



HAL
open science

Apprentissage par transfert pour discriminer le bois/feuille des Unmanned aerial vehicle Laser Scanning (ULS)

Yuchen BAI, Jean-Baptiste Durand, Grégoire Vincent, Florence Forbes

► To cite this version:

Yuchen BAI, Jean-Baptiste Durand, Grégoire Vincent, Florence Forbes. Apprentissage par transfert pour discriminer le bois/feuille des Unmanned aerial vehicle Laser Scanning (ULS). TRIDIFOR 2022 - Méthodes innovantes d'analyse de données 3D en forêt, GIP ECOFOR, Sep 2022, en ligne, France. hal-04837364

HAL Id: hal-04837364

<https://inria.hal.science/hal-04837364v1>

Submitted on 13 Dec 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License



Apprentissage par transfert pour discriminer le bois/feuille des Unmanned aerial vehicle Laser Scanning (ULS)

PhD student:
Yuchen BAI

Supervisor:
Jean-Baptiste DURAND
Grégoire VINCENT
Florence FORBES





Outline

- ❑ Introduction
- ❑ Three identified questions
- ❑ Prototype model
- ❑ Perspectives

Introduction

- Airborne Laser Scanning (ALS):

 - RIEGL LMSQ780

 - point density : $\sim 70/\text{m}^2$, pulse density : $\sim 40/\text{m}^2$

- Terrestrial Laser Scanning (TLS):

 - RIEGL VZ-400

 - point density : $> 200,000/\text{m}^2$, pulse density : $> 200,000/\text{m}^2$

- Unmanned aerial vehicle (UAV) Laser Scanning (ULS):

 - RIEGL miniVUX-1UAV

 - point density : $> 750/\text{m}^2$, pulse density : $> 500/\text{m}^2$

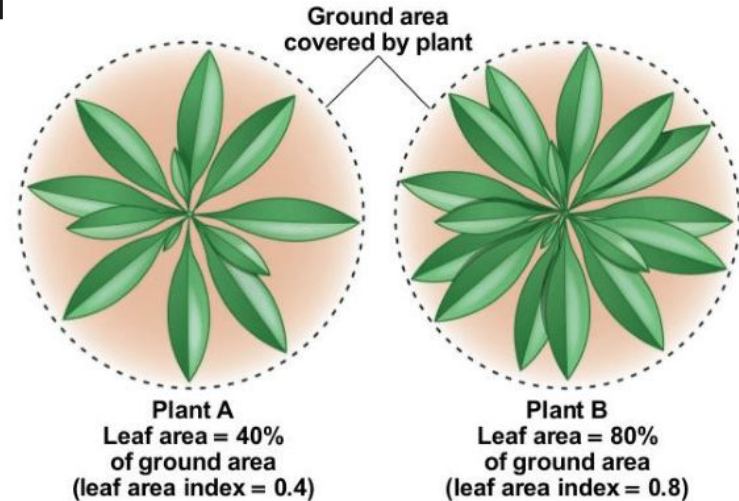
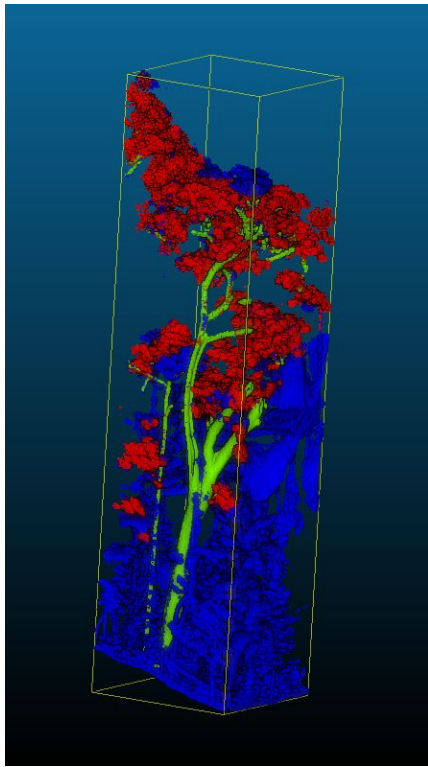
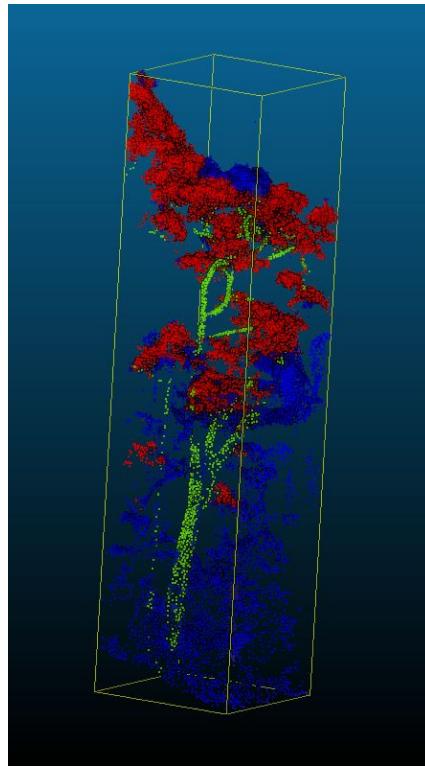


Figure 1. Leaf Area Index (LAI)

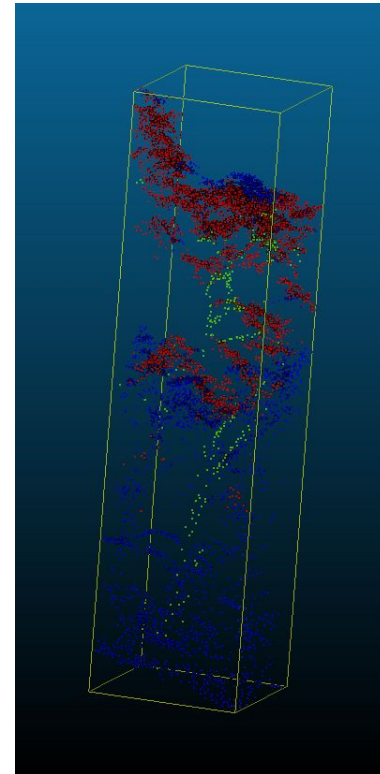
The leaf area density (**LAD**) is defined as the total one-sided leaf area of photosynthetic tissue per unit canopy volume. The Leaf area index (**LAI**) is then derived by integrating the leaf area density over the canopy height. It corresponds to the one sided leaf area per unit horizontal ground surface area.



(a) Terrestrial Laser Scanning (TLS)



(b) Unmanned Aerial Vehicle (UAV) Laser Scanning (ULS)



(c) Aerial Laser Scanning (ALS)

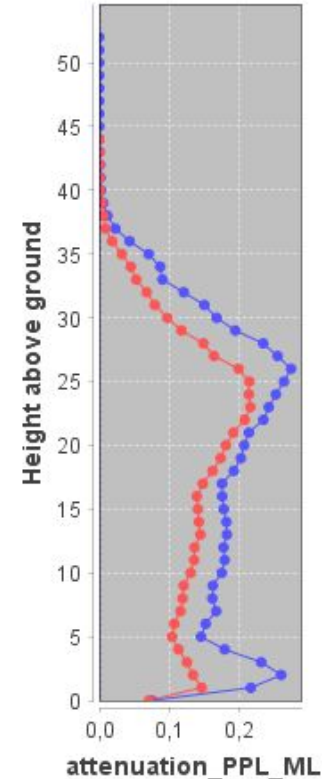


Figure 2. LiDAR (Light Detection and Ranging) in a cuboid, 20 m x 20m x 50m, *“More is different”* -- P. W. Anderson

Three identified questions

- ❑ How to do semantic segmentation (wood-leaf discrimination) on ULS data?
 - ❑ TLS -> dense, small footprint and fewer mixed points
 - ❑ no good approach for drone data (Unmanned aerial vehicle Laser Scanning (ULS))
- ❑ How to get a more accurate and unbiased point estimator of vegetation density?
 - ❑ clumping (aggregate distribution of leaves),
 - ❑ occlusion (limited penetration and incomplete exploration of voxels)
 - ❑ develop estimates which are less biased and have lower variance
- ❑ How to correct the censorship bias of undetected targets?
 - ❑ undetected hits bias the calculated signal attenuation and the estimated leaf areas.
 - ❑ model the effect of this censorship and correct this bias
 - ❑ in dense canopies, systematic overestimation of Leaf Area Index (LAI) 10~30% in magnitude
 - ❑ simulation!

attenuation_PPL_ML
E profile



—• TLS.vox

—• ULS_195956.vox

State of the Arts methods

- Methods based on geometric and spatial properties
 - Treeseg[1], LeWos[2], Quantitative Structure Models (QSMs) based[3], Superpoint based[4], Itakura et al. [10]
- Methods based on Random forest :
 - 'Deep Points Consolidation algorithm'[5]
- Methods based on Deep Learning :
 - Point-wise methods : FSCT[6], Deep-RBN[7], Morel et al.'s method [8]
 - Voxel-wise methods : Windrim et al.'s method[9], Yang et al. [11]

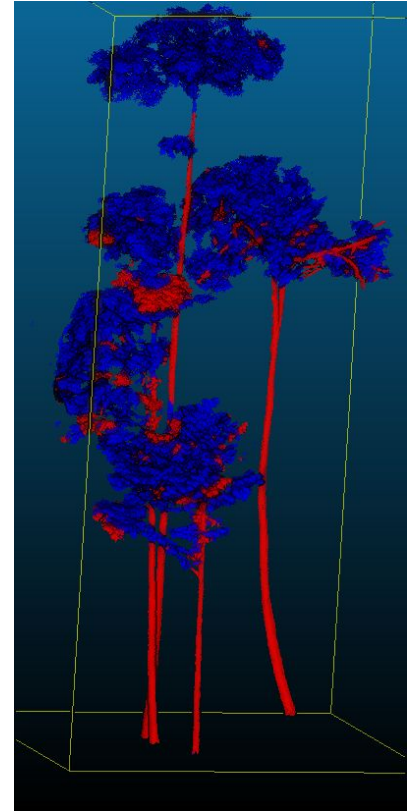


Fig 3. Lewos on TLS data

Prototype model : Point-Voxel based neural network

Semantic segmentation on ULS dataset

How to get training data?

- ❑ **label transfer from TLS to ULS data**
- ❑ training with TLS and fine-tuning with ULS
- ❑ downsample the TLS data

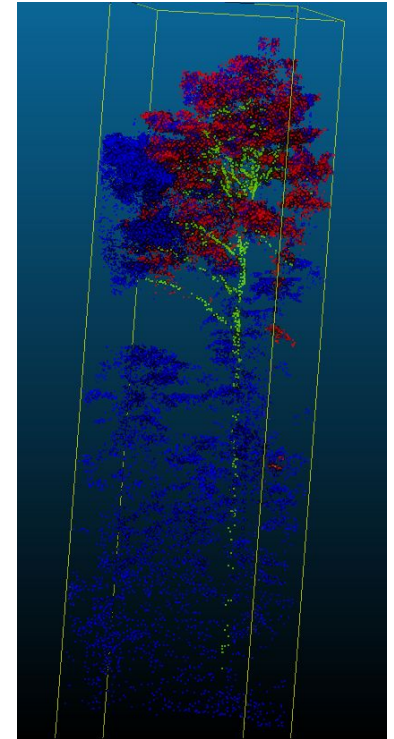


Fig 4. ULS-like data

Use Canopy Height Model (CHM) to qualify co-registration quality

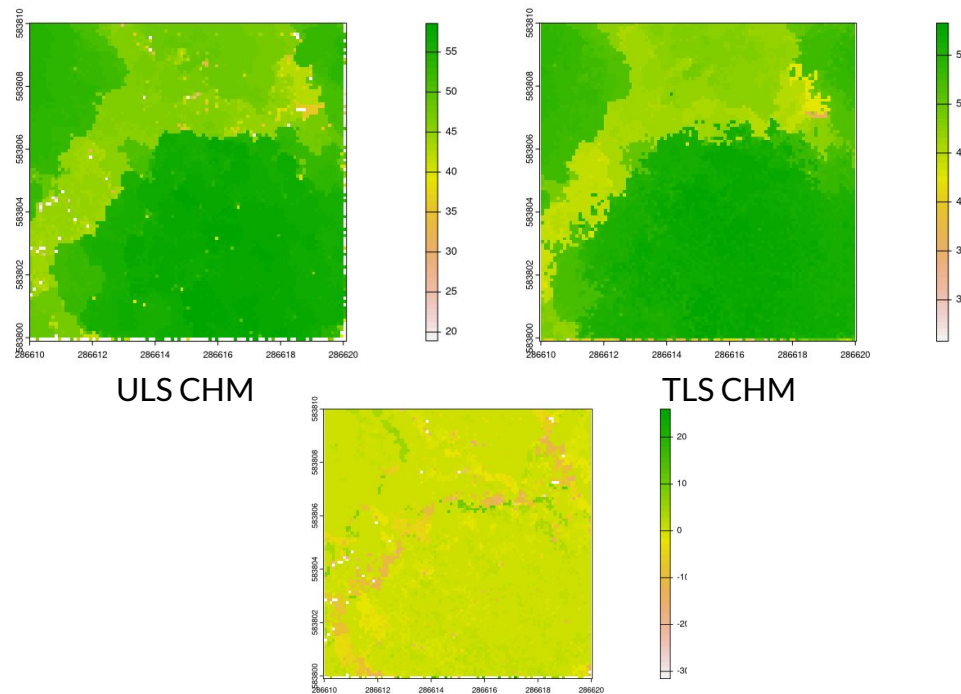
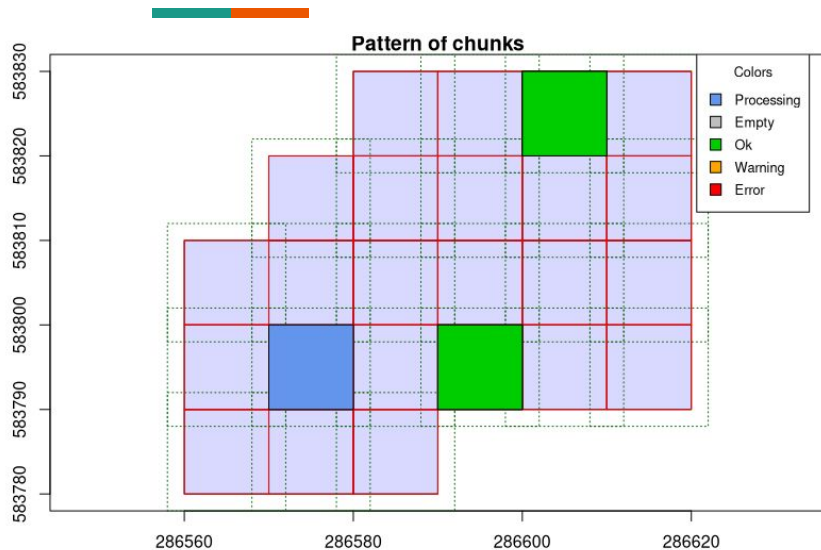
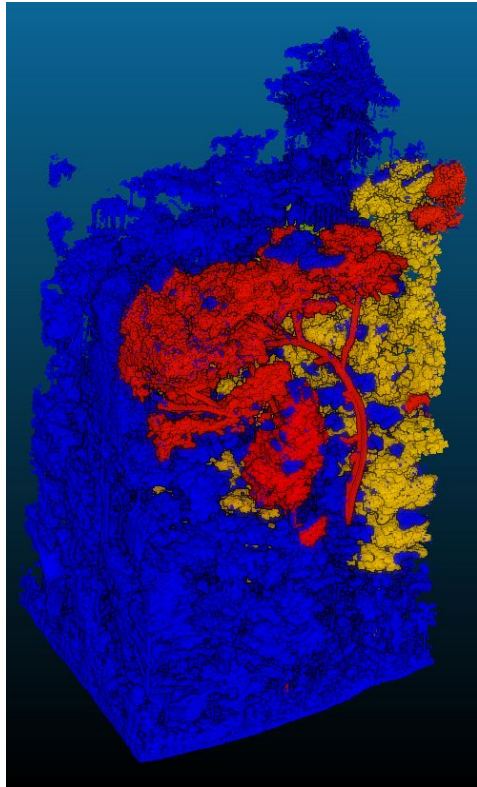
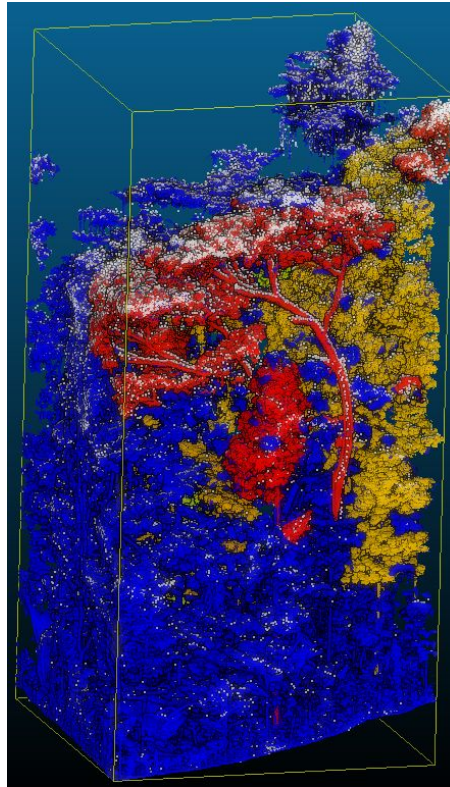


