remote sensing

MDPI

Article

Beyond Measurement: Extracting Vegetation Height from High Resolution Imagery with Deep Learning

David Radke [1,*], Daniel Radke [2,3] and John Radke [4,5]

1 David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, ON N2L 3G1, Canada
2 Department of Computer Science, Universität des Saarlandes, 66123 Saarbrücken, Germany; dradke@mpi-inf.mpg.de
3 Max-Planck-Institut für Informatik, 66123 Saarbrücken, Germany
4 Landscape Architecture and Environmental Planning, City and Regional Planning, University of California, Berkeley, CA 94720-2000, USA; ratt@berkeley.edu
5 Center for Catastrophic Risk Management, University of California, Berkeley, CA 94720-1922, USA
* Correspondence: dtradke@uwaterloo.ca

Received: 16 October 2020; Accepted: 16 November 2020; Published: 19 November 2020

Abstract: Measuring and monitoring the height of vegetation provides important insights into forest age and habitat quality. These are essential for the accuracy of applications that are highly reliant on up-to-date and accurate vegetation data. Current vegetation sensing practices involve ground survey, photogrammetry, synthetic aperture radar (SAR), and airborne light detection and ranging sensors (LiDAR). While these methods provide high resolution and accuracy, their hardware and collection effort prohibits highly recurrent and widespread collection. In response to the limitations of current methods, we designed Y-NET, a novel deep learning model to generate high resolution models of vegetation from highly recurrent multispectral aerial imagery and elevation data. Y-NET's architecture uses convolutional layers to learn correlations between different input features and vegetation height, generating an accurate vegetation surface model (VSM) at 1×1 m resolution. We evaluated Y-NET on 235 km^2 of the East San Francisco Bay Area and find that Y-NET achieves low error from LiDAR when tested on new locations. Y-NET also achieves an R^2 of 0.83 and can effectively model complex vegetation through side-by-side visual comparisons. Furthermore, we show that Y-NET is able to identify instances of vegetation growth and mitigation by comparing aerial imagery and LiDAR collected at different times.

Keywords: deep learning; artificial intelligence; vegetation surface modeling

1. Introduction

Frequent and accurate assessments of vegetation are critical for managing forest biomass, designing effective mitigation strategies in anticipation of wildland fires, and managing vegetation in the wildland urban interface (WUI) [1–4]. Moreover, quality information about forest structure and biomass helps analysts develop smarter logging strategies, predict economic yield, and analyze the effects of vegetation in socio-technical systems. Vegetation, of all the common geographic information systems (GIS) data layers, is arguably the most dynamic due to constantly changing shape and density, a trend that appears to be accelerating with climate change [5]. An accurate model of vegetation height, which we refer to as a *vegetation surface model* (VSM), provides important insights for environmental planning, risk management, and economic benefits.

Current methods of remote sensing for detailed vegetation information include manual ground surveys, aerial photogrammetry, synthetic aperture radar (SAR), and light detection and ranging (LiDAR) [6–9]. While these methods are widely used, they are all subject to multiple

limitations. Ground surveys are often time-consuming and slow, while aerial photogrammetry requires premeditated collection of multiple images at specific angles. This is not only time-consuming, but difficult to scale across large swaths [10]. X-band SAR has been used to generate VSMs using radio waves, although the process requires special antennas installed on aircraft or spacecraft. The accuracy and resolution of SAR is heavily dependent on the relationship between stem biomass and characteristics of the vegetation. This relationship is unique to specific vegetation species, preventing techniques from being truly general [8,11–14]. Both airborne and terrestrial-based **LiDAR** involve actively measuring and recording complex three-dimensional (3D) vegetation to generate an accurate VSM from a LiDAR *point cloud* (Figure 1a). LiDAR is the current state-of-the-art for VSMs, although has the trade-off of being limited to a range of hundreds of meters. This relatively short range requires low-altitude aircraft and imposes similar restrictions to photogrammetry in terms of scalability and recurrence [15,16]. Although the cost of LiDAR scanners is bound to decrease in the future, the time consumption of collection will remain costly, continuing to limit the recurrence of widespread airborne LiDAR. Furthermore, LiDAR data contains bird strikes, power-lines, and water body absorption, which may not represent the true landscape. While algorithms exist to mitigate known sources of errors, post-processing can introduce other sources of noise in the LiDAR point cloud. Only a fraction of the continental United States has been scanned by LiDAR due to the immense cost and collection burden, and most areas are scanned only once with no projected re-scan rate [17].

The limitations of current methods have limited the potential for applications, due to low recurrence rates of accurate VSMs. With the rise in popularity of deep machine learning, we postulate that new intelligent methods of remote sensing must overcome current time and cost limitations. We propose extracting detailed information from frequent and relatively inexpensive sources of data, which will further broaden the scope of applications that require accurate and frequent remote sensing [18].

Multispectral aerial imagery is relatively inexpensive and well suited for recording two-dimensional (2D) data of the Earth's surface at different spatial resolutions (n-m resolution maps to a $n \times n$ m square on the ground). Satellites and planes have captured images of the Earth for decades, and today companies like *DigitalGlobe* [19] and *Planet* [20] collect high resolution imagery of earth with high recurrence. In addition, elevation layers such as digital elevation models (DEMs) which are publicly available at various spatial resolutions through the United States Geological Survey (USGS) [21] and other organizations worldwide. We refer to DEMs and other raster layers derived from elevation analysis with the term *terrain data*.

Figure 1. (**a**) Light detection and ranging (LiDAR) scan of a tree; (**b**) 1-meter resolution National Agriculture Imagery Program (NAIP) imagery; (**c**) 30-meter resolution Landsat imagery.

Previous work has explored the possibility of extracting a VSM from low-resolution multispectral images (Figure 1c) [22–24]; however, current practical applications often require higher resolution data. For example, wildland fire suppression is most effective when direct attack methods are used and enables firefighters to work close to the fire perimeter. Current state-of-the-art fire models use a 30 × 30 m spatial resolution imagery to predict fire spread [25,26], causing a spatial disconnect between models and fire fighting strategies on the ground. Furthermore, since current methods

have limited range and recurrence, up-to-date information from current practices would require premeditated scanning strategies from recently before the fire, when smoke occlusion is not an issue. This would require accurate predictions of where fires will happen beforehand, which is not feasible. Contrarily, highly recurrent aerial imagery would likely have a recent image prior to the fire.

We developed Y-NET, a novel deep learning model to overcome the scanning recurrence and range limitations of current methods, and to improve the spatial resolution of previous related work. Y-NET generates a VSM with 1×1 m resolution given multispectral 2D aerial imagery (Figure 1b) and terrain data. Y-NET's architecture is designed with *a priori* knowledge of the input data to learn from imagery and terrain inputs separately. Y-NET combines the compressed data together in latent space for accurate vegetation height estimations. To validate the efficacy of our approach, we use aerial imagery obtained from the United States National Agriculture Imagery Program (NAIP) [27], and DEMs and LiDAR scans from the USGS [21], all of which are freely available to the public. We evaluated Y-NET in the East San Francisco Bay Area, a highly-researched and high-risk area of wildland fire, where mitigation strategists are actively looking for high resolution modeling through multi-million dollar government funded research grants [28–30].

Y-NET uses supervised learning, meaning the model trains on landscape A for which LiDAR data are included as a ground truth measure. Y-NET is then evaluated on landscape B, where the model only receives imagery and terrain data to recreate the 3D structure of B. Regions in landscape A are separated into commonly named and disjoint *training* and *validation* datasets for training. Landscape B composes the *testing* dataset, which is completely spatially separate from landscape A. To evaluate the performance of Y-NET, we compare the generated VSM for landscape B with the corresponding LiDAR and calculate the pixel-wise error in height, consistent with other work [14,31,32]. We find empirically that Y-NET achieves high accuracy and R^2 value with 1×1 m spatial resolution data by formalizing the problem as being similar to semantic image segmentation (similar to past work [33–35]).

While we evaluated Y-NET for *spatial generalizability* (training on A, testing on B), we expect that *temporal generalizability* may also be possible given available data (training on A_t, testing on A_{t+i} where t and i represent time). However, a proper evaluation of temporal generalizability requires high-resolution multispectral imagery and LiDAR from two distinct times for the same location. We observed two LiDAR scanning missions within our study site, one from 2007 and one from 2018. Since four-band multispectral NAIP imagery was not collected in our study site in 2007, we were unable to perform a proper temporal generalizability evaluation. However, in Section 4 we visually compared Y-NET's VSM with 2016 four-band NAIP imagery to the 2007 LiDAR in our study site, and show that Y-NET can effectively identify vegetation mitigation and growth given temporal differences. We leave a full in-depth analysis of temporal generalizability to future work reliant on available data.

The contributions of our work are three-fold: (i) we developed Y-NET, a novel deep learning model capable of highly accurate VSM generation given four-band multispectral imagery and terrain data, (ii) we evaluated Y-NET on real-data from a highly-studied region at high risk of wildland fire, both statistically and visually for spatial generalizability, and (iii) we visually evaluated the potential for temporal generalizability of Y-NET and show it can identify vegetation mitigation and growth.

2. Materials and Methods

2.1. Study Site

Our study was conducted in the East San Francisco Bay Area (Figure 2), chosen for the dense WUI, high vulnerability to wildland fire, and the focus of past vegetation studies [3,4,30,36,37]. Our study site covers approximately 235 km^2 and contains a broad range of vegetation heights, from 0 to 77 m tall composed of grasslands, shrublands, woodlands, and forest. This region is well monitored and studied for remote sensing, environmental planning, and natural disaster research. In addition to the available LiDAR we used to benchmark Y-NET, our study site is located in a Mediterranean climate, which is known to be at high risk of wildland fire, and requires strict vegetation mitigation strategies [38–41].

Helping support up-to-date vegetation monitoring for applications such as wildland fire modeling and vegetation mitigation in areas where VSM information is crucial is the main motivation of our work.

