CNN-based Object Segmentation in Urban LIDAR With Missing Points

Allan Zelener
The Graduate Center, CUNY
New York City, USA
azelener@gradcenter.cuny.edu

Ioannis Stamos
Hunter College & Graduate Center of CUNY
New York City, USA
istamos@hunter.cuny.edu

International Conference on 3D Vision (3DV), Stanford University, CA, October 25 - 28, 2016

Abstract

We examine the task of point-level object segmentation in outdoor urban LIDAR scans. A key challenge in this area is the problem of missing points in the scans due to technical limitations of the LIDAR sensors. Our core contributions are demonstrating the benefit of reframing the segmentation task over the scan acquisition grid as opposed to considering only the acquired 3D point cloud and developing a pipeline for training and applying a convolutional neural network to accomplish this segmentation on large scale LIDAR scenes. By labeling missing points in the scanning grid we show that we can train our classifier to achieve a more accurate and complete segmentation mask for the vehicle object category which is particularly prone to missing points. Additionally we show that the choice of input features maps to the CNN significantly effect the accuracy of the segmentation and these features should be chosen to fully encapsulate the 3D scene structure. We evaluate our model on a LIDAR dataset collected by Google Street View cars over a large area of New York City.

1. Introduction

Growing interest in applications including mapping and autonomous vehicle navigation have lead to continued efforts in acquiring large-scale 3D range scans of the navigable world. Many companies have now invested in fleets of cars equipped with LIDAR scanners to acquire the raw 3D data for these applications. This then motivates the development of algorithms to generate semantic information, such as object segmentation, to connect the raw data to higher level applications.

Previously the task of object segmentation in 3D point clouds has been posed as assigning a class label to each 3D point in a scene. However LIDAR scans may be imperfect and contain missing data because certain surfaces such as glossy metal or transparent glass may reflect or refract the emitted light and prevent measurement in a particular scanning direction. But despite there being no valid measurement, the sensor's pose and orientation is still known. In this work our key contributions are highlighting how this information may still be used and reframing the object segmentation task for LIDAR scenes as assigning class labels to each sensor grid point, including both acquired 3D points and missing grid points. We focus specifically on the task of vehicle segmentation in urban scenes due to the ubiquitous presence of vehicles and high frequency of missing points on vehicle surfaces. We also describe the details of our procedure for labeling missing points, which cannot be visualized and labeled in simple 3D renderings of the LIDAR scene.

In this work we propose an end-to-end convolutional neural network (CNN) architecture for predicting semantic labels in LIDAR range scans, this is to our knowledge the first application of this model to large scale LIDAR scenes. Directly applying a standard CNN model is not possible because LIDAR scans contain many thousands of scanlines and the conventional solution of downscaling may impact the real 3D measurements. To resolve this problem another key contribution of this work is to describe a preprocessing pipeline for generating normalized patches of low-level features from large-scale LIDAR scenes. Utilizing this pipeline, diagrammed in Figure 1, we can efficiently train and apply our model despite the irregular dimensions of large LIDAR scenes.

Additionally we experiment with different choices of low-level feature inputs and show that these features can significantly effect the resulting segmentation accuracy. Simply using 3D positions as inputs would be sensitive to the choice of origin in the registered coordinate system and only using depth from the scanner would not fully capture the relative 3D structure between neighboring points and the mobile sensor. By using additional feature maps that model these 3D relations, as well as the positions of missing points, we demonstrate improved overall segmentation accuracy.

We evaluate our proposed model on a large urban dataset acquired by Google Street View cars equipped with LIDAR sensors which we have annotated with over 1000 in-

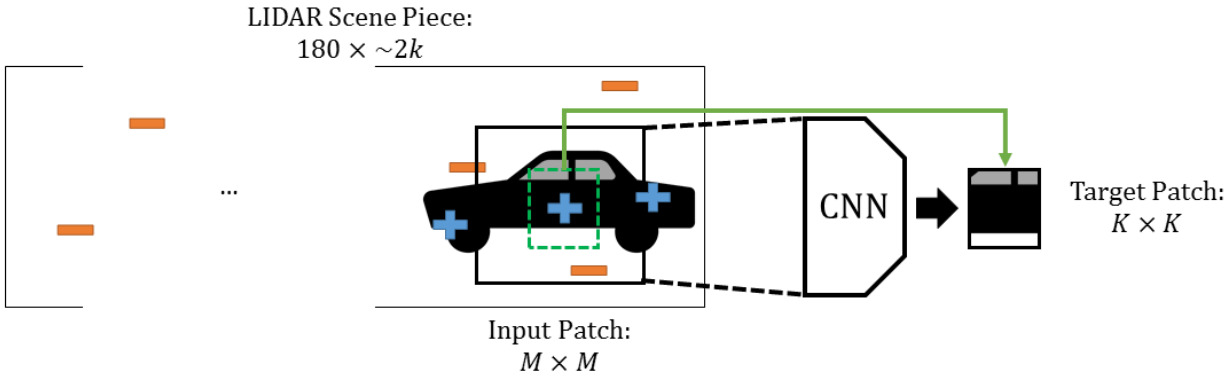

Figure 1. System Overview. During training we sample positive and negative locations in large pieces of the LIDAR scene. For each sampled position we extract an input patch of low-level features and using our CNN model predict labels for a target patch centered on the same location. Note that the gray windows on the car are likely to be missing points and should be labeled with the positive class in our system. At test time we use a sliding window to densely segment a scene.