Figure 5. CHM comparison

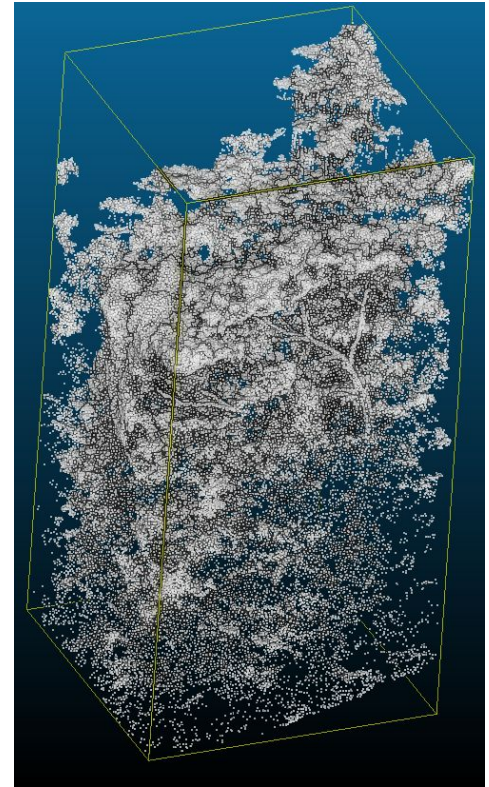
$$\text{Diff CHM} = \text{ULS CHM} - \text{TLS CHM}$$



(a) TLS

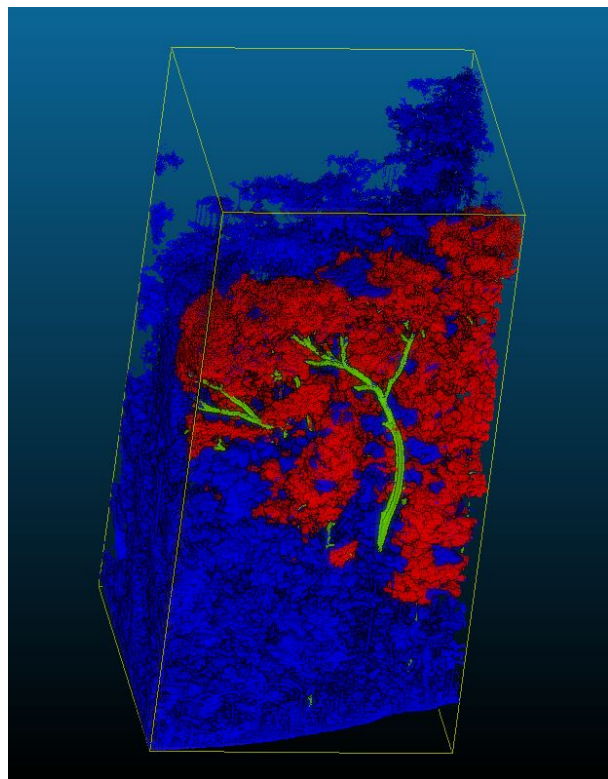


(b) overlap

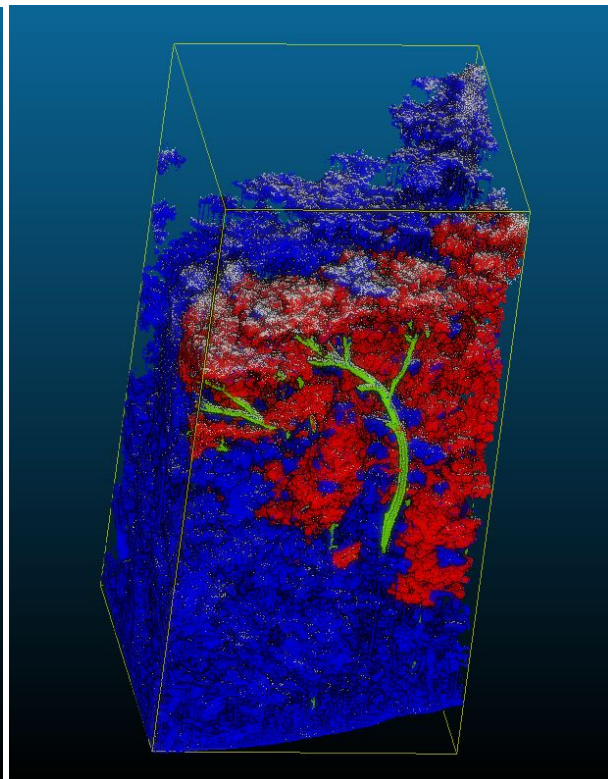


(c) ULS

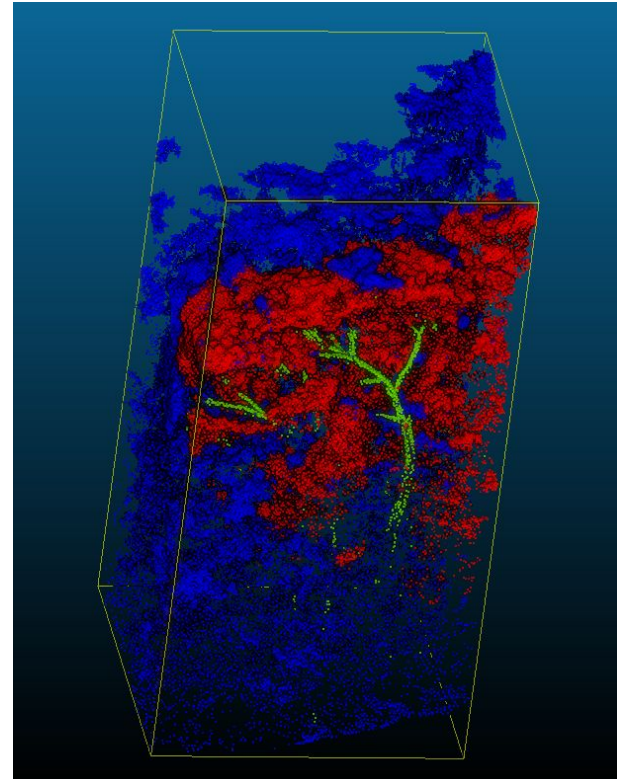
Figure 6. TLS and ULS co-registration



(a) TLS



(b) overlap



(c) ULS apres
transfer label

Figure 7. TLS and ULS co-registration

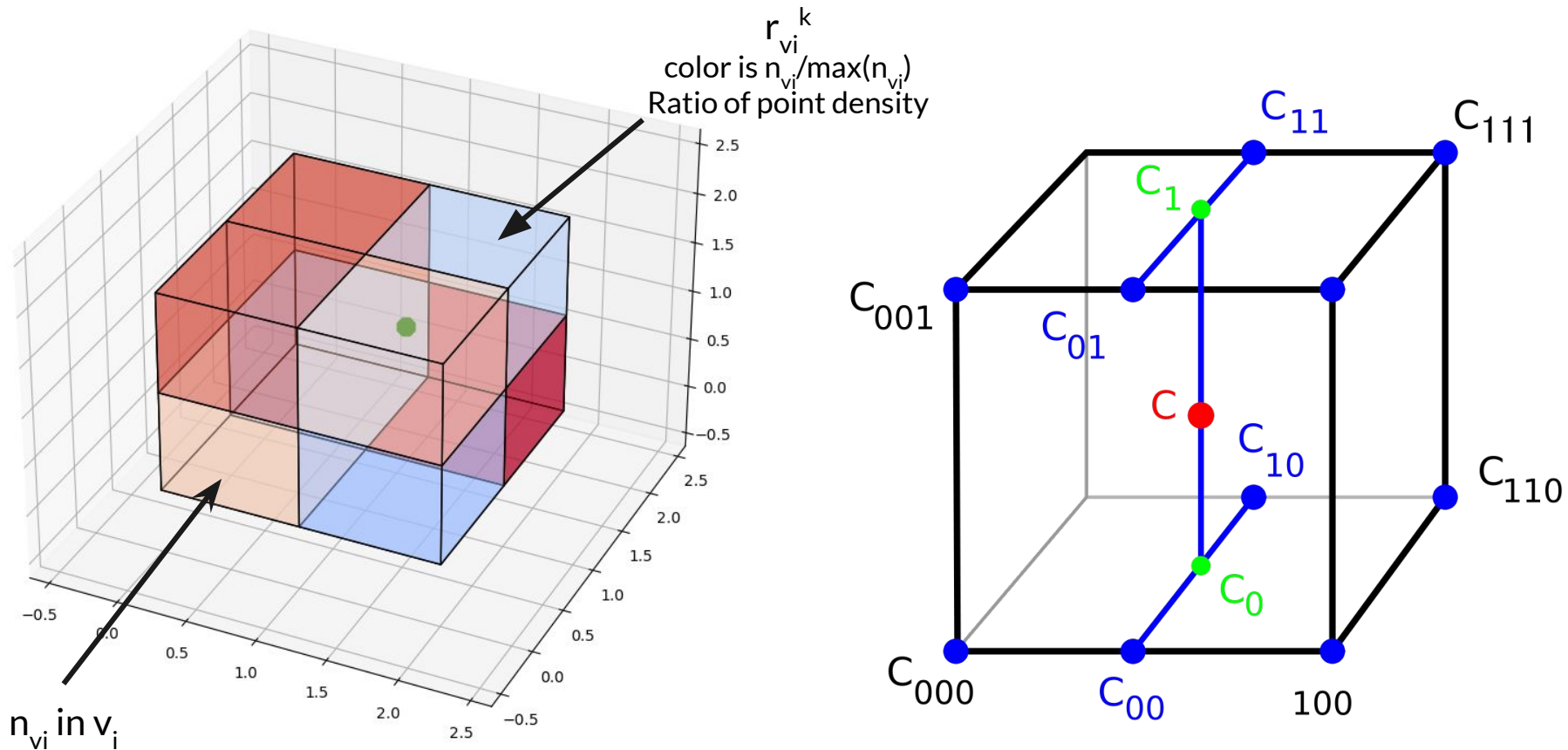
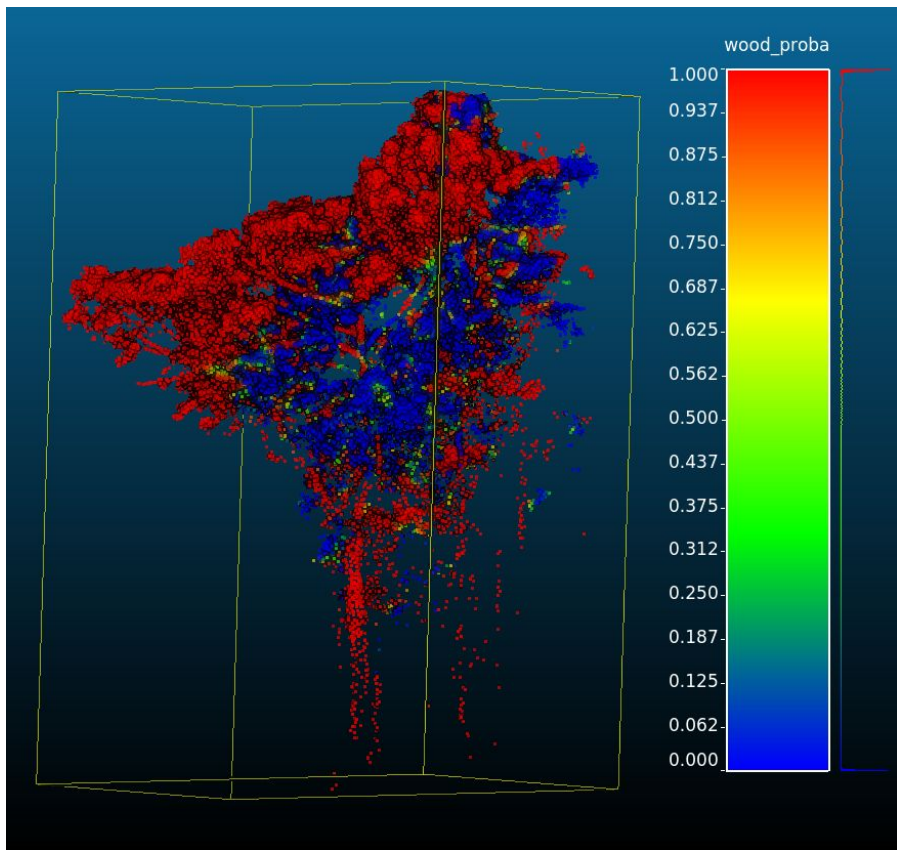
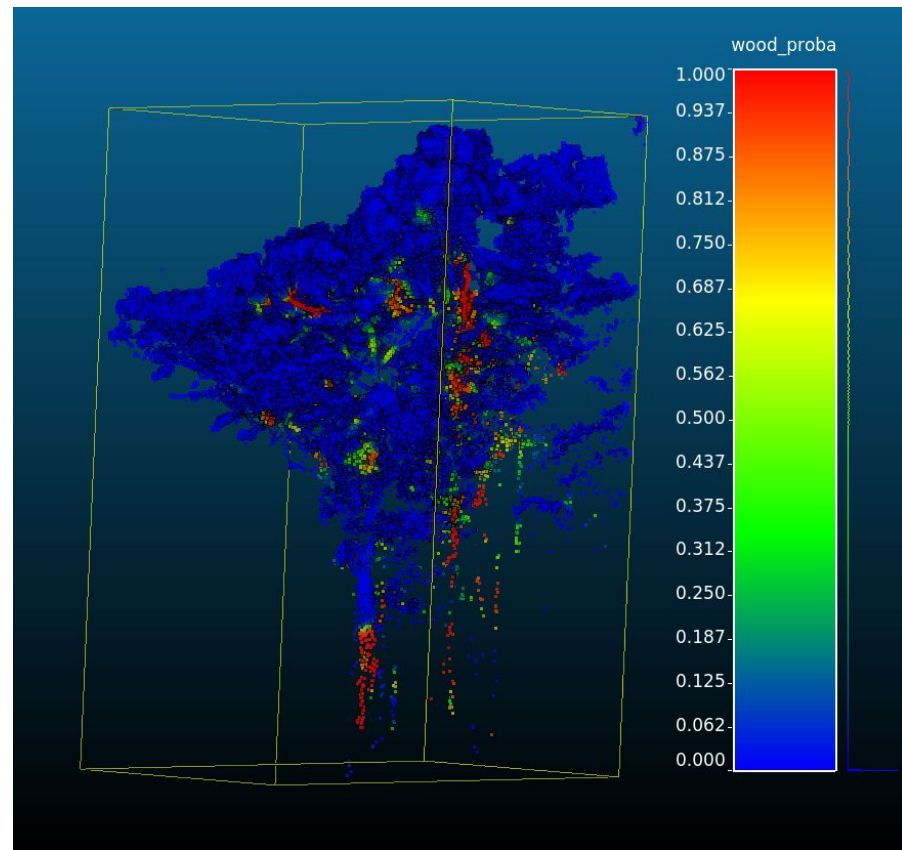