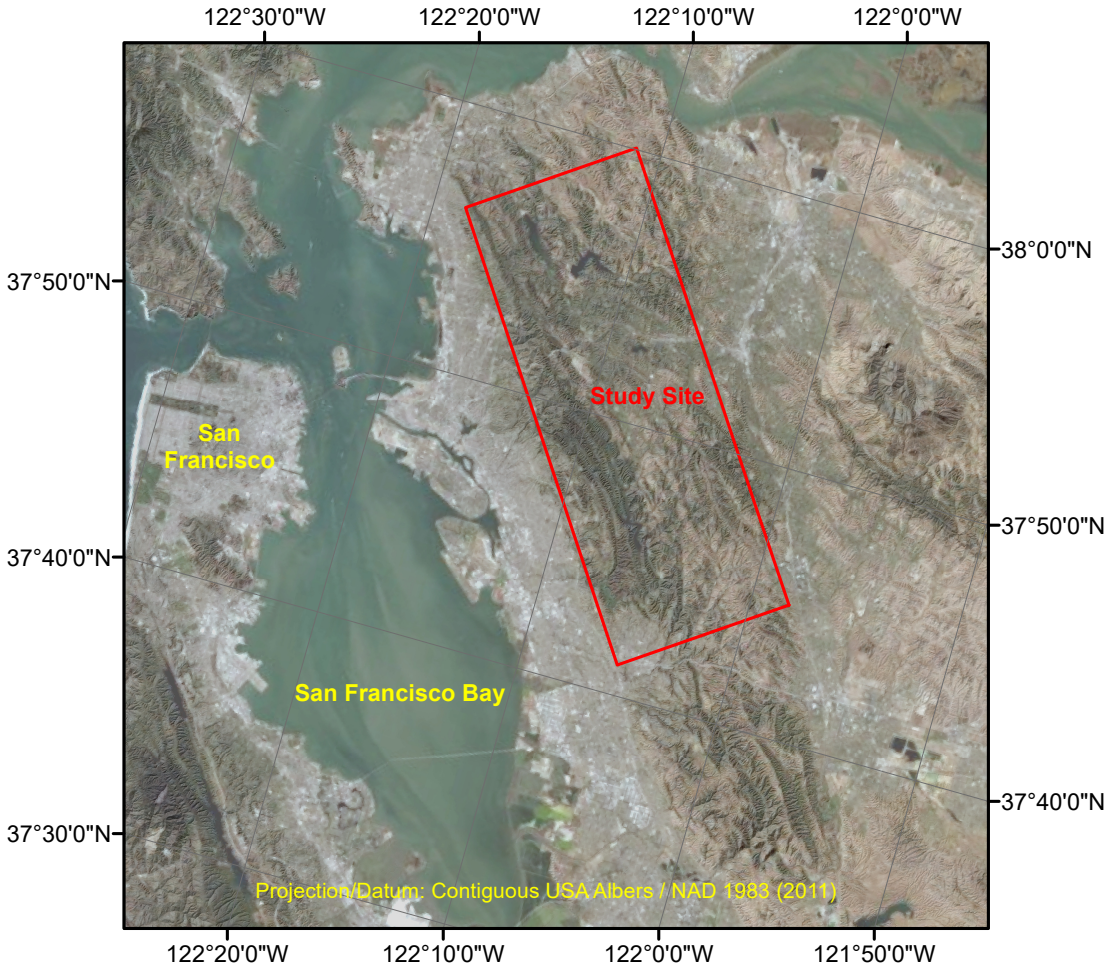

Figure 2. Our study site is located in the East San Francisco Bay Area, covering roughly 235 km^2 of the highly-researched wildland urban interface (WUI) that is at high risk of wildland fire.

2.2. Data Features and Processing

Y-NET utilizes nine layers of input data, divided into two groups: visual data (six layers) and terrain data (three layers). In practice, these features may be collected for any available date; however, due to available LiDAR, which we used to train and evaluate our model, we selected data from 2018. The study site was divided into non-overlapping tiles of 64 × 64 pixels. For example, the study area (235 km^2) has over 235 million 1 × 1 m pixels for each layer of input data, amounting to 57,373 tiles of size 64 × 64 × 9 features. We divided this dataset into spatially disjoint training, validation, and testing datasets, explained further in Section 2.5. The *validation* dataset was used to assist the training phase, and we emphasize that the model was not evaluated on this dataset and was not exposed to LiDAR during our evaluation or future deployment.

2.2.1. Visual Data

We collected 1 × 1 m resolution multispectral aerial imagery from a 23 July 2018 United States NAIP imaging mission, and project it to the NAD_1983_2011_Contiguous_USA_Albers coordinate system and Albers datum using the ArcGIS Project tool from ESRI [42]. NAIP has **red** (0.6–0.7 µ), **blue** (0.4–0.5 µ), **green** (0.5–0.6 µ), and **near-infrared** (NIR, 0.8–0.9 µ) bands of imagery which are commonly used to identify important vegetation signatures. In addition to the bands themselves,

we combine the red and NIR bands to generate another input layer, **NDVI** using ArcGIS's Raster Calculator tool [43,44]. NDVI represents the normalized vegetation health within each pixel so that $NDVI \in [-1, 1]$, where a higher value is synonymous with healthier vegetation.

To ensure that Y-NET was only exposed to pixels which contain vegetation, we included a layer to mask out structures and roads that may harm training of the model. We used the iterative self-organizing (ISO) cluster unsupervised classification tool in ArcGIS [45] to perform an unsupervised classification on the four NAIP raster bands, separating pixels that are vegetation from water, roads, and structures, similar to recent work [46]. As insurance, we concatenated the generated mask with: road footprints delineated from a GPS verified ground based street centerline survey, and building footprints generated by the AI team at Microsoft using a popular machine learning segmentation model, U-NET, with satellite imagery [47]. We defined our mask as the **footprints** layer, which contains pixels with binary values pertaining to either vegetation or non-vegetation.

The four NAIP bands (red, blue, green, and NIR), NDVI derived from NAIP, and footprints layer detailed above make up the model inputs composed from aerial imagery, and are passed to the visual branch of Y-NET, discussed further in Section 2.5.

2.2.2. Terrain Data

The ground surface elevation and shape can have a major impact on the type and condition of vegetation, therefore we denote the second class of input data to our model as terrain data. Our definition of terrain data includes a **DEM** obtained from the USGS, and **slope degree** and **surface aspect** layers derived from the DEM using the Slope and Aspect tools provided within ArcGIS [48,49]. DEMs are remotely sensed raster images with raw elevation metric values for each pixel, measured as the elevation of the ground surface above the mean sea-level elevation, ignoring vegetation and human-built structures on the ground. The USGS DEM has 1×1 m spatial resolution, a vertical accuracy of 0.086 m for non-vegetated vertical accuracy (NVA) at a 95% confidence level, and a vertical accuracy 0.27 m for vegetated vertical accuracy (VVA) at a 95% confidence level. Pixels in the aspect layer a represent the compass direction in degrees from north that the ground faces, so that $a \in [0, 359]$. Pixels in the slope layer s represent the degree that the ground is tilted from flat within each pixel, so that $s \in [0, 90]$.

2.2.3. LiDAR Derived Dataset Labels

While training a supervised deep machine learning model, a model must be exposed to labels defined as ground truth in order to learn correlations and a distribution over real data, ideally generalizing to produce labels for unseen instances. We used airborne LiDAR as the source of our ground truth as it is the state-of-the-art 3D remote sensing technology. LiDAR lasers record reflections for every object they contact to create a *point cloud* in 3D space, the last surface generally being the Earth's surface. We collected LiDAR data for our study site from the USGS scanned from December 2017 to April 2018 by Quantum Spatial, Inc. using a VQ1560i laser scanning system from Riegl of Horn, Austria [50,51], scanned at approximately the same time as the NAIP imagery. The collected LiDAR data has the horizontal projection of Contiguous USA Albers, meters, using the NAD 1983(2011) datum, and the vertical datum of NAVD88 GEOID 12B, meters, and a nominal pulse spacing (NPS) of 0.71 m. Using the *first-pulse* of the LiDAR, we generated a digital surface model (DSM) representing the elevation above mean sea-level of the first object that the lasers contacted on their path to the ground. We calculated the DSM using the USGS LiDAR, which was classified by the American Society for Photogrammetry and Remote Sensing (ASPRS) standard LiDAR point classification. To generate the DSM, we used a natural neighbor interpolation that is less susceptible to small changes in the triangulation of Voronoi neighbors using ArcGIS's LAS to Raster tool [52,53]. This follows Sibson's [54] area-stealing interpolation and avoids peaks, pits, ridges, valleys, passes, and pales that have not already been represented by the input samples [55]. We performed pixel-wise

subtraction between the DSM and DEM and generate a layer where each 1×1 m pixel value represents the height of the object within each pixel.

In theory, this should result in a perfect representation of vegetation height, although we identify a high amount of noise in the 2018 LiDAR dataset with unrealistic heights, seen in box *b* of Figure 3. In response, we deployed a data processing algorithm to eliminate LiDAR noise and better represent the true landscape. Our process first identifies pixels whose heights appear to be anomalies, usually appearing as small groups or single pixels with heights that differ by orders of magnitude from their neighbors. We transformed the pixels into points with coordinates (i, j, z), where i and j are the row and column of the pixels in the study site, and z is the height in meters. We run a DBSCAN clustering algorithm [56], and points labeled as "noise" represent potentially anomalous pixels. DBSCAN has two input arguments, $eps \in \mathbb{R}^+$ and $min\_samples \in \mathbb{N}$. In the point representation of our dataset, the algorithm labels a point as noise if it does not contain at least *min_samples* points within distance *eps*. This proves to be a good identifier of anomalies, as these points will be isolated in 3D space (immediate x, y neighborhood has extremely different height values), while most other points will have most neighbors close in 3D space (immediate x, y neighborhood has similar height values). As we are concerned with identifying anomalies in our dataset, we refer to [56] for a further understanding of how DBSCAN defines entire clusters. After identifying the anomalous points and their corresponding pixels, the median height of the pixel's 3×3 neighborhood was assigned as the new pixel height value.

An example of the performance of this algorithm is shown in Figure 3, where we filtered out noise in the layer representing the heights of objects, and further assign values as the median of neighborhoods over the spatial layer. DBSCAN effectively removes random potential anomalies, but preserves the power lines from the LiDAR data due to those clusters of pixels being similar in height. Known techniques like the Hough transform have been shown to remove linear shaped data from LiDAR [57], though we leave this further processing to future work and discuss the implications of not removing power lines in Section 4.

