Tree Species Mapping Using Voxel Density Rasters With Convolutional Autoencoders

Gunner Stone [1] **Alireza Tavakkoli** [1]

Abstract

Cataloging forest biodiversity in a stem mapping is an integral part in creating a comprehensive forest inventory. This laborious procedure aids in both quantifying the amount of carbon taken out of the atmosphere and determining fire prevention strategies. Through combining cutting edge remote sensing imagery and deep learning classifiers, this research proposes to help drastically reduce the time spent manually analysing metrics about a region and its ecosystem by automating the cataloging of tree species.

1. Introduction

Forests are an integral component in climate change mitigation because of their ability to absorb carbon into their biomass and soil. Conversely, wildfires worsen the climate crisis by producing excess CO2 emissions. One of the details recorded in a forest inventory is the species biodiversity within a forest. Cataloguing biodiversity across a region helps map potential fire-hazards by differentiating more flammable tree species from others. Additionally, different species have different carbon-densities, which are an important factor when calculating an overall forest's carbon sink(1).

Therefore, keeping track of biodiversity allows environmental scientists to better monitor and maintain forest health, more accurately evaluate the overall forest's contribution to climate health, and provide better guides for terrestrial studies surrounding fire prevention. Traditionally, cataloging biodiversity is a challenging task because of the extensive training required for a human to make correct species classifications. Additionally, cataloging species is a laborious process that requires someone to physically be on site labeling trees manually. Automating this process would have significant implications for saving expenses in both training time and manual labeling.

Currently, a technological solution that aims to take the labor cost away from forest inventorying utilizes terrestrial and airborne laser scanning (TLS/ALS) to create a visualized tree distribution through a plot of lidar points. This technology can be used to generate point-cloud data for a region of forest which can save forestry professionals the labor of manually measuring everything themselves. This information can later be put into a computer to create a visualization of the data. Despite having a mostly complete way of automating a forest inventory, a necessary function this technology is not yet advanced enough to incorporate automatically is species-mapping. Automatic tree species identification from raw unstructured point-cloud data is a complex problem that has not yet been solved. The goal of this project is to accurately identify tree species in a generalized and robust way utilizing terrestrial and airborne lidar.

Solving the species mapping challenge will have favorable implications for the feasibility of a completely automatic tree inventorying process. Solving this problem will help drastically reduce the time spent manually analysing metrics about a region and its ecosystem by automatically generating a forest inventory, complete with species identification.

2. Related Works

Currently, many tree species classification (TSC) algorithms suffer in computational performance as a result of the high dimensionality required to achieve quality results. A common strategy to combat this issue is to resample the point cloud to a fraction of its original size and to then perform classification on the resample. However, performing this resampling comes at the cost of losing valuable information about the original scans(2). TSC algorithms also suffer from a lack of usable sample data to train on. Small amounts of training data often produce models that fail to generalize and are also prone to overfitting. This section reviews different implementations of tree species identification, where they fall short, and where they can be improved.

Typically, traditional machine learning aims at learning how

[1]Department of Computer Science and Engineering, University of Reno, Nevada, United States of America. Correspondence to: Gunner Stone <gunnerstone@unr.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

different feature parameters contribute towards making linear/nonlinear classification decisions. One of the simplest ways for humans to determine the species of tree is by observing the features of its leaf structure(3; 4). Unfortunately this strategy is not possible when a tree is out of season and its leaves have shed. This warrants the need for a general solution that works independent of a visible leaf structure. Extensive work has already been done to extract explicit tree structure (ETS) from lidar(5; 6). These explicit tree structures consist of many leaf-independant features such as: tree height, crown height, crown volume, and bark texture just to name a few. Surprisingly, classifying species using explicit tree features has resulted in lackluster results. ETS and similar variations have been used as feature parameters in the following models LOOCV-SVM, KNN, Ada Boost, RF, NB, and LDA. But while being both quick to train and make classifications, none of these traditional classifiers are robust enough to produce reliable results on generalized data(2). From these results it can be implied that ETS features alone are not enough for a model to classify trees apart from one another.