Figure 9. Trilinear interpolation

It approximates the value of a function at an intermediate point (x,y,z) within the local axial rectangular prism linearly, using function data on the lattice points. Find 8 nearest voxels in the space.



(a) overpredict wood



(b) overpredict leaf

Figure 10. Preliminary results : overprediction!

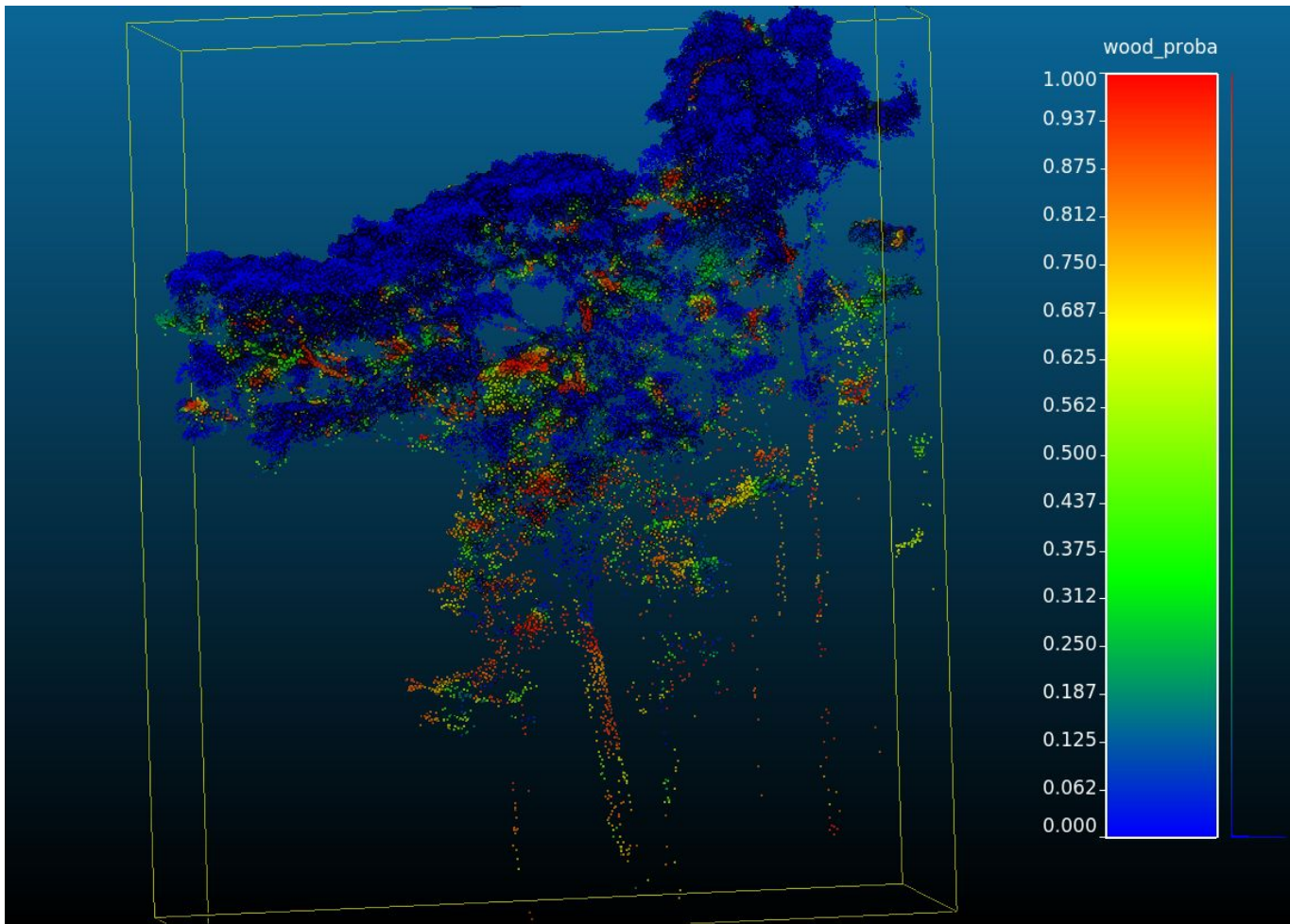


Figure 11. Failure prediction for some parts, need to improve

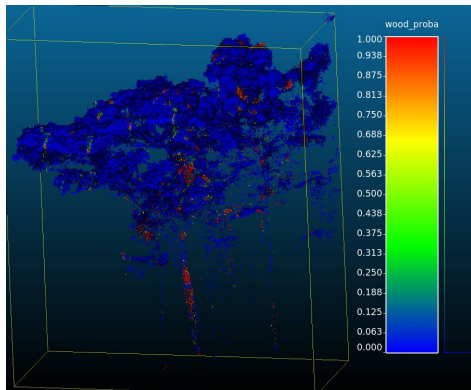


Fig 12-1. Training on tls, p@intensity, v@ratio point density, 0.2m, acc<50%

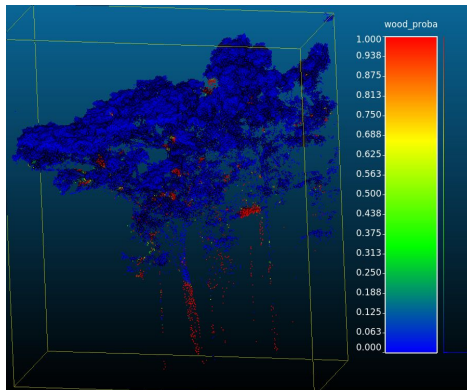


Fig 12-2. p@elevation, v@ratio point density, 0.2m, acc<50%

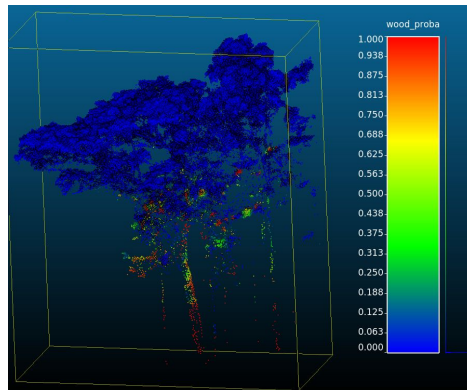


Fig 12-3. p@intensity, v@ratio point density, 0.1m, acc<50%

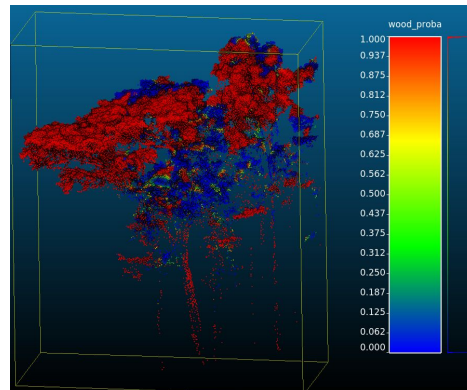


Fig 12-4. p@intensity, v@ratio point density, 0.1m, without rescaling, acc<70%

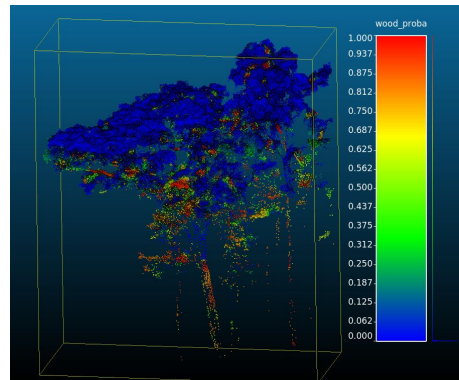


Fig 12-5. p@intensity, v@ratio point density, 0.1m, rescaling, acc=85%

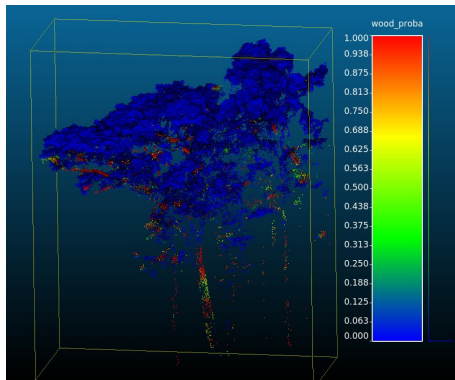


Fig 12-6. p@intensity, v@ratio point density, 0.1m, rescaling, acc=80%

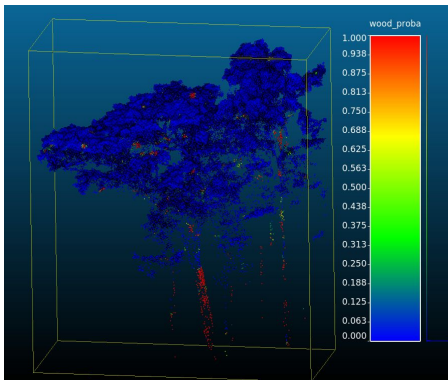


Fig 12-7. p@intensity, v@ratio return number, 0.1m, rescaling, acc<50%

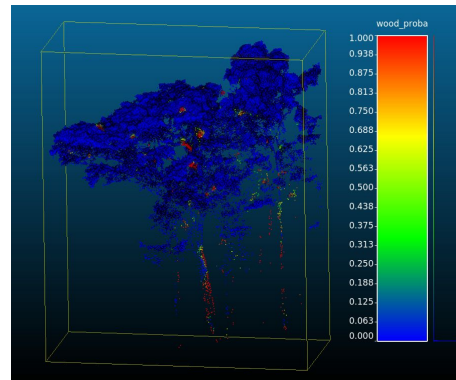


Fig 12-8. p@intensity, v@std intensity, 0.1m, rescaling, acc<50%

Fig 12: Still no desired result

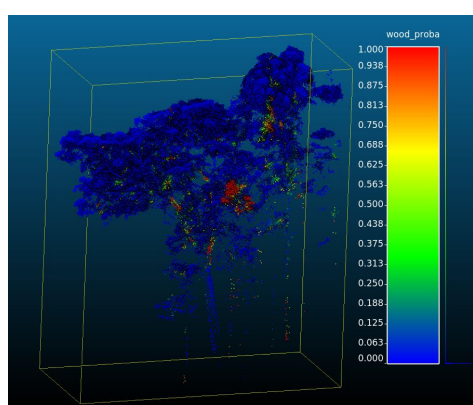


Fig 12-9. p@intensity, v@occupacy
1 or 0, 0.1m, acc<50%

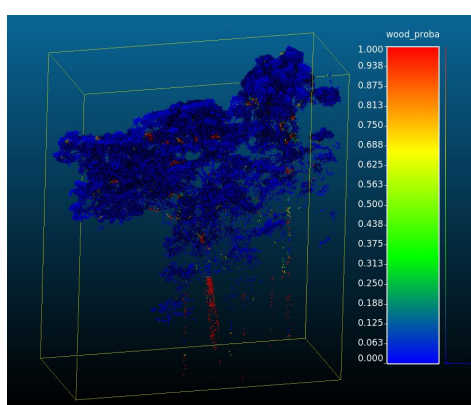


Fig 12-10. p@intensity, v@intensity
moyen, 0.1m, acc<50%

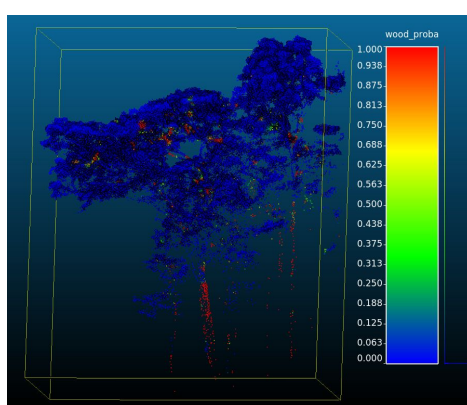


Fig 12-11. p@intensity + elevation,
v@ratio point number, 0.1m, acc<50%

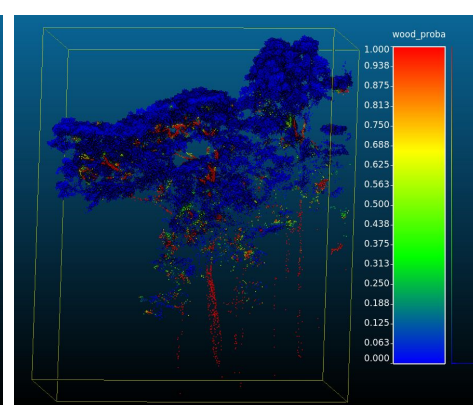


Fig 12-12. p@intensity + elevation,
v@ratio point number, 0.1m, **pointnet**,
acc=90%