dividual vehicles. This dataset consists of several runs of LIDAR scans acquired by two sensors mounted on either side of a moving car, stitching together scanlines from the moving sensors produces long push-broom LIDAR scenes. We compare several choices of low-level feature inputs and show that our proposed model can produce accurate vehicle label predictions on both measured 3D points and unmeasured missing points.

2. Related Work

Semantic segmentation and object detection have been extensively covered in the literature and in this section we limit comparison to other works on 3D urban range scans and related CNN approaches in RGB and RGB-D images.

Earlier work on 3D object detection focused on matching handcrafted locally computed 3D features on points with fixed supports such as spin images [15, 22, 19] or shape contexts [9, 31]. While these features are useful for matching object instances using distinctive keypoints [20], for the classification task these features have been combined with bag-of-words representations that model distributions over a dictionary of feature codewords. In this case the local invariance properties of these features may no longer be desirable, leading to the bag-of-words representation being dominated by features common across object categories. For example, since objects are often only partially observed from a limited set of viewpoints some works [24, 28] were able to improve performance by using viewpoint dependent features. Our work shares this more egocentric view of 3D vision by working directly on the scanning grid as opposed to the more common approach of working in a globally registered coordinate system.

More recent works have adopted unsupervised segmentation strategies to generate supersegments of 3D points, similar to superpixels, by clustering 3D points or voxels [11, 34, 21, 8], or fitting geometric primitives like planes [32, 17, 35]. This allows computation of segment-level features that may be more distinctive than local point features. These segments can then be combined in some structure model, for example using graphical models [3, 25, 2] or hierarchical segmentation trees [23, 33] over adjacent segments, to smooth classification results by modeling local context and give a structured representation of the final semantic or instance segmentation. However the limitation of these approaches is that it is difficult to correct for errors in the initial unsupervised segmentation which may propagate to the final segmentation. Also when clustering 3D points, the missing grid points are not considered and often the adjacency information from the scanning grid is discarded. We note that Dohan *et al*. [8] also evaluate their segmentation algorithm on parts of the same dataset as we do, but our results are not directly comparable since they perform additional preprocessing to remove ground and facade planes, which is reasonable for offline processing but would need to be handled differently for online applications. The same kind of removal could be applied as a post-processing step to our system but we wish to demonstrate the results of a fully supervised approach.

In 2D vision, the most recent state-of-the-art results on a variety of classification [16, 14] and segmentation [18, 7] tasks have been achieved using convolutional neural networks. Compared to handcrafted features, CNNs can explore a richer space of feature transformations that are optimized for the target task with the tradeoff that a CNN requires model architecture engineering. While sometimes combined with structure models [6, 26], CNNs can achieve competitive performance when directly optimizing for the task of segmentation.

Corresponding 3D applications of CNNs most related to

the system we present in this paper have been in the domain of RGB-D images, typically acquired indoors using infrared structured light sensors. These systems often leverage the availability of well-registered RGB channels to reuse pre-trained RGB models on much larger datasets. Completely missing depth information is also characterized differently in this setting, typically caused by disparity occlusions due to distances between RGB camera, IR emitter and IR sensor rather than surface material properties. One line of research has focused on unsupervised feature learning [27, 4, 5] due to the limited availability of labeled 3D data, but our work is more comparable to supervised methods. We considered the architecture of Song and Xiao [29] which contains 3D convolutions. Urban LIDAR scenes however are less cluttered than indoor scenes and the cost of 3D convolutions may outweigh the benefit in such sparse conditions, but it is an alternative worth exploring in future work. Most similar to our approach is the depth feature extraction component of Gupta *et al*. [12] that uses a similar set of input feature mappings including depth, height, and angle maps for each pixel in an RGB-D scene. Whereas they restrict themselves to three feature maps in order to initialize a deep model trained on the ImageNet dataset, we build upon this work by showing that additional feature maps can significantly improve performance when trained from scratch and also by explicitly handling missing points from the LIDAR grid.

3. Data Preparation

Here we describe the necessary steps for preparing large LIDAR scenes for use with our CNN model. This includes our procedure of labeling points and missing point positions with the target class label, sampling patches of the scene as input to the CNN, and computing low-level input features on each patch.