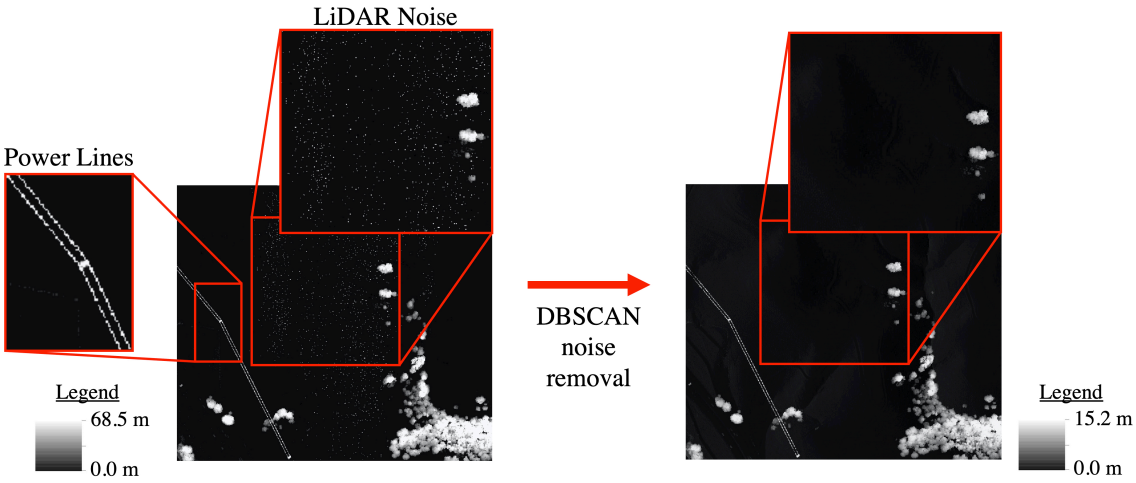

Figure 3. We analyze the raw normalized-digital surface model (DSM) (nDSM) of an area on the left with power lines and sensor noise. The nDSM on the right is the result after the DBSCAN algorithm removes the sensor noise, however does not remove power lines which we validated visually.

We refer to this final layer as the **normalized-DSM (nDSM)**, and used it as the labeled data for supervised training of our model, as well as the ground truth height in our evaluation to measure Y-NET's performance. Discussed next, the labels were used to update the weight parameters of our neural network during the training phase by the process of *backpropogation* by calculating the difference between Y-NET's predictions and the corresponding label at each iteration of training.

2.3. Deep Learning Background

The field of machine learning is built on the fundamental idea of learning directly from experience and data itself [58]. The hierarchy of how a computer identifies low-level visual representations (i.e., identifying edges first, then corners, then object parts in images) combine for a rich and complex understanding of an image; the culmination of which forms the definition of *deep learning* [59]. In essence, training a deep learning model is the process of approximating a theoretical true underlying distribution of observed data, P_{data}. Although a perfect representation of P_{data} is theoretically impossible, we can optimize the parameters of an approximating function P_{model} given observed data samples. For example, in least-squares linear regression we find the optimal parameters (slope, intercept) to fit a line (P_{model}) to observed samples, best approximating the true underlying distribution (P_{data}). Using a neural network with parameters θ as P_{model}, we approximate P_{data} by observing data (i.e., images, text, and numerical vectors) to learn important features and fit θ to hopefully generalize well to unseen data from the same distribution.

The process of *training* a neural network model can be defined as an optimization problem which minimizes a defined loss function (A loss function is sometimes also called a cost function or objective function), visualized as a landscape which represents how good θ of P_{model} is at approximating P_{data}. The global lowest point of the loss function is equivalent to finding the best possible θ to estimate P_{data}. Minimizing this function is often done through the algorithm of *gradient descent* [60], which continuously makes small changes to θ to decrease future error by the process of backpropagation [61]. Backpropagation continuously calculates the derivative of the loss function given θ, and updates θ to give more accurate estimations in the future. This is equivalent to taking a step in the downward direction of the loss function according to the calculated gradient. There are a variety of loss functions available, but our model presented in Section 2.5 minimizes the *mean squared error* (MSE) loss function, defined for i estimations as:

$$MSE = \frac{1}{m}\sum_{i}(\hat{y}_{test} - y_{test})_i^2 \tag{1}$$

where $\hat{y}_{test}$ is the set of estimations from P_{model} and y_{test} is the set of ground truth labels.

Unlike feedforward regression or classification deep learning models, autoencoders are composed of two modules: an encoder to map inputs to a compressed feature mapping h and a decoder to map h back to the original shape, shown in Figure 4. These modules are used to compress input data and preserve important representations for a defined goal, an example is image segmentation [62–64]. Autoencoders consist of any combination of fully connected layers, convolutional layers [65–68], and pooling or dropout layers [69,70]. While fully connected neural networks are used for single dimension inputs, convolutional layers use a sliding kernel to preserve relativity of the input and are typically used for images, videos, or sensor data. Dropout layers are used to ignore a subset of parameters in the network at each iteration of training, ultimately reducing the reliance between individual variables in θ. An example of using autoencoders for image segmentation is training a model ϕ given input data with n features so that $\phi : \mathbb{R}^{\alpha} \rightarrow \mathbb{R}$, where $\alpha = 3$ given a natural image (a natural image is one taken from a camera, or how humans perceive the world instead of a PDF or scanned document) (RGB), seen in Figure 4. Further details of autoencoders and representation learning are beyond the scope of this paper, and we refer the reader to [59] for a deeper understanding.

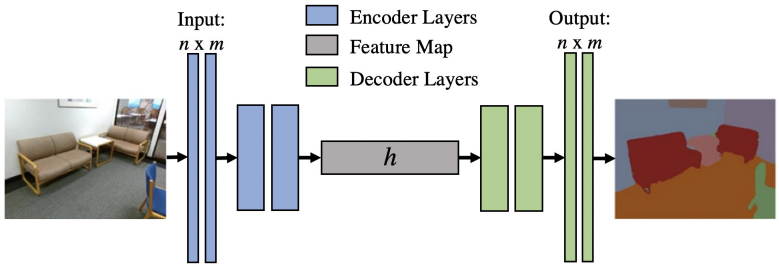

Figure 4. Visualization of an autoencoder architecture. Image segmentation used as an input–output images adapted from [71].

2.4. U-NET

Popular applications for deep learning autoencoders include autonomous driving, object detection, and medical imaging. A popular model that achieves state-of-the-art performance for tumor identification is called U-NET, named for the shape of the model's architecture [72]. U-NET is an extension of the classic convolutional network designed to learn information from fewer training examples by using both context-aware upsampling and high-resolution data in the decoder. Mentioned in Section 2.2, U-NET was also used by Microsoft AI to identify the buildings we include in our footprints mask. U-NET uses convolutional and pooling layers to encode features in h, and convolutional and upsampling layers to decode h, with a final layer using a *sigmoid* activation function. Sigmoid compresses estimations for each pixel to saturate to either 0 or 1, creating a distribution with two sets in the output image, for example a *tumor* or *background*. This can be thought of as performing a classification for every pixel in an input image.

The main difference in the U-NET architecture is the use of *skip connections*, which directly copies high-resolution data from the encoder to the decoder to be paired with context-aware upsampling data. Skip connection also provide a shortcut for gradient updates of θ to travel directly back to early layer of the network. Recent work has used U-NET for one-dimensional (1D) data anomaly detection, and created an abstraction called MU-NET for multivariate 1D sensor data [73]. MU-NET has a separate encoder for each variable and concatenates each h together before passing the data to a single decoder; however, their code is not public and they do not provide detail of the concatenation or skip connection configuration coming from multiple encoders. To highlight the power of our model's architecture, we included U-NET in our evaluation and replace the output activation function with a linear transformation to allow for height estimation greater than a value of one.

2.5. Our Model: Y-NET

2.5.1. Architecture and Training

Y-NET uses a novel architecture to generate a VSM from the multispectral aerial imagery and terrain data presented in Section 2.2. Y-NET uses an autoencoder framework, used to disentangle important features of input data and map the output to the same dimensions of the input. Whereas autoencoders are typically used for classification, we designed Y-NET to generate regression estimations pertaining to vegetation height.

Figure 5 shows the Y-NET architecture, where the grey boxes represent convolutional layers, empty boxes represent dropout layers, plus signs represent concatenation, and the blue, red, and green arrows represent max pooling layers, upsampling layers, and skip connections respectively. While convolutional layers, dropout layers, and skip connections were discussed in Section 2.3, we detail the functionality of concatenation, max pooling, and upsampling layers in this section.

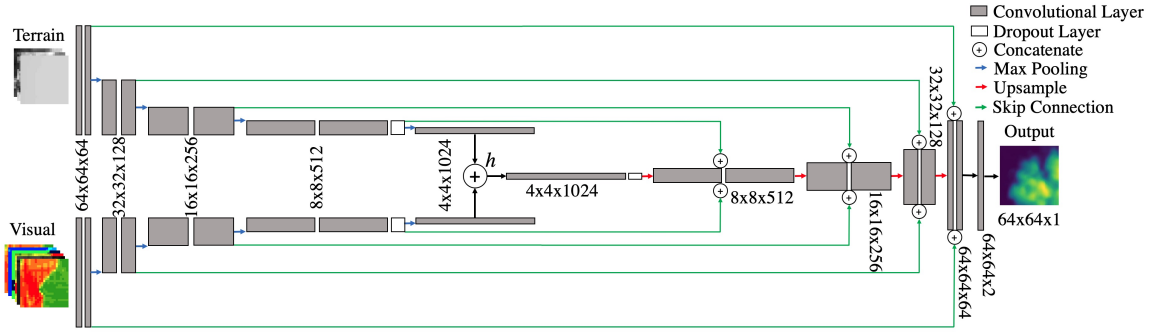

Figure 5. The Y-NET architecture receives terrain and visual data into two separate encoder branches, concatenates their latent representations, and passes through a single decoder. The output of Y-NET is a height estimation in $\mathbb{R}^1$ for every pixel in a 64×64 pixel input tile location.

We designed Y-NET with two identical input encoder branches to leverage the unique properties of the visual and terrain data separately, similar to feature grouping in other work [74]. We list the output dimension of each convolutional layer throughout the network to better understand how the data are transformed. The model receives samples of terrain data that are $64 \times 64 \times 3$ into the terrain branch, and samples of visual data that are $64 \times 64 \times 6$ into the visual branch. Each encoder passes the data through two convolutional layers at a time, followed by a max pooling layer, which decreases the height and width dimensions of the data by assigning the maximum value within a 3×3 neighborhood. The convolutional layers increase the third dimension through nonlinear transformations, shown in each stage of the model.