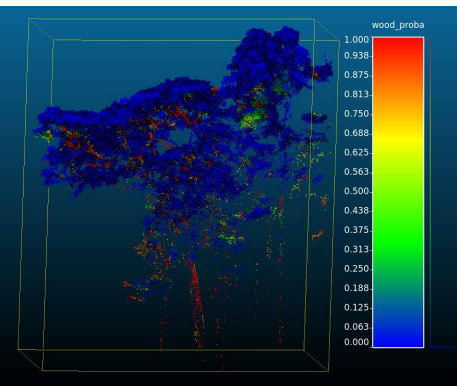


Fig 12-13. p@intensity + elevation,
v@ratio point number, 0.1m, pointnet,
acc=92.061%

Fig 12-14. p@intensity, v@ratio
point number, 0.1m, pointnet, [class
weights added](#)

Fig 12-15. p@intensity, v@ratio
point number, 0.1m, pointnet,
ensure the proportion of wood
points when sampling

Fig 12: Result on test data, improvement needed

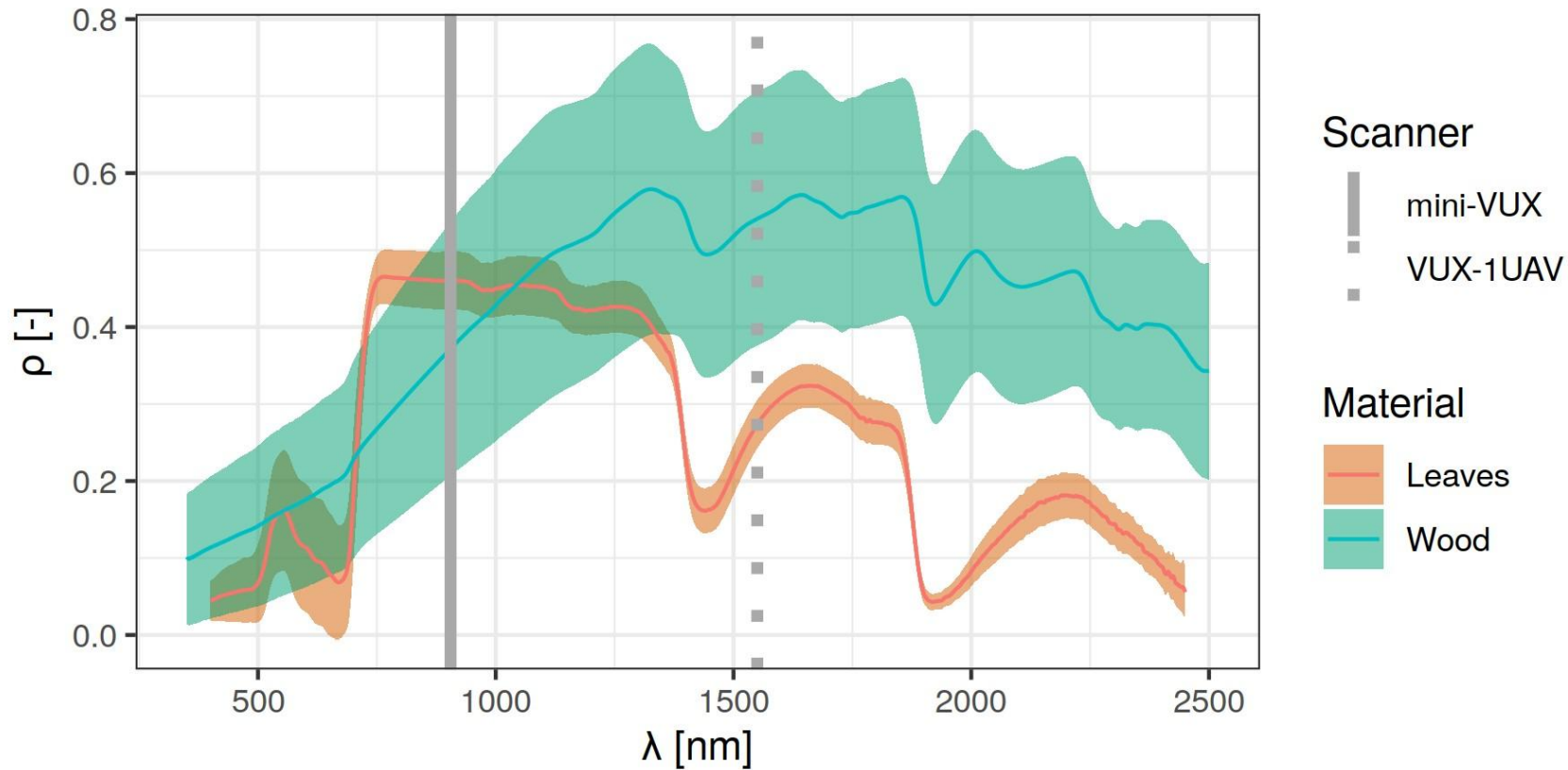
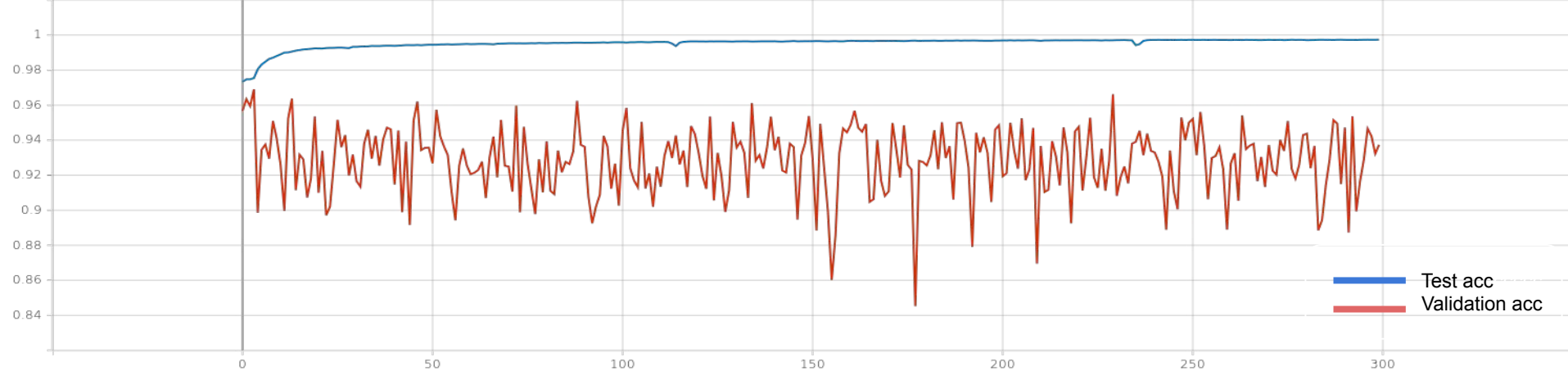
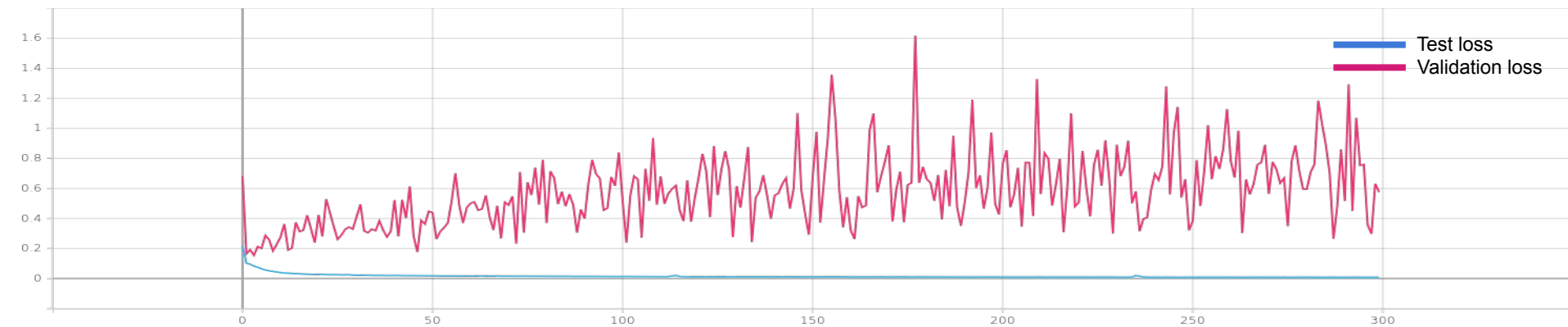


Figure 13. reflectance ratio (intensity), ULS (905 nm) vs TLS (1550 nm), *Greg Vincent*



(a) accuracy , 300 epochs



(b) loss, 300 epochs

Figure 14. Plot loss and accuracy for training 300 epochs

2.4 Matthew Coefficient Correlation (MCC)

The MCC can be calculated directly from the **confusion matrix** using the formula:

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

Fig 13. Matthew Coefficient Correlation / Phi coefficient

		Predicted condition	
		Positive (PP)	Negative (PN)
Actual condition	Total population = P + N		
	Positive (P)	True positive (TP)	False negative (FN)
	Negative (N)	False positive (FP)	True negative (TN)

validation matthew correlation coefficient - avg

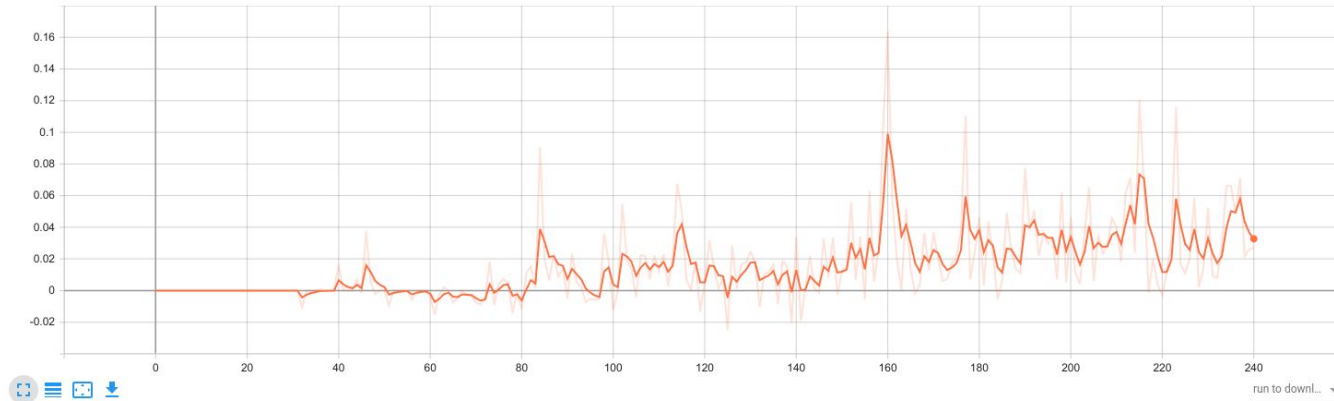


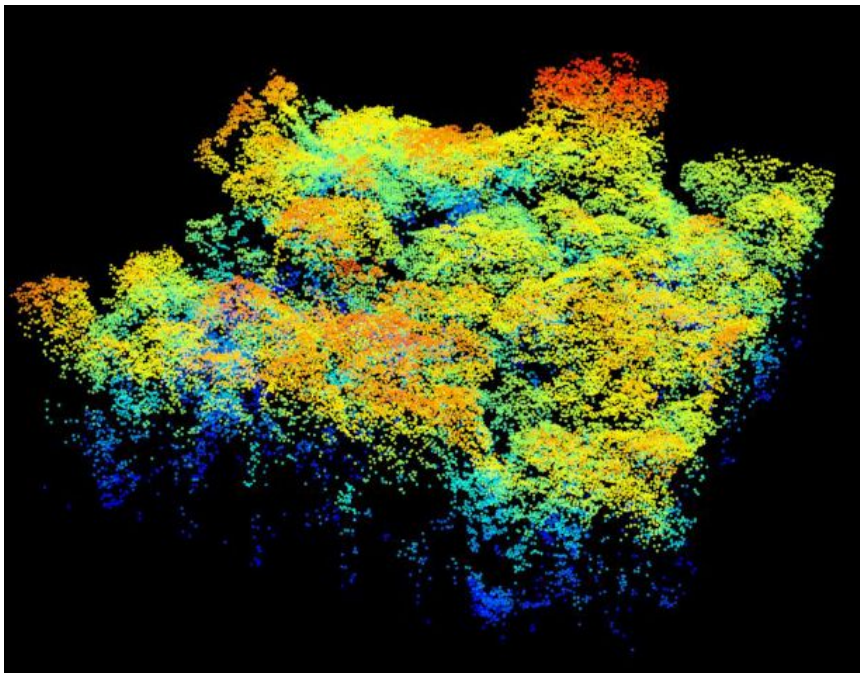
Fig 15. MCC is a good metric for class imbalance case

		Prediction	
	Total test point = 100000	Wood	Leaf
True situation	Wood	328	3182
	Leaf	4325	92165