3.1. Labeling Procedure

Initially each 3D point is labeled using a 3D interface that allows several freeform tools for the selection of points [30]. For example, in order to simplify labeling objects our tool allows labeling points in a volume above a plane fit to a selection of points, *e.g*. the ground plane. However it is not possible to label missing points in this view, as show in Figure 2 since there is no valid location to render them.

In order to label the missing points once object points have been labeled in 3D, we reproject all the points to have unit distance from the sensor and interpolate the missing points using the known fixed angle between points along each scanline. This gives a well-defined location to render the missing points and produces a circular 2D image of each scanline with respect to the sensor position. Using the 3D object labels as a guide, it is now possible to fill in any missing point positions within an object's surface with the appropriate label, as can be seen in Figure 3.

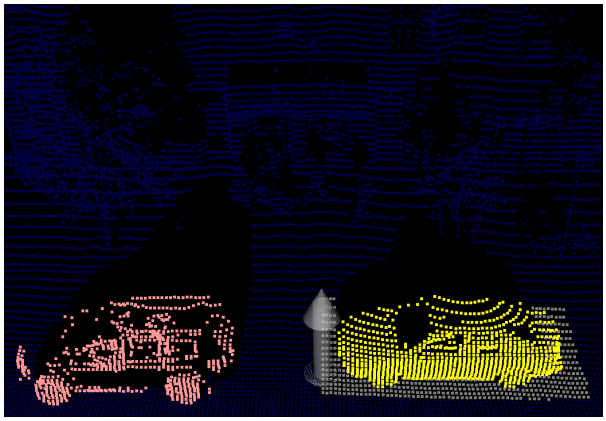

Figure 2. Part of a 3D scene containing two cars. While missing data due to occlusions and sensor range are obvious, it is not entirely clear from this view where missing points are located in relation to 3D points. We also show how selecting all points above a fit ground plane makes it possible to quickly and accurately label the 3D object points.

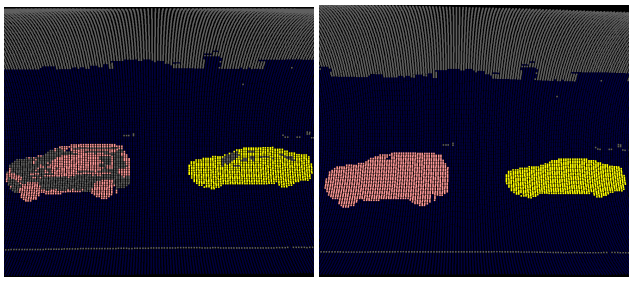

Figure 3. Labeling missing points (best viewed in color). Left: 2D reprojection with missing points on cars and above buildings visualized in gray. Note that some cars only have missing points on windows while others are more heavily effected. Right: Missing points within boundaries of the car are labeled.

3.2. Patch Sampling

Since an individual LIDAR scene may be many thousands of scanlines long, it is necessary to generate input of an appropriate size for the CNN. We avoid interpolated resizing, common in RGB images, since it can distort the 3D points and interact with the missing points.

We first break each full LIDAR run into smaller pieces of several thousand scanlines, avoiding segmenting target objects when possible. During training, for each of T pieces we sample up to $\frac{N}{2}$ positions that contain a target label and $\frac{N}{2}$ unlabeled background positions. This biased sampling helps balance the distribution of positive and negative samples for training a standard classifier, which is necessary in our case since vehicle points are a minority of scene points.

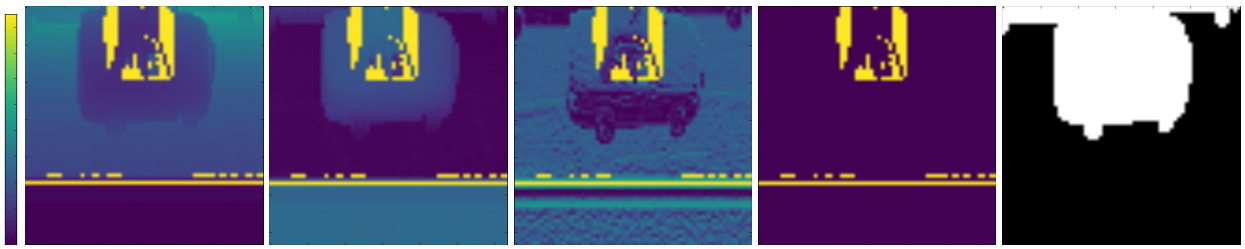

Figure 4. Input low-level features. Color values from navy (low) to yellow (high) follow the *viridis* color map shown on the far left. Left to right: Depth, height, signed angle, missing mask, and ground truth labels in black and white.

The size of one training epoch is set to be NT. Centered on each sampled position we generate an $M \times M$ patch of input features and a $K \times K$ patch of labels where $K \leq M$. These patches are continuously generated throughout the training procedure and minibatches of size 32 are dequeued from a random shuffle queue. In order to reduce preprocessing computation and memory usage, one set of NT sample patches is reused for 20 training epochs but is still randomly shuffled between epochs.