The latent representations from each encoder both have size $4 \times 4 \times 1024$, and are concatenated, or joined, along their third axis to create h, and move through the single decoder on the right of Figure 5. Skip connections copy information from the output of each encoding stage directly to the correlated decoder stage, also providing a direct path for gradient update information to flow to early layers of the network during training. Y-NET generates a regression output distribution of vegetation heights in $\mathbb{R}^1$ (size $n \times m \times 1$) given the input features of $\mathbb{R}^\alpha$ ($n \times m \times \alpha$), where α is an arbitrary number of input layers for each pixel; $m = n = 64$ pixels and $\alpha = 9$ in our experiments.

We divided the dataset described in Section 2.2 into disjoint sets, using 56% for training, 14% for validation, and 30% for testing. For evaluation, the output distribution of each tile in the testing dataset was compared with the corresponding nDSM, and we calculated the absolute error value between the estimated height and the ground truth value. Again, we stress the importance of understanding that the validation dataset was used to determine the duration of training, and the evaluation only uses the testing dataset. As part of our underlying research, we queried vegetation researchers and fire experts to specify ranges of vegetation heights that are relevant to vegetation modeling and wildland fire fighting. We show the defined ranges of interest and dataset sizes in pixels (1×1 m) in Table 1, and note that the ranges are also consistent with past studies uses the imperial system of measurement [30,75].

Table 1. Amount of pixels (millions) in each dataset, presented in specified height ranges of interest. The training and validation datasets were used in Y-NET's training process, while the testing dataset was used for our evaluation.

Range (m)	Train	Validate	Test
0.0–0.6	57.68	14.43	30.63
0.6–1.8	7.05	1.74	3.77
1.8–6.0	15.01	3.72	8.08
6.0–15.3	36.46	9.04	19.48
15.3–24.4	12.63	3.12	6.82
24.4+	2.93	0.75	1.64
Total	131.76	32.80	70.42

For the deep learning details of Y-NET, each convolutional layer makes use of the rectified linear unit (ReLU) activation function [76–78]. Three dropout layers with a dropout rate of 50% were included to help improve the learning performance of individual parameters throughout the network. The model was trained using the Adam optimizer with a learning rate of $10e^{-4}$ and the MSE loss function using the early-stopping method. Early-stopping uses the validation dataset to assess the performance of the model after each training epoch, stopping training when the model begins to overfit to the training dataset. Y-NET was implemented using the TensorFlow deep learning library [79].

2.5.2. Motivation and Inspiration

While Y-NET uses an autoencoder framework, the actual architecture of the model is quite unique. The inspiration behind Y-NET's architecture came from the U-NET model, which is an extension of the so-called "fully convolutional network" [80] designed to learn important information from fewer training examples. Unlike three-band images, our input layers have far more than just red, green, and blue channels. We were also able to form input layer two groups to separate inputs, which likely have little affect on the other (visual and terrain data). The kernel in convolutional layers encodes all input features together, and since we expected significant correlations between the visual and terrain input layers to be unlikely, we defined two separate encoder branches which feed into one decoder, forming a "Y" shape seen in Figure 5.

The higher resolution features from both encoders were combined with the context-aware upsampled output in the decoder through skip connections. Combining these data features together helps with feature localization in the eventual output. Incorporating the higher resolution from both encoders ensures that both visual and terrain input layers improve the localization of 3D features. Given our application domain, we expected that the culmination of dual encoders and using all skip connections would result in higher localization and feature estimation accuracy over other architectures.

2.5.3. Workflow & Implementation

The workflow of using Y-NET in practice includes minor GIS formatting of remotely sensed data for locations of interest, defined by a rectangular polygon over a landscape. Currently, our workflow includes manually collecting the data from a desired source and clipping each layer to the exact same coordinates and dimensions over each location. We also used the ArcGIS Slope and Aspect tools to extract the slope and aspect layers from the DEM, and performed unsupervised classification for the footprints layer, which must be visually validated by the user for accurate coverage, usually just for a subset of the location. The NDVI layer can be calculated using ArcGIS's Raster Calculator, or using NumPy array calculations from the red and NIR NAIP bands. The size of the location only effects the time for the preprocessing algorithms to run, though typically does not take long. After preprocessing each location of interest, the user simply includes them in a directory that Y-NET's code can access, automatically completing the remaining preprocessing steps of creating tiles and constructing the necessary datasets. When training a new Y-NET model, the user can choose to include all locations as a single input or omit an entire location. Including all locations results in a random sampling of tiles for each dataset, while specifying a location to omitting from results in that location becoming the entire testing dataset. Training the Y-NET model for our evaluation takes about 2 hours using an NVIDIA TITAN V GPU, and deploying it on the testing dataset takes only a few minutes. Once a Y-NET model is trained, it can be given any new location and generate a VSM in a few minutes, without retraining. The amount of time needed to generate a VSM is dependent on the size of the testing location, however our evaluation completed in minutes. We acknowledge that an end-to-end workflow could include automatic data preprocessing given a user-defined polygon, increasing the speed of the application workflow even more, however leave this implementation for future work.

2.6. Evaluation

As no previous work creates a VSM at 1×1 m resolution from a multispectral image and terrain layers, our evaluation includes Y-NET, Y-NET without skip connections (Y-NET-noskip), the state-of-the-art segmentation U-NET model with a linear output layer, and an autoencoder. While photogrammetry and SAR have the ability to create a VSM, their collection processes require multiple special passes with specific measurements to be accurate and still compare to LiDAR as a benchmark for accuracy. Therefore, we compared each model against LiDAR (nDSM) and evaluated for median absolute error in meters, mean absolute error in meters, root-mean-squared-error (RMSE) in meters, and R^2. Let Y represent the set of pixels in the testing dataset. R^2 is calculated by

$$R^2 = 1 - \sum_{i=0}^{Y} \frac{(y_i - f_i)^2}{(y_i - \bar{y})^2} \tag{2}$$

where y_i is the ground truth height of a pixel, $\bar{y} = \frac{1}{Y}\sum_{i=0}^{Y} y_i$, and f_i is the height estimation from Y-NET for pixel i. An R^2 value closer to 1 suggests the model fits to the ground truth landscape better, and negative values mean the model consistently predicts $y_i - f_i > y_i - \bar{y}$.

Secondly, we plotted the cumulative density function (CDF) for Y-NET and the separate height ranges from Table 1. A CDF details the percentage of estimations (y-axis) that are within a certain amount of error in meters (x-axis) from the ground truth, the nDSM in our case. For example, the CDF for an oracle with no error would show a perfectly vertical line ranging from 0.0–1.0 along the y-axis. We also conducted an extensive visual analysis to inspect the VSM from Y-NET on the testing dataset.

3. Results

3.1. Statistical Results

Table 2 shows the statistical results of every model in our evaluation, where Y-NET performs best for each metric. We find empirically that Y-NET achieves an R^2 value of 0.830, compared to 0.774 for Y-NET-noskip, and negative values for U-NET and autoencoder. U-NET and autoencoder fail to learn any important representations from the input data, despite countless efforts of tweaking their architectures. Both models make trivial predictions for every pixel in the dataset, showing that the standard autoencoder framework is unable to generalize towards estimating vegetation height. Figure 6 shows a colorized representation of the confusion matrices for Y-NET and Y-NET-noskip, normalized to improve visibility where the darkest blue corresponds with the maximum value of that row, or highest correlation between Y-NET and the nDSM. We included a 45° diagonal line to correspond with perfect estimations. Whereas Y-NET-noskip appears to under-estimate vegetation below 5 m and over-estimate the height of vegetation from 5 to 15 m tall, Y-NET's estimations are noticeably more aligned with the diagonal, and are less variable below 30 m, supporting Y-NET's higher R^2 value. We find it interesting that both models suffer from noisy estimations around $nDSM = 60$ m, which we attribute to LiDAR noise missed by the DBSCAN algorithm and power lines, discussed further in Section 4.

Table 2. Statistical performance of each model given the testing dataset (70.42 million pixels). Y-NET outperforms all other models for every metric in our evaluation. The best result for each metric is in bold.

Model	Median Error (m)	Mean Error (m)	RMSE (m)	R^2
Y-NET	**0.58**	**1.62**	**3.14**	**0.830**
Y-NET-noskip	0.82	2.00	3.62	0.774
U-NET	5.01	6.03	7.67	−0.015
Autoencoder	5.01	6.03	7.67	−0.015

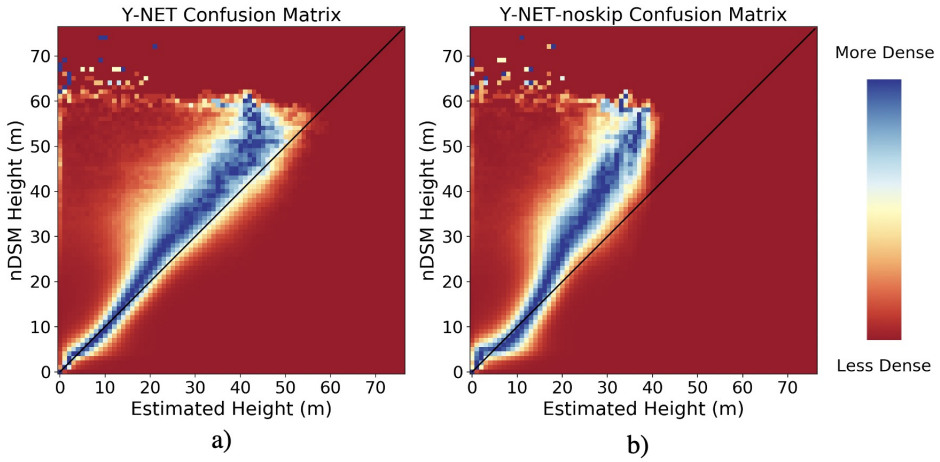

Figure 6. (**a**) Confusion matrix of Y-NET's estimations. (**b**) Confusion matrix of Y-NET-noskip's estimations.

Figures 7a,b show the CDF graphs for Y-NET on the testing dataset and the specified height ranges, respectively. Y-NET achieves a median error of just 0.07 m in the height range 0–0.6 m, and over 80% of the estimations in this range are also within just 0.24 m of the nDSM. We suspect this result is actually an overestimation due to the mandatory grassland mitigation deadline within our study site which occurs between the collection of LiDAR and NAIP [81].