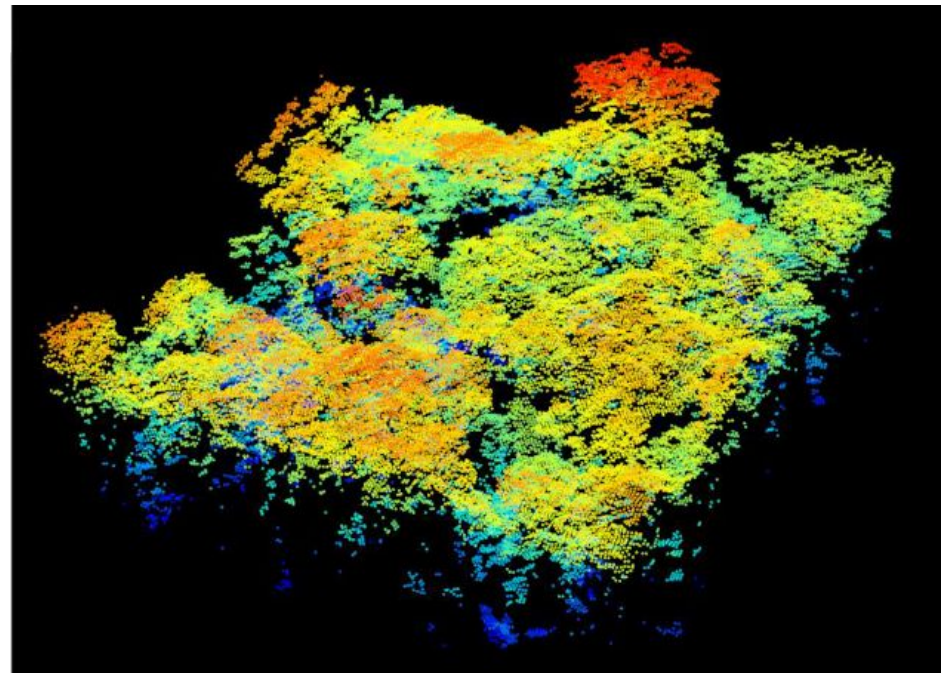
Figure 13. Confusion Matrix for the best result so far (sample size=5000 point, epoch = 546)

Perspectives

- ❑ Improving existing DL prototype model
- ❑ Limits of current estimators of vegetation density
 - ❑ estimation of clumping parameter required
 - ❑ study spatial dependencies by using hierarchical Bayesian models.
- ❑ The censorship bias of undetected interceptions
 - ❑ undetected interceptions biases the calculated signal attenuation and the estimated leaf areas
 - ❑ model the effect and correct this bias



(a) Airborne Riegl LMS-Q780



(b) DART-RC (Ray-Carlo): simulated ALS data

Figure 16. DART simulation

DART has 3 major modes, we may use DART-RC which simulates LiDAR signals with a Ray-Carlo (RC) approach that combines ray tracking and forward Monte Carlo (MC) methods.

Reference

1. A. Burt, M. Disney, and K. Calders, “Extracting individual trees from lidar point clouds using treeseg,” *Methods in Ecology and Evolution*, vol. 10, no. 3, pp. 438–445, 2019.
2. D. Wang, S. Momo Takoudjou, and E. Casella, “Lewos: A universal leaf-wood classification method to facilitate the 3d modelling of large tropical trees using terrestrial lidar,” *Methods in Ecology and Evolution*, vol. 11, no. 3, pp. 376–389, 2020.
3. P. Raunonen, E. Casella, K. Calders, S. Murphy, M. Åkerblom, and M. Kaasalainen, “Massive-scale tree modelling from tls data,” *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. II-3/W4, pp. 189–196, 2015.
4. D. Wang, “Unsupervised semantic and instance segmentation of forest point clouds,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 165, pp. 86–97, 2020.
5. S. T. Digumarti, J. Nieto, C. Cadena, R. Siegwart, and P. Beardsley, “Automatic segmentation of tree structure from point cloud data,” Oct 2018.
6. S. Krisanski, M. S. Taskhiri, S. Gonzalez Aracil, D. Herries, A. Muneri, M. B. Gurung, J. Mont-gomery, and P. Turner, “Forest structural complexity tool—an open source, fully-automated tool for measuring forest point clouds,” *Remote Sensing*, vol. 13, no. 22, 2021.
7. H. Liu, X. Shen, L. Cao, T. Yun, Z. Zhang, X. Fu, X. Chen, and F. Liu, “Deep learning in forest structural parameter estimation using airborne lidar data,” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 14, pp. 1603–1618, 2021.
8. J. Morel, A. Bac, and T. Kanai, “Segmentation of unbalanced and in-homogeneous point clouds and its application to 3d scanned trees,” Sep 2020.
9. L. Windrim and M. Bryson, “Detection, segmentation, and model fitting of individual tree stems from airborne laser scanning of forests using deep learning,” *Remote Sensing*, vol. 12, no. 9, 2020.
10. K. Itakura, S. Miyatani, and F. Hosoi, “Estimating tree structural parameters via automatic tree segmentation from lidar point cloud data,” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 15, pp. 555–564, 2022.
11. J. Yang, Z. Kang, S. Cheng, Z. Yang, and P. H. Akwensi, “An individual tree segmentation method based on watershed algorithm and three-dimensional spatial distribution analysis from airborne lidar point clouds,” 2020.

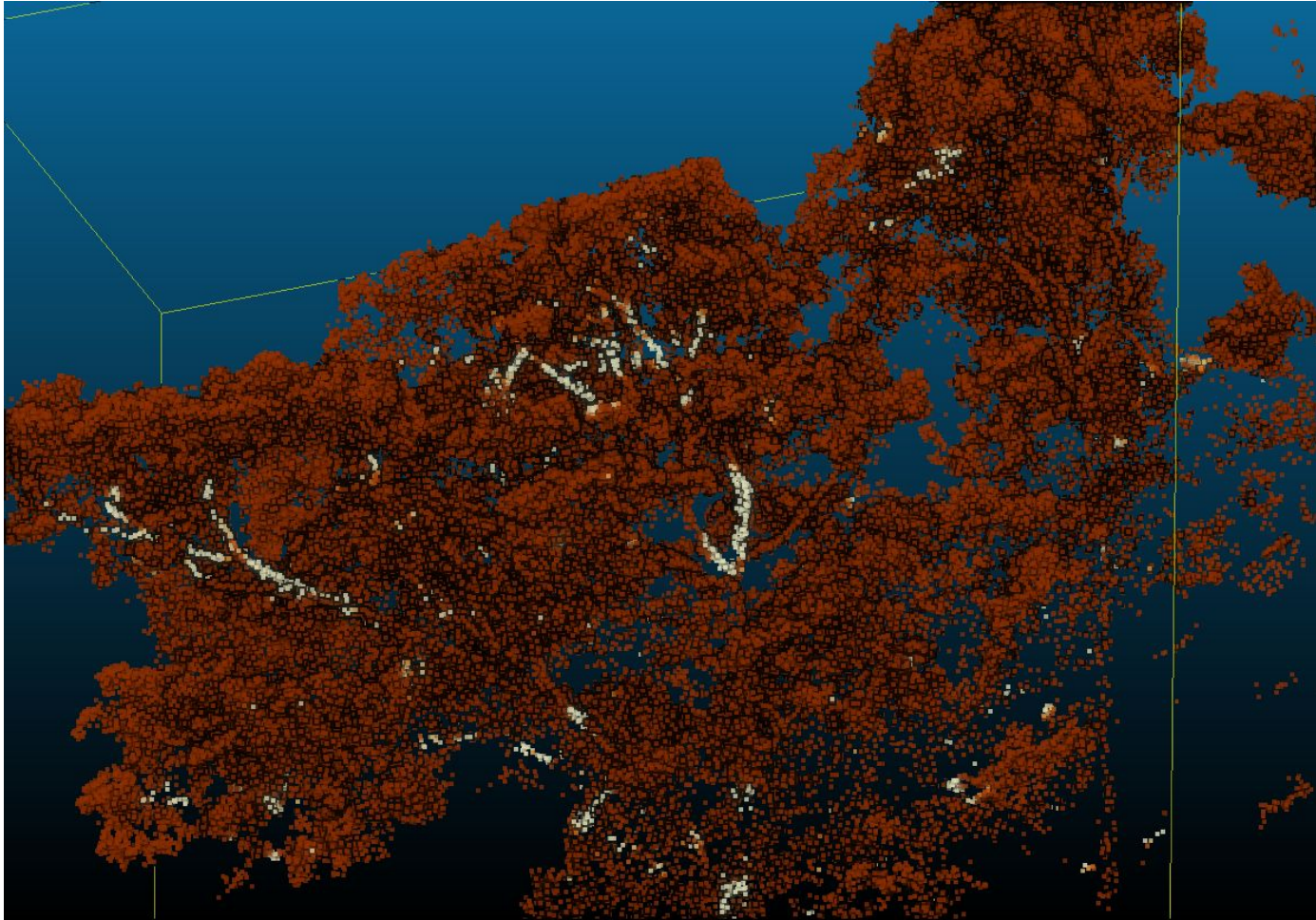
Annexe A. Compare 12 different methods

	Algo	Train Set			Test Set				
		Name	Number of tree	Number of points	Type de LiDAR	Name	Number of tree	Number of points in each class	Type de LiDAR
Geometric	treeseg	1 ha of tropical forest (moist, Terra Firma, lowland, mixed species, old growth) in Nouragues Nature Reserve, French Guiana/ 0.25 ha of Eucalyptus spp. open forest in Karawatha Forest Park. Australia	425/40	4 billion/50 million	TLS				
	automatic reconstruction for plot level	30-m diameter English oak plot and a 80-m diameter Australian eucalyptus plot		124 million/71 million	TLS				
	new classification algo	Mulligans Flat Nature Reserve (Australian Capital Territory, Australia)	28	1,379,173 points	MLS				
	robot detect tronc superpoint, geometric only	any forest Synthetic dataset	30	20 million triangle mesh	TLS				
	Random forest	simulated data							
	LeWoS	Eastern estimating of Cameroon	61	50000/second	TLS				
	Supervised	a novel individual tree segmentation method	Qishan scenic area, 4 places, 1947.16, 44,596.64, 60,601.78 and 14,780.11 square meters 50% training, 50% testing	501 (trees), 168 (trees)/334 (buildings), 426 (trees), and 166 (trees) 10240 after the data augmentation	700,000 scan point clouds per second 1511.30 pts m ⁻² , 1002.17 pts m ⁻² , 722.31 pts m ⁻² , and individual tree point clouds 20.000 to 100.000	ULS(unmanned aerial vehicle)	Qishan scenic area	522, 160, 456, and 167	
FSCT		6 20m ² plots and 1 circular 40 m Diameter plots	177 trees		TLS				
an innovative method					TLS	a semi-deciduous forest of Eastern Cameroon			TLS
australia, helicopter		two Radiata pine forests in NSW, Australia Tumut, NSW, Australia (collected in November 2016) and Carabost, NSW, Australia (collected in February, 2018)	400 stems/ha and 600 stems/ha - 60 + 70	300–700 points per m ²	ALS (helicopter)		- ~10 + ~15		ALS
3D CNN		eight different sites across the Northern New England/Acadian Forest	714 trees/ha	9 pls/m ²	ALS				

Annexe A-bis. Compare 12 different methods

	Algo	Descriptor/Method	Performance	Notes	Articles	Author	Year	conference/Journal
Geometric	treeseq	Euclidean clustering, principal component analysis, region-based segmentation, shape fitting and connectivity testing	96 % and 70 %	Not full automatically 30% required further manual segmentation	Extracting individual trees from lidar point clouds using treeseq	Andrew Burt et al.	2018	Methods in Ecology and Evolution
	automatic reconstruction for plot level	build quantitative structure models(QSMs) for every tree, this method is based on morphological rules, a cover-set approach, and geometric primitives.	N/A	quantitative structure model of every tree Calculate the biomass, not classification	MASSIVE-SCALE TREE MODELLING FROM TLS DATA	P. Raumonen et al.	2015	ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences
	new classification algo	structural characteristics of the vegetation objects	98 % for trees and 80 % for vegetation	data set is too small, recognize tree and vegetation not classification, and failed to correctly classify three of the trees and three of the elevated vegetation objects	Deriving comprehensive forest structure information from mobile laser scanning observations using automated point cloud classification	Suzanne M. Marselis et al.	2016	Environmental Modelling & Software
	robot detect tronc	3-D geometry analysis of point clouds and geometric primitives fitting	N/A	Tech report, let robot to be able to traverse the perceived terrain. Not too related to us.	Automatic Three-Dimensional Point Cloud Processing for Forest Inventory	Jean-François Lalonde et	2006	N/A
	superpoint, geometric only	Super point graph + unsupervised	87.7%	based on super point, the conception that I have seen in Paris Simulated data not real	Unsupervised semantic and instance segmentation of forest point clouds	Di WANG	2020	ISPRS Journal of Photogrammetry and Remote Sensing
	Random forest	Random forest classifier Deep Points Consolidation, meso-skeleton	~ 91%	the algorithm works at the level of the input point cloud itself, preserving quantitative accuracy in the resulting model. Use labelled data manual thresholds or heuristics based on tree allometry do not scale well across different species of trees and	Automatic Segmentation of Tree Structure From Point Cloud Data	Sundara Tejaswi Digumarti	2018	IEEE
	LeWoS	graph based point cloud segmentation technique Manual fine-tuning some thresholds, class probability Graph-structured class regularization operates on class probability	~ 91%	STOA?	LeWoS: A Universal Leaf-wood Classification Method to Facilitate the 3D Modelling of Large Tropical Trees Using Terrestrial LiDAR	Di WANG	2019	Methods in Ecology and Evolution
	Supervised	a novel individual tree segmentation method	Deep learning, supervised, PointNet++	~ 90%	similar to FSCT, See more details about data on the page 7 of 22	Individual Tree Crown Segmentation Directly from UAV-Borne LiDAR Data Using the PointNet of Deep Learning	Xinxin Chen et al.	2021
FSCT		Deep learning, supervised, PointNet++	~ 95.4%	FSCT	Sensor Agnostic Semantic Segmentation of Structurally Diverse and Complex Forest Point Clouds Using Deep Learning	Sean Krisanski et al.	2021	MDPI
an innovative method		Deep learning, supervised, PointNet++ Poisson disk sampling, local PCA,	~ 90%	subsample the raw point clouds by using Poisson disk sampling our method classifies 90 to 95% of the points as good as a human	Segmentation of unbalanced and in-homogeneous point clouds and its application to 3D scanned trees	Jules Morel	2020	Springer Nature
australia, helicopter		low-flying aircraft at high-resolutions (hundreds of points per m2) A stem segmentation approach extended by incorporating voxel representations that include LiDAR return intensity into the learning representation	~ 72% - 92%	Individual tree need manually label	Detection, Segmentation, and Model Fitting of Individual Tree Stems from Airborne Laser Scanning of Forests Using Deep Learning	Lloyd Windrim	2020	MDPI
3D CNN		All LiDAR data were acquired between 2012 and 2016 in leaf-on conditions between June and August			The Use of Three-Dimensional Convolutional Neural Networks to Interpret LiDAR for Forest Inventory	Elias Ayrey	2018	MDPI

