Towards open-source LOD2 modelling using convolutional neural networks

H. Muftah¹ · T. S. L. Rowan¹ · A. P. Butler¹

Received: 18 January 2021 / Accepted: 3 April 2021 / Published online: 13 May 2021
© The Author(s) 2021

Abstract

The aim of this paper is to classify and segment roofs using vertical aerial imagery to generate three-dimensional (3D) models. Such models can be used, for example, to evaluate the rainfall runoff from properties for rainwater harvesting and in assessing solar energy and roof insulation options. Aerial orthophotos and building footprints are used to extract individual roofs and bounding boxes, which are then fed into one neural network for classification and then another for segmentation. The approach initially implements transfer learning on a pre-trained VGG16 model. The first step achieves an accuracy of 95.39% as well as a F_1 score of 95%. The classified images are segmented using a fully convolutional network semantic segmentation model. The mask of the segmented roof planes is used to extract the coordinates of the roof edges and the nexus points using the Harris corner detector algorithm. The coordinates of the corners are then used to plot a 3D Level of Detail 2 (LOD2) representation of the building and the roof height is determined by calculating the maximum and minimum height of a Digital Surface Model LiDAR point cloud and known building height data. Subsequently the wireframe plot can be used to compute the roof area. This model achieved an accuracy of 80.2%, 96.1%, 96.0%, 85.1% and 91.1% for flat, hip, gable, cross-hip and mansard roofs, respectively.

Keywords Convolutional neural networks · Deep learning · LOD2 modelling · Building architecture reconstruction

Introduction

Population growth has led to increasing demands on water and energy. The use of roofs for rainwater harvesting and capturing solar energy provides opportunities to help meet these demands. Assessing the potential impact of these opportunities requires knowing roof types and their structure over a wide area. This can be achieved using publicly accessible aerial photographs and LiDAR data, however, to do this at scale, the process needs to be automated and inherent challenges when using such data overcome. The work reported here describes how machine learning has been used to classify and segment roofs using vertical aerial imagery to generate three-dimensional (3D) models.

A number of methods have been investigated for the solution of this problem. These can be split into four main groups: plane-fitting (Vosselman and Dijkman 2001;

Mongus et al. 2014; Dorninger and Pfeifer 2008), morphological (Zhang et al. 2003; Pingel et al. 2013), classical machine learning (Jutzi and Gross 2009; Lodha et al. 2006; Ducic et al. 2006) and more recently deep learning (Pirotti et al. 2019; Zhao et al. 2018; Castagno and Atkins 2018). In line with later methods, image classification and segmentation are applied here using deep learning. This is done to assess if modern advances in deep learning can overcome current deficiencies in open-source data that have hitherto prevented model creation of this type. Roof classification, segmentation, and geometric reconstruction techniques are all utilised.

Roof classification and reconstruction can be categorised into two distinct groups; model-based and data-driven approaches. Model-based approaches require prior knowledge of buildings to form a group of building models to which data points are fitted (Castagno and Atkins 2018). This approach outperforms data-driven approaches in cases where data points are limited, as it is robust, and the topology of the roof is always correct. This method, however, is constrained by the range of defined building models, resulting in an inability to model complex structures accurately.

✉ T. S. L. Rowan
t.rowan@imperial.ac.uk

¹ Civil and Environmental Engineering Department, Imperial College London, London, UK

In the data-driven approach, points are allocated to planar surfaces to construct 3D building models, whilst the roof is constructed using roof surfaces derived from segmentation algorithms. A third option, comprising a combination of the data-driven and model-based approaches, is used to exploit the strengths of each method (Alidoost and Arefi 2016). In this paper, a model-based approach is used to classify roof types, and a fusion of model and data-driven approaches are used to construct a 3D LOD2 model of buildings.

This multistep approach is not without precedent, Castagno and Atkins (2018) incorporated classical machine learning and deep learning to develop a 2-stage classification of roof shapes using a fusion of LiDAR data and satellite images. In the first stage, a convolution neural network (CNN) is used to extract a reduced feature set, which is used as an input to a classical machine-learning support vector machine (SVM) and a random forest classifier. Transfer learning was carried out using a pre-trained CNN model, this was done due to the lack of training data and was used in combination with image augmentation. Similar work has also been carried out in Partovi et al. (2017) also used transfer learning in two different methodologies, the first fine-tunes a pre-trained VGGNet model (Simonyan and Zisserman 2014) on the ImageNet dataset in the Caffe framework. The second links the results of deep features extracted from the final fully connected layer output of three pre-trained large CNNs into a new feature vector.

Once classified, roof reconstruction requires a well-structured approach, as data quality, blocking features and camera angles can all cause problems in geometry mapping. As seen in Pirotti et al. (2019) applied a Convolutional Neural Network (CNN) using TensorFlow to segment roof and facade of 3D buildings. Descriptors were extracted using a convolutional-type strategy based on nearest neighbour point clouds and geometric features. Zhao, Pang and Wang (2018) put forward a multi-scale CNN, composed of a multiple k single-scale CNN. For each LiDAR point, three contextual images at three scales of the following attributes were made: height, intensity, and roughness.

Vosselman and Dijkman (2001) employed two different strategies for the 3D reconstruction of buildings using LiDAR data through the extension of the Hough transform. The first strategy detects intersection and height jump lines to refine initial ground plan partitioning. The second strategy fits all detected planar surfaces to five roof models, the initial models are later refined by analysing the remaining cloud points. Jochem et al. (2009) extracted building points for building reconstruction directly from a 3D point cloud by separating object and terrain points. Points were classified according to surface roughness and similar normal vectors. Finally, segmentation is employed through seed point selection and region growing. Matikainen et al. (2003) applied region-based segmentation to the Digital Surface Model

(DSM) which used bottom-up region merging and a local optimisation process to restrict the growth of a defined heterogeneity criterion.

Plane-fitting employs traditional segmentation methods such as edge detection, thresholding and region growing. However, these methods are less efficient than deep learning at image segmentation as they necessitate human intervention and employ inflexible algorithms. Similarly, classical machine-learning algorithms encounter computational difficulties when working with a high-dimensional dataset, conversely, deep learning is able to process high-dimensional data such as RGB (Castagno and Atkins 2018). Plane-fitting methods solely employ a data-driven approach, in this paper, roof segmentation applies semantic segmentation, a model-driven approach to guarantee correct roof typology. Although data-driven approaches excel over model-driven approaches in the reconstruction of complex roofs, complex roofs were not encountered often in this research.