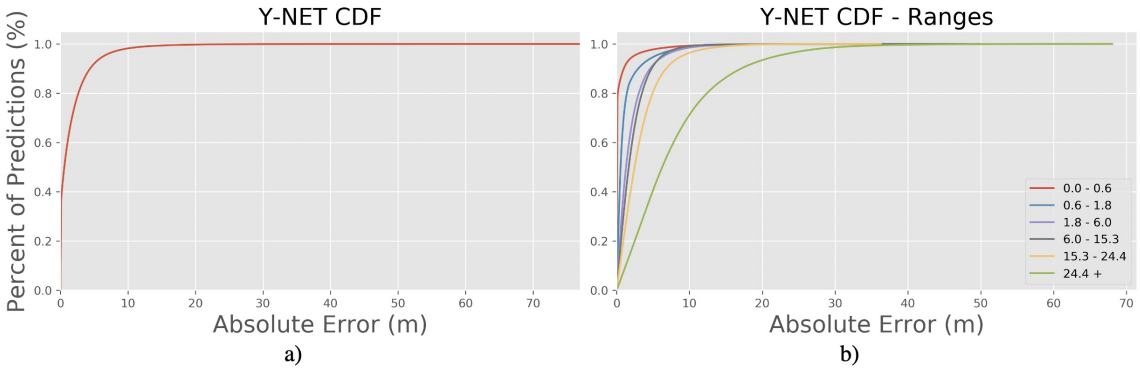

Figure 7. (**a**) Cumulative density function (CDF) of Y-NET for the test dataset. (**b**) CDF of Y-NET separated by height ranges for the test dataset.

We also highlight that the most important aspect of Figure 7b is that Y-NET effectively differentiates grasslands from other vegetation types. While Y-NET generally achieves low median estimation error, the long CDF tails are further evidence of anomalous estimation errors caused by data noise and power lines.

In summary, while U-NET and Autoencoder fail to learn effective representations, both versions of Y-NET learn associations between the input data layers and vegetation height to a certain degree. This result speaks to the importance of the data grouping architecture given *a priori* knowledge about the input features, and further to the skip connection configuration for localization and gradient flow within Y-NET. In particular, since the data layers from imagery are more similar than those of the terrain data, encoding them separately limits the amount of confusion when comparing dissimilar input features.

3.2. Visual Results

While statistical performance is important, we understand that a visual analysis of the VSM output from Y-NET can also reveal helpful notions of performance. We further emphasize that while Y-NET is trained by comparing with the nDSM for tile locations in the training dataset, Y-NET is never exposed

to the nDSM for tiles in the testing dataset, which are entirely different physical locations than in the training dataset. For example, the tiles shown in Figures 8 and 9 are examples from the testing dataset where Y-NET receives the visual data and terrain data for this new location and generates a VSM.

Figure 8 shows six example tiles from the testing dataset, with rows representing data layers and columns representing different tile locations. Row 1 is the NAIP imagery which we include for visual context as to what is located in each tile. Row 2 shows the nDSM, row 3 shows the VSM generated by Y-NET, and row 4 shows the difference between the two (rows 2–4 encoded using the color bar at the bottom of each column).

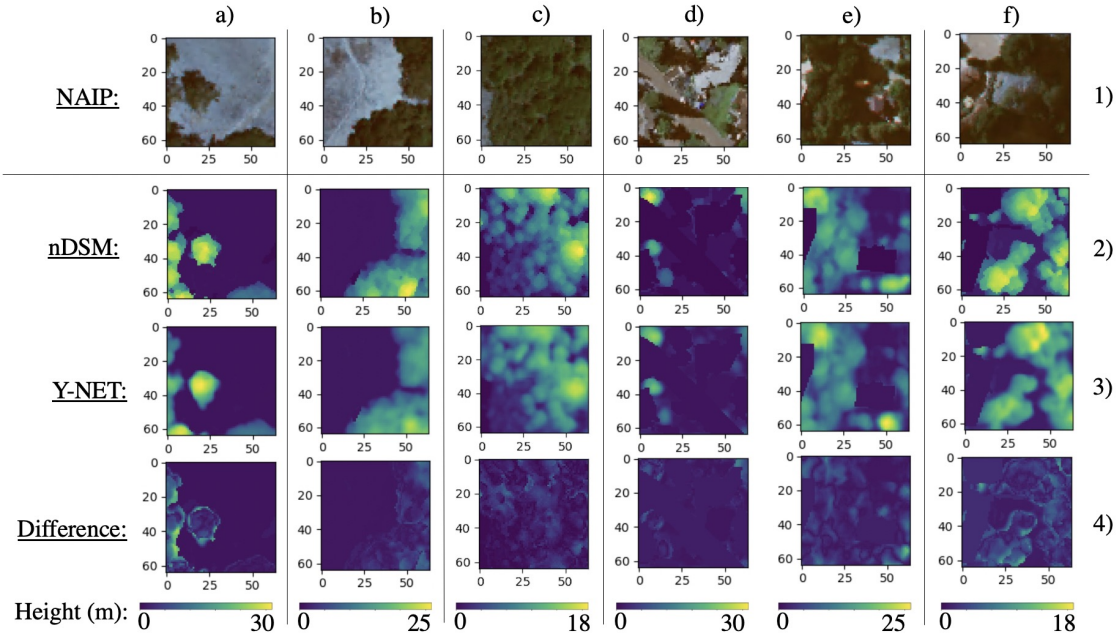

Figure 8. Visualization of Y-NET's output for tiles. Each color bar only pertains to rows 2–4 of a column. The column labels a–f are used with the row numbers to reference each image.

To relate this data to the workflow from Section 2, Y-NET receives multiple 2D layers of visual data and terrain data similar to row 1, and generates the output in row 3, where each pixel value represents a height estimation. By calculating the difference between rows 2 and 3, we can calculate the performance of the model on the testing dataset. We chose these samples from the test dataset to help visualize the variance in the testing dataset distribution: low-forested areas (*a*, *b*), dense tall forest vegetation (*c*), and the wildland–urban interface (*d*, *e*, *f*). Analyzing each tile, we can see that Y-NET can effectively model the vegetation height similar to the nDSM for each location despite a wide distribution of landscape types, vegetation density, and shadowing present in the visual data. Perhaps the most important part of Figure 8 is row 4 for tiles *a* and *f*, where we notice estimation error is often greatest around the perimeters of tree canopies. We discuss the potential cause of this being relief displacement in Section 4.

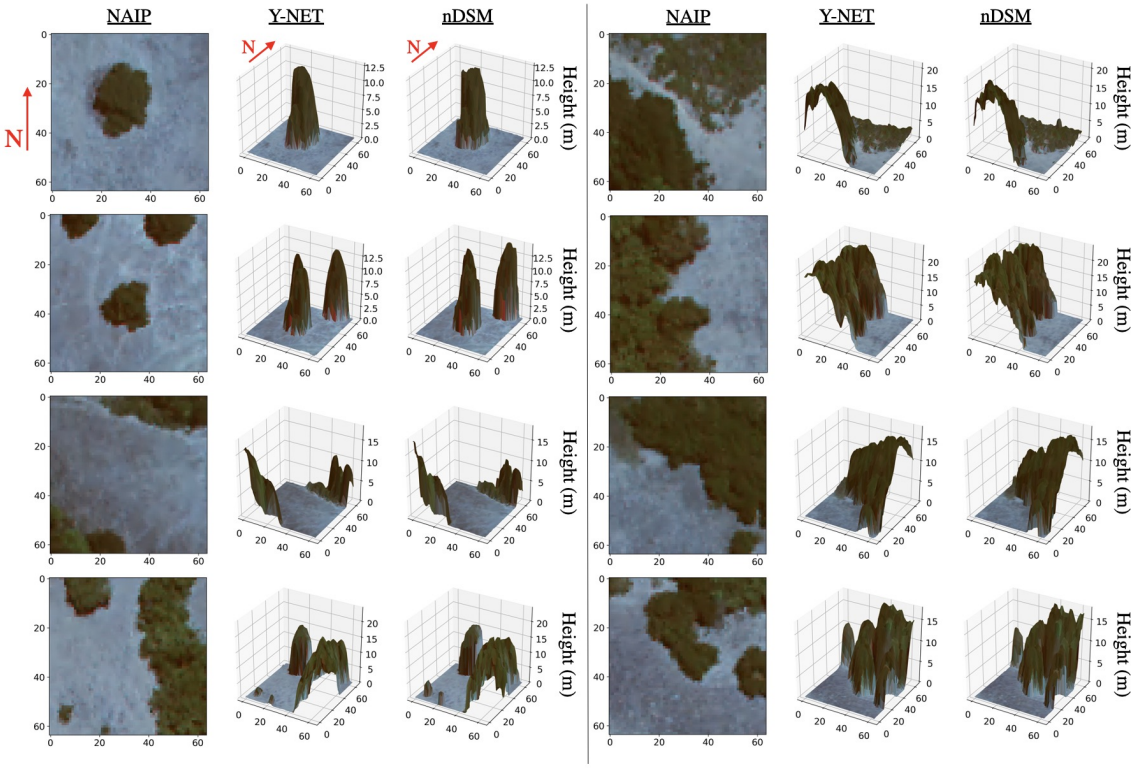

Figure 9. 3D comparisons of Y-NET's VSM and the corresponding nDSM. We include multiple examples with different vegetation densities and height to show the modeling performance of Y-NET over a broad range of examples.

Figure 9 shows more examples from the testing dataset where we project the NAIP image over the 3D VSM generated by Y-NET and the nDSM for each tile. We select these tile locations to show 3D models with varying amounts of distribution variability, similar to Figure 8. Comparing each tile location we can see that Y-NET generates an effective VSM for tiles with varying vegetation densities from the publicly available input data. This evaluation shows the high performing spatial generalizability of Y-NET, being able to train on locations *A* and generate highly accurate and representative VSMs for locations *B*. As only a fraction of the world has been measured at least once with advanced methods such as LiDAR, high performing spatial generalizability shows that Y-NET is capable of producing accurate VSMs over large areas of land which may not have been scanned with LiDAR before. Widespread availability to high resolution 3D spatial data will help support important applications such as wildland fire fighting and mitigation strategies, one motivation behind this research, among many others.

4. Discussion

Although we envision Y-NET to be capable given frequently collected multispectral imagery from satellites, we validate our model using NAIP imagery collected by aircraft because it is in the public domain. As NAIP imagery is obtained from lower elevation flights, there is expected relief displacement [82] for the canopies of trees, which is radial from the center of the image, an imperfection not as prevalent in satellite imagery [83]. With relief displacement, tree canopies are projected slightly further away from the tree center, and the displacement is increased with steeper topography. Although NAIP uses *orthoimagery* to account for this, the performance of the USGS's algorithm to correct relief displacement may cause slight misalignment between NAIP and LiDAR in the same projection. We hypothesize this may be the cause of the error seen around the circumference of tree canopies shown in Figure 8 for tiles *a* and *f*, and could be minimized when using privately collected satellite imagery.