3.3. Input Features

Since 3D point positions vary throughout a scene depending on the global coordinate system, it becomes necessary to generate normalized features for each patch independent of the sampled position. Similar to Gupta *et al*. [12] we adopt depth from the sensor and height along the sensor-up (vertical) direction as reliable measures, although our view of height is more egocentric based on relative heights within a patch rather than geocentric and fixed to the ground elevation. In addition to the angle feature of [12] which measures the angle of elevation between a point and the sensor position, we also consider the signed angle feature described in Stamos *et al*. [30] which measures the elevation of the vector formed by two consecutive points and indicates the convexity or concavity of three consecutive points. While the signed angle requires additional points as support and may be undefined for some points due to missing data and edge cases, it may be a better measure of local curvature than the sensor position dependent angle of [12].

Each of the depth, height, angle, and signed angle features is normalized by the mean and standard deviation of the non-missing points within each patch and clipped to the range $[-6, 6]$ to control for outliers. Most points normally lie well within this range however some patches may be cropped in a way that outlier points very close or very far from the sensor take extreme values. This produces similar features throughout the scene, regardless of the average distance of surfaces within a patch from the sensor. For missing points their value is set to 6, the maximum value in the clip range.

The final feature we consider is a simple $0/1$ mask indicating which scanning grid locations correspond to the missing points. This results in an $M \times M \times 4$ patch of features as input to the CNN. An example set of features for a given patch can be seen in Figure 4.

4. Model

Our model follows a now commonly used architecture for convolutional neural networks. In this section we describe the layout of our model's architecture, the regression parameterization for segmentation, and details of the optimization including regularization and initialization methods. This model establishes a baseline for training end-to-end CNNs on urban LIDAR scenes.

Our model consists of a sequence of convolutional layers with max-pooling followed by a sequence of fully connected linear layers. In our experiments we set the number of layers to two 5×5 convolutional and two linear layers. This model is relatively shallow compared to modern 2D image based models, but was useful in establishing a baseline and experimenting with various low-level feature inputs since best practices for LIDAR images are not yet established.

In order to accomplish single class segmentation our model predicts a $K \times K$ block of labels for a window of points centered on the $M \times M$ input patch. We parameterize this as K^2 independent binary classification tasks utilizing logistic regression on the final representation produced by the CNN on the entire $M \times M$ patch. We typically set $K < M$ so we only predict on locations that have sufficient feature support within the $M \times M$ patch. In preliminary experiments errors were more likely on the edges when using the entire $M \times M$ window. To label a full scene we simply use a sliding window with stride K. In this way, different choices of M and K allow trade-offs between CNN model size and number of evaluations needed to cover an entire scene.

The total loss of the model is the sum of the binary cross entropy losses of each logistic regression plus an L2-regularization penalty on the weights of the linear layers

with scaling coefficient $\lambda = 0.001$,

$$-\sum_{k=1}^{K^2} y_k \log(p_k) + (1 - y_k) \log(1 - p_k) + \frac{\lambda}{2} \sum_{l=1}^{L} ||W_l||_2^2,$$

where y_k is 1 if the kth point in the target grid is positive and 0 otherwise, p_k is the probability of the kth point being the positive class, and W_l are the weights of the lth linear layer.

For additional regularization we also apply dropout with $p = 0.5$ on the final layer weights. The weights for the layers with ReLU activation are initialized using the method of He *et al*. [13] and the weights for the final layer with sigmoid activation use the Xavier initialization [10]. The model is trained by stochastic gradient descent with momentum of 0.9 and initial learning rate 0.01. The learning rate is decayed using an exponential schedule every 350 epochs by a rate of 0.95.

5. Experiments

Our proposed model is evaluated on a subset of the large R5 Google Street View dataset which includes a collection of 20 runs through lower Manhattan covering approximately 100 city blocks, that we have annotated with vehicle labels. This dataset was acquired by cars equipped with two LIDAR sensors on either side of the car capable of acquiring 180 point scanlines with points in 1 degree increments. A summary of the labeled data, containing over 40 million grid points, can be seen in Table 1.

For training we use the majority of the run denoted NYC 0, a relatively large run containing many vehicles, consisting of $T = 25$ pieces while reserving two of the pieces for in-sample testing. Note that these in-sample test scenes are not included in our quantitative results since the relatively superior results on these scenes may be due to in-sample bias. We set the number of samples per piece to $N = 256$ to allow for one training epoch's worth of data to fit comfortably in memory. For these experiments we set the patch size to $M = 64$ and the target size to $K = 8$. We use the TensorFlow [1] library to define out models and train each model for $10,000$ epochs which takes approximately 28 hours on a single Titan X GPU.

