PFG (2023) 91:125–147
https://doi.org/10.1007/s41064-023-00233-3

ORIGINAL ARTICLE

Optimised U-Net for Land Use–Land Cover Classification Using Aerial Photography

Andrew Clark[1,2] · Stuart Phinn[2] · Peter Scarth[2]

Received: 17 August 2022 / Accepted: 13 January 2023 / Published online: 13 February 2023

Abstract
Convolutional Neural Networks (CNN) consist of various hyper-parameters which need to be specified or can be altered when defining a deep learning architecture. There are numerous studies which have tested different types of networks (e.g. U-Net, DeepLabv3+) or created new architectures, benchmarked against well-known test datasets. However, there is a lack of real-world mapping applications demonstrating the effects of changing network hyper-parameters on model performance for land use and land cover (LULC) semantic segmentation. In this paper, we analysed the effects on training time and classification accuracy by altering parameters such as the number of initial convolutional filters, kernel size, network depth, kernel initialiser and activation functions, loss and loss optimiser functions, and learning rate. We achieved this using a well-known top performing architecture, the U-Net, in conjunction with LULC training data and two multispectral aerial images from North Queensland, Australia. A 2018 image was used to train and test CNN models with different parameters and a 2015 image was used for assessing the optimised parameters. We found more complex models with a larger number of filters and larger kernel size produce classifications of higher accuracy but take longer to train. Using an accuracy-time ranking formula, we found using 56 initial filters with kernel size of 5×5 provide the best compromise between training time and accuracy. When fully training a model using these parameters and testing on the 2015 image, we achieved a kappa score of 0.84. This compares to the original U-Net parameters which achieved a kappa score of 0.73.

Keywords Convolutional neural network · Deep learning · Semantic segmentation · Land use · Land cover · Aerial imagery

1 Introduction

Image classification for analysis of land use and land cover (LULC) was one of the earliest fundamental remote-sensing methods (Jensen 2005). There have been many studies exploring classification techniques for LULC mapping, but identifying the ideal classifier for any given application is still unclear (Pandey et al. 2021). Traditional approaches, such as maximum likelihood, fuzzy logic, and object-oriented classifications, are referred to as shallow learning. These methods extract data based on spatial, spectral, textural, morphological, and other cues (Ball et al. 2017). In contrast, deep learning is multi-layered, learns from the data itself, and results can be significantly more accurate than shallow learning (Deng 2014) and can outperform manual human editing (Zhang et al. 2016).

For LULC classification within high-resolution imagery, the most accurate classification algorithms are Decision Trees (DT), Support Vector Machines (SVM), Random Forest (RF), and CNN (Ma et al. 2019). Many studies have compared deep learning techniques to shallow learning algorithms such as RF and SVM. Kussul et al. (2017) found an ensemble of CNN outperformed fully connected multilayer perceptrons and RF classifiers for land cover and crop types using Landsat 8 and Sentinel-2 imagery. Stoian et al. (2019) compared an altered U-Net with RF for mapping various land uses with a time-series of Sentinel-2 data. They found that U-Net had a similar or better accuracy than RF but took longer to train; however, the study did not use a Graphics Processing Unit (GPU) which would have made the training of the U-Net more efficient (Krizhevsky et al. 2012). Freudenberg et al. (2019) trained U-Net using a GPU to detect palm tree plantations in Worldview imagery and

✉ Andrew Clark
Andrew.Clark@uqconnect.edu.au

1 Remote Sensing Research Centre, The University of Queensland, St Lucia, Brisbane, QLD, Australia

2 Applied Agricultural Remote Sensing Centre, The University of New England, Armidale, NSW, Australia

found U-Net to be > 10% more accurate and 100 times more efficient than AlexNet.

The integration of textural properties through object-based segmentation techniques has significantly improved classification results for remote-sensing applications (Blaschke et al. 2015). Classifications that use spatial as well as spectral information outperform those based on purely on pixels (Pandey et al. 2021). As a result, CNN classifications assisted by enhanced computing power in recent years (Ma et al. 2016) have higher accuracy compared to SVM and RF (Ma et al. 2019).

