POINT-WISE CLASSIFICATION OF HIGH-DENSITY UAV-LIDAR DATA USING GRADIENT BOOSTING MACHINES

E. Sevgen*, S. Abdikan

Dept. of Geomatics Engineering, Hacettepe University, 06800, Beytepe, Ankara, Turkey
(eraysevgen, sayginabdikan)@hacettepe.edu.tr

KEY WORDS: UAV-LiDAR, multi-scale Hand-crafted features, Point Cloud, Gradient Boosting Machine, LightGBM, Classification

ABSTRACT:

Point-wise classification of 3D point clouds is a challenging task in point cloud processing, whereas, in particular, its application to high-density point clouds needs special attention because a large number of point clouds affect computational efficiency negatively. Although deep learning based models have been gaining popularity in recent years and have reached state-of-the-art results in accuracy for point-wise classification, their requirements of the high number of training samples and computational resources make those models inefficient for high-density 3D point clouds. However, traditional machine learning classifiers require less training samples, so they are capable of reducing computational requirements, even considering the latest machine learning classifiers, particularly in ensemble learning of gradient boosting machines, the results can compete with deep learning models. In this study, we are studying the point-wise classification of high-density UAV LiDAR data and focusing on efficient feature extraction and a recent state-of-the-art gradient boosting machine learning classifier, LightGBM. Our proposed framework includes the following steps: at first, we are using point cloud sampling for creating sub-sampled point clouds, then we are calculating the features based on those scales implemented on GPU. Finally, we are using the LightGBM classifier for training and testing. For the evaluation of our framework, we used a publicly available benchmark dataset, Hessigheim 3D. According to the results, we achieved an overall accuracy of 87.59% and an average F1 score of 75.92%. Our framework has promising results and scores closer to deep learning models. However, more distinctive features are required to obtain more accurate results.

1. INTRODUCTION

Three-dimensional (3D) point clouds are one of the main data sources in remote sensing and related fields. It is a common practice to use 3D point cloud-derived products (e.g., Digital Terrain Models or Digital Surface Models) as an additional dataset for height information along with remote sensing images. On the other hand, their direct processing has been gaining its popularity because *i*) denser and more accurate 3D point clouds are available, *ii*) more information can be obtained from 3D point clouds, *iii*) more representative objects are possible in 3D point clouds, *iv*) emerging fields (such as autonomous cars) require direct 3D point cloud processing. One of the foremost steps of point-cloud processing is labeling each point in the point cloud into a predefined semantic category, namely point-wise classification. A variety of remote sensing applications are based on point-wise classification, such as change detection, ground filtering, and urban monitoring. Thus, point-wise classification for 3D point clouds is an important topic in 3D point cloud processing and is an open area considering recent emerging fields.

3D point clouds include a few problematic issues: *variable size of objects*: a variety of objects exist in the data, for example cars and buildings may need different considerations for feature extraction and object recognition; *heterogeneous density and incomplete objects:* because of occlusion, a part of the objects may not be represented in the point clouds, whereas heterogeneous density is a natural property of point clouds; *irregular structure:* a point cloud irregularly represents objects on the surface; *high density:* since advancements in sensor technology and methods, a high number of points are possible in point clouds, which makes the processing computationally burdensome. In this study, we propose a feature calculation framework considering different sizes of objects in high-density UAV LiDAR dataset. A machine learning (ML) classifier then labels each point in the point cloud using those features in our framework.

ML based models have started to play a key role in the literature for point-wise classification of 3D point clouds similar to other remote sensing tasks in recent years. Even though deep learning (DL) of neural networks based end-to-end approaches, which profit from automatic feature extraction and selection, have started to dominate 3D point cloud processing, hand-crafted features still maintain their importance in point-wise classification, since they have generally been preferred as an additional dimension for increasing accuracy in many DL models. Instead of using them in a deep framework, which requires high computational resources and large amounts of training samples, a careful selection of neighborhood definitions and sampling strategies is going to reduce the cost of deep models and approaching state-of-the-art results, as deep models have. Therefore, in this study, we use a recent state-of-the-art ML classifier using hand-crafted features for point-wise classification.