Deep learning frameworks have been looked at for their utility in finding hidden patterns within large amounts of data. This capability lends itself well to a tree lidar scan. Rather than classifying using a list of ETS features, deep learning frameworks use the entire point cloud as a composite of millions of 3D-coordinate feature parameters. Unlike traditional machine learning techniques, deep learning models, notably PointNet++(7), produce results that reliably hold up to those of a human. However, these TSC deep learning models suffer heavily from overfitting(2). Because of how dependent these types of models are on having large amounts of data samples, insufficiently sized datasets have heavily contributed to these models overfitting the training data. It is apparent that there exists a lack of high quality pre-labeled data. This lack of data is one of the more highlighted reasons behind not achieving more reliable and robust results(2).

This paper's method for tree species classification proposes a novel solution that addresses the challenges that these other implementations face. It allows for learned parametric feature reduction, scales well with small amounts of training data, and has competitive classification scores.

3. Methods

The proposed work creates a latent representation for a tree density raster by mapping it to a low dimensional manifold using a convolutional autoencoder. The autoencoder is trained jointly with a categorical classifier to help direct the encoding half of the autoencoder to produce a latent space with discriminatory properties such that it prioritizes encoding both the visual structure of a tree and its species into the low dimensional manifold. Thus, the latent space manifold can be used to determine what species a given tree density raster most represents.

3.1. The Original Dataset

Because such a big challenge to this species classification problem is a lack of data, this section goes into detail about how this paper's dataset was created. The dataset consists of several individually segmented TLS scans for the following 7 tree species of Western North American: Black Oak, Douglas Fir, Jeffrey Pine, Lodgepole Pine, Ponderosa Pine, Sugar Pine, and White Fir.

In total there were 0 scans of Black Oaks, 1 scan of Douglas Firs, 2 scans of Jeffrey Pines, 2 scans of Lodgepole Pines, 13 scans of Ponderosa Pines, 2 scans of Sugar Pines, and 5 scans of White Firs.

3.2. Density Rasters Explained

One of the goals of feature engineering is to select a small subspace of the original data while, at the same time, preserving important information relevant to the task at hand. The chaotic and unstructured nature of raw lidar points in 3 dimensional coordinate space does not lend itself well to machine learning. It is computationally expensive, if not impossible to get a comprehensive understanding of a singular point in 3d-coordinate space and its relationship to the structure of its nearest neighbors. Therefore, it is necessary to discretize the unstructured 3d-coordinate space into one where a network could make more meaningful inferences about the input features.

One of the ways this can be accomplished is through the use of rasters. Rasters in this context are essentially 2d photographs of the lidar points in 3d-coordinate space. These photographs would be taken at some camera position and angle. The closer a point is to the camera, the brighter it is, while the further away from the camera, the darker. However, generating a raster this way causes information about points not seen by the camera to be lost. This is where a density raster can be utilized.

A density raster can by imagined as a planar sweep of the raw unstructured data. The sweeping plane is divided into MxM evenly sized squares. If one of the squares hits a lidar point along its path, that square's total points goes up by 1. This planar sweep generates a comprehensive 2d raster of the original coordinate space input where each pixel's intensity is determined by the number of points hit during the sweep.