Furthermore, it is also important to note that while the LiDAR in our study was recorded from December 2017 to May 2018, the NAIP imagery was taken in July 2018. According to the Moraga-Orinda Fire District webpage, the government-mandated vegetation mitigation deadline within our study site is 15 June [81]. This means that the grasslands are expected to be taller in the LiDAR than what is present in the NAIP image; therefore, we suspect the 80th percentile error for 0–0.6 of 0.24 m to actually be slightly exaggerated. While collecting LiDAR and imagery on the same day would allow for an ideal performance analysis, the time consumption and cost of collecting LiDAR makes this infeasible for large swaths of land.

4.1. Long CDF Tails

Perhaps more beneficial to our evaluation than observing overall performance metrics is to analyze instances where Y-NET's predictions are the most *statistically inaccurate*. As shown by the long tails of the CDFs in Figure 7, a small amount of estimations are extremely incorrect, essentially skewing our statistical results in Table 2. We emphasize *statistically inaccurate* because we observe a small set of examples in our testing dataset with high error, and find that they are likely due to sources of noise that were not removed by our DBSCAN algorithm.

We identify two main instances of noise across the 2018 LiDAR dataset for our study site shown in Figure 10: LiDAR processing noise and power lines. We use a height color map ranging from blue (short) to yellow (tall) to highlight the anomalous variations more than using the overlaid NAIP image. Seen on the left of Figure 10, we observe random irregularities of extremely high values which are not characteristic of the landscape and causes high performance error for our model.

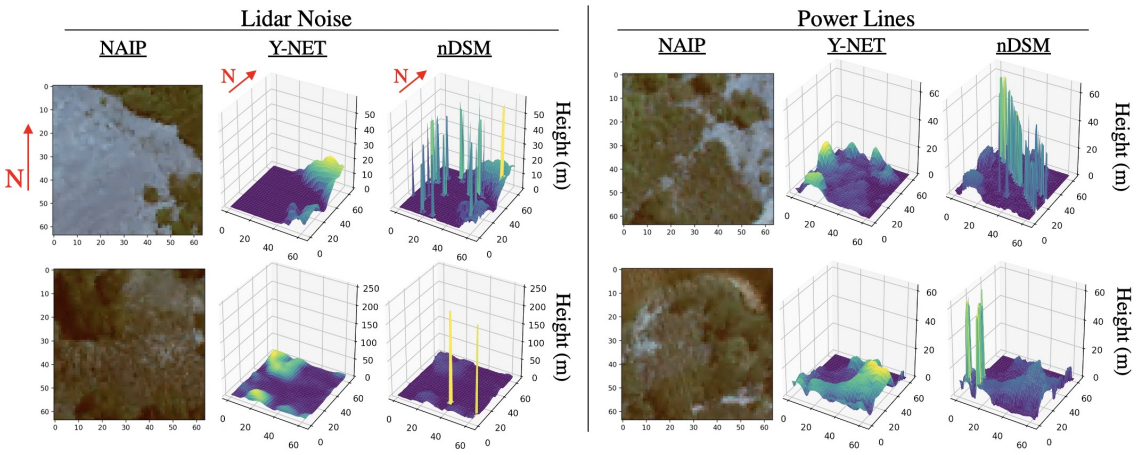

Figure 10. Examples of noise in our study site caused by LiDAR anomalies and power lines.

The second source of high estimation error we identify as power lines, shown on the right of Figure 10. As power lines are physical objects above the Earth's surface, they are recorded as being the first object the LiDAR laser contacted, and have enough neighboring pixels around the same height to avoid being identified as noise. Furthermore, power lines are too small in the NAIP imagery to be removed by our footprints masking layer, causing the nDSM height for these pixels to be extremely high. The calculated error for Y-NET in instances with power lines is enormous, despite the Y-NET modeling the vegetation surface below, shown in Figure 10.

4.2. Vegetation Mitigation and Growth

While our evaluation shows the ability for Y-NET to generalize to spatially disjoint locations, we acknowledge Y-NET could be used in practice to identify temporal changes of vegetation, such as mitigation efforts to reduce fire risk and vegetation growth. For example, temporal generalizability involves training on a landscape at one time A_t, and generating a VSM for the same landscape given

future 1×1 m resolution multispectral images A_{t+i}, where i is a difference in time. In order to properly evaluate temporal generalizability, LiDAR and multispectral imagery must be available for the same location at two different times. As this data does not exist for our study site, we are unable to statistically evaluate the efficacy of Y-NET for temporal generalizability.

As a result, we perform a secondary evaluation of Y-NET with 2016 NAIP imagery and visually compare the generated VSM with LiDAR (nDSM) from 2007 to draw interesting temporal conclusions. The 2016 NAIP data also shows Y-NET can generalize to images collected with different environmental and atmospheric conditions present during different imaging missions. Figure 11 contains three tile locations from this evaluation, including the red band of NAIP as it enhances tree canopies. Columns *a* and *b* show two instances of mitigation highlighted by red boxes, where trees had been removed between 2007 and 2016. We clearly see large differences in row 4 between Y-NET and the 2007 nDSM, however the red band in row 1 shows no tree present in that location in 2016.

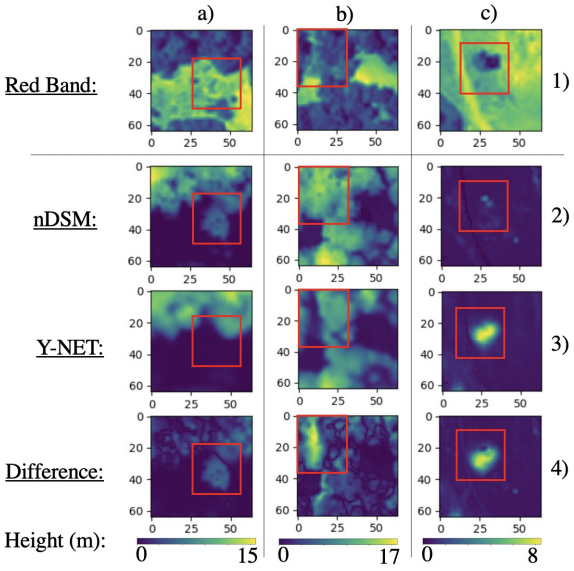

Figure 11. Examples of Y-NET with 2006-2007 LiDAR and 2016 NAIP imagery. The column labels (**a**–**c**) are used with the row numbers to reference each image.

Column *c* shows an instance of vegetation growth, where row 1 shows a mature tree that casts a significant shadow, however the 2007 nDSM shows vegetation with a height of only about 3 m. We suspect the tree grew significantly from 2007 to 2016, and Y-NET identifies the tree with a height of 8 m in 2016, visually more representative of the vegetation in the image.

4.3. Impact

We expect Y-NET to have a significant impact on how 3D data is obtained and to support extensive applications which benefit from highly recurrent and accurate 3D data. While we evaluated the spatial generalizability of Y-NET, we also acknowledge that Y-NET can be used for temporal generalizability and visually validate this, although the data for a proper evaluation are not currently available. While LiDAR data are sparse in time, we have shown that once a Y-NET model is trained, it can generate an accurate VSM given spatially disjoint data sources and identify mitigation and growth given different imagery despite new shadowing and environmental conditions.

While image-based methods such as photogrammetry allow for good 3D models to be generated more easily than LiDAR, they still require multiple photos with precise angular specifications, leading to higher collection costs when compared to Y-NET's input data. Furthermore, in the event of a natural disaster, such as a wildland fire, photogrammetry would only be effective if the region were mapped in hindsight due to smoke occlusions at the time of a fire, wheres aerial imagery that Y-NET uses is constantly collected over large swaths of land. Furthermore, Y-NET allows for VSM recurrence

at the timescale of multispectral imagery collection, allowing for spatially widespread 3D modeling of vegetation with only increasing recurrence as more satellites are deployed. Applications that would directly benefit from constantly updated VSMs include wildfire mitigation and modeling, biomass and ecosystem health, and forest economic yield forecasts. While providing higher frequency of measurements with lower cost than existing methods, we expect new previously unfathomable applications to emerge as well.

4.4. Input Feature Ablation Study

Figure 12 details the results of an input feature ablation study with Y-NET, where we trained and evaluated Y-NET while removing a single input feature each time. Doing so reveals the contribution of each input layer to Y-NET's performance. We label each bar where "!Red" represents NAIP's red band being removed from the input. All values recorded are the difference from the performance of Y-NET with all layers included, therefore downward facing bars mean the model yielded less error for that height range, suggesting those layers hinder performance when included and less important to the output.

We note the red imagery band and slope terrain layers are most important when determining the height of short vegetation under 1.8 m, though adversely affect Y-NET for tall vegetation. The NIR imagery band is most important for medium height vegetation, while the DEM, aspect, green, NIR, and NDVI are all significant for the tallest vegetation. With regards to overall statistical metrics not shown in Figure 12, the NIR band causes the highest increase in mean error, RMSE, and also decreased R^2 value the most. Parallel with Figure 12, the red imagery and slope terrain layers have the greatest impact on the median error, since this value is consistently below 1.8 m.

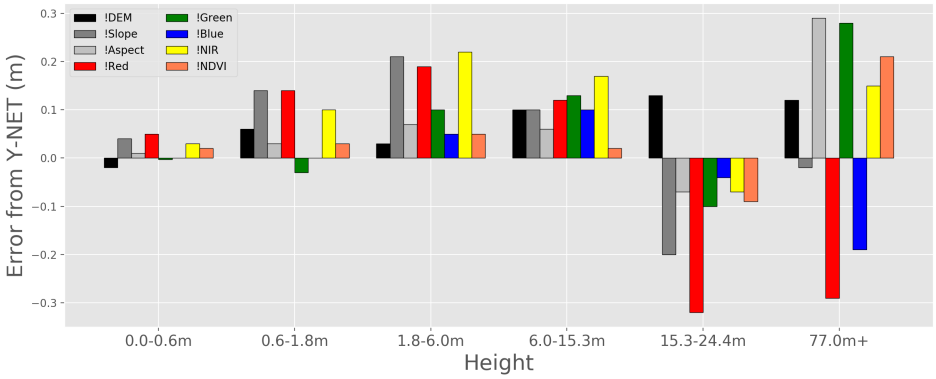

Figure 12. Performance of Y-NET when input layers are removed compared to all included. Absolute error in meters.