At first, our study efficiently extracts the features from a high-density 3D point cloud dataset. Since multiple scales allow one to aggregate more information from a variety of sizes of objects in the dataset, we are following a multi-scale feature extraction approach. Using a single type of neighborhood, we extract the features for each level prioritizing the computational cost. Second, we consider a recent ensemble learning method in ML, particularly gradient-boosting machines, which is increasing its popularity in tabular data processing, even superseding deep

* Corresponding author

models. We take into account their efficiency as well as their performance and high potential usage in point cloud analysis. In the last step, we compare our results with DL models on a UAV-LiDAR-based benchmark dataset.

The rest of the study is organized as follows: we present the related work in Section 2, then we describe our methodology in Section 3. In Section 4, we define our dataset and give our experimental results on this dataset. We compare our results with previous studies and summarize the study in Section 5.

2. RELATED WORKS

There exist various studies for point-wise classification of 3D point clouds using a type of ML method. Those ML approaches can be categorized into traditional models, where features are extracted manually and then a classifier is applied, and deep models, where feature extraction and classification are performed together without any requirement of feature engineering. One of the review studies (Xie et al., 2020) gives overall literature on the 3D point cloud classification, and we give a brief overview of the 3D point cloud classification based on the above categorization.

A typical traditional ML framework for 3D point cloud classification includes a few steps: neighborhood definition, feature extraction and selection, and classification. Therefore, the 3D point cloud classification using ML methods follows this framework based on the different aspects of the above steps (Weinmann et al., 2015).

Taking into account high-density LiDAR datasets, multi-scale feature extraction has been performed using sampling techniques. For this purpose, voxel sampling with the k-nearest neighbor search (Hackel et al., 2016) and grid sampling with radius search (Thomas et al., 2018) have previously been used in the classification of point clouds. The disadvantage of these sampling methods is that they produce regular samples from the irregular point cloud. However, irregular sampling methods produce more representative details. Random sampling, the cheapest method for memory and computational complexity, is the fastest method for irregular sampling; thus, it may cause losing important information, which leads to fewer distinctive features. Instead, Poisson sampling still produces irregular samples but also takes into account the spatial distribution of the points on the surface.

A few neighborhoods have been proposed in the literature. Spherical and k-nearest neighbors are the most commonly used types, where the points in a distance of the sphere are of interest and a particular number of neighbors are found without radius. A combination of those two is popular in DL frameworks and guarantees the k number of points in an r radius. In addition to that, the cylindrical neighborhood is preferred (Niemeyer et al., 2014), which is a 2D projected version of the 3D radius search. The important factor in the neighborhood definition is to select a suitable parameter, which is r or k. Once selected as a large value, the computational cost is increasing, while it can be small and details may be lost. Empirical selection is generally preferred in the literature and the selection depends on the size of objects in the dataset.

In addition to those traditional ML models, deep neural networks have made a tremendous attack on 3D point cloud analysis in recent years. It is common to create feature images from point clouds and use them in a regular Convolutional Neural Network (CNN); moreover, voxelization is another approach for data regularization and is preferred in the literature. Raw point cloud based methods, on the other hand, allow one to directly process point cloud data in a DL model. At first, the study of PointNet (Qi et al., 2017a) has shown the permeation invariant neural network for direct 3D point cloud processing in DL, but it lacks hierarchical information aggregation; therefore, the same authors proposed the PointNet++(Qi et al., 2017b) deep network proposing hierarchical feature extraction by down/up sampling and information aggregation. Then, inspired by the convolution of the images, KpConv (Thomas et al., 2019), proposes learning the local structure of the point clouds in the kernel. The advantage of this type of neural networks is that they can capture more local information than PointNet++ type deep models. More details on DL models in 3D point cloud classification are presented in (Zhang et al., 2019; Bello et al., 2020; Guo et al., 2020).

In this study, we are considering the efficient processing of high-density UAV-LiDAR data using a traditional ML framework. The selection of the classifier was performed according to the latest state-of-the-art ML classifiers. We also compare the results with deep models, and take one of the highly used ML classifiers in the 3D point cloud processing, Random Forest, as a baseline.