Deep learning is a subset of machine learning and a learning algorithm based on neural networks (Schmidhuber 2015). These networks consist of many layers which can transform images into categories through learning high-level features (Litjens et al. 2017). Neural networks are described as networks which simulate the processes of the human brain—interconnected neurons which process incoming information (Jensen 2005). The solution is obtained by nonalgorithmic and unstructured methods by adjustment of weights connecting the neurons in the network (Rao & Rao 2003). They have the advantage of not requiring normal distributed data and can adaptively simulate complex and non-linear patterns (Jensen 2005; Zhu et al. 2017) such as found in high-resolution aerial photography.

Convolutional Neural Networks (CNN) are situated at the fringe between machine learning and computer vision, combining the power of deep learning with contextual image analysis. CNN have gained momentum for image classification, since the AlexNet architecture won the ImageNet contest by a wide margin in 2012 (Krizhevsky et al. 2012).

CNN have been used in applications, such as number plate reading (Mondal et al. 2017), facial recognition (Almabdy & Elrefaei 2019), and aerial image classification (Khalel & El-Saban 2018; Othman et al. 2016). Many previous studies applying deep learning to earth observation data have focused on the classification or categorisation of image chips but do not address the problem of per-pixel classification (Castelluccio et al. 2015; Cheng et al. 2015; Penatti et al. 2015). Although an advancement in computer vision for earth observation, these classifications are not useful in natural resource management applications as they do not define precise spatial location and extent of LULC features. More recent studies have addressed this problem, and per-pixel classifications using deep learning seem to be outperforming the traditional approaches in high-resolution imagery (Ma et al. 2019).

CNN are best suited to higher resolution imagery and a number of studies have found U-Net performs better using this imagery compared to medium resolution sensors such as Sentinel-2 (Liu et al. 2019; Wurm et al. 2019). Medium-resolution images (10–30 m) lack the fine structural components found in higher resolution imagery; it is difficult to apply CNN classification without this additional information on texture (Sharma et al. 2017). Wurm et al. (2019) found higher accuracy when using Quickbird imagery compared to lower resolution Sentinel-2 imagery for the detection of slums in Mumbai, India. Romero et al. (2016) found deep CNN architectures performed significantly better than shallow CNN, kernel-based Principal Component Analysis and spectral classification for land-use classification in aerial photography, multispectral and hyperspectral imagery.

As the benefits of CNN are realised within the earth observation community, the drive to find the optimal configuration for LULC classification has resulted in many suggested architectures for a wide range of applications (Huang et al. 2018; Zhang et al. 2018). Adding further complexity to the subject, existing architectures such as Inception (Szegedy et al. 2015) and ResNet (He et al. 2016) used for traditional photography classification have also been successfully adapted for mapping LULC using multispectral remote-sensing imagery (Ghosh et al. 2020; Liu et al. 2019; Mahdianpari et al. 2018).

The most important issue with some CNN architectures is they contain large receptive fields and evaluation of contextual information over multiple scales can result in classifications at a lower resolution than the original image. A remote-sensing and deep learning review article by Ma et al. (2019) suggested that creating symmetric architectures using unconvoluted layers and skip connections, such as U-Net (Ronneberger et al. 2015), results in classifications with the same spatial resolution as the input.

Originally developed for biomedical imaging (Ronneberger et al. 2015), U-Net has been adopted for use with optical earth observation data and has become one of the most popular architectures (Neupane et al. 2021) with overall accuracies of > 90% (Du et al. 2019; Flood et al. 2019; Gurumurthy et al. 2019; Kestur et al. 2019; Khryashchev et al. 2018; Wagner et al. 2019; Wei et al. 2019). There are numerous studies which have redesigned the U-Net specifically for extracting features from earth observation data (Kim et al. 2019; Ren et al. 2020); however, this has further complicated the issue as to which architecture is best to use for general earth observation applications.

Additional complexities exist when applying the U-Net architecture to earth observation data with selection of appropriate algorithmic options for kernel initialisation, activation, loss, loss optimiser, and learning rate. Further to this, the U-Net architecture can be defined with different numbers of initial filters, kernel size, and depth (number of layers); all of which can have profound impacts on how well a model learns the training data, if it is able to learn at all. For example, CNNs can experience a vanishing gradient problem when using certain activation functions with a small range of values (e.g., sigmoid) resulting in insufficient learning (Sun et al. 2019).

Globally consistent archives of regularly updated moderate-to-high spatial resolution satellite imagery are becoming increasingly available (e.g., Google Earth Engine Amazon Web Services and Microsoft Planetary Computer data stores). The data are only one part of the mapping solution—appropriate mapping methods, data storage, and computational platforms at scale are all required to conduct useful data analysis within the field "big data analysis" (Hey et al. 2009). Users have recognised the need for supply of imagery and increased their demand for spatial information accordingly (i.e., larger areas of interest, more-rapid response, and more-detailed classifications), but how the imagery relates to big data analysis is an open research question.