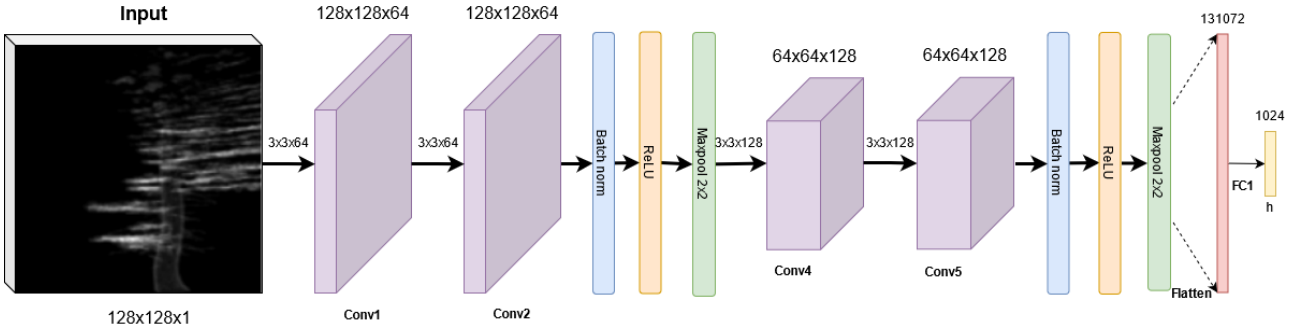

Figure 1. The model's architecture showing the encoding portion of the convolutional autoencoder with latent space manifold h.

3.3. Density Raster Pipeline

For a given TLS scan of a tree, a 3d meshgrid was constructed with dimensions MxMxM where M is the desired resolution of the 2d rasters. This 3d meshgrid is fit to only encompass the points of a scan. Therefore, the constructed meshgrid aims to segment a tree into M equally sized chunks along each of its three dimensions. Once a meshgrid is snugly fit on top of a tree scan, all points within that scan are fit to the closest neighboring point in the meshgrid. Because the meshgrid divides all dimensions into M equally sized chunks, this step normalizes height, width, and depth.

Fitting scan points to their nearest neighboring meshgrid point using euclidean distance is not feasible for large amounts of unstructured data and therefore is intractable in this case. To solve this computational challenge, a KD-Tree data structure was created from the meshgrid to save on computational costs by cutting this point-fitting process down to O(N log(N)) time.

This fitting process is necessary because fitting original scan coordinates to a 3d meshgrid converts a TLS scan from being represented by millions of uncorrelated points in a real number coordinate space to discrete indices within the constrained meshgrid.

Given both the meshgrid and indices where mappings were found, a 3d boolean matrix can be created where positive and negative mappings are indicated with a binary state. This 3d boolean matrix can be visualized as a voxelized tree structure as shown in Figure 2.

From this voxelized tree structure a 2d raster can be created by calculating the voxel densities along the depth of a specified dimension. Different voxel density rasters can be generated by rotating the voxel matrix along one of its axes. To maintain a side profile of a tree in the rasters, this rotation was only done by rotating along the y axis. It's important to note that lossless matrix rotations can only be achieved on rotations of 90, 180, 270. All others are only approximations and are also more computationally challenging to find. For this project, 8 different rotations were found for a 3d voxel matrix in increments of 45 degrees.

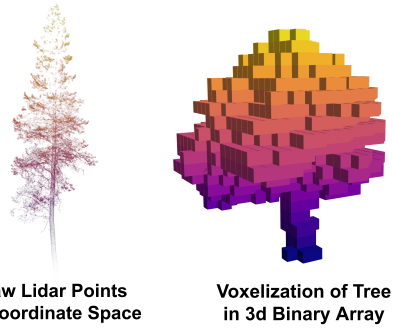

Figure 2. This illustration visualizes the millions of original unstructured lidar datapoints of a Lodgepole Pine and its respective lower dimensional representation in binary voxel space with dimensions 20x20x20.

Additionally, more rasters of a tree can be created by adding vertical angular variation. Vertical variation can be accomplished by rotating about either the x or z axes. The issue faced by adding additional vertical variation in the rasters at this stage is that it would add more data loss by rotating the matrix by a non lossless degree amount less than 90. Because the vertically rotated 3d matrix is defaulting this stage to an approximation, carrying out the 45 degree rotations about the y axis would result in a raster that is an approximation of an approximation. In order to solve this data-loss challenge, the vertical variations were done to the original 3d coordinate points of the TLS scan before the discretization process.