Our study mainly differs from the previous studies (Thomas et al., 2018; Hackel et al., 2016) based on our subsampling technique, where our output is still irregular data format and the ML classifier, where we are using gradient boosting based ensemble learning method.

3. METHODOLOGY

We are following a traditional ML framework for 3D point cloud classification, focusing on the efficiently multi-scale feature extraction for a high density UAV-LiDAR point cloud and the application of one of the gradient boosting machine classifiers, LightGBM, to the extracted feature set. The proposed framework is shown schematically in Figure 1. It includes the steps: *i*) in the first step, the data have been sampled using Poisson sampling, where in each step the number of points is reduced based on the predefined distance; *ii*) multi-scale features have been computed for each scale based on radius neighborhood; *iii*) in the last step, the LightGBM classifier has been applied to the training data, where it is important to note that considering the class imbalance, an equal number of points are selected in the data. The details of the proposed framework are discussed in the following paragraphs.

Neighborhood determination is a time-consuming operation in feature extraction; considering dense data, it is computationally cumbersome. On the other hand, all points in a neighborhood are not needed for the feature calculation process. Therefore, it is common to work with sub-sampled versions of the point cloud data, while it is a factor for the reduction of computational cost, and at the same time may cause information loss. As shown in Figure 2, the efficiency is obtained by sampling.

In the literature, there exist a few sampling methods based on these categories: *Regular* and *Irregular*. Regular sampling methods produce regular sampling from irregular point clouds. 3D voxelization is commonly used in DL, even for regularizing the point cloud data for further CNN-like neural networks.

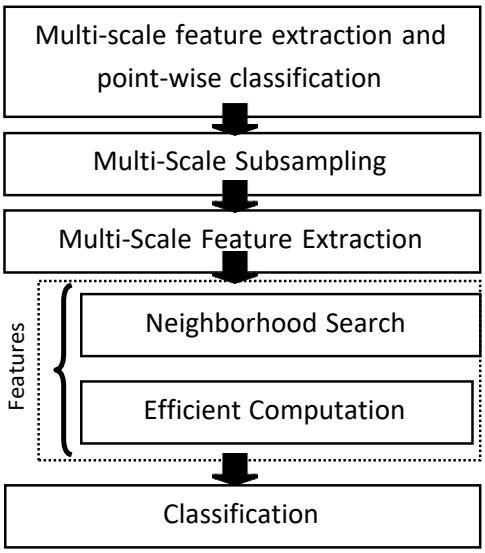

Figure 1. The proposed framework for this study.

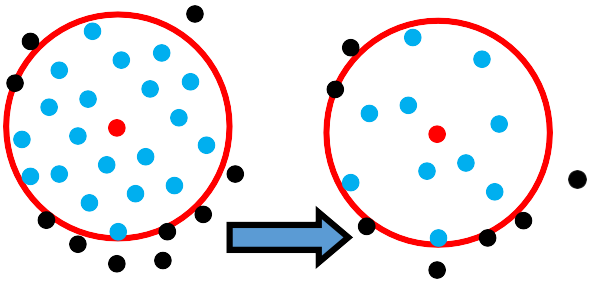

Figure 2. The sampling reduces the computational cost of feature calculation.

A 2D version also exists and projects the 3D point cloud onto a 2D surface. The irregular sampling methods, as the name implies, produce irregular point clouds. Random sampling is the first method in this group that reduces the number of points to the desired number by random selection. It is the fastest method, but it is highly possible to lose useful information. Another similar method is Poisson sampling, which takes advantage of the Euclidean distance between each point in the point cloud. It also keeps spatial objects' details, as well as is a computationally faster method. Therefore, we select Poisson sampling in this framework.