The challenge presented in this work is the use of open-source data and whether this deep-learning approach can provide useful and accurate models. The open-source nature of the data can cause issues in terms of quality as discussed in this paper. Though, approaches and techniques in deep learning are constantly evolving and should overcome this issue. This paper sets out the first steps of a fully automated modelling process. The four stages of model creation are discussed:

- Data cropping, across the various three datasets (Aerial Imagery, LiDAR, and Map).
- Classification using a CNN
- Segmentation using a fully convolutional network (FCN) semantic segmentation model.
- Reconstruction of the roof using the Harris corner detector algorithm, LiDAR based, and height weighting.

This paper is structured such that introduction and motivation is given in the first section. The methodology used is given in the second section. In the third section, the classification CNN development is detailed; in the fourth section, the segmentation FCN evolutions are presented. Results are discussed in the fifth section and conclusions are given in the sixth section.

Methodology

In this work, multiple CNNs are used to classify and then segment data and other algorithms are used to reconstruct the geometry. This workflow is summarised in Fig. 1. Crucially—and novel we utilise all the data that are available—not in training but in data preparation for the deep-learning steps, i.e. the individual buildings are cropped from aerial

Fig. 1 A workflow of the building geometry reconstruction method implemented. Black pathway for teaching, and red for usage

images using maps and recorded building heights to ascertain cutoffs for data heights. The method is further outlined in this section, with data collection, CNN training and other algorithms used described below.

Model evaluation criteria

To evaluate the quantitative performance of the image classification model, the training, validation, and test

accuracy as well as the loss are calculated. Accuracy is calculated based on True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN).

For example, if an object is positive and is identified as positive, it is a TP; if it is identified as negative, it is a FN. If the object is negative and is identified as negative, it is a TN, and if it is identified as positive, it is FP (Fawcett 2006). Accuracy is defined as follows:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \quad (1)$$

Accuracy, however, is not the best metric to use where there is a class imbalance, as a high classification accuracy may be achieved when some classes appear more often than others. A variety of metrics will be used in this work. Additional metrics are necessary, therefore, to evaluate the model's performance. Consequently, precision and recall metrics are also used to quantify the information retrieval accuracy of the model. These are

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad (2)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \quad (3)$$

The F_1 score calculates the harmonic mean of the precision and recall (Fawcett 2006):

$$F_1 \text{ score} = 2 \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \quad (4)$$

The loss is calculated using the categorical cross-entropy loss function and is the objective function optimised in the model. It is the difference between the predicted and true distribution, and is used in optimising the network. Loss ranges from 0 to infinity and a lower loss indicates lower error in the predictions. Whilst accuracy is discrete and is easy to interpret, loss is a continuous

variable and gives a better representation of the model's performance (Rusiecki 2019). Loss is defined as follows:

$$E = -\frac{1}{N} \sum_{j=0}^N \sum_{i=0}^M (y_{ij} \log(\hat{y}_{ij})), \quad (5)$$

where N is the number of training data, M is the number of classes, y is the observed values of the predicted variables and $\hat{y}$ is the predicted values.

In semantic segmentation, the loss is also calculated using categorical cross-entropy; however, another measure of accuracy is the Dice coefficient (Dice 1945). The Dice coefficient is calculated by multiplying the pixels that overlap in the ground truth mask and the predicted mask by two and dividing it by the sum of the pixels in the ground truth and the predicted mask:

$$\text{Dice} = \frac{2|y \cap \hat{y}|}{|y| + |\hat{y}|}. \quad (6)$$

Data collection—classification

Aerial images were collected from EDINA Digimap Service, a map and data delivery service. Vertical 1×1 km aerial orthophotos with a 25 cm resolution (EDINA Aerial Digimap Service 2018) and Ordnance Survey MasterMap® Topography Layer tiles were downloaded (EDINA Digimap Ordnance Survey Service 2017). The MasterMap® provides the building footprint, a bounding box is generated using minboundrect MATLAB (D'Errico 2021) and applied to aerial images to separate individual buildings, Fig. 2. A buffer of 10 pixels is applied to the bounding box to capture the complete roof. The processed images of the roofs are manually sorted into five classes representative of common roof shapes in London. The roof shapes include; flat, hip, cross-hip, gable, and mansard. The classified images are used in both image classification and semantic segmentation discussed in Sects. 2.4 and 2.6. The data in this study were

Fig. 2 A workflow of the building geometry reconstruction method implemented

primarily gathered from the following boroughs in London: Brent, City of Westminster, Ealing, Hammersmith and Fulham, Harrow, Hillingdon, and Kensington and Chelsea. Inner and outer boroughs were chosen to obtain a representative sample of roof types. Based on the data collected from this study, the most popular roof type in London is gable making up 42.08% (7135) followed by hip 30.78% (5219), flat 24.29% (4119), cross-hip 1.93% (328), and finally mansard 0.92% (156), as shown in Fig. 2. Examples of each roof type are given in Fig. 3. 16,958 images were collected over an area of 15 km².

The quality of the images is compromised as a result of some blurring produced during image capture. Furthermore, some of the images are not adjusted correctly to the topography and camera tilt when transforming from aerial images to orthophotos. It is important, therefore, to find a balance between clear vertical images that contain minimal occlusion and variation from trees, shadows and other obstructions, resulting in a higher model accuracy, and images that contain these variations to allow for a more generalised model. The performance of a CNN model rapidly deteriorates with lower image resolution. Chevalier et al. (2015), for example, found that performance dropped by 20% when the resolution was decreased from 100×100 to 20×20 pixels. Due to the low quality of open-source LiDAR DSM (EDINA LIDAR Digimap Service 2016a, b), which has a resolution of 50 cm per pixel, it was not selected for classification and segmentation in this study.

One challenge this dataset poses is the lack of uniformity in orientation and image size. For the purposes of this work, we define orientation as the angle between true north and the major (longer) axis of the building. Orientation is resolved in a two-part approach: first, during image convolution,

rotation and inversion of images enable classification and segmentation to be learnt at any angle. Second, the algorithms for reconstruction are written to handle buildings of any orientation. To test the impact of image resolution, the dataset was split into two sets comprising of resolutions greater and less than 64×64, as shown in Fig. 4. The training set sizes were equalised to establish a fair test.

