointNet network that aims to better capture local features of point clouds. The motivation behind this extension network was that the original PointNet architecture was only able to learn global geometric features of point clouds, and not local ones. Specifically, PointNet++

is a set of multi-scaled, cascaded PointNet neural networks [8], thus enabling learning of both global and local geometric features. As other research in deep learning and neural network architecture design has consistently shown that the ability to detect local features in spatial data is important to overall task performance, the authors modified the original PointNet architecture to develop PointNet++.

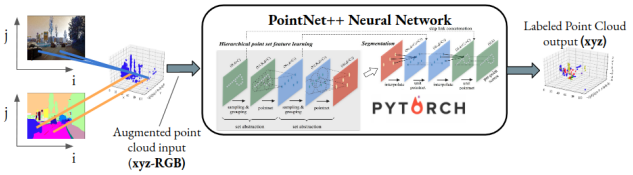

Figure 4. Our customized, sparse data fusion-based PointNet++ neural network architecture.

3.6. Class Imbalance Correction Techniques

In addition to studying the applicability of sparse data fusion as a setting for semantic scene segmentation, the other main research focus of this project was to investigate the effects of various class-imbalance correction techniques on class accuracies and intersection over union (IoU; also known as the Jaccard Index), a useful metric for evaluating the performance of semantic segmentation and object detection systems Intersection over Union, for a given ground truth class c, is defined as:

$$IoU(c) = \frac{|\{i : P(i) = c\} \cap \{(i : L(i) = c\}|}{|\{i : P(i) = c\} \cup \{i : L(i) = c\}|} \quad (2)$$

Where c corresponds to the ground truth class, $P \in \mathbb{R}^N$ corresponds to the predicted label mask for a point cloud with N points (indexed by i), and $L \in \mathbb{R}^N$ corresponds to the ground truth label mask for a point cloud with N points (indexed by i).

With our main metrics for evaluation defined, we are now ready to discuss each of these class imbalance correction frameworks.

3.6.1 Class Aggregation

This implementation is conceptually the most basic, and though it may help to improve overall performance across all classes, it introduces substantial limitations into inference models by essentially decreasing the level of granularity over which the model is designed for. Specifically, this technique simply abstracts classes together into more general hierarchical classes (i.e. the classes "cars" and "trucks" would be merged into the more general "vehicle" class). In section 4.2, our two-class and six-class segmentation models represent class aggregations of our full 55-class segmentation model.

3.6.2 Weighted Cross-Entropy

Another class imbalance correction method that is typically utilized in multi-class classification settings is weighted cross-entropy, which intuitively weights the losses associated with incorrectly predicting rarer examples more heavily than losses associated with more common examples.

$$\text{wNLL}(p, y) = \sum_{t=1}^{C} w_t p_t \log(y_t) \quad (3)$$

Where p_t and y_t denote the probability estimate and ground truth of class t, respectively, C denotes the number of ground truth classes, and w_t denotes the weight for class t, which is typically a ratio of the frequency of class t to the frequency of the most common ground truth class.

For our implementations, we elected to threshold these weights $w_t \leq 10 \ \forall \ t \in |C|$. We did not test the efficacy of this threshold in full, and for a more optimized system, more testing should be conducted on what class weighting threshold, if any, yields optimal behavior.

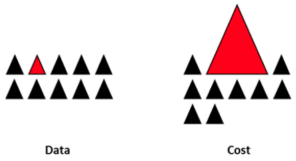

Figure 5. Intuitive notion of weighted cross-entropy: the less frequently a class appears in the training dataset, the larger losses associated with this class should be. Weighted cross-entropy prevents a binary or multi-class classification system from simply predicting the most common class for all predictions.