However, 3D point clouds include a variety of objects of varying sizes so that information loss is possible in a single neighborhood (Figure 3). Thus, it is common to use multi-scale neighborhoods to capture more information from various sizes of objects. In a dense UAV-LiDAR dataset, using multiple radius search on the raw data is going to cause a considerable computational problem. Instead, working with sub-sampled versions of the point cloud led to an efficient way to compute features. At each scale, while the radius is increasing, the number of points in the neighborhood remains close to each other at each level; for that reason, starting from small to large objects, a useful feature set is obtained. As shown in Figure 4, the characteristics of objects of different sizes are captured efficiently.

After the neighbors of the points in the point cloud are calculated, the features are computed using that information. Since our goal is to easily compute the features, we prefer lightweight features, which are calculated in one step and do not require additional computational steps. Therefore, we use eigenvalue-based features in the first step. They are based on the covariance matrix of the neighborhood. Eigenvalues and eigenvectors are derived from the covariance matrix by eigendecomposition. Then, the eigenvalue and eigenvector-based features, namely, geometric features, are calculated. Another lightweight feature set is based on the height values of the neighbors. The mean and variance of neighbors are added to the height feature set. Additionally, because they were provided, we added color and intensity values to our feature set with their mean and variance values. In total, we have 31 features per scale, which consists of 18 geometric, 5 height, 6 colors, and 2 intensity features. As a sum, we add the xyz coordinate values, RGB, and intensity values.

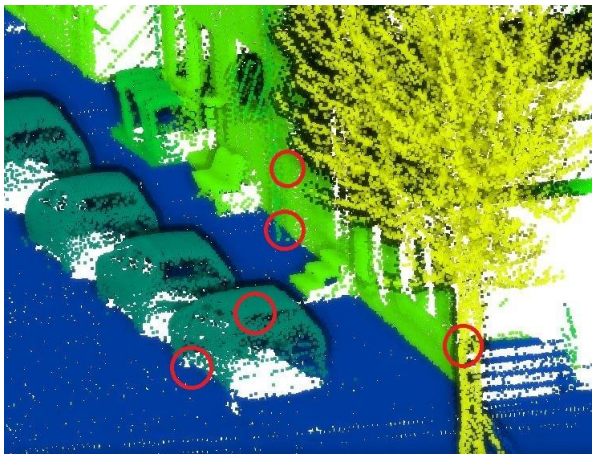

Figure 3. Various size of objects in the 3D point cloud datarequires multi-scale neighborhood.

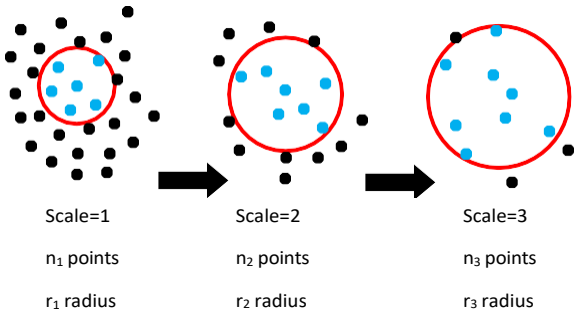

Figure 4. Multiple scales reduce the computational cost. It should be note that $n_1 > n_2 > n_3$ and $r_1 < r_2 < r_3$.

In the last step, we apply our ML classifier to the feature set. RF (Breiman, 2001) classifier is one of the most popular classifiers in 3D point cloud classification. RF is a bagging-based ensemble learning method, which relies on a collection of weak learners of decision trees. On the other hand, boosting based ensemble learning classifiers have been gaining popularity, particularly in the tabular data processing. Although they have been proposed in other studies, their potential has not been investigated in the point-wise classification of 3D point clouds before.

Gradient boosting machines have various versions in the literature. Gradient boosting is based on iteratively updating the

previous tree based on the results. Therefore, at each iteration, a weak learner decision tree turns out to be a strong learner. Among other implementations, XgBoost (Chen and Guestrin, 2016) and CatBoost (Prokhorenkova et al., 2018), we select LightGBM (Ke et al., 2017), because of the property of Gradient-based One-Side Sampling (GOSS), which saves the memory in processing and leads to faster training.

In the next section, we define the application of this framework to a dense UAV-LiDAR point cloud.

4. EXPERIMENT AND RESULTS

In this section, we describe the dataset in the next subsection, then we give details of our implementation, and then in the subsection we give the results compared with other methods in the literature.