The smaller resolution training set achieved faster convergence than the larger resolution images, which exhibit no learning and an increase in loss. This can be put down to two factors, the first is down sampling carried out by inter-area interpolation when downscaling image size, resulting in loss of information. Roofs with a resolution of 64×64 have an area of at least 256 m², assuming the roof makes up the entire image. This implies that these buildings are more likely to be institutional, industrial, or commercial. Roof types belonging to large non-residential buildings are typically more complex and can consist of multiple roof classes, consequently, it is more difficult to classify them. The negative impact of downsizing remains minimal by setting the input size to 64×64 as only 162 images had a resolution greater than 64×64 compared to 16,792 images with a lower resolution.

The images are pre-processed with the same pre-processing method used when training VGG16. The data are zero-centred by subtracting the mean RGB value from each pixel. The average is made equal to zero, which prevents distortions arising from differences in means (Patro and Sahu 2015). This is important to maintain data within an appropriate range to control backpropagation gradients. Feature scaling normalisation is also used to scale values between 0 and 1, this is beneficial in classification problems using neural network backpropagation as it accelerates the learning

Fig. 3 Roof types prepared for segmentation where the bottom row shows the ground truth segmentation mask for: **a** flat, **b** hip, **c** gable, **d** cross-hip, and **e** mansard

phase and has been shown to be useful for generalisation (Krizhevsky et al. 2012). To achieve this, the image RGB values are divided by 255 (Al Shalabi et al. 2006). Image augmentation is performed directly on the original dataset and is stored on the CPU. The dataset is augmented randomly using Keras ImageDataGenerator with the following augmentations; shifting the width and height by 10%, a zoom range of 0.2, a shear range of 0.1, rotation by 10 degrees and flipping horizontally and vertically. The image dataset is split into training, testing and validation sets where 20% of the training images are first allocated to testing and from the remaining training images 20% are allocated to the validation set.

Image augmentation is performed directly on the original dataset and is stored on the CPU. The dataset is augmented randomly using Keras ImageDataGenerator with the following augmentations; shifting the width and height by 10%, a zoom range of 0.2, a shear range of 0.1, rotation by 10 degrees and flipping horizontally and vertically.

Data collection—segmentation

As shown in the previous subsection, care must be taken when collecting data for deep learning. As high accuracy is required for the reconstruction of building geometry, the images for segmentation are resized to 256×256 pixels. This is because setting the resolution too low leads to a coarser segmentation. No form of pre-processing is applied to the images. The images and masks are augmented by multiplying all pixels within the image by a value sampled from 0.9 and 1.1 to brighten or darken the image. To expand the training set, half of the images are inverted about the horizontal axis and inverted about the vertical axis. In addition, they are rotated randomly between 0° and 90° and scaled between 90 and 110%. This approach helps to combat the various orientation issues that can occur with roof data analysis. In addition, as scaled and rotated images do not require

re-segmentation by hand it does not incur extra work. A Gaussian blur is applied to half the images with a $\sigma_x = 0$ and $\sigma_y = 8$, a low-pass filter to attenuate noise (Van Vliet et al. 1998). Image augmentation in this case was found to reduce loss by 59.66% and improved accuracy by 2.04%.

Shuffling

Due to the non-convex nature of the loss function where it exhibits numerous local minima, the gradient descent algorithm, an optimisation algorithm, is used to minimise the objective function by moving iteratively in the direction of steepest descent (Ruder 2016). Local minima rather than the global minima are typically found in static data where the local minima locations remain unchanged with each iteration. Through shuffling, the network is trained with unfamiliar samples leading to faster learning. This method can only be applied to stochastic learning as the order of the data is insignificant (Bengio 2012). This was applied to the image classification and the semantic segmentation models.

Harris detection and geometry reconstruction

The mask image is pre-processed by converting it to grey-scale, and this image is used to calculate the Harris response, as summarised in Fig. 5, and is outputted in an image form equal to the size of the mask. An image patch size of 5 is chosen with a Sobel kernel size of 3 and a k of 0.04. These values were chosen based on values used in the literature. Thresholding is applied next, where the threshold is equal to $0.1 \max(R)$. If the pixel value is greater than the threshold, it is assigned a maximum value of 255, otherwise it is assigned a value of 0. The corner coordinates are refined using the centroid locations, this is important for corners detected multiple times. An early stopping criterion is set with a maximum number of 100 iterations and if the centroid has moved at least 0.001 pixels. The colours of the segments

Fig. 4 Accuracy and loss results for varying sized image groups

were changed as the corners of the green segment were not detected as well. This method has proved to be more successful than the Hough transform, which sometimes predicts overlapping lines within the same edge or the edge lines are disconnected if the edge of the roof was not predicted perfectly in the mask.

Classification CNN training

The crux of this work lies in the optimisation of the training to determine whether a CNN network can overcome the relatively poor data size and resolution of the dataset presented. Training was carried out using an i7-7700k CPU, a GTX 1070 Ti GPU and 16 GB RAM. Here, the transfer-learning VGG16 model was chosen with ImageNet weights, the classification layer is removed, as it is unique to the classification task. ImageNet is an annotated image database with 1.2 million images categorised into 1000 classes. The model is built using the Python library Keras, a TensorFlow high-level application programming interface. Tensor-flow is ‘an end-to-end open-source platform for machine learning’, and probably the most widely used today. Keras was designed to dramatically speed up the development of deep-learning models, it is minimalistic but expandable and written in python, which remove a lot of the barriers to fast deep-learning deployment. A model head is attached to the network, consisting of a flattened, dense, dropout and another

dense layer. The model is trained over 1000 epochs using an early stopping and model checkpoint call back to halt training when accuracy and loss stop improving, all trainable convolutional block layers are frozen. The model head weights are randomly initialised, this is necessary to warm up the model head by optimising the weights to the specific training dataset. Training with backpropagation trains in a layer by layer manner, training the first convolutional layers in the first iterations and higher layers converge with time. The trained weights are saved and loaded into a fine-tuning model where the network’s architecture is replicated, the five convolutional blocks are unfrozen to allow for errors to backpropagate through the network. Fine-tuning the model immediately without initialisation of weights will lead to overfitting with a small dataset, transfer learning also improves the generalisation performance of the network (Yosinski et al. 2014). Keskar and Socher (2017) suggest a cross-training strategy by switching from Adam to stochastic gradient descent (SGD) to narrow the generalisation gap. In this work, Adam is used during transfer learning before switching to SGD during fine-tuning.