The objective of this study is to assist in the development of a framework to use existing LULC information from one year to automatically classify LULC features from another point in time. Developing a framework to automatically map LULC would be of benefit to mapping programs globally. The automated and efficient classification of LULC features from high spatial resolution imagery will be extremely valuable to inform response to natural disasters or biosecurity incidents, as well as studies on agricultural productivity and sustainability, land-use planning, monitoring of and investment in natural resources, biodiversity conservation, and improving water availability and quality.

In this paper, we provided some clarity on the use of U-Net for earth observation applications by running a series of trials with varying parameters of the U-Net to determine the optimal architecture configuration and algorithmic choice for LULC classification in aerial photography. Optimising the U-Net architecture for LULC mapping improves understanding and application of CNN for operational LULC mapping across large regions and assists in addressing the increased demand for rapid large-area classification using high spatial resolution imagery.

2 Method

2.1 Project Area

The project area focused on an agricultural region on the Atherton Tablelands in North Queensland, Australia (Fig. 1). The area consists of the World Heritage listed Wet Tropics rainforests in the east and encompasses the towns of Mareeba in the north, Atherton in the south, and Dimbulah to the west.

The major land uses of the area include production from relatively natural environments, conservation and natural environments, and production from irrigated agriculture and plantations (Table 1) (DSITI 2017).

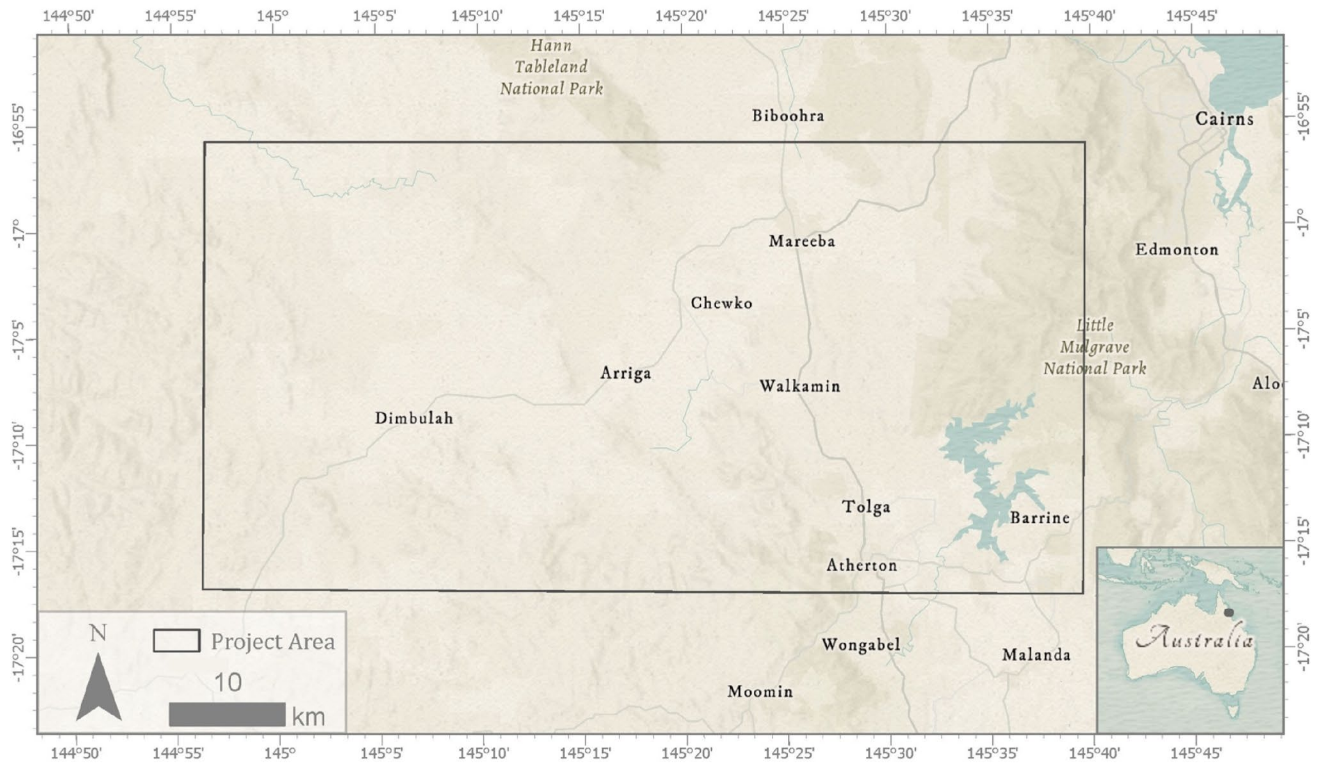


Fig. 1 Project area located in north Queensland, Australia. Basemap provided by ESRI