3.6.3 Focal Loss

One popular technique that has been used in more traditional RGB scene segmentation [3] is *Focal Loss* [6], which we can conceptualize as a variant of weighted cross-entropy for each point (which at this level, becomes multi-class classification). This loss formula is given by the following:

$$FL(p_t) = -(1 - p_t)^{\gamma} \log(p_t) \quad (4)$$

Where p_t in this context refers to the probability of a point being of ground truth label t, and γ refers to a tunable hyperparameter that is used to correct for class imbalance; the larger γ is, the less penalty there will be for guesses on other classes, which prevents a single most frequent ground truth class from dominating the predictions made by the multi-class classification system. We experiment with the use of this loss function below.

3.6.4 Incremental Transfer Learning

The final class imbalance correction approach we leverage is a technique known as Incremental Transfer Learning [1]. The idea behind this approach is similar to that of $N \rightarrow N+1$ Transfer Learning [5], in which a neural network learns an increasingly complex multi-class classification mapping through the addition of single ground truth classes at a time. The technique taken in Abdou et al., which we also investigate in this project, involves training a point-cloud-based semantic segmentation system in stages, beginning with the least represented class, and progressively adding the next-least-represented class after each iteration, until all classes are represented. The approach we take is given below:

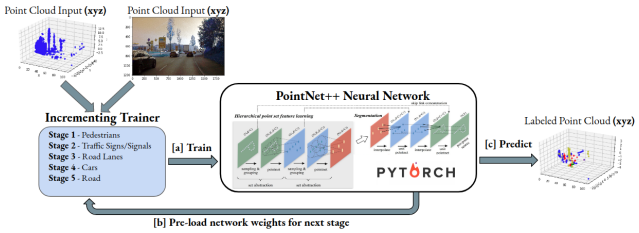

Figure 6. Incremental Transfer Learning framework that we integrate into our sparse data fusion framework. Note that the weights between each training stage are saved, and trained again without any weight freezing [1]

.

4. Implementation

4.1. Computing Environment

Due to the high data volume nature of this project, we utilized Amazon AWS's Elastic Computing Cloud (EC2). Our table of computing specifications is provided below:

Parameter	Specification
Instance Type	p2.xlarge
Operating System	Ubuntu 18.04.3 LTS
Number of GPUs	1
GPU Model	Tesla K80
Machine Learning Library	PyTorch

Table 1. Specifications for our GPU computing environment.

4.2. Segmentation Models

We utilized several different models for demonstrating the effectiveness of sparse data fusion in tandem with a PointNet++ model. These models are briefly discussed below, and are at different granularities of ground truth classes. These classes were aggregated manually through a pre-defined mapping from a set of more granular classes to a set of less granular classes.

4.2.1 Baseline: Binary Road Segmentation

We begin with a model in which our PointNet++ network learned to distinguish between road and non-road objects. Though this framework is by no means amenable to functioning as a fully autonomous guidance and control system, it serves as a good baseline proof of concept for our network, and has applications for vehicular linear dynamical models [2]. Our next level of granularity chooses to focus on six different ground truth classes.

4.2.2 Coarse Autonomous Perception: 6-Class Scene Segmentation

We now increase the complexity of our model to distinguish not only between road and non-road points, but between 4 other classes as well:

Class Name	Total Percentage of PC Points
"Other"	51.1%
"Road"	39.2%
"Vehicle"	6.6%
"Pedestrian"	0.3%
"Road Signs/Signals"	0.7%
"Lanes"	2.0%

Table 2. Ground truth classes in our six-class segmentation model, along with their total percentage of points in the training set.

As we can see, even with just six ground truth classes, there is significant label imbalance. This greatly affects our system's scene segmentation performance.

4.2.3 Full-Class Model: 55 Class Segmentation System

Finally, we wanted to test the capabilities of our sparse fusion system on all 55 ground truth classes present in the A2D2 dataset, which more closely resembles a realistic autonomous driving framework. Class imbalance was even more pervasive here, with some labels having zero instances across our entire 10000 point cloud dataset (e.g. "zebra crossing signs"), while the road class occupied more than half of all segmented ground truth labels.

With our models and class imbalance correction procedures defined, we are now ready to discuss our experiments and results.