This is computationally easily done within a 3d coordinate space using vector calculus. While this process does make vertical rotation lossless, performing this variation at the beginning of the pipeline forces the voxel-fitting step to be repeated. This decision makes a trade-off between data generation time and data quality.

3.4. Model Architecture

The architecture, shown in Figure 1. uses a convolutional autoencoder to create the manifold representation. The encoding and decoding components were inspired by VGG architecture.

The encoding mapping takes the input raster and runs it through several layers of 3x3 convolutional kernels, expanding the channel depth along the way. Batch normalization, ReLU activations, and Max Pooling are used to keep the activation outputs and size of the network parameters under control. The encoder flattens the last convolutional output and downscales it using a dense layer whose dimensions are determined by a bottleneck size hyper-parameter. Larger bottleneck sizes preserve higher quality reconstructions but don't learn economical representations of the input space or its labels, which is important in this application.

The decoding architecture is essentially a reverse of the encoding, replacing convolutional kernels for convolutional transposes, etc. It is however worth noting that the final activation function for the decoder needs to be a sigmoid. This activation allows the decoder to output a gray-scale raster that contains pixel values between 0 and 1.

The classifier architecture shown in Figure 3 is essentially a small multi-layer perceptron that takes the manifold output from the encoder and maps it to 7 output nodes using softmax as the final activation.

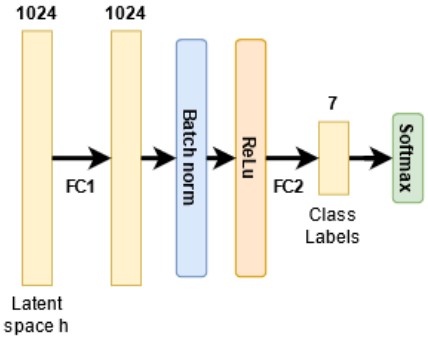

Figure 3. A diagram of the multi-layer perceptron classifying latent manifold h into 7 different species classes.

The model uses L2 loss for the raster reconstruction and categorical cross entropy for the species classification. Both of these losses are added together to compose the total loss. The network's total loss should represent its abilities in both classifying a given tree species as well as its ability to reconstruct the original input. This model was trained for 100 epochs using 90% LR reduction on validation loss plateau with a patience threshold of 3. The best validation loss checkpoint model was saved and then run on the test set of rasters.

4. Experimental Results

600 tree density rasters were generated from 25 different trees across 7 species. The data was then split into 60% train, 20% validation, and 20% test. The tree density rasters started out with a dimensionality of 128x128, which was then reduced using the convolutional autoencoder to a 1024 one-dimensional vector. The convolutional autoencoder and the species classifier were trained together over 100 epochs. The learning rate was set to decay by 90% every time the validation loss plateaued. The weights and biases that produced the model with the lowest validation accuracy were saved and ran on the test-set to produce the results shown in Figure 4.

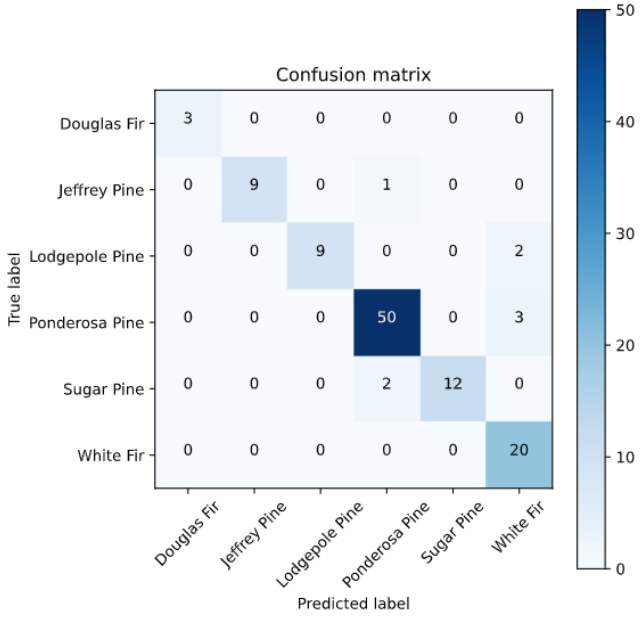

Figure 4. A confusion matrix showing the predictions of the species classifier on the test set.