Learning rate

The learning rate is a hyperparameter in CNNs and requires fine-tuning to achieve the fastest convergence and highest accuracy. Most CNN model applications carry out back-propagation, as proposed by Rumelhart et al. (1986). This method takes an input vector and produces an output vector which is compared with the target value. Learning takes place if there is a difference between these two values and the weights are adjusted to reduce the difference accordingly. The equation for weight update is

$$w(t+1) = w(t) - \eta \frac{\partial \mathcal{F}(w)}{\partial w}, \quad (7)$$

where w is the weight, η is the learning rate, t is the number of iterations, and $\mathcal{F}$ is an (arbitrary) error function. The negative differential of the loss function multiplied by the learning rate gives the direction in which the weight is updated to achieve the greatest decrease in the loss function. The minimum error rate is achieved when the bias and variance are minimised. Unfortunately, often convergence to these limits can be slow. One of the reasons for this is the magnitude of the learning rate as modifying the weight by a fixed portion of the partial derivative of the error function with respect to the weight only results in a small adjustment to the weight. This is attributable to the shape of the error function. If the shape is flat, the derivative is small whereas a highly curved shape means the derivative is large and can result in overshooting of the local minima (Jacobs 1988). Choosing the correct learning rate is important, if it is set too large

Fig. 5 Classification of image pixels using eigenvalues λ_1 and λ_2 of matrix M obtained in Harris detector algorithm

the training may fail to converge or may even diverge. If the learning rate is too small, then the training may be very slow.

Cyclical learning rate

The learning rate schedule adjusts the learning rate between epochs, and this is done through learning rate annealing. The learning rate is initially large to speed up learning and avoid false local minima, and this also restricts the learning of noise in the dataset. The learning rate subsequently decays as the local minima is approached to suppress oscillations and enhance complex pattern learning. Recently, Smith (2017) proposed a method in which the learning rate is varied cyclically between a minimum base_{lr} and a maximum max_{lr} called the cyclical learning rate (CLR). This method is easy to implement, as it does not require fine-tuning the learning rate parameter through trial and error. It has also been shown to achieve higher accuracy compared to a fixed learning rate. A triangular learning rate policy is adopted, where the learning rate is varied linearly within a band. An input of the step size is required, and this represents the number of iterations in a half cycle. The cyclical learning rate is adjusted as follows:

$$\eta_t = \text{base}_{lr} + (\text{max}_{lr} - \text{base}_{lr})(\max(0, 1 - x)), \quad (8)$$

where x is equal to:

$$x = \left\lfloor \frac{t}{\text{step size}} - 2(\text{cycle}) + 1 \right\rfloor, \quad (9)$$

In addition, cycle is defined as follows:

$$\text{cycle} = \left\lfloor \frac{1 + t}{2(\text{step size})} \right\rfloor. \quad (10)$$

To determine max_{lr} and base_{lr} , the learning rate was increased exponentially following each batch update

between $1e-10$ and 1. The loss for the transfer-learning model is recorded and plotted, and this is shown in Fig. 6 and Table 1. The figure shows that from $1e-10$ to $1e-7$ the loss is static; the learning rate is too low for the network to learn.

At $1e-6$, the network begins to learn as the loss starts decreasing slowly and the learning rate meets the minimum threshold. Between $1e-5$ and $1e-3$ lies the optimal learning rate as the loss decreases rapidly. The learning rate begins to increase past $1e-3$, indicating it is too large for successful learning to take place. The base_{lr} and max_{lr} values are, therefore, set to $1e-5$ and $1e-3$, this linearly alternates periodically with a step size of 8. The CLR schedule using the triangular policy is shown in Fig. 6.

To find the optimal learning rate for Adam during transfer learning, three cases were analysed; first, the rate was set to a constant value of $1e-5$, second at $1e-3$ and finally the rate was varied cyclically between these two values. These rates were tested over 100 epochs and the corresponding validation accuracies and losses are shown in Fig. 7 and Table 2.

The results show that using Adam with a CLR learning schedule achieves the fastest convergence compared to a constant learning rate. The graphs show that a learning rate of $1e-3$ results in oscillations in the performance of the model. This suggests that the rate is too large and hence the weight updates are too large as well, resulting in divergence

Table 1 Validation accuracy and loss obtained

Optimisation	Validation accuracy	Validation loss
Adam ($lr = 1e-3$)	0.8468	0.4862
Adam ($lr = 1e-5$)	0.7990	0.5284
Adam CLR ($\text{max}_{lr} = 1e-3$ and $\text{base}_{lr} = 1e-5$)	0.8524	0.4449

Fig. 6 (Left) training loss as a function of an exponentially increasing learning rate from $1e-10$ to 1 for the transfer-learning model. (Right) CLR schedule using triangular policy varied within $1e-5$ and $1e-3$

Fig. 7 Model loss and accuracy curves without regularisation for different learning rate schedules

Table 2 Validation accuracy and loss obtained with dropout

Optimisation	Validation accuracy	Percentage difference (%)	Validation loss	Percentage difference (%)
Adam ($lr = 1e-3$)	0.8432	−0.43	0.4383	−10.36
Adam ($lr = 1e-5$)	0.8027	0.46	0.5344	1.13
Adam CLR ($\max_{lr} = 1e-3$ and $\text{base}_{lr} = 1e-5$)	0.8520	−0.05	0.4232	−5.00

from the minimum loss. For a learning rate of $1e-3$ combined with CLR, overfitting of the training dataset is evident and the validation curves plateau indicating that no learning is taking place past 15 epochs. Regularisation is, therefore, necessary and methods for avoiding overfitting are discussed in the next subsection.

Regularisation

Regularisers apply a penalty to the parameters in the model as the model complexity increases. This leads to a decrease in the weight matrices as it implies that neural networks with smaller weights result in simpler models, reducing the likelihood of overfitting. The penalties are added to the loss functions as described below.