While removing layers from Y-NET occasionally leads to better outcomes for specific height ranges, removing an input feature never increases Y-NET's performance across all heights or evaluation metrics. Thus, we conclude that all input layers are beneficial for generating accurate VSMs, and the original Y-NET model with all layers is superior.

4.5. Future Work

While Y-NET is an important first step, we believe there is an abundance of future work with highly recurrent VSMs and in the intersection of artificial intelligence and remote sensing. Although Y-NET achieves good performance at high resolution, wildfire modeling software at high resolution has been lacking [17], causing a spatial disconnect with actual fire fighting efforts. Perhaps existing fire modeling software such as Farsite [26] or FireCast [25] could be re-purposed to model fires at 1×1 m resolution and leverage Y-NET's VSM for improved accuracy where recent LiDAR is not available.

In this paper, we have only discussed supervised machine learning, although we believe WUI fire modeling and mitigation strategies would also benefit from reinforcement learning given an up-to-date VSM from Y-NET. Reinforcement learning and remote sensing is an intersection that is almost entirely unexplored, and we envision an abundance of interesting applications emerging from the generalizability of Y-NET with reinforcement learning algorithms. We refer the reader to [84] for more information on reinforcement learning. With data driven approaches emerging as the dominant force of research and development, we hope a proper evaluation of Y-NET's temporal generalizability will be possible with future data to use as ground truth.

5. Conclusions

We propose Y-NET, a novel deep learning model to generate a high resolution VSM from readily available visual data and terrain data. The motivation behind our work stems from cost, complexity, and periodicity limitations of LiDAR for widespread remote sensing, and the need for current and future modeling software to transition towards higher resolution. Among many domains, wildland fire fighting and modeling is one that would directly benefit from up-to-date high resolution vegetation data, especially since Y-NET can generate a VSM given aerial imagery just before the time of a fire. We also expect the VSMs Y-NET generates to unlock a class of unprecedented applications which rely on up-to-date vegetation modeling.

Our method improves on past attempts to use deep learning for remote sensing by moving to 1×1 m resolution and achieving a high R^2 value with low error from LiDAR. We empirically show that grouping similar input features into separate encoder branches and including skip connections in Y-NET drastically improves estimation performance. By visually evaluating Y-NET, we validate the models robustness given tiles with varying heights and densities of vegetation, ranging from single trees, to forests and grasslands, to the WUI. Furthermore, we identify instances of noise in existing LiDAR, deploy a DBSCAN clustering algorithm to mitigate that noise, and show that Y-NET can effectively model vegetation height and identify instances of vegetation growth and mitigation. Finally, we assert that Y-NET is a first step to using deep learning for VSMs. The abundance of available spatial data is attractive for data-driven techniques such as neural networks, and we argue the intersection between deep learning and GIS is currently underutilized.

Author Contributions: D.R. (David Radke): conceptualization, methodology, software, evaluation, writing—review and editing, resources. D.R. (Daniel Radke): data curation, methodology, writing—review and editing, resources. J.R.: conceptualization, data curation, writing—review and editing, resources. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially funded by the Beatrix Farrand Research Fund, University of California, Berkeley, USA.

Acknowledgments: We would like to thank Meredith and Susan Radke for their constant support and editing throughout the long COVID-19 summer of 2020. We also thank Greg Biging, Tim Brecht, and Kate Larson for their feedback and collaboration throughout the development process. We would like to acknowledge the editor and reviewers for their valuable and constructive comments which helped us improve our manuscript.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Franklin, S.E. *Remote Sensing for Sustainable Forest Management*; Lewis Publishers: Boca Raton, FL, USA, 2001.
2. Skole, D.; Tucker, C. Tropical Deforestation and Habitat Fragmentation in the Amazon: Satellite Data from 1978 to 1988. *Science* **1993**, *260*, 1905–1910. [CrossRef] [PubMed]
3. Russell, W.H.; McBride, J.R. Landscape scale vegetation-type conversion and fire hazard in the San Francisco bay area open spaces. *Landsc. Urban Plan.* **2003**, *64*, 201–208. [CrossRef]
4. Keeley, J.E. Fire history of the San Francisco East Bay region and implications for landscape patterns. *Int. J. Wildland Fire* **2005**, *14*, 285–296. [CrossRef]
5. Westerling, A.L.; Hidalgo, H.G.; Cayan, D.R.; Swetnam, T.W. Warming and Earlier Spring Increase Western U.S. Forest Wildfire Activity. *Science* **2006**, *313*, 940–943. [CrossRef]

6. Flores, L.; Martinez, L. Land Cover Estimation in Small Areas Using Ground Survey and Remote Sensing. *Remote Sens. Environ.* **2000**, *74*, 240–248. [CrossRef]
7. Mikita, T.; Janata, P.; Surovy, P. Forest Stand Inventory Based on Combined Aerial and Terrestrial Close-Range Photogrammetry. *Forests* **2016**, *7*, 165. [CrossRef]
8. Solberg, S.; Astrup, R.; Gobakken, T.; Næsset, E.; Weydahl, D.J. Estimating spruce and pine biomass with interferometric X-band SAR. *Remote Sens. Environ.* **2010**, *114*, 2353–2360. [CrossRef]
9. Greaves, H.E.; Vierling, L.A.; Eitel, J.U.; Boelman, N.T.; Magney, T.S.; Prager, C.M.; Griffin, K.L. Estimating aboveground biomass and leaf area of low-stature Arctic shrubs with terrestrial LiDAR. *Remote Sens. Environ.* **2015**, *164*, 26–35. [CrossRef]
10. Nikolakopoulos, K.G.; Soura, K.; Koukouvelas, I.K.; Argyropoulos, N.G. UAV vs. classical aerial photogrammetry for archaeological studies. *J. Archaeol. Sci. Rep.* **2017**, *14*, 758–773. [CrossRef]
11. Solberg, S.; Astrup, R.; Bollandsås, O.M.; Næsset, E.; Weydahl, D.J. Deriving forest monitoring variables from X-band InSAR SRTM height. *Can. J. Remote Sens.* **2010**, *36*, 68–79, [CrossRef]
12. Schuster, C.; Ali, I.; Lohmann, P.; Frick, A.; Forster, M.; Kleinschmit, B. Towards Detecting Swath Events in TerraSAR-X Time Series to Establish NATURA 2000 Grassland Habitat Swath Management as Monitoring Parameter. *Remote Sens.* **2011**, *3*, 1308–1322. [CrossRef]
13. Lopez-Sanchez, J.M.; Ballester-Berman, J.D.; Hajnsek, I. First Results of Rice Monitoring Practices in Spain by Means of Time Series of TerraSAR-X Dual-Pol Images. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2011**, *4*, 412–422. [CrossRef]
14. Perko, R.; Raggam, H.; Deutscher, J.; Gutjahr, K.; Schardt, M. Forest Assessment Using High Resolution SAR Data in X-Band. *Remote Sens.* **2011**, *3*, 792–815. [CrossRef]
15. Hyde, P.; Dubayah, R.; Walker, W.; Blair, J.; Hofton, M.; Hunsaker, C. Mapping forest structure for wildlife habitat analysis using multi-sensor (LiDAR, SAR/InSAR, ETM+, Quickbird) synergy. *Remote Sens. Environ.* **2006**, *102*, 63–73. [CrossRef]
16. Maltamo, M.; Malinen, J.; Packalén, P.; Suvanto, A.; Kangas, J. Nonparametric estimation of stem volume using airborne laser scanning, aerial photography, and stand-register data. *Can. J. Remote Sens.* **2006**, *36*, 426–436. [CrossRef]
17. Radke, J.D.; Biging, G.S.; Roberts, K.; Schmidt-Poolman, M.; Foster, H.; Roe, E.; Ju, Y.; Lindbergh, S.; Beach, T.; Maier, L.; et al. *Assessing Extreme Weather-Related Vulnerability and Identifying Resilience Options for California's Interdependent Transportation Fuel Sector*; California's Fourth Climate Change Assessment, California Energy Commission: Sacramento, CA, USA, 2018; Publication Number: CCCA4-CEC-2018-012.
18. Buckley, J. Isebrands, T.S. Practical field methods of estimating canopy cover, PAR, and LAI in Michigan Oak and pine stands. *North. J. Appl. For.* **1999**, *16*, 25–32. [CrossRef]
19. Digital Globe. Available online: https://www.digitalglobe.com (accessed on 29 September 2020).
20. Planet. Available online: https://www.planet.com (accessed on 29 September 2020).
21. United States Geological Survey. Available online: https://www.usgs.gov (accessed on 29 September 2020).
22. Stojanova, D.; Panov, P.; Gjorgjioski, V.; Kobler, A.; Džeroski, S. Estimating Vegetation Height and Canopy Cover from Remotely Sensed Data with Machine Learning. *Ecol. Inform.* **2010**, *5*, 256–266. [CrossRef]
23. Gu, C.; Clevers, J.G.P.W.; Liu, X.; Tian, X.; Li, Z.; Li, Z. Predicting Forest Height using the GOST, Landsat 7 ETM+, and Airborne LiDAR for Sloping Terrains in the Greater Khingan Mountains of China. *ISPRS* **2018**, *137*, 97–111. [CrossRef]
24. Dżeroski, S.; Kobler, A.; Gjorgjioski, V.; Panov, P. Predicting forest stand height and canopy cover from Landsat and Lidar data using data mining techniques. In Proceedings of the Poster Presentation at Second NASA Data Mining Workshop: Issues and Applications in Earth Science, Pasadena, CA, USA, 23–24 May 2006.
25. Radke, D.; Hessler, A.; Ellsworth, D. FireCast: Leveraging Deep Learning to Predict Wildfire Spread. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19, Macao, Macao, 10–16 August 2019; pp. 4575–4581; [CrossRef]
26. Finney, M.A. *FARSITE: Fire Area Simulator—Model Development and Evaluation*; US Forest Service: Missoula, MT, USA, 1998.
27. National Agriculture Imagery Program. Available online: https://www.sciencebase.gov/catalog/item/51355312e4b0e1603e4fed62 (accessed on 29 September 2020).