4.1 Dataset

We would like to test our feature set and LightGBM classifier performance compared with other methods; we select a publicly available dataset: Hessigheim 3D (H3D) dataset[1] (Kölle et al., 2021). It was collected from a RIEGL Ricopter platform mounted on a Riegl VUX-1LR LiDAR sensor in Hessigheim, Germany. Additionally, two Sony Alpha 6000 cameras captured image data at the same time. The dataset consists of four phases and we are interested in the phase of March 2018. Data are split into training, validation, and testing by default. Training and validation include intensity and RGB values, while the testing tile does not have class labels. Verification of the algorithm is performed through the benchmark website.

The dataset consists of elevation classes, namely, low vegetation, impervious surface, vehicle, urban furniture, roof, facade, shrub, tree, soil/gravel, vertical surface, and chimney. The distribution of the training and validation data class is given in Figure 5. The dataset includes 60M,14M, and 50M points for training, validation, and testing splits.

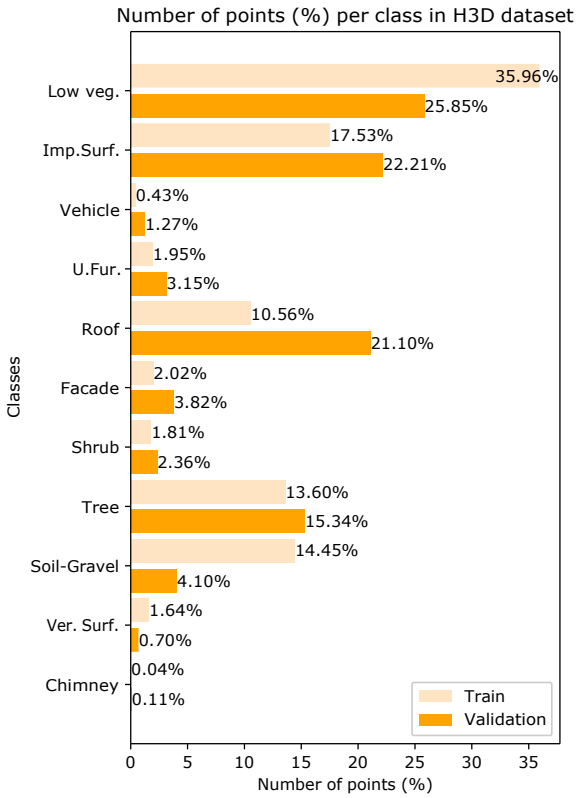

Figure 5. Class distribution in H3D dataset. Testing labels are not shared publicly.

4.2 Implementation

We implemented the proposed methodology using a few tools: *i*) multi-scale sampling with Poission sampling is performed using Point Distribution Library (PDAL) (PDAL contributors, 2022). *ii*) Feature calculation includes two phases: in the first phase we find the neighbors on the GPU using the Point Cloud Library (Rusu and Cousins, 2011), then, at the second phase, we calculate the features on the CPU in fully parallel. The implementation script is written in CPP. *iii*) The classification and evaluation scripts written in the Python programming language using the LightGBM official repository and the scikit-learn (Pedregosa et al., 2011) evaluation functions. Visualization and other data-intensive operations are performed on CloudCompare[2].

4.3 Evaluation

We are evaluating our results by the following metrics.

$$OA = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

$$P = \frac{TP}{TP + FP} \quad (2)$$

$$R = \frac{TP}{TP + FN} \quad (3)$$

,

$$F1 = \frac{2 \times TP}{2 \times TP + FP + FN} \quad (4)$$

where TP, TN, FP, FN represent true positive, true negative, false positive, false negative, respectively. We are also reporting qualitative visuals and confusion matrix which are provided by the dataset provider after submitting our results.

4.4 Results

We used the training and validation dataset for parameter estimation, then the trained model is used for testing labels. The visuals and the confusion matrix were obtained from the benchmark website.