Dropout

The first form of regularisation explored was dropout, this randomly eliminates units from a network within a

hidden layer with a set probability. This method is used to provide a more efficient alternative of separately training networks which is computationally costly during training and testing. Hinton et al. (2012) found that a dropout of 0.5 applied in all hidden layers achieves a lower error than applying it to one hidden layer. Dropout can approximately increase the number of iterations by twofold, but training time per epoch decreases (Krizhevsky et al. 2012). It helps in generalising the model by forcing it to stop relying on any input node as there is a random probability of it being omitted. Therefore, weights assigned to certain features by the network are low, reducing the bias a model may place on a particular input. Using dropout with a steady learning rate accelerates the convergence of the models without leading to overfitting. Although the performance of the CLR drops, it is no longer overfitting. The gradients of the loss function for all scenarios suggest that the training was stopped prematurely and that there is room for improvement through further training. These results are shown in Table 3.

Table 3 Validation accuracy and loss obtained with L1-regularisers where $\lambda = 0.01$

Optimisation	Validation accuracy	Percentage difference (%)	Validation loss	Percentage difference (%)
Adam (lr = 1e-3)	0.7548	-11.49	1.9795	121.12
Adam (lr = 1e-5)	0.7246	-9.77	0.9355	55.62
Adam CLR (max _{lr} = 1e-3 and base _{lr} = 1e-5)	0.7750	-9.51	0.8226	59.60

Fig. 8 Gradient histogram of dense layers 1 and 2 excluding bias. Left: using $\lambda = 0.01$, right: using $\lambda = 0.0001$

L1-regularisers

The second form of regularisation is L1-regularisers; it minimises the sum of squares conditional on a constant being greater than the sum of absolute values of the coefficients (Tibshirani 1996). This method shrinks and sets less important coefficients to zero to preserve better features. The λ was set to 0.01 and trained over all three cases, the results are shown in Table 3. Weights are updated as follows using the L1-regulariser:

$$w^* = \arg \min_w \sum_{i=1}^N \left(y_i - \sum_j w_j \hat{y}_{ij} \right)^2 + \lambda \sum_j |w_j|, \quad (11)$$

where λ is the regularisation parameter.

The loss function demonstrates that no learning is taking place in all cases, the gradient of the dense layers was plotted to examine this further in Fig. 8. In the two dense layers, the gradient is not flowing backwards through the layers during training and is close to zero in dense layer 1. L1 shrinks irrelevant weights but in this case, the weights

Table 4 Validation accuracy and loss obtained with L1-regularisers where $\lambda = 0.0001$

Optimisation	Validation accuracy	Percentage difference (%)	Validation loss	Percentage difference (%)
Adam (lr = 1e-3)	0.8347	-1.44	0.5634	14.71
Adam (lr = 1e-5)	0.8166	2.18	0.9765	59.55
Adam CLR (max _{lr} = 1e-3 and base _{lr} = 1e-5)	0.8420	-1.23	0.5107	13.77

have been set too low. As a result, λ is decreased to a value of 0.0001, the gradient backpropagating to dense layer 1 is now greater, prompting weight updates resulting in model optimisation. The loss and accuracy for the updated λ values are shown in Table 4.

Step decay

Step decay is the most popular form of learning rate decay; the learning rate is kept constant for a number of steps D and then decreased (Ge et al. 2019):

$$\eta_{T+1} = \eta_T \times F^{\left\lfloor \frac{(1+T)}{D} \right\rfloor}, \quad (13)$$

where F is a factor controlling the rate and T is the number of epochs.

Similar tests were run to find the optimal learning rate for SGD in the fine-tuning model, constant learning rates of $8e-5$ and $1e-4$, CLR in the range $8e-5$ and $1e-4$ with step size of 8 and step decay with initial learning rate of $1e-4$, F of 0.5 and D of 5 were examined. The two values were chosen by plotting loss against learning rate. The results are shown in Fig. 9 and Table 5.

Step decay converges the slowest; however, it is the only method that did not lead to overfitting of the training dataset, in this case, CLR was the fastest to converge.

Nesterov momentum

Nesterov momentum (Nesterov 1983) has recently gained popularity in optimisation problems. Similar to classical momentum this is a first-order optimisation method and is characterised by a convergence rate of $O\left(\frac{1}{t^2} + \frac{1}{\sqrt{bt}}\right)$, compared to in SGD $O\left(\frac{1}{t} + \frac{1}{\sqrt{bt}}\right)$, where b is the minibatch size (Sutskever et al. 2013). It is calculated using the following equation:

Table 5 Validation accuracy and loss obtained for the fine-tuning model

Optimisation	Validation accuracy	validation loss
SGD (lr = $1e-4$)	0.8725	0.3895
SGD (lr = $8e-5$)	0.8688	0.4048
SGD CLR ($\max_{lr} = 1e-4$ and $\text{base}_{lr} = 8e-5$)	0.8736	0.3861
SGD step decay	0.8589	0.4194

$$\Delta w(t) = \eta \frac{dE(w)}{d(w + p\Delta w(t-1))} + p\Delta w(t-1). \quad (15)$$

In some cases, Nesterov momentum did not improve the rate of convergence in comparison to classical momentum; this is because Nesterov only guarantees accelerating convergence in convex gradient descent (Goodfellow et al. 2016). Although Nesterov achieves faster convergences in the start as the $\frac{1}{t^2}$ term dominates, in the long term, SGD and Nesterov methods are equivalently effective as the $\frac{1}{\sqrt{bt}}$ term begins to dominate (Sutskever et al. 2013). In the fine-tuning model, CLR was ultimately used and accelerated by Nesterov momentum where p is equal to 0.9. The results are shown in Table 6.

Semantic CNN segmentation training

For roof segmentation, semantic segmentation is carried out using U-Net (Ronneberger et al. 2015) with a ResNet34 (He et al. 2016) backbone pre-trained on ImageNet. Semantic

Fig. 9 Model loss and accuracy curves with Nesterov momentum for different learning schedules

Table 6 Validation accuracy and loss obtained with Nesterov momentum

Optimisation	Validation accuracy	Percentage difference (%)	Validation loss	Percentage difference (%)
SGD ($\text{lr} = 1\text{e}-4$)	0.8920	2.21	0.3545	− 9.41
SGD ($\text{lr} = 8\text{e}-5$)	0.8876	2.14	0.3508	− 14.29
SGD CLR ($\text{max}_{\text{lr}} = 1\text{e}-4$ and $\text{base}_{\text{lr}} = 8\text{e}-5$)	0.8861	1.42	0.3643	− 5.81
SGD step decay	0.8821	2.67	0.3689	− 12.81

segmentation is a subclass of image segmentation that provides dense predictions, a method by which each pixel class is predicted (Sercu and Goel 2016). Unlike CNNs which are unable to manage different input sizes due to the fully connected layers, U-Net is a modified FCN that can process input images of varying dimensions. U-Net architecture is used for this task as it is designed for biomedical image processing in which images are not readily available. Data augmentation is applied to compensate for the small training dataset through elastic deformations, used in conjunction with traditional image augmentation. The network can learn the invariances within the applied deformations without it being reflected in the annotated images (Ronneberger et al. 2015).