28. Gov. Newson Visits East Bay Hills to Highlight Wildfire Danger, Prevention Efforts. Available online: https://www.mercurynews.com/2019/04/23/gov-newsom-visits-east-bay-hills-to-highlight-wildfire-danger-prevention-efforts/ (accessed on 29 September 2020).
29. Moraga-Orinda Fire District Awarded Major State Wildfire Prevention Grant. Available online: https://www.dailydispatch.com/StateNews/CA/2019/March/20/MoragaOrinda.Fire.District.awarded.major.state.wildfire.prevention.grant.aspx (accessed on 29 September 2020).
30. McBride, J.R.; Kent, J. The failute of planning to address the urban interface and intermix fire-hazard problems in the San Francisco Bay Area. *Int. J. Wildland Fire* **2019**, *28*, 1–3. [CrossRef]
31. Luo, S.; Wang, C.; Pan, F.; Xi, X.; Li, G.; Nie, S.; Xia, S. Estimation of wetland vegetation height and leaf area index using airborne laser scanning data. *Ecol. Indic.* **2015**, *48*, 550–559. [CrossRef]
32. Lee, S.K.; Yoon, S.Y.; Won, J.S. Vegetation Height Estimate in Rice Fields Using Single Polarization TanDEM-X Science Phase Data. *Remote Sens.* **2018**, *10*, 1702. [CrossRef]
33. Li, X.; Shao, G. Object-Based Land-Cover Mapping with High Resolution Aerial Photography at a County Scale in Midwestern USA. *Am. J. Remote Sens.* **2014**, *6*, 11372–11390. [CrossRef]
34. Lassiter, A.; Darbari, M. Assessing alternative methods for unsupervised segmentation of urban vegetation in very high-resolution multispectral aerial imagery. *PLoS ONE* **2020**, *15*, e0230856. [CrossRef] [PubMed]
35. Li, W.; Fu, H.; Yu, L.; Cracknell, A. Deep Learning Based Oil Palm Tree Detection and Counting for High-Resolution Remote Sensing Images. *Remote Sens.* **2016**, *9*, 22–35. [CrossRef]
36. Radke, J. Modeling Urban/Wildland Interface Fire Hazards within a Geographic Information System. *Geogr. Inf. Sci.* **1995**, *1*, 9–21, [CrossRef]
37. Rowntree, L.B. Afforestation, Fire, and Vegetation management in the East Bay Hills of the San Francisco Bay Area. *Yearb. Assoc. Pac. Coast Geogr.* **1994**, *56*, 7–30. [CrossRef]
38. Beer, T. The interaction of wind and fire. *Bound.-Layer Meteorol* **1991**, *54*, 287–308. [CrossRef]
39. Lecina-Diaz, J.; Alvarez, A.; Retana, J. Extreme Fire Severity Patterns in Topographic, Convective and Wind-Driven Historical Wildfires of Mediterranean Pine Forests. *PLoS ONE* **2014**, *9*, e85127. [CrossRef]
40. Duff, T.J.; Keane, R.E.; Penman, T.D.; Tolhurst, K.G. Revisiting Wildland Fire Fuel Quantification Methods: The Challenge of Understanding a Dynamic, Biotic Entity. *Forests* **2017**, *8*, 351. [CrossRef]
41. Tolhurst, K. *Report on Land and Fuel Management in Victoria in relation to the Bushfires of 7th February 2009*; Royal Commission: Victoria, Australia, 2010.
42. ESRI ArcGIS Project Tool. Available online: https://desktop.arcgis.com/en/arcmap/latest/tools/data-management-toolbox/project.htm (accessed on 29 September 2020).
43. Carlson, T.N.; Ripley, D.A. On the Relation between NDVI, Fractional Vegetation Cover, and Leaf Area Index. *Remote Sens. Environ.* **1997**, *62*, 241–252. [CrossRef]
44. ArcGIS Raster Calculator. Available online: https://desktop.arcgis.com/en/arcmap/10.3/tools/spatial-analyst-toolbox/how-raster-calculator-works.htm (accessed on 29 September 2020).
45. ArcGIS ISO Cluster Unsupervised Classification. Available online: https://desktop.arcgis.com/en/arcmap/10.3/tools/spatial-analyst-toolbox/iso-cluster-unsupervised-classification.htm (accessed on 29 September 2020).
46. Norzaki, N.; Tahar, K.N. A comparative study of template matching, ISO cluster segmentation, and tree canopy segmentation for homogeneous tree counting. *Int. J. Remote Sens.* **2019**, *40*, 7477–7499, [CrossRef]
47. Microsoft U-NET Building Segmentation. Available online: https://azure.microsoft.com/en-us/blog/how-to-extract-building-footprints-from-satellite-images-using-deep-learning/ (accessed on 29 September 2020).
48. ESRI ArcGIS Slope Tool. Available online: https://desktop.arcgis.com/en/arcmap/10.3/tools/spatial-analyst-toolbox/how-slope-works.htm (accessed on 29 September 2020).
49. ESRI ArcGIS Aspect Tool. Available online: https://desktop.arcgis.com/en/arcmap/10.3/tools/spatial-analyst-toolbox/how-aspect-works.htm (accessed on 29 September 2020).
50. USGS LiDAR Point Cloud CA NoCAL Wildfire B5b 2018. Available online: https://www.sciencebase.gov/catalog/item/5e74a80fe4b01d50926c3033 (accessed on 29 September 2020).
51. USGS LiDAR Point Cloud CA NoCAL Wildfire B5b 2018 Metadata. Available online: https://www.sciencebase.gov/catalog/file/get/5e74a80fe4b01d50926c3033?f=__disk__a3%2F8e%2Fa2%2Fa38ea2b6e0756b7e532a24f68b7f1b5a07fc2a51&transform=1&allowOpen=true (accessed on 29 September 2020).
52. ESRI Creating raster DEMs and DSMs from Large Lidar Point Collections. Available online: https://desktop.arcgis.com/en/arcmap/10.3/manage-data/las-dataset/lidar-solutions-creating-raster-dems-and-dsms-from-large-lidar-point-collections.htm (accessed on 29 September 2020).

53. ESRI LAS to Raster Tool. Available online: https://desktop.arcgis.com/en/arcmap/latest/manage-data/raster-and-images/las-to-raster-function.htm (accessed on 29 September 2020).
54. Barnett, V. *Interpreting Multivariate Data*; Journal of the American Statistical Association: Alexandria, VA, USA, 1983; Volume 78, p. 735.
55. Warntz, W. A New Map of the Surface of Population Potentials for the United States, 1960. *Geogr. Rev.* **1964**, *54*, 170–184. [CrossRef]
56. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise*; International Conference on Knowledge Discovery and Data Mining: Montreal, QC, Canada, 1996; pp. 226–231.
57. Habib, A.; Morgan, M.; Kim, E.M.; Cheng, R. Linear features in photogrammetric activities. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2004**, *35*, 610–615.
58. LeCun, Y.; Gengio, Y.; Hinton, G. Deep Learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
59. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016. Available online: http://www.deeplearningbook.org (accessed on 15 April 2020).
60. Cauchy, A. *Methode generale pour la resolution de systemes d'equations simultanees*; Compte Rendu des Deances de L'academie des Sciences: Paris, France, 1847; pp. 81–221.
61. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning Representations by Back-Propagating Errors. *Nature* **1986**, *323*, 533–536. [CrossRef]
62. Lecun, Y. Modeles Connexionnistes de L'apprentissage (Connectionist Learning Models). Ph.D. Thesis, Pierre and Marie Curie University, Paris, France, 1987.
63. Bourlard, H.; Kamp, Y. Auto-association by multilayer perceptrons and singular value decomposition. *Biol. Cybern.* **1988**, *59*, 291–294. [CrossRef] [PubMed]
64. Hinton, G.; Zemel, R.S. *Autoencoders, Minimum Description Length, and Helmholtz Free Energy*; NIPS: Denver, CO, USA, 1993; pp. 3–10.
65. Lecun, Y. *Generalization and Network Design Strategies*; Technical Report CRG-TR-89-4; University of Toronto: Toronto, ON, Canada, 1989; pp. 326–345.
66. LeCun, Y.; Bengio, Y. Convolutional Networks for images, Speech, and Time-Series. *Brain Theory Neural Netw.* **1995**, *3361*, 1995.
67. Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
68. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems 25*; Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2012; pp. 1097–1105.
69. Zhou, Y.T.; Chellappa, R. Computation of optical flow using a neural network. In Proceedings of the IEEE 1988 International Conference on Neural Networks, San Diego, CA, USA, 24–27 July 1988; Volume 2, pp. 71–78.
70. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
71. Badrinarayanan, V.; Kendall, A.; Cipolla, R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 2481–2495. [CrossRef]
72. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015*; Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 234–241.
73. Wen, T.; Keyes, R. Time series anomaly detection using convolutional neural networks and transfer learning. *arXiv* **2019**, arXiv:1905.13628.
74. Yi, S.; Ju, J.; Yoon, M.K.; Choi, J. Grouped Convolutional Neural Networks for Multivariate Time Series. *arXiv* **2017**, arXiv:1703.09938.
75. Wright, C.S.; Wihnanek, R.E. *Stereo Photo Series for Quantifying Natural Fuels*; United States Department of Agriculture: Corvallis, OR, USA, 2014,
76. Jarrett, K.; Kavukcuoglu, K.; Ranzato, M.; LeCun, Y. What is the best multi-stage architecture for object recognition? In Proceedings of the 2009 IEEE 12th International Conference on Computer Vision, Kyoto, Japan, 29 September–2 October 2009; pp. 2146–2153.

77. Nair, V.; Hinton, G.E. Rectified Linear Units Improve Restricted Boltzmann Machines. In Proceedings of the International Conference on Machine Learning, Haifa, Israel, 21–24 June 2010.
78. Glorot, X.; Bordes, A.; Bengio, Y. Deep Sparse Rectifier Neural Networks. In Proceedings of the 14th International Conference on Artificial Intelligence and Statistics, Ft. Lauderdale, FL, USA, 11–13 April 2011; pp. 315–323.
79. Tensorflow. Available online: https://www.tensorflow.org (accessed on 29 September 2020).
80. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 3431–3440.
81. Moraga-Orinda Fire District Website. Available online: https://www.mofd.org/our-district/fuels-mitigation-fire-prevention/hazardous-wildfire-fuels-reduction-program (accessed on 29 September 2020).
82. Relief Displacement. Available online: https://engineering.purdue.edu/~bethel/elem3.pdf (accessed on 29 September 2020).
83. Sheng, Y.; Gong, P.; Biging, G. True Orthoimage Production for Forested Areas from Large Scale Aerial Photographs. *Am. Soc. Photogramm. Remote Sens.* **2003**, *3*, 259–266. [CrossRef]
84. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; A Bradford Book: Cambridge, MA, USA, 2018.