Table 1 Project area primary land uses (DSITI 2017)

Primary land use	Hectares	Proportion (%)
Production from relatively natural environments	201,625	67.3
Conservation and natural environments	40,404	13.5
Production from irrigated agriculture and plantations	37,939	12.7
Intensive uses	10,133	3.4
Water	7219	2.4
Production from dryland agriculture and plantations	2143	0.7
Total	299,463	100.0

2.2 Multispectral Aerial Imagery

Orthorectified aerial images of the study area are periodically captured through the Queensland Government Spatial Imagery Subscription Plan. For this project, we used the 2018 orthorectified mosaic with a spatial resolution of 20 cm for training the models and referred to as the 2018 training image (Fig. 2b). The image was acquired between 1st and 27th August 2018 mostly using a Vexcel Ultracam Eagle camera except for the southwestern corner which was captured by and A3 Edge camera. The two cameras have different spectral properties which can be seen in Fig. 2b.

To test the accuracy and transferability of the models, a 2015 mosaic captured between 17th July and 14th October using an A3 Edge camera at a spatial resolution of 25 cm was used and referred to as the 2015 test image (Fig. 2a).

Both images consisted of three bands (red, green, and blue) and were resampled to 50 cm using cubic convolution for spatial resolution consistency between the images and to minimise the amount of data required for training and inference.

2.3 Training Data Collection

The nine LULC classes selected for this project included banana plantations, sugarcane crops, mature tree crops, young tree crops, tea tree plantations, forestry plantations, berry crops, and vineyards with all remaining areas classified as other uses. These classes were chosen based on the ability to identify the class within the RGB imagery and in the field, and existing data from previous work (Clark & McKechnie 2020). These data were manually digitised using the 2015 test image and 2018 training image within a Geographic Information System (GIS). Table 2 lists the total class area and proportion of the project area for 2015 and 2018. The data were stored as polygons within an ESRI shapefile format.

A field verification trip to the project area was undertaken in January 2020 and collected 1,582 observations on LULC for the eight main classes along with other land uses. The data collected in the field were used to ensure that the training data were accurate with the temporal difference between the image acquisition and field observation (17 months) taken into consideration.

2.4 Training Data Generation

Semantic segmentation using CNNs involves a patch-based training approach where image chips of a certain size are fed to the model and convolutional filters attempt to extract abstract information from the data to ultimately define features according to colour, texture, and other contextual information. The purpose of the training data generation stage was to provide the model with many different examples of LULC features.

LULC data intrinsically contain an imbalance in class area which was evident within our project (Table 2). As a result, any systematic approach (e.g., grid based) to data sampling will contain a disproportion of data between large and small area classes. To counteract this imbalance, a stratified random sampling approach was implemented.

To achieve this, we calculated the number of patches per class by multiplying the total number of patches by the log of the class area divided by the sum of the log of the project area. The result was rounded up to the nearest integer (Eq. 1)

$$N_{cp} = \left\lceil N_p \frac{\log(a_c)}{\sum_n^i \log(a_c)} \right\rceil, \tag{1}$$

where N_{cp} is the number of class patches, N_p is the total number of required training patches, and a_c is the class area.

For each feature, we distributed the number of class patches based on the proportion of the area that the feature represents rounded up to the nearest integer (Eq. 2)

$$N_{fp} = \left\lceil N_{cp} \frac{a_f}{a_c} \right\rceil, \tag{2}$$

where N_{fp} is the number of feature patches, N_{cp} is the number of class patches, a_f is the feature area, and a_c is the class area.

Once we calculated the number of patches per feature, we randomly generated patch locations ensuring that the centre of the patch was located within the feature. Table 3 shows the distribution of training patches per class. Although the centre of each class patch was within the target feature, patches also contained other surrounding class features.