The training data are sampled into 8 levels, then 25K points for each class, a total of 275K, are randomly selected. For those samples, the aforementioned feature sets are calculated in the radius neighborhood starting from 0.25m. After having the features, the LightGBM classifier is run with the optimum parameters. The last prediction for the benchmark also includes 15K points from the validation dataset, which means a total of 440K points.

According to the testing results depicted in Figure 6 as a confusion matrix, our framework has 87.59% in overall accuracy

[1] https://ifpwww.ifp.uni-stuttgart.de/benchmark/hessigheim/Default.aspx
[2] https://www.danielgm.net/cc/

and 75. 92% in the average F1 measure. We interpret the classes’ precision and recall scores, where precision shows how the framework works with the predicted values, and recall is an indicator of how the framework performs with actual labels, as well as the confusion between classes. The highest precision/recall scores were obtained in the roof and tree classes; however, the gravel class has the lowest recall score and the shrub and urban furniture classes have the lowest precision score. The class of low vegetation, impervious surfaces, roof, tree, and chimney has a minimum number of samples confused with other classes. However, the remaining classes have more confusion samples than the others. 11% and 7% of the class of cars were mislabeled as roof and urban furniture, respectively. Similarly, in the class of urban furniture, 11% of the samples labeled facade and 12% of the samples labeled shrub. The facade, shrub, and vertical surface classes share similar confusion patterns at some level. On the contrary, in the gravel class, we have severe confusion with particularly low vegetation and impervious surfaces. This class was mislabeled as 45% and 37% as low vegetation and impervious surfaces, respectively. For the same class, we have 71% precision and 14% recall scores, indicating that we labeled some predicted values as gravel with 71%, but only 14% of them were correct. It is clear that those classes require more distinctive color information.

The benchmark website also provides visuals of the testing labels; unfortunately, the testing labels are not shared with the public. We can see the same confusion patterns in the visuals in Figure 7. Red circles show examples of confusion with car classes, while blue and green circles indicate confusion with urban furniture. More visuals can be obtained through actual labels.

Overall, the results indicate that our feature set and ML classifier perform well on this dataset, although some classes suffer from confusion with other classes with low precision and recall scores. In the next section, we will evaluate our framework compared to previous approaches.

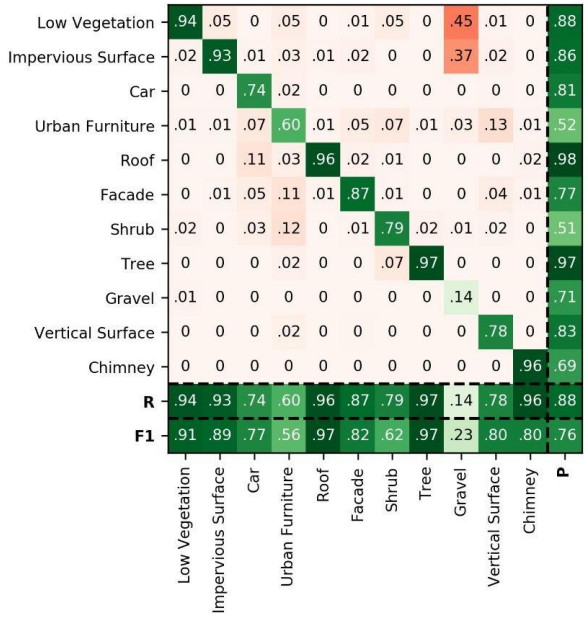

Figure 6. Confusion matrix for the results of the predicted points in the test area.

5. DISCUSSION AND CONCLUSION

We compare the results of our framework with previous studies compiled from the results section of the benchmark website (Table 1). We only took the published results, as we were unable to see the details of the unpublished approach. It should be noted that taking published results was our only criterion, although there are a few worse or better results than our framework on the benchmark website.

We compare our results with RF in the next subsection, then with DL models in Section 5.2. Next, we do a summary drawing at the end of Section 5.3 for future work.

5.1 Comparing with RF

According to the benchmark explanation (Kölle et al., 2021), our approach and the RF on the benchmark share a similar approach, multi-scale feature extraction then using a ML classifier. Differences were in multi-scale levels, feature sets, and ML classifier.