Annexe B. Small branches can also be detected



Annexe C. merged TLS data

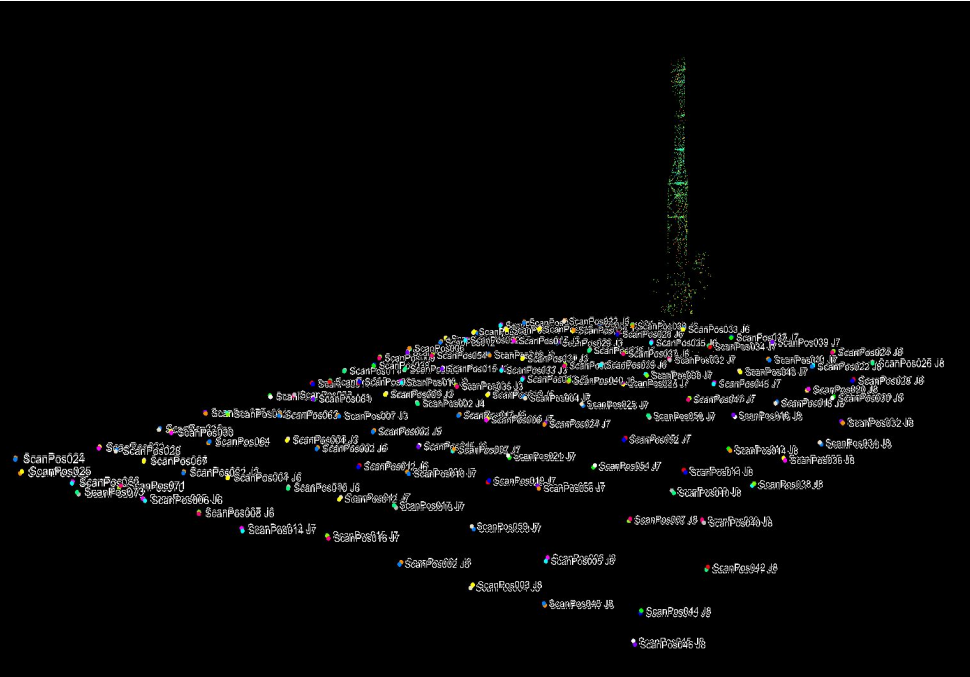


Figure 2. Different scanning position under tower

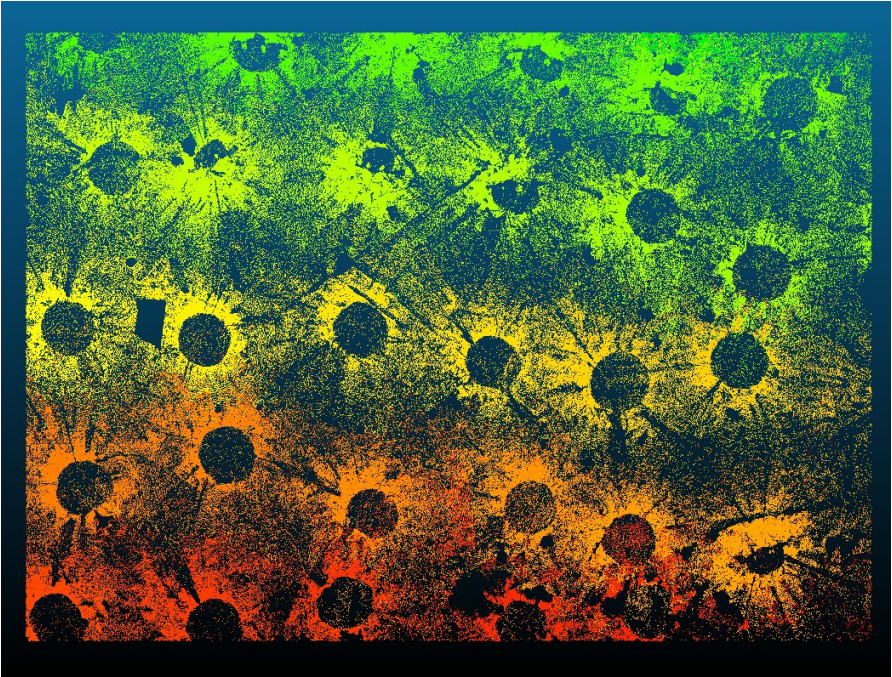
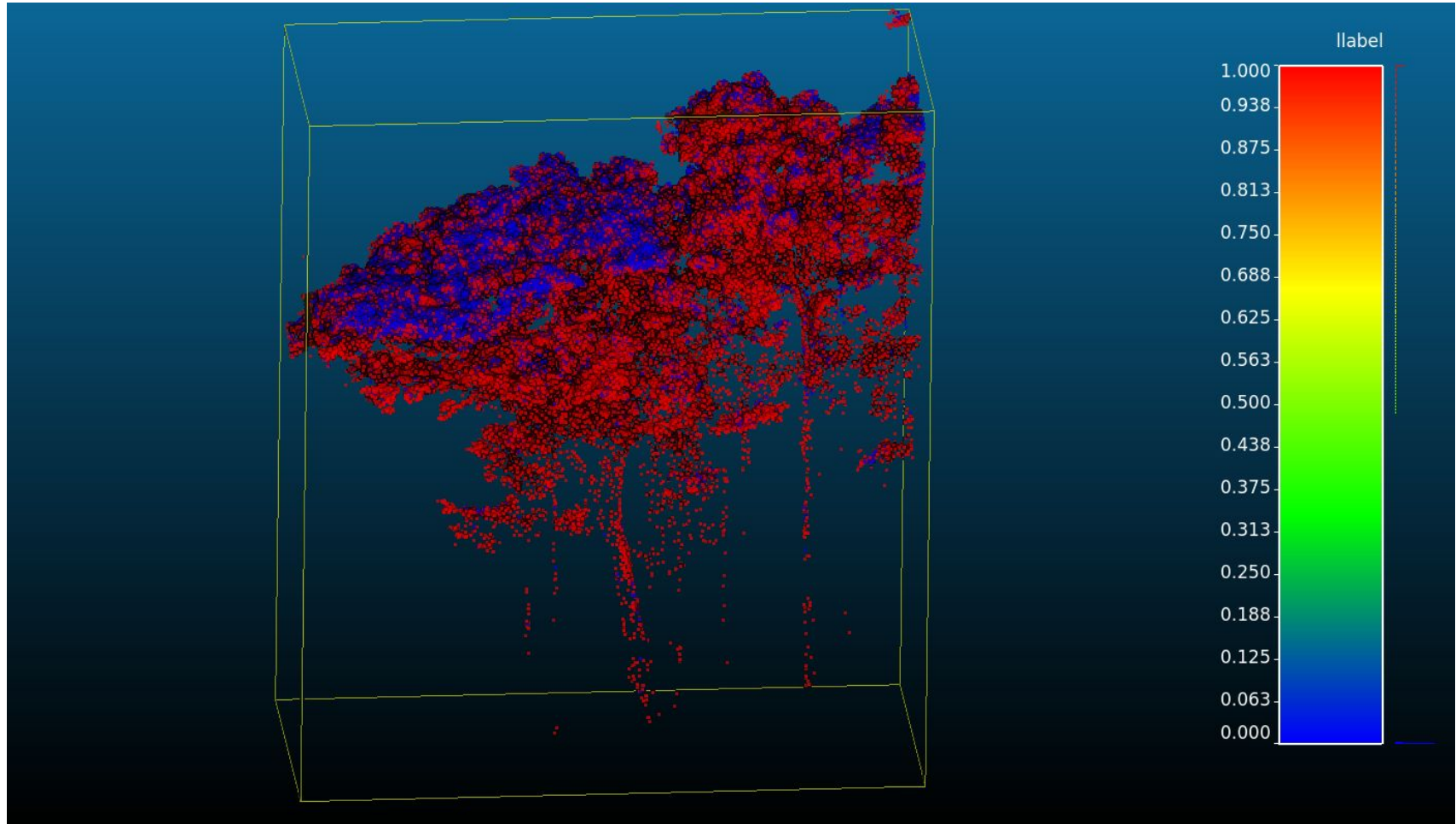


Figure 3. A part of TLS Digital Terrain Model (DTM)

Annexe D. Training on TLS predict on ULS



Prototype model : Point-Voxel based neural network

Semantic segmentation on ULS data



- ❑ label transfer from TLS to ULS data
- ❑ training with TLS and fine-tuning with ULS
- ❑ **Subsample ULS-like data from the TLS data**

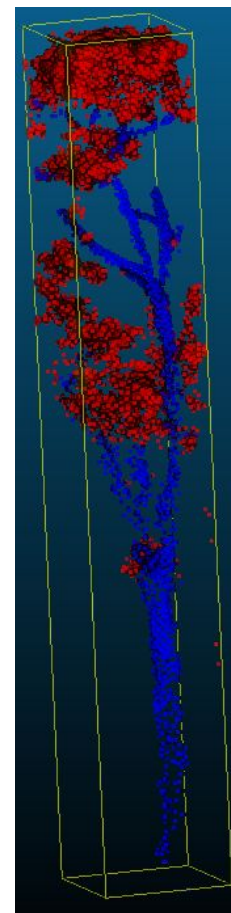
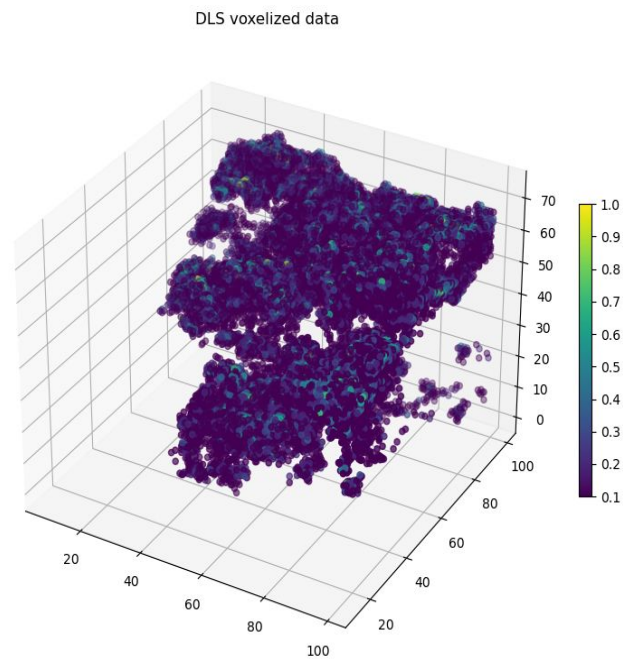
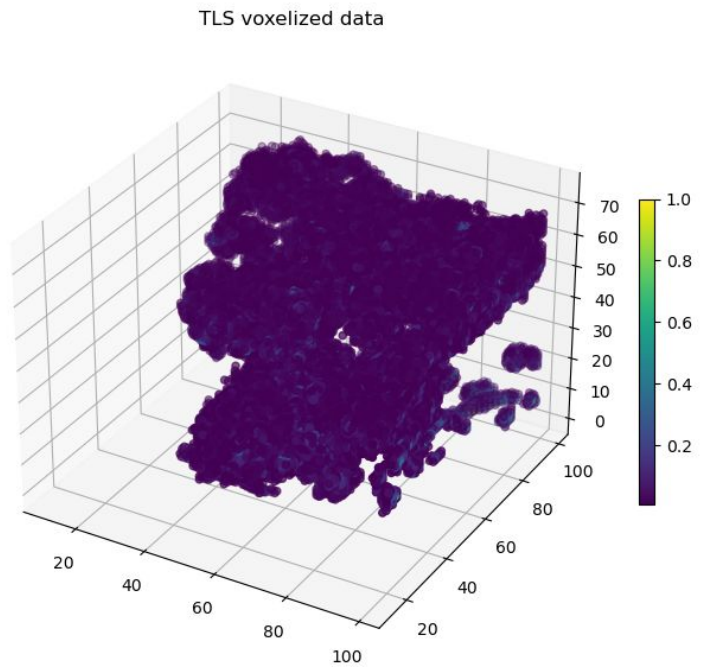


Figure 7. Subsample ULS-like data from the TLS data

1.3 Subsample ULS-like data from the TLS data

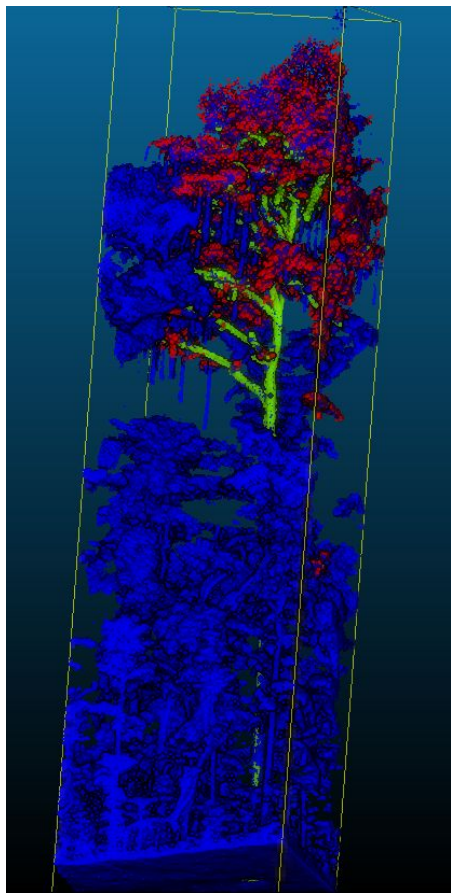


Fig 4: TLS 36.laz

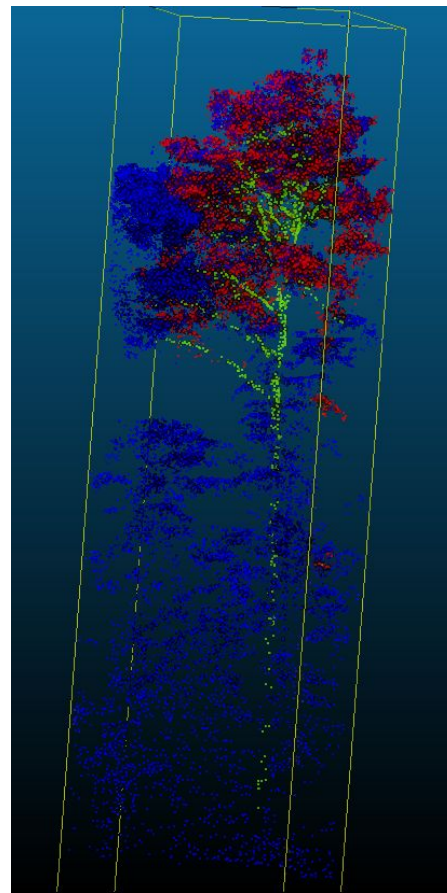
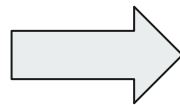