To evaluate how our approach generalizes to completely new scenes we use three relatively smaller runs with fewer vehicles denoted NYC 1, 11, and 19. See Table 1 for a full summary of the number of vehicles and scanlines for each of these runs.

We train a new model for each of a select number of combinations of the input feature maps that we consider: normalized depth (D), normalized relative height (H), angle with up-axis (A), signed angle (S), and missing mask (M). A summary of all the models we've trained can be found in Table 2. We note that DHASM, the model containing all of the candidate feature maps, yields the highest average precision score. However it has a relative dip in performance compared to many of the other models in a narrow range towards the middle of the curve, as seen in Figure 5. It is difficult to explain this behavior but one possibility is that

Run + Side	Vehicles	Scanlines
NYC 0 Side 1	295	41572
NYC 0 Side 2	359	41572
NYC 1 Side 1	138	78101
NYC 11 Side 1	126	12787
NYC 11 Side 2	77	12787
NYC 19 Side 1	49	19144
NYC 19 Side 2	37	19144
Total	1081	225107

Table 1. Number of labeled vehicles and scanlines per run. Note that only parts of Side 1 of NYC 1 were used in our experiments. Despite having the most scanlines, NYC 1 does not contain proportionally more vehicles.

Features	Test AP
D	77.49
DHA	86.40
DHS	84.54
DHAM	84.72
DHSM	86.58
DHASM	**86.74**

Table 2. Average precision of different feature combinations. D denotes depth, H denotes height, A denotes angle, S denotes signed angle, and M denotes the missing mask. The model containing all feature maps gives the best overall performance.

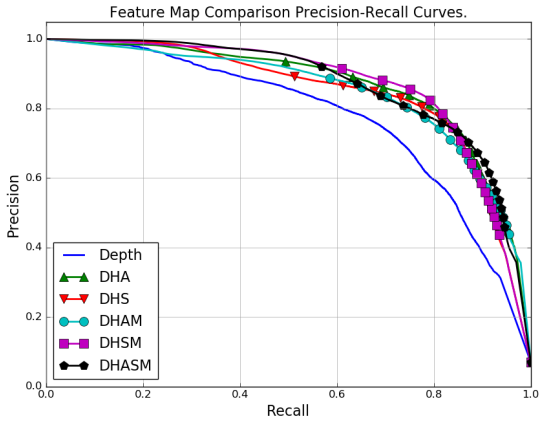

Figure 5. Precision-Recall curves for feature map comparison. The top performing combinations of features throughout all possible sensitivity settings are DHSM and DHASM, which utilize our proposed signed angle and missing mask feature maps.

Features	Test AP
DHSM-NML	82.71
DHSM	84.80
DHASM-NML	83.85
DHASM	**84.92**

Table 3. Average precision on non-missing labeled points only. NML denotes a model trained with no missing point labels for the vehicle class.

the fixed size of the CNN across our experiments does not have enough capacity to fully utilize the increased number of input feature maps. The next best model in this region, DHSM, uses our proposed signed angle feature and perhaps indicates some confusion caused by the presence of both angle based representations. We would also like to note here that while our evaluation may not be directly comparable to Dohan *et al*. [8] due to differences in preprocessing and subsets of the dataset used, our approach clearly outperforms their point-level segmentation baseline and is comparable to their full approach without explicitly generating hypothesis segmentations, demonstrating the strength of CNN-based feature representations.

Additionally we tested the efficacy of labeling missing points by comparing our top two models against equivalent versions trained without labels for the missing points. To fairly compare these models we only consider predictions on the non-missing points. As can be seen in Table 3, simply training a model with labeled missing points leads to a significant increase in average precision scores even on those points that are not missing themselves. Again the full combination of features, DHASM, suffers a relative dip in the curves in Figure 6, but the best models to use at all settings are still those trained with missing labels.

In order to generate visualizations for a qualitative evaluation we selected the DHASM model and chose a confidence threshold corresponding to a recall of 0.85 on the test set, this setting resulted in a threshold of 0.46 with test precision 0.73. We observed high quality segmentation on our relatively simple in-sample test scenes as shown in Figure 7. General quality of segmentation for small conventional vehicles were maintained for out-of-sample scenes, e.g. Figure 8, but additional errors were introduced due to the higher frequency of more challenging vehicles like trucks and previously unobserved styles of facades and vegetation.

6. Conclusion and Future Work

In this work we have presented a CNN model and training pipeline for segmentation of large scale LIDAR scenes acquired by vehicle-mounted sensors. In our evaluation we have shown that our model which has been designed to explicitly handle missing points through additional labeling is able to produce a superior segmentation versus an equivalent model without these labels. Furthermore we've shown that the choice of input features is a significant factor for this task and utilizing the additional feature maps we propose that represent the missing points and 3D structure of the scene can further improve performance.