Taking into account the global scores produced from the confusion matrix of the two approaches, our framework improves the F1 scores $\approx$ 1% over RF and passes by a narrow margin in overall accuracy. Moreover, in class level F1 scores, our framework is superseding the RF in almost all classes; we have similar results for the class of chimney, and RF has better F1 scores in shrub and gravel classes.

The reason behind that is that our feature set includes only height information coming from the neighborhood search, but RF takes the height features from the digital elevation model, which makes the feature set more distinctive. Since we would like to see the effect of the classifier, we used only basic height models with more features, it is clear that the result would be better compared to RF. As a drawback, we used eight scales, whereas RF has four scales.

We can conclude that gradient boosting machines have reached better results than RF even with basic but more multi-scale features. However, a comprehensive study of ablation will allow us to understand which part of the framework contributes more to the results.

5.2 Comparing with DL Models

In this subsection, we compare our results with DL models from the baseline paper (Kölle et al., 2021), namely Sparse Convolution Neural Network, which is based on voxelization and 3D CNN approach, PointNet++, which is based on processing raw point clouds hierarchically using PointNet at each level, and KpConv, which is based on kernel-based convolution on the point clouds. In addition to those, we use (Gao et al., 2022) for comparison, which uses a type of kernel-based convolution but with self-attention modules.

In overall accuracy, the result of our framework is on the same page as the others. On the other hand, in average F1 scores, SCN has the highest score, and our framework is in second place. However, it should be noted that KpConv reached 0% in chimney, thus considering this our framework actually cannot supersede its F1 score. It is clear that our framework superseeds PN++ and (Gao et al., 2022) in the overall scores. At the class level, in contrast, our framework has better results in

Figure 7. A general visualization of predicted points (left) and a closer look at the results (right) from the test area

Class	SCN	RF	KpConv	(Gao et al., 2022)	PN++	Ours
Low Veg.	**92.31**	90.36	88.57	87.62	78.11	90.88
Imp. Surf.	88.14	88.55	88.93	85.62	72.07	**89.40**
Vehicle	63.51	66.89	**82.10**	52.40	31.78	77.28
Urban Fur.	57.17	51.55	**63.89**	36.71	13.65	55.76
Roof	96.86	96.06	**97.13**	95.48	73.98	97.05
Facade	83.19	78.47	**85.13**	69.30	47.79	81.88
Shrub	68.59	67.25	**75.24**	47.39	28.34	62.06
Tree	96.98	95.91	**97.38**	94.28	71.80	97.10
Soil/Gravel	44.81	**47.91**	42.68	25.08	9.65	23.17
Vert. Surf.	78.20	59.73	**80.87**	65.94	21.67	80.27
Chimney	73.61	**80.65**	0.00	38.59	4.39	80.28
Mean	**76.67**	74.85	72.90	63.49	41.20	75.92
OA	**88.42**	87.43	87.69	84.20	68.50	87.59

Table 1. F1 scores for each class and mean F1 and Overall Accuracy. The bold font indicates the first place. Sparse ConvolutionalNetwork (SCN), RF (RF), and PointNet++(PN++) are from the baseline paper

some classes, but cannot compete with KpConv. In almost all classes, KpConv performed better than the rest.

It seems clear that our framework can compete with PN++ but not with the KpConv architecture. Additionally, on the website there are the best results that have not been published yet. We can indicate that our model needs more attention to compete with DL models.

5.3 Conclusion

In this study, we used a traditional ML framework for point-wise classification of 3D point clouds for a highly dense UAV-LiDAR dataset. Because the computational cost is increasing in highly dense point clouds, we took advantage of multi-scale feature extraction for subsampling of the point cloud. Moreover, we implement the neighborhood functions on GPU and parallel feature calculation on the CPU. After having features, we used a gradient boosting machine classifier, LightGBM, for the classification of point clouds. Our framework uses efficiency for high-dense point clouds.

We tested our framework on a publicly open benchmark dataset, H3D. Based on the results, our framework can achieve good results as well as other traditional ML approaches; however, it falls behind DL-based models. It is clear that with more features, the results will improve accuracy.