Fig 5: ULS-like data from TLS 36.laz

2.3 Adam or SGD + Nesterov momentum?

moving average for the step in lieu of the gradient. Mathematically, the Adam update equation can be represented as:

$$w_k = w_{k-1} - \alpha_{k-1} \cdot \frac{\sqrt{1 - \beta_2^k}}{1 - \beta_1^k} \cdot \frac{m_{k-1}}{\sqrt{v_{k-1} + \epsilon}}, \quad \text{where} \quad (2)$$

$$\begin{aligned} m_{k-1} &= \beta_1 m_{k-2} + (1 - \beta_1) \hat{\nabla} f(w_{k-1}), \\ v_{k-1} &= \beta_2 v_{k-2} + (1 - \beta_2) \hat{\nabla} f(w_{k-1})^2. \end{aligned} \quad (3)$$

Fig 9. Adam update equation

[*] Nitish Shirish Keskar, Richard Socher, “Improving Generalization Performance by Switching from Adam to SGD”

2.3 Adam or SGD + Nesterov momentum?

For ease of analysis, we rewrite update rules of MaSS in Eq.1 in the following equivalent form (introducing an additional variable \mathbf{v}):

$$\begin{cases} \mathbf{w}_{t+1} \leftarrow \mathbf{u}_t - \eta_1 \tilde{\nabla} f(\mathbf{u}_t), \\ \mathbf{u}_{t+1} \leftarrow (1 + \gamma) \mathbf{w}_{t+1} - \gamma \mathbf{w}_t + \eta_2 \tilde{\nabla} f(\mathbf{u}_t) \end{cases} \iff \begin{cases} \mathbf{w}_{t+1} \leftarrow \mathbf{u}_t - \eta \tilde{\nabla} f(\mathbf{u}_t), \\ \mathbf{v}_{t+1} \leftarrow (1 - \alpha) \mathbf{v}_t + \alpha \mathbf{u}_t - \delta \tilde{\nabla} f(\mathbf{u}_t), \\ \mathbf{u}_{t+1} \leftarrow \frac{\alpha}{1 + \alpha} \mathbf{v}_{t+1} + \frac{1}{1 + \alpha} \mathbf{w}_{t+1}. \end{cases} \quad (8)$$

There is a bijection between the hyper-parameters (η_1, η_2, γ) and (η, α, δ) , which is given by:

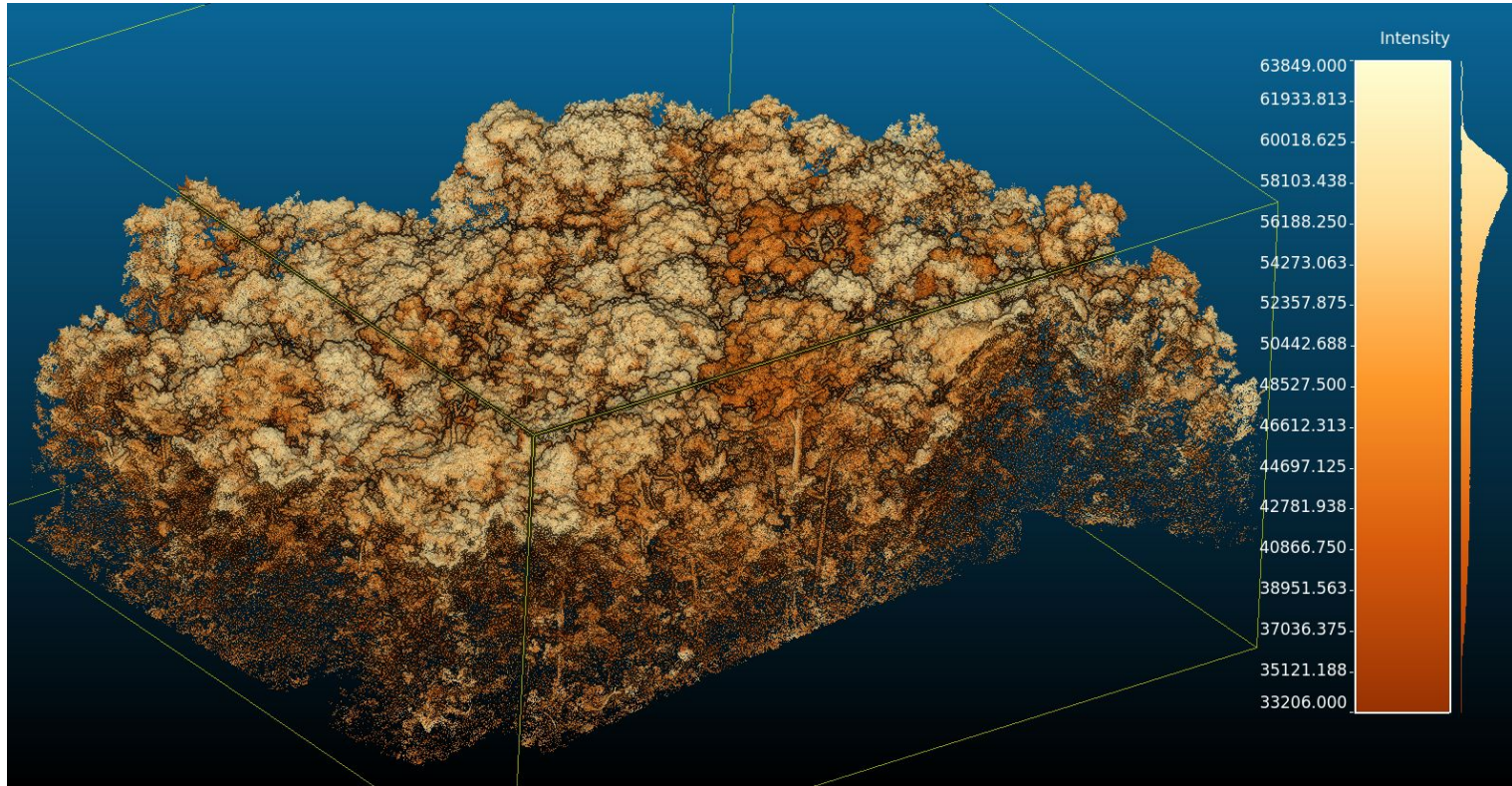
$$\gamma = (1 - \alpha)/(1 + \alpha), \quad \eta_1 = \eta, \quad \eta_2 = (\eta - \alpha\delta)/(1 + \alpha). \quad (9)$$

Remark 3 (SGD+Nesterov). In the literature, the Nesterov's method is sometimes written in a similar form as the R.H.S. of Eq.8. Since SGD+Nesterov has no compensation term, δ has to be fixed as η/α , which is consistent with the parameter setting in [15].

Fig 10. Nesterove SGD update equation

[*] Chaoyue Liu, Mikhail Belkin, "Accelerating SGD with momentum for over-parameterized learning"

Annexe 2. ULS merged data (intensity)



acc train (epoche avg) vs val

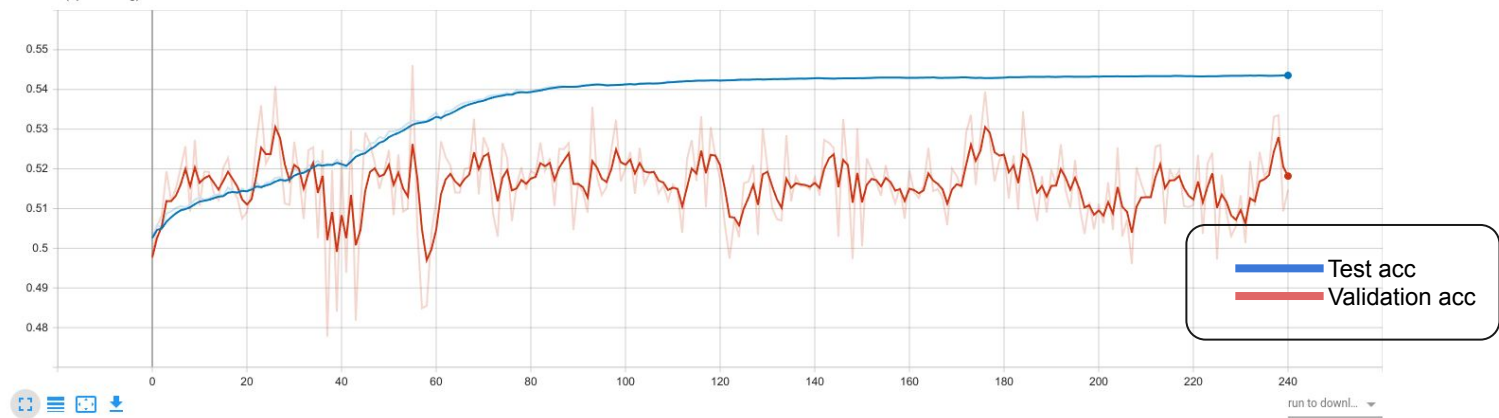


Fig 11. Accuracy of 300 epochs on new dataset, SGD + Nesterov, class weights

loss train (epoch avg) vs val

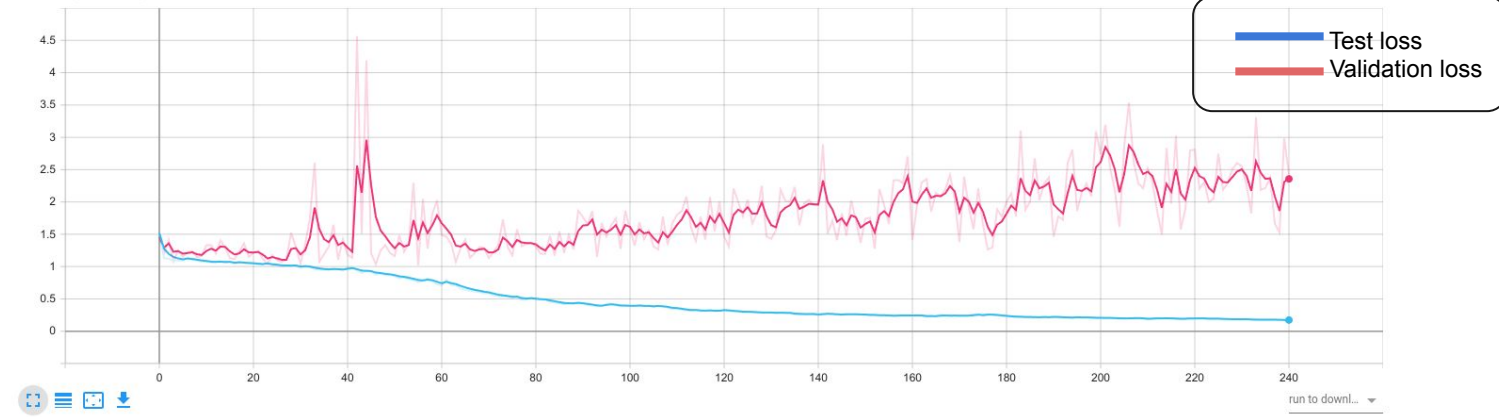
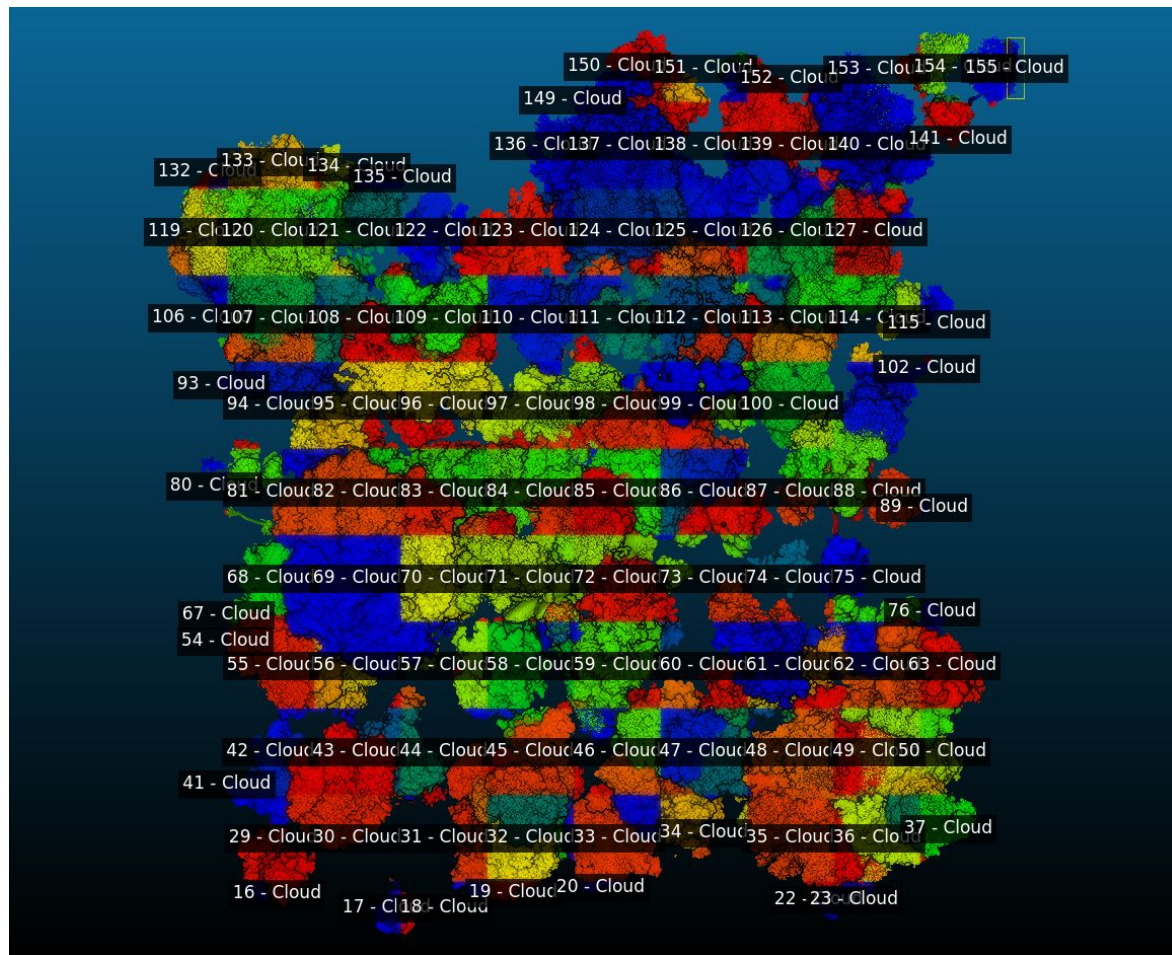
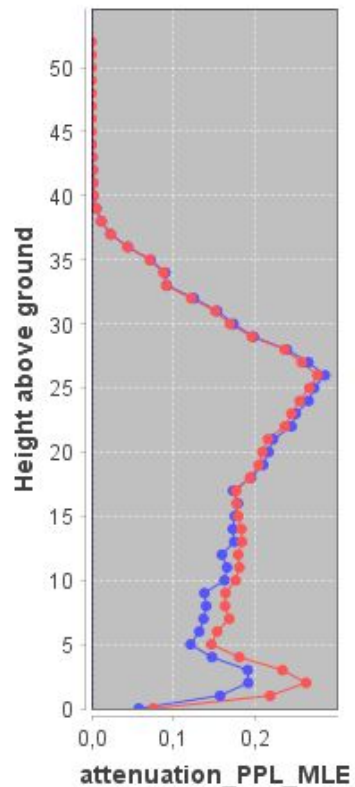