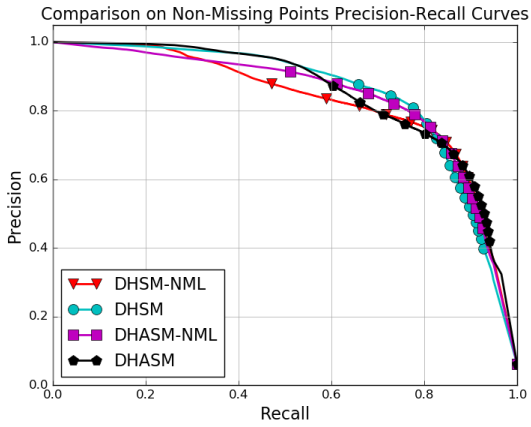

Figure 6. Precision-Recall Curves for comparing efficacy of missing point labels. Here we see that models trained with missing point labels generally outperform those models without those labels, even on the non-missing points.

There exist several directions for future work. Further experimentation with additional feature maps can refine which features are necessary to sufficiently model the 3D structure of the scene for a CNN with 2D spatial convolutions. We would also like to address additional tasks with this model, including multi-class object-level segmentation and pose estimation. It may also be possible to further model missing points by measuring their true values in controlled scans or synthetic data in order to impute depth values for missing points.

Acknowledgement

We express our gratitude to Google for providing funding and data (our special thanks to Steve Hsu for attending to all of our needs, as well as Art Pope, Aleksey Golovinskiy and Vivek Verma). We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan-X GPU used for this research. We also thank Thomas Flynn for developing the labeling software and Ilona Michalowska, for additional development and for leading the labeling effort, as well as James Kluz, Ahmed Elsaeyed, and Himanshu Tanwar for carefully labeling the dataset. Finally, this work was also partially supported by CUNY bridge and PSC-CUNY funds.