The confusion matrix easily allowed for the precision and recall to be calculated for each species. This gives a more comprehensive overview of the classifier's performance considering the dataset was imbalanced. Precision and recall values for each species are listed in Table 1.

Because the dataset is imbalanced, a raw accuracy metric is unable to effectively describe the overall performance of this model. Given the precision and recall values across all categories, the average F_1-score can be calculated. The F_1-score is a better performance metric instead of accuracy for more imbalanced datasets, so it is used here as an indicator for the models overall performance.

The F_1-score for this model ended up being **0.938**.

Table 1. Summary of precision and recall from test set

Classification Performance Measurements		
Species	**Precision**	**Recall**
Black Oak	1.0	1.0
Douglas Fir	1.0	1.0
Jeffrey Pine	1.0	0.9
Lodgepole Pine	1.0	0.818
Ponderosa Pine	0.943	0.943
Sugar Pine	1.0	0.857
White Fir	0.8	1.0

5. Conclusion

This research uses a combination of a convolutional autoencoder and a multi-layer perceptron to create a lightweight model that is able to accurately distinguish between seven different tree species using their latent manifold representations. The inference and parsing times of this proposed model are competitive with other deep classifiers and therefore would allow this model to easily integrate into large-scale classification tasks. The proposed model additionally offers reliable classifications that compare with more heavy-weight deep belief models like PointNet++ and IncResV2.

Future work can expand upon this paper by attempting to either increase the classification performance, allow for more versatility in integrating additional species, or increase the time and space efficiencies of the model's pipeline.

6. Acknowledgements

I'd like to thank the members of the GEARS Lab at University of Nevada, Reno for their assistance in data collection, segmentation, and labeling, without which this project would not have been possible.

References

[1] Brian J. Clough, Miranda T. Curzon, Grant M. Domke, Matthew B. Russell, and Christopher W. Woodall, "Climate-driven trends in stem wood density of tree species in the eastern united states: Ecological impact and implications for national forest carbon assessments," *Global Ecology and Biogeography*, vol. 26, no. 10, pp. 1153–1164, 2017.

[2] Zhouxin Xi, Chris Hopkinson, Stewart B. Rood, and Derek R. Peddle, "See the forest and the trees: Effective machine and deep learning algorithms for wood filtering and tree species classification from terrestrial laser scanning," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 168, pp. 1–16, 2020.

[3] O. Mzoughi, I. Yahiaoui, N. Boujemaa, and E. Zagrouba, "Advanced tree species identification using multiple leaf parts image queries," in *2013 IEEE International Conference on Image Processing*, 2013, pp. 3967–3971.

[4] I. Yahiaoui, O. Mzoughi, and N. Boujemaa, ," in *Leaf Shape Descriptor for Tree Species Identification*. 2012, pp. 254–259, IEEE.

[5] Yi Lin and Martin Herold, "Tree species classification based on explicit tree structure feature parameters derived from static terrestrial laser scanning data," *Agricultural and Forest Meteorology*, vol. 216, pp. 105–114, 2016.

[6] Markku Åkerblom, Pasi Raumonen, Raisa Mäkipää, and Mikko Kaasalainen, "Automatic tree species recognition with quantitative structure models," *Remote Sensing of Environment*, vol. 191, pp. 1–12, 2017.

[7] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," 2017.