Fig 12. Loss of 300 epochs on new dataset, SGD + Nesterov, class weights

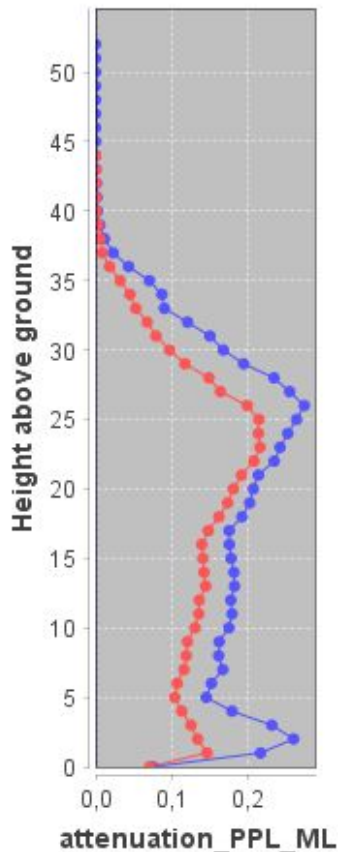
Annexe 3. Tls parcels with labels



attenuation_PPL_MLE
profile



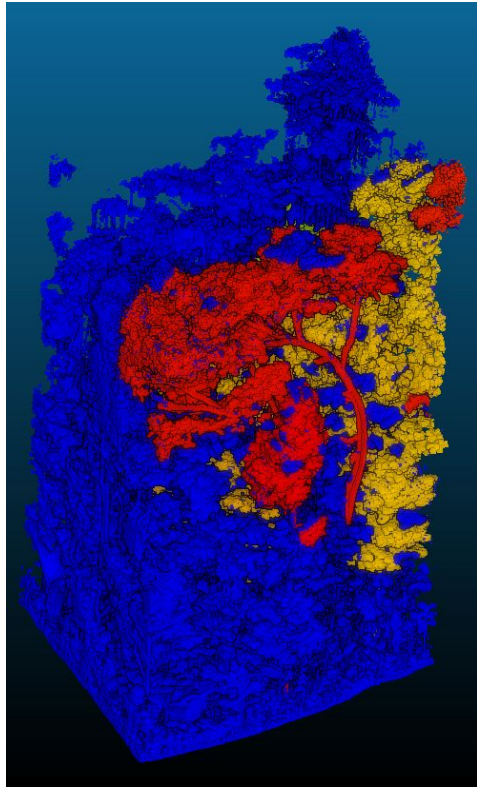
attenuation_PPL_ML
E profile



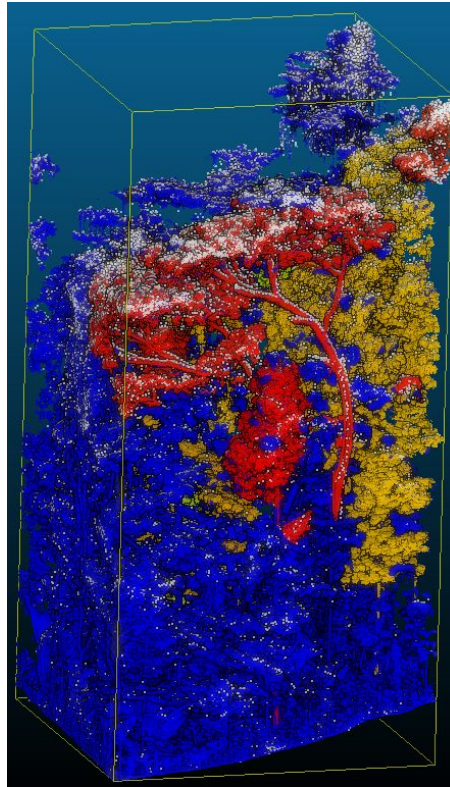
— ULS_195956.vox
— ULS_195956_dec90pct.vox

— TLS.vox
— ULS_195956.vox

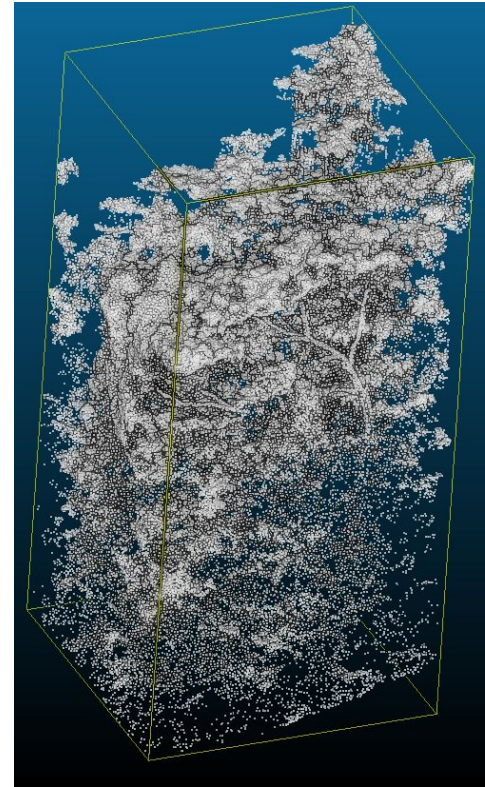
Attenuation



(a) TLS



(b) overlap



(c) ULS

Figure 4. TLS and ULS co-registration

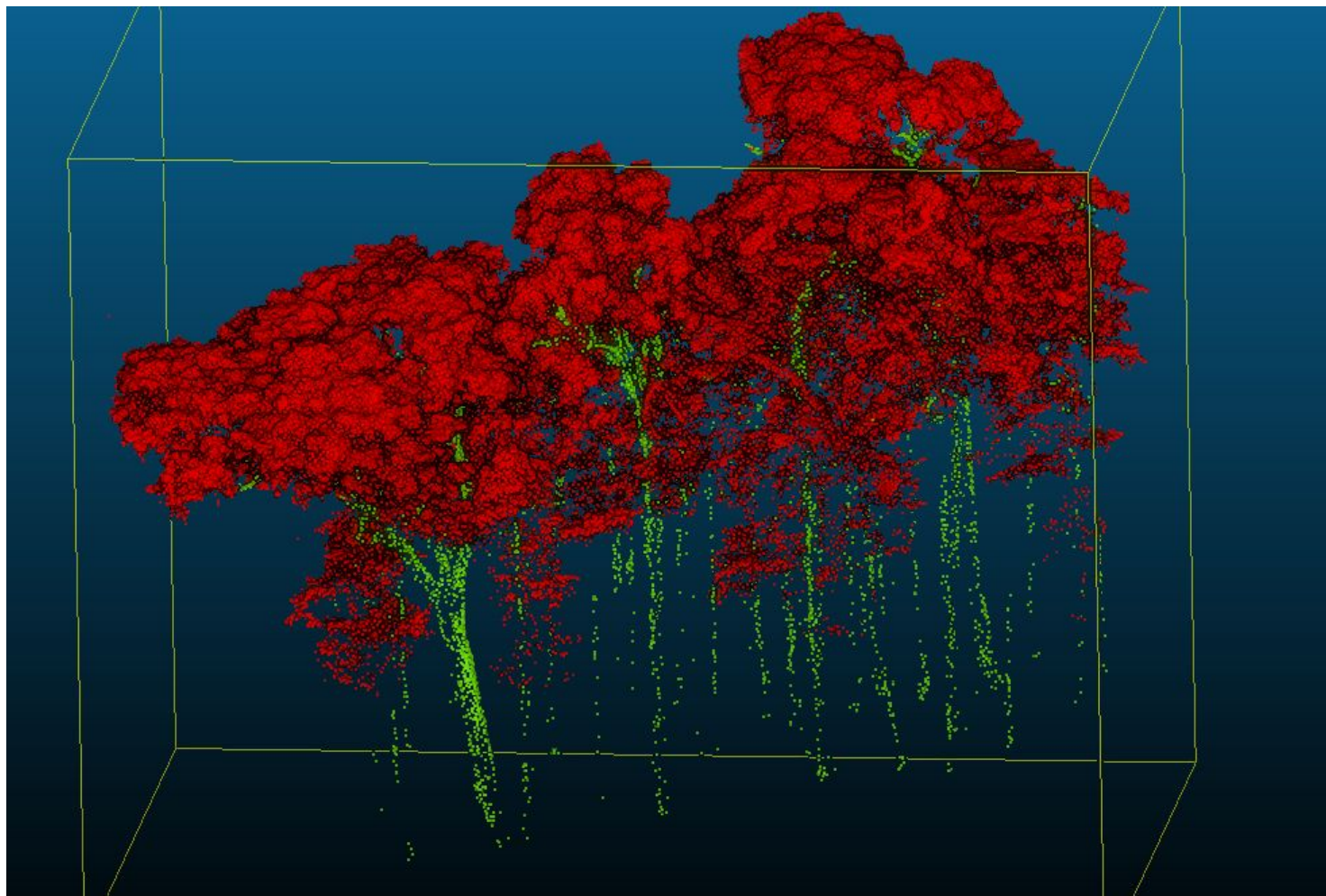


Figure 6. label transfer from TLS to ULS data



NEURAL NETWORKS FOR LEAF/WOOD DISCRIMINATION IN LIDAR FOR LEAF AREA DENSITY ESTIMATION

PhD student:
Yuchen BAI

Supervisor:
Jean-Baptiste DURAND
Grégoire VINCENT
Florence FORBES



2.2 Weighted loss/class weights for wood&leaf

```
>> TrainDataSet is prepared: detach() numpy(). transpose(0, 2, 1) reshape(self.batch_size*self.sample_size, 2)
>>> device=cpu print("test label.detach() numpy(). transpose(0, 2, 1) reshape(batch_size*sample_size, 2) shape")
>>> samples.shape=(60, 5000, 5) detach(). numpy()[0]. T.astype('int64')
>>> samples_cuboid_index.shape=60
>> So checkpoint folder exist, the path is: /home/yuchen/Documents/PhD/phd_mission/src/checkpoints
>> So gradient clipping folder exist, the path is: /home/yuchen/Documents/PhD/phd_mission/src/checkpoints
len(self.train_loader.dataset)= 212
No checkpoints found at /home/yuchen/Documents/PhD/phd_mission/src/checkpoints
===== Start epoch 0 =====
epoch : loss = 0.7638216018676758, weighted_loss = 13.978540420532227
[e=0]>>> [Training] - Current test loss: 0.7638216018676758 - test accuracy: 0.502
epoch : loss = 0.6371403336524963, weighted_loss = 7.388162612915039
[e=0]>>> [Training] - Current test loss: 0.6371403336524963 - test accuracy: 0.5047
epoch : loss = 0.5406445264816284, weighted_loss = 3.8675711154937744
[e=0]>>> [Training] - Current test loss: 0.5406445264816284 - test accuracy: 0.49625
epoch : loss = 0.41227322816848755, weighted_loss = 19.190902709960938
[e=0]>>> [Training] - Current test loss: 0.41227322816848755 - test accuracy: 0.50155
epoch : loss = 0.32395246624946594, weighted_loss = 2.7720727920532227
[e=0]>>> [Training] - Current test loss: 0.32395246624946594 - test accuracy: 0.5053
epoch : loss = 0.1903519630432129, weighted_loss = 7.873373985290527
[e=0]>>> [Training] - Current test loss: 0.1903519630432129 - test accuracy: 0.49925
epoch : loss = 0.12450125813484192, weighted_loss = 3.9948694705963135
[e=0]>>> [Training] - Current test loss: 0.12450125813484192 - test accuracy: 0.49805
epoch : loss = 0.1914239078760147, weighted_loss = 1.1349207162857056
[e=0]>>> [Training] - Current test loss: 0.1914239078760147 - test accuracy: 0.50235
epoch : loss = 0.05195620283484459, weighted_loss = 1.7309350967407227
```

Fig 7: wood point predictions are dominant

```
sklearn.utils.class_weight.compute_class_weight(class_weight, *, classes, y)
```

[source]

Estimate class weights for unbalanced datasets.

Parameter	class_weight : dict, 'balanced' or None
s:	If 'balanced', class weights will be given by $n_samples / (n_classes * np.bincount(y))$. If a dictionary is given, keys are classes and values are corresponding class weights. If None is given, the class weights will be uniform.
classes :	ndarray Array of the classes occurring in the data, as given by <code>np.unique(y_org)</code> with <code>y_org</code> the original class labels.
y :	array-like of shape (n_samples,) Array of original class labels per sample.
Returns:	class_weight_vect : ndarray of shape (n_classes,) Array with <code>class_weight_vect[i]</code> the weight for i-th class.

References

The "balanced" heuristic is inspired by Logistic Regression in Rare Events Data, King, Zen, 2001.

Fig 8: Class weights given by sklearn are better than preset