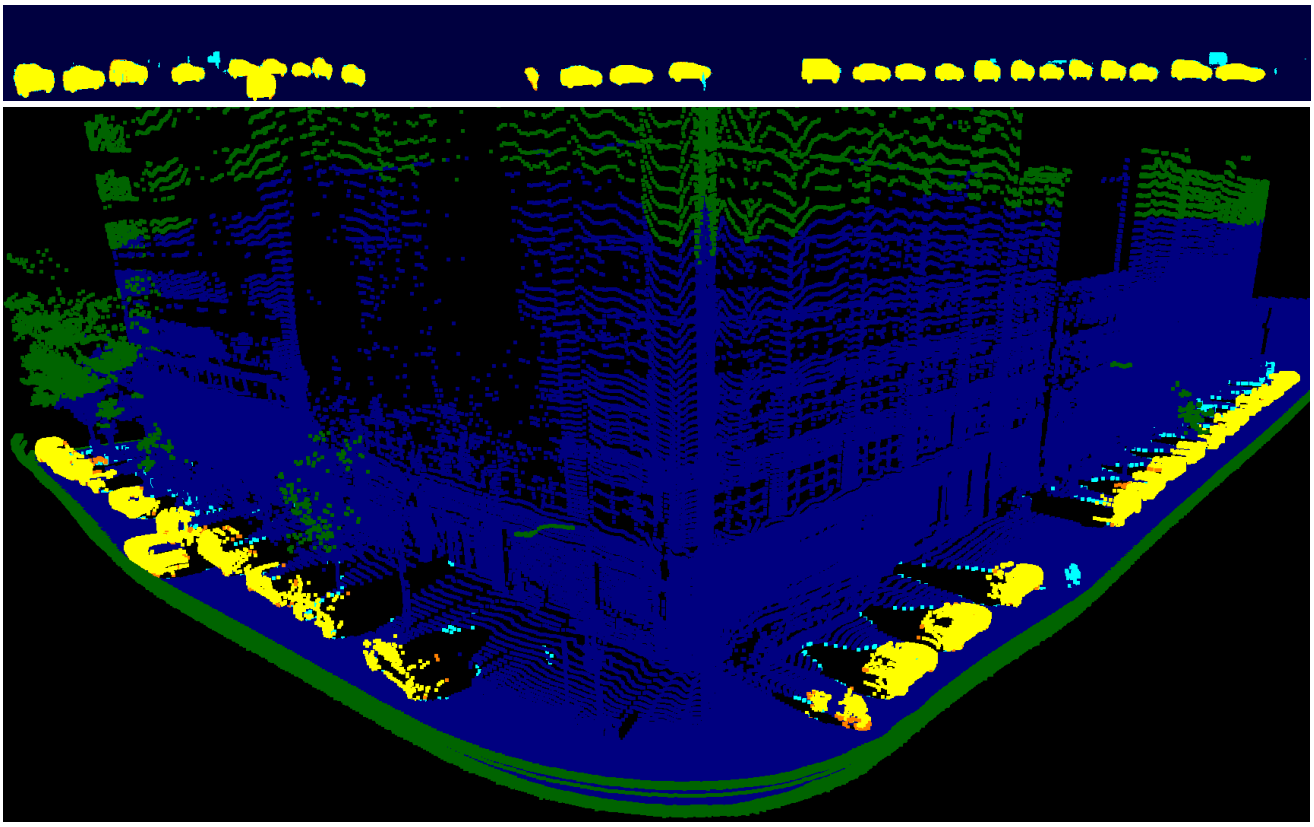

Figure 7. Results on NYC 0 in-sample test scene. Colors correspond to True Positives - Yellow, True Negatives - Dark Blue, False Positives - Cyan, False Negatives - Orange. Green denotes boundary points that were not classified. Top: 2D representation of test scene showing classification including missing points. Bottom: The same scene rendered in 3D. For this scene most errors correspond to either object boundaries or small foreground objects like phone booths or people that may look locally similar to a partially observed vehicle in LIDAR.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. 5

[2] A. Anand, H. S. Koppula, T. Joachims, and A. Saxena. Contextually guided semantic labeling and search for 3D point clouds. *Int. J. Rob. Res.*, 32(1):19–34, Jan. 2013. 2

[3] D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, and A. Ng. Discriminative learning of markov random fields for segmentation of 3D scan data. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 169–176. IEEE, 2005. 2

[4] L. Bo, X. Ren, and D. Fox. Unsupervised feature learning for RGB-D based object recognition. In *Experimental Robotics*, pages 387–402. Springer, 2013. 3

[5] Y. Cheng, R. Cai, X. Zhao, and K. Huang. Convolutional fisher kernels for RGB-D object recognition. In *3D Vision (3DV), 2015 International Conference on*, pages 135–143. IEEE, 2015. 3

[6] C. Couprie, C. Farabet, L. Najman, and Y. LeCun. Indoor semantic segmentation using depth information. In *International Conference on Learning Representations (ICLR)*, 2013. 2

[7] J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. *Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[8] D. Dohan, B. Matejek, and T. Funkhouser. Learning hierarchical semantic segmentations of lidar data. In *3D Vision (3DV), 2015 International Conference on*, pages 273–281. IEEE, 2015. 2, 6

[9] A. Frome, D. Huber, R. Kolluri, T. Bulow, and J. Malik. Recognizing objects in range data using regional point descriptors. In *European Conference on Computer Vision (ECCV)*, pages Vol III: 224–237, 2004. 2

[10] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, volume 9, pages 249–256, 2010. 5