All the roof types, except flat, were analysed to extract their geometries. In this section, we will examine one of these Hip style roofs, as they have some complex geometry and are a good proxy for the errors found in all roof styles.

Training

For hip roofs, the five classes are ‘hip dark’, ‘hip light’, ‘side dark’, ‘side light’ and ‘background’. In total, 195 images were segmented, labelled and split into 187 and 8 for training and validation, respectively. This is the maximum training ratio as the batch size is equal to 8. The training was carried out over 1500 epochs, with a model checkpoint call-back for the maximum Dice coefficient. The best model was

obtained and saved at 1090 epochs with a validation Dice of 96.10% and a validation loss of 0.1489. The roof segmentation progress during training is displayed in Fig. 10. Shadows cast on the upper right corner make it difficult to distinguish the roof boundary correctly, creating an uneven edge in the final segmentation mask.

Testing

The trained model was tested on randomly sampled roofs that are representative of the entire dataset, the masks along with the overlays are displayed in Fig. 11. The displayed images include a plain roof, a dual colour roof with neighbouring buildings, an incomplete roof and a roof with dormers. Although this model achieves a very high Dice score of 96.10%, most of the inaccuracy is attributed to the incorrect detection of the edges. However, for the purpose of this research, the effective detection of edges, typically requiring a much higher accuracy is not needed. Instead, semantic segmentation is used to define the coordinates of the roof corners and the nexus points. Furthermore, segmentation of dormers and chimneys is not necessary as it does not substantially impact on model applications.

Errors

Figure 12 shows common issues that are found in masks that were predicted less successfully, Figures a.3, b.3 and c.3

Fig. 10 Sequential snapshots of semantic segmentation training of the hip roof model, taken at epochs 1, 10, 50, 200, 300, 500, and 1050. Where red is ‘hip light’, green is ‘hip dark’, yellow is ‘side dark’, and blue is ‘side light’

Fig. 11 Results obtained from testing images of hip roofs; top row: the original image, centre row: the mask segments overlaying the source images, bottom row: predicted masks

Fig. 12 Results obtained from testing images of hip roofs; top row: the original image, centre row: the mask segments overlaying the source images, bottom row: predicted mask

show one of the planes was predicted as two classes bleeding into each other. This may be due to the effect of shadows on part of the roof and can be tackled through a mix of image augmentation to darken and lighten the images and training on roofs where the effects of shadows are more prominent. Figure 12, b.3 shows that one of the hip segments spills over onto the adjoining roof. This is most likely due to the similar colour of the adjoining roof; however, it is unclear why the building to the left is segmented as part of the same roof. These problems can be solved by expanding the training dataset to improve its prediction with unevenly coloured roofs and images with crowded roof structures.

Results and discussion

The results obtained in this paper show an improvement when compared to papers that have used deep learning in similar applications. The dataset used by Partovi et al. (2017) contains nearly 10,000 training images achieved a precision and recall of only 76%. Axelsson et al. (2018) achieved the highest accuracy of 96.65% over two classes flat and ridge where ridge is equivalent to gable roof, these two classes are very distinct, the starting accuracy in Axelsson et al. (2018) is 50%, much higher than other papers which start training at an accuracy of 12.5–14.3%. Alidoost and Arefi (2016) achieved a very high accuracy using 700 tiles with approximately 100 tiles per class, and it is the only paper where there was no class imbalance.

To tackle an imbalance in classes, both Axelsson et al. (2018) and Partovi et al. (2017) added more images to the dataset by simply copying the images and augmenting images to artificially inflate the dataset, respectively, this was done prior to training as opposed to during training as done in this paper. The accuracy, precision and recall obtained in this paper are greater than all papers with the exception of Axelsson et al. (2018). However, due to the difference in the number of classes, resolution, geographical location and number of images, it is difficult to draw a fair comparison between the studies.

Table 7 Classification metrics for transfer-learning CNN

	Test images	Precision	Recall	F_1 score
Flat	805	0.93	0.96	0.95
Hip	1072	0.94	0.98	0.96
Cross-hip	65	1.00	0.58	0.74
Gable	1413	0.97	0.95	0.96
Mansard	37	0.67	0.43	0.52
Accuracy	3392	—	—	0.95
Macro-average	3392	0.90	0.78	0.82
Weighted average	3392	0.95	0.95	0.95

Table 8 Validation results for semantic segmentation

Roof	Validation dice (%)	Validation loss
Hip	96.10	0.1489
Cross-hip	85.12	1.1433
Gable	95.94	0.1294
Mansard	91.13	0.4170

The results achieved in this paper are displayed in Tables 7 and 8. To visualise the model's performance, a new tile is selected from the London borough of Harrow. Each roof image is given a prediction and a probability, this is displayed on the main tile image, Fig. 13. The bounding boxes are assigned different colours to represent each label. Houses within the same area typically share similar roof types, an all-inclusive tile is difficult to find, therefore, only four classes are displayed which make up 99.08% of the dataset used in training. The figure shows that the model predicted 88.95% of roofs correctly, the errors are predominantly associated with bounding boxes cropping demolished sheds. This is lower than the accuracy achieved on the test dataset where individual images are examined and removed in cases where only a small part of the roof is visible, the roof is demolished or is not detectable. To avoid this, an up-to-date building footprint is necessary. Misclassifications have a low probability, setting a minimum probability threshold would help in eliminating errors. Other major errors include flat shed false negative (FN) classified as gable; this may be as a result of the coarse resolution of shed structures. Another

Fig. 13 Ariel image for a portion of OS tile TQ1785 with classification labels and certainties applied

misclassification is hip roof FN classified as gable due to the small size of the bounding boxes cropping a small part of the roof, which resembles a gable roof structure.