5. Results

For evaluating our framework with different class imbalance correction techniques, we will focus chiefly on aggregate accuracy (total fraction of points whose ground truth class was correctly predicted), as well as intersection over union (or the Jaccard Index/Similarity Coefficient), as defined above.

Using our computing environment as described above in tandem with the aforementioned models and class imbalance correction mechanisms, we were able to run the following experiments and achieve the following results:

Model	Avg. Acc. (%)	Avg. IoU
55 classes, wNLL	67.2%	6.0%
55 classes, Focal Loss	64.2%	3.2%
6 classes, wNLL	79.7%	32.5%
6 classes, wNLL, and TL	29.0%	14.2%
2 classes, Road Detection	90.2%	79.9%

Table 3. Table of our results for both accuracy and intersection over union (IoU) metrics. Please note the following abbreviations: wNLL: weighted cross-entropy, TL: transfer learning (incremental transfer learning).

6. Conclusions

Our implementations demonstrate that sparse point cloud data fusion is a promising computer vision setting that is amenable for accurate, real-time semantic scene segmentation. The GitHub repository for this project can be accessed here[1].

6.1. Strengths of Framework

As discussed in the introduction, sparse data fusion can substantially reduce the needed spatial processing for semantic segmentation, making this framework more robust to higher resolution/more data-rich environments than standard RGB image scene segmentation. Our average inference time on each batch of 4 data-fused point clouds was approximately 1.25 seconds, which was achieved through the use of a single Tesla K80 GPU. This rapid inference time makes this framework a viable candidate for real-time semantic segmentation applications, such as autonomous driving. Additionally, our baseline framework, though it can only distinguish between road and non-road classes, performs with 90% accuracy and 80% IoU, which is comparable to state-of-the-art binary road segmentation [7] models.

6.2. Limitations of Framework

Though this sparse data fusion PointNet++ framework performs well for few-class models, such as our binary road segmentation model and six-class segmentation model, this framework significantly lacks robustness to class imbalanced datasets, even after attempted correction via class re-weighting, incremental transfer learning, and focal loss. This under-performance became especially salient after we tested this framework on the A2D2 dataset with all 55 unbalanced ground truth classes.

This sparse data fusion framework enables efficient and accurate real-time inference in semantic scene segmentation settings. Once problems arising from class imbalance are addressed, this framework has even more potential to deliver robust semantic scene segmentation solutions to fields ranging from real-time satellite mapping to autonomous navigation.

[1] https://github.com/rmsander/sparse-fusion-scene-seg

7. Acknowledgements

I would like to thank the 6.869 team at MIT for providing me with critical instruction, support, and AWS access for this project.

References

[1] Mohammed Abdou et al. “Weighted Self-Incremental Transfer Learning for 3D-Semantic Segmentation”. In: (2018).

[2] Jesús Balado et al. “Road Environment Semantic Segmentation with Deep Learning from MLS Point Cloud Data”. In: *Sensors* 19.16 (2019), p. 3466.

[3] Kento Doi and Akira Iwasaki. “The Effect of Focal Loss in Semantic Segmentation of High Resolution Aerial Image”. In: *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*. IEEE. 2018, pp. 6919–6922.

[4] Di Feng et al. “Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges”. In: *arXiv preprint arXiv:1902.07830* (2019).

[5] Ilja Kuzborskij, Francesco Orabona, and Barbara Caputo. “From n to n+ 1: Multiclass transfer incremental learning”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 3358–3365.

[6] Tsung-Yi Lin et al. “Focal loss for dense object detection”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2980–2988.

[7] Gabriel L Oliveira, Wolfram Burgard, and Thomas Brox. “Efficient deep models for monocular road segmentation”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 4885–4891.

[8] Charles Ruizhongtai Qi et al. “Pointnet++: Deep hierarchical feature learning on point sets in a metric space”. In: *Advances in neural information processing systems*. 2017, pp. 5099–5108.