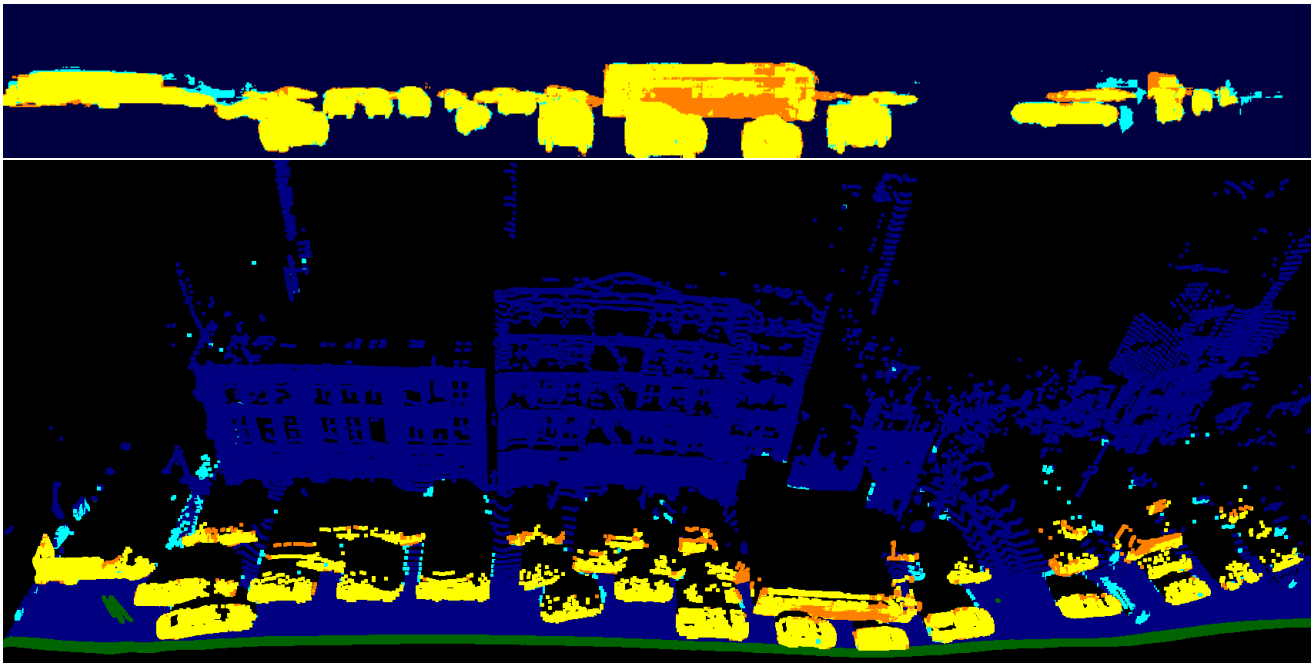

Figure 8. Results on NYC 1 out-of-sample test scene. Color key is the same as Figure 7. Relatively high accuracy is still maintained on this challenging high traffic out-of-sample test scene. Notable mistakes in this scene include parts of large vehicles, like trucks and buses, with mostly planar surfaces that may look locally similar to facades, as well as impatient pedestrians crossing the street through traffic.

[11] A. Golovinskiy, V. G. Kim, and T. Funkhouser. Shape-based recognition of 3D point clouds in urban environments. In *2009 IEEE 12th International Conference on Computer Vision*, pages 2154–2161. IEEE, 2009. 2

[12] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik. Learning rich features from RGB-D images for object detection and segmentation. In *European Conference on Computer Vision (ECCV)*. Springer, 2014. 3, 4

[13] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034, 2015. 5

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[15] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):433–449, 1999. 2

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 2

[17] K. Lai and D. Fox. Object recognition in 3D point clouds using web data and domain adaptation. *The International Journal of Robotics Research*, 29(8):1019–1037, 2010. 2

[18] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015. 2

[19] B. Matei, Y. Shan, H. Sawhney, Y. Tan, R. Kumar, D. Huber, and M. Hebert. Rapid object indexing using locality sensitive hashing and joint 3D-signature space estimation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 28(7):1111–1126, July 2006. 2

[20] A. Mian, M. Bennamoun, and R. Owens. On the repeatability and quality of keypoints for local feature-based 3D object retrieval from cluttered scenes. *International Journal of Computer Vision*, 89(2-3):348–361, 2010. 2

[21] D. Munoz, J. A. Bagnell, and M. Hebert. Co-inference machines for multi-modal scene analysis. In *European Conference on Computer Vision (ECCV)*, 2012. 2

[22] A. Patterson, P. Mordohai, and K. Daniilidis. Object detection from large-scale 3D datasets using bottom-up and top-down descriptors. In *European Conference on Computer Vision (ECCV)*, pages 553–566, 2008. 2

[23] X. Ren, L. Bo, and D. Fox. RGB-(D) scene labeling: Features and algorithms. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2759–2766. IEEE, 2012. 2

[24] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu. Fast 3D recognition and pose using the viewpoint feature histogram. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 2155–2162. IEEE, 2010. 2

[25] N. Silberman and R. Fergus. Indoor scene segmentation using a structured light sensor. In *Computer Vision Workshops*

(ICCV Workshops), 2011 IEEE International Conference on, pages 601–608. IEEE, 2011. 2

[26] N. Silberman, D. Sontag, and R. Fergus. Instance segmentation of indoor scenes using a coverage loss. In *European Conference on Computer Vision (ECCV)*, pages 616–631. Springer, 2014. 2

[27] R. Socher, B. Huval, B. Bath, C. D. Manning, and A. Y. Ng. Convolutional-recursive deep learning for 3D object classification. In *Advances in Neural Information Processing Systems*, pages 665–673, 2012. 3

[28] S. Song and J. Xiao. Sliding shapes for 3D object detection in depth images. In *European Conference on Computer Vision (ECCV)*, pages 634–651. Springer, 2014. 2

[29] S. Song and J. Xiao. Deep sliding shapes for amodal 3D object detection in RGB-D images. *Computer Vision and Pattern Recognition (CVPR)*, 2016. 3

[30] I. Stamos, O. Hadjiliadis, H. Zhang, and T. Flynn. Online algorithms for classification of urban objects in 3D point clouds. In *International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission*, 2012. 4

[31] F. Tombari, S. Salti, and L. Di Stefano. Unique signatures of histograms for local surface description. In *European Conference on Computer Vision (ECCV)*, pages 356–369. Springer, 2010. 2

[32] A. Toshev, P. Mordohai, and B. Taskar. Detecting and parsing architecture at city scale from range data. In *Computer Vision and Pattern Recognition (CVPR)*, 2010. 2

[33] C. Wu, I. Lenz, and A. Saxena. Hierarchical semantic labeling for task-relevant RGB-D perception. In *Robotics: Science and Systems (RSS)*, 2014. 2

[34] X. Xiong, D. Munoz, J. A. Bagnell, and M. Hebert. 3-D scene analysis via sequenced predictions over points and regions. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011. 2

[35] A. Zelener, P. Mordohai, and I. Stamos. Classification of vehicle parts in unstructured 3D point clouds. In *3D Vision (3DV), 2014 International Conference on*, volume 1, pages 147–154. IEEE, 2014. 2