There are several possible methods available to improve the accuracy and loss achieved by both image classification and semantic segmentation models. Better accuracy can be achieved in image classification using input images with dimensions equal to the image dimensions used during the pre-training of the model. In this case, VGG16 was trained using images with dimensions of 256×256 . The highest resolution provided by getmapping, an Aerial photography provider to Digimap in the UK, is 12.5 cm, double the resolution available for this research. Though these data are not open source, it is hoped that this quality of data may in time be readily available, as the higher resolution would allow the input image dimensions to be increased to 128×128 . The only drawback is that whilst this would increase the accuracy achieved, it is limited by the hardware capabilities needed to process large numbers of structures, in order to create the wireframe outputs shown in Fig. 14.

Shadows, small structures on roofs, and vegetation occlude the roof, and this introduces variations in the dataset which leads to reduced accuracy. This can be predominantly tackled by increasing the model's generalisation ability through the addition of more varied data.

Although complex roofs are uncommon, the model is based on a model-driven approach and is unable to classify roofs that do not fit within the discrete classes assigned. It will, therefore, fail to extract the geometry of these roof structures. Creating a complex class label to allocate unknown roof structures is a simple fix; however, this cannot be translated to semantic segmentation.

Conclusion

In this paper, a deep learning and algorithmic strategy for roof classification using RGB vertical aerial imagery has been proposed. The data are pre-processed using map data to first identify the buildings in Ariel Photography. Then transfer learning using the application and fine-tuning of a VGG16 model pre-trained on ImageNet is implemented. The model head architecture was optimised by exploring

regularisation methods, convergence acceleration algorithms, learning rates and annealing. After development of image classification for this problem, the transfer-learning model utilised CLR with a dropout layer in the model head of 0.5. While the fine-tuning model utilised CLR accelerated using Nesterov momentum. The classified images are segmented using transfer learning of U-Net with a pre-trained ResNet34 backbone to build a semantic segmentation model. The hyperparameters batch size and number of iterations were optimised for the best performing models. Some of these decisions and options for optimisation have been discussed and compared above. This model was found to offer a high degree of accuracy from a small dataset. The semantic segmentation models achieved a Dice score of 96.10%, 95.95%, 85.12% and 91.13% for roof types hip, gable, cross-hip and mansard, respectively. The mask segments enable the extraction of corner and nexus coordinates using the Harris corner detection algorithm. DSM, a raster elevation model, provides building height information, which is integrated into the coordinates to reconstruct a 3D LOD2 model of the buildings. Given the performance of the model on a small region of London, the scope of the work can be extended to a larger region within London and the rest of the UK. The availability of more training data will allow for the first strategy to be revisited and further improved. In future work, focus should be made on improving the segmentation of the current roof classes as well as complex roof structure.

Appendix

This Appendix briefly gives details the authors felt would be useful in replicating the work.

Image classification—training

The number of steps per epoch is calculated by dividing the number of training images by the batch size. The batch size, the number of images passed through one iteration, is set to 32.

Fig. 14 The raw (left), segmented (including corner ID) image (centre) and the reconstructed for a gable roof (right)

In the transfer-learning model, cyclical learning rate (CLR) (Smith 2017) varies between a maximum and minimum learning rates of $1e-3$ and $1e-5$, respectively. This is used in conjunction with a dropout layer of 0.5 in the model head. In the fine-tuning model, CLR is also used varying between $1e-4$ and $8e-5$. It is accelerated using Nesterov momentum (Nesterov 1983) with p of 0.9.

Roof segmentation—data collection and labelling

Images for each class were placed in a folder, filenames were numbered from 0 to n , where n is the total number of images. Folders for images and annotations were created, the image annotations were created using LabelMe, a graphical image annotation tool. As this is a time-consuming process, it was only possible to annotate 101, 195, 100, 78, and 51 images for hip, flat, cross-hip, gable and mansard, respectively.

Roof segmentation—training

Transfer learning is applied using U-Net with a ResNet34 backbone pre-trained on ImageNet. The model layers are frozen, and the output head is warmed up by training over 50 epochs. The layers in the model are then set to trainable and trained over 1500 epochs with a batch size of 8 and Adam learning rate set to $1e-3$.

Funding The research was supported by Natural Environmental Research Council (United Kingdom) Grant no. NE/S003495/1.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Al Shalabi L, Shaaban Z, Kasasbeh B (2006) Data mining: a preprocessing engine. *J Comput Sci* 2(9):735–739
- Alidoost F, Arefi H (2016) Knowledge based 3D building model recognition using convolutional neural networks from lidar and aerial imageries. *Int Arch Photogramm Remote Sens Spatial Inf Sci*. 41:833
- Axelsson M, Soderman U, Berg A, Lithen T (2018) Roof type classification using deep convolutional neural networks on low resolution photogrammetric point clouds from aerial imagery. In: 2018 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, pp 1293–1297
- Bengio Y (2012) Practical recommendations for gradient-based training of deep architectures. *Anonymous neural networks: tricks of the trade*. Springer, New York, pp 437–478
- Castagno J, Atkins E (2018) Roof shape classification from LiDAR and satellite image data fusion using supervised learning. *Sensors* 18(11):3960
- Chevalier M, Thome N, Cord M, Fournier J, Henaff G, Dusch E (2015) LR-CNN for fine-grained classification with varying resolution. In: 2015 IEEE International Conference on Image Processing (ICIP), IEEE, pp 3101–3105
- D'Errico J (2021) A suite of minimal bounding objects, MATLAB Central File Exchange. <https://www.mathworks.com/matlabcentral/fileexchange/34767-a-suite-of-minimal-bounding-objects>. Accessed 9 May 2021
- Dice LR (1945) Measures of the amount of ecologic association between species. Wiley, New York
- Dorninger P, Pfeifer N (2008) A comprehensive automated 3D approach for building extraction, reconstruction, and regularization from airborne laser scanning point clouds. *Sensors* 8(11):7323–7343
- Ducic V, Hollaus M, Ullrich A, Wagner W, Melzer T (2006) 3D vegetation mapping and classification using full-waveform laser scanning. In: EARSeL and ISPRS workshop on 3D remote sensing in forestry. Citeseer
- EDINA Aerial Digimap Service (2018) High Resolution (25cm) Vertical Aerial Imagery [tq1086, tq1287, tq1389, tq1483, tq1580, tq1683, tq1781, tq1783, tq1785, tq1886, tq2280, tq2480, tq2481, tq2482, tq1990]. Getmapping. Scale: 1:500. <https://digimap.edina.ac.uk>. Accessed May 2020
- EDINA Digimap Ordnance Survey Service (2017) OS MasterMap® Topography Layer [TIFF geospatial data] [tq1086, tq1287, tq1389, tq1483, tq1580, tq1683, tq1781, tq1783, tq1785, tq1886, tq2280, tq2480, tq2481, tq2482, tq1990]. Ordnance Survey (GB). Scale: 1:1000. <https://digimap.edina.ac.uk>. Accessed May 2020
- EDINA LIDAR Digimap Service (2016a) Lidar Composite Digital Surface Model England 1m resolution [ASC geospatial data] [tq1580, tq1781, tq1785]. Open Government Licence. Scale: 1:4000. <https://digimap.edina.ac.uk>. Accessed May 2020
- EDINA LIDAR Digimap Service (2016b) Lidar Composite Digital Terrain Model England 1m resolution [ASC geospatial data] [tq1580, tq1781, tq1785]. Open Government Licence. Scale: 1:4000. <https://digimap.edina.ac.uk>. Accessed May 2020
- Fawcett T (2006) An introduction to ROC analysis. *Pattern Recogn Lett* 27(8):861–874
- Ge R, Kakade SM, Kidambi R, Netrapalli P (2019) The step decay schedule: A near optimal, geometrically decaying learning rate procedure
- Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT press, Cambridge
- He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
- Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov RR (2012) Improving neural networks by preventing co-adaptation of feature detectors. [arXiv:1207.0580](https://arxiv.org/abs/1207.0580)
- Jacobs RA (1988) Increased rates of convergence through learning rate adaptation. *Neural Networks* 1(4):295–307
- Jochem A, Höfle B, Rutzinger M, Pfeifer N (2009) Automatic roof plane detection and analysis in airborne lidar point clouds for solar potential assessment. *Sensors* 9(7):5241–5262