As a future study, we have shown that using a traditional ML framework can compete with DL models. Adding more feature sets is going to help to increase the results, for the large-scale high-density point sets, the efficiency needs to be taken into account. In addition, integration with traditional ML approaches and the DL model is going to change the results positively, as well as distinguish the classes.

ACKNOWLEDGEMENT

We would like to thank the University of Stuttgart for providing a dataset and hosting the H3D benchmark.

REFERENCES

Bello, S. A., Yu, S., Wang, C., Adam, J. M., Li, J., 2020. Deep learning on 3D point clouds. *Remote Sensing*, 12(11), 1729.

Breiman, L., 2001. Random Forests. *Machine Learning*, 45(1), 5-32.

Chen, T., Guestrin, C., 2016. Xgboost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, Association for Computing Machinery, New York, NY, USA, 785–794.

Gao, F., Yan, Y., Lin, H., Shi, R., 2022. PIIE-DSA-Net for 3D Semantic Segmentation of Urban Indoor and Outdoor Datasets. *Remote Sensing*, 14(15).

Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., Bennamoun, M., 2020. Deep learning for 3d point clouds: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(12), 4338–4364.

Hackel, T., Wegner, J. D., Schindler, K., 2016. Fast Semantic Segmentation of 3D Point Clouds with Strongly Varying Density. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, III-3, 177–184.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.-Y., 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 3146–3154.

Kölle, M., Laupheimer, D., Schmohl, S., Haala, N., Rottensteiner, F., Wegner, J. D., Ledoux, H., 2021. The Hessigheim 3D (H3D) benchmark on semantic segmentation of high-resolution 3D point clouds and textured meshes from UAV LiDAR and Multi-View-Stereo. *ISPRS Open Journal of Photogrammetry and Remote Sensing*, 1.

Niemeyer, J., Rottensteiner, F., Soergel, U., 2014. Contextual classification of lidar data and building object detection in urban areas. *ISPRS Journal of Photogrammetry and Remote Sensing*, 87, 152–165. https://linkinghub.elsevier.com/retrieve/pii/S0924271613002359.

PDAL contributors, 2022. PDAL: The Point Data Abstraction Library.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., Gulin, A., 2018. Catboost: Unbiased boosting with categorical features. *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS’18, Curran Associates Inc., Red Hook, NY, USA, 6639–6649.

Qi, C. R., Su, H., Mo, K., Guibas, L. J., 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Qi, C. R., Yi, L., Su, H., Guibas, L. J., 2017b. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30.

Rusu, R. B., Cousins, S., 2011. 3D is here: Point Cloud Library (PCL). *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China.

Thomas, H., Deschaud, J.-E., Marcotegui, B., Goulette, F., Gall, Y. L., 2018. Semantic Classification of 3D Point Clouds with Multiscale Spherical Neighborhoods. *arXiv:1808.00495 [cs]*. arXiv: 1808.00495.

Thomas, H., Qi, C. R., Deschaud, J.-E., Marcotegui, B., Goulette, F., Guibas, L. J., 2019. Kpconv: Flexible and deformable convolution for point clouds. *Proceedings of the IEEE/CVF international conference on computer vision*, 6411–6420.

Weinmann, M., Jutzi, B., Hinz, S., Mallet, C., 2015. Semantic point cloud interpretation based on optimal neighborhoods, relevant features and efficient classifiers. *ISPRS Journal of Photogrammetry and Remote Sensing*, 105, 286–304. https://linkinghub.elsevier.com/retrieve/pii/S0924271615000349.

Xie, Y., Tian, J., Zhu, X. X., 2020. Linking Points With Labels in 3D: A Review of Point Cloud Semantic Segmentation. *IEEE Geoscience and Remote Sensing Magazine*, 8(4), 38–59. https://ieeexplore.ieee.org/document/9028090/.

Zhang, J., Zhao, X., Chen, Z., Lu, Z., 2019. A review of deep learning-based semantic segmentation for point cloud. *IEEE Access*, 7, 179118–179133.