- Jutzi B, Gross H (2009) Nearest neighbour classification on laser point clouds to gain object structures from buildings. *Int Arch Photogramm Remote Sens Spatial Inf Sci* 38(Part 1):4–7
- Keskar NS, Socher R (2017) Improving generalization performance by switching from adam to sgd. *arXiv Preprint arXiv:1712.07628*
- Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. *Adv Neural Inf Process Syst* 25:1097–1105
- Lodha SK, Kreps EJ, Helmbold DP, Fitzpatrick D (2006) Aerial LiDAR data classification using support vector machines (SVM). In: *Third international symposium on 3D data processing, visualization, and transmission (3DPVT'06)*. IEEE, pp 567–574
- Matikainen L, Hyypä J, Hyypä H (2003) Automatic detection of buildings from laser scanner data for map updating. *Int Arch Photogramm Remote Sens Spatial Inf Sci* 34(3/W13):218–224
- Mongus D, Lukač N, Žalik B (2014) Ground and building extraction from LiDAR data based on differential morphological profiles and locally fitted surfaces. <http://www.sciencedirect.com/science/article/pii/S0924271613002840>
- Nesterov YE (1983) A method for solving the convex programming problem with convergence rate $O(1/k^2)$. *Dokl. akad. nauk Sssr*, pp 543–547
- Partovi T, Fraundorfer F, Azimi S, Marmanis D, Reinartz P (2017) Roof Type Selection based on patch-based classification using deep learning for high Resolution Satellite Imagery. *Int Arch Photogramm Remote Sens Spatial Inf Sci ISPRS Arch* 42(W1):653–657
- Patro S, Sahu KK (2015) Normalization: a preprocessing stage. *arXiv Preprint arXiv:1503.06462*
- Pingel TJ, Clarke KC, McBride WA (2013) An improved simple morphological filter for the terrain classification of airborne LIDAR data. *ISPRS J Photogramm Remote Sens* 77:21–30
- Pirotti F, Zanchetta C, Previtali M, Della Torre S (2019) Detection of building roofs and facades from aerial laser scanning data using deep learning. In: *2nd international conference of geomatics and restoration, GEORES 2019*. Copernicus GmbH, pp 975–980
- Ronneberger O, Fischer P, Brox T (2015) U-net: Convolutional networks for biomedical image segmentation. In: *International conference on medical image computing and computer-assisted intervention*. Springer, New York, pp 234–241
- Ruder S (2016) An overview of gradient descent optimization algorithms. *arXiv Preprint arXiv:1609.04747*
- Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. *Nature* 323(6088):533–536
- Rusiecki A (2019) Trimmed categorical cross-entropy for deep learning with label noise. *Electron Lett* 55(6):319–320
- Sercu T, Goel V (2016) Dense prediction on sequences with time-dilated convolutions for speech recognition. *arXiv. arXiv:1611.09288*.
- Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. *arXiv Preprint arXiv:1409.1556*
- Smith LN (2017) Cyclical learning rates for training neural networks. In: *2017 IEEE winter conference on applications of computer vision (WACV)*, IEEE, pp 464–472
- Sutskever I, Martens J, Dahl G, Hinton G (2013) On the importance of initialization and momentum in deep learning. In: *International conference on machine learning*, pp 1139–1147
- TensorFlow (2017) [www.TensorFlow.org](http://www.tensorflow.org). Accessed May 2020
- Tibshirani R (1996) Regression shrinkage and selection via the lasso. *J R Stat Soc Ser B (Methodol)* 58(1):267–288
- Van Vliet LJ, Young IT, Verbeek PW (1998) Recursive Gaussian derivative filters. In: *Proceedings. Fourteenth international conference on pattern recognition (Cat. No. 98EX170)*, IEEE, pp 509–514
- Vosselman G, Dijkman S (2001) 3D building model reconstruction from point clouds and ground plans. *Int Arch Photogramm Remote Sens Spatial Inf Sci* 34(3/W4):37–44
- Yosinski J, Clune J, Bengio Y, Lipson H (2014) How transferable are features in deep neural networks? *Adv Neural Inf Process Syst* 27:3320–3328
- Zhang K, Chen S, Whitman D, Shyu M, Yan J, Zhang C (2003) A progressive morphological filter for removing nonground measurements from airborne LIDAR data. *IEEE Trans Geosci Remote Sens* 41(4):872–882
- Zhao R, Pang M, Wang J (2018) Classifying airborne LiDAR point clouds via deep features learned by a multi-scale convolutional neural network. *Int J Geogr Inf Sci* 32(5):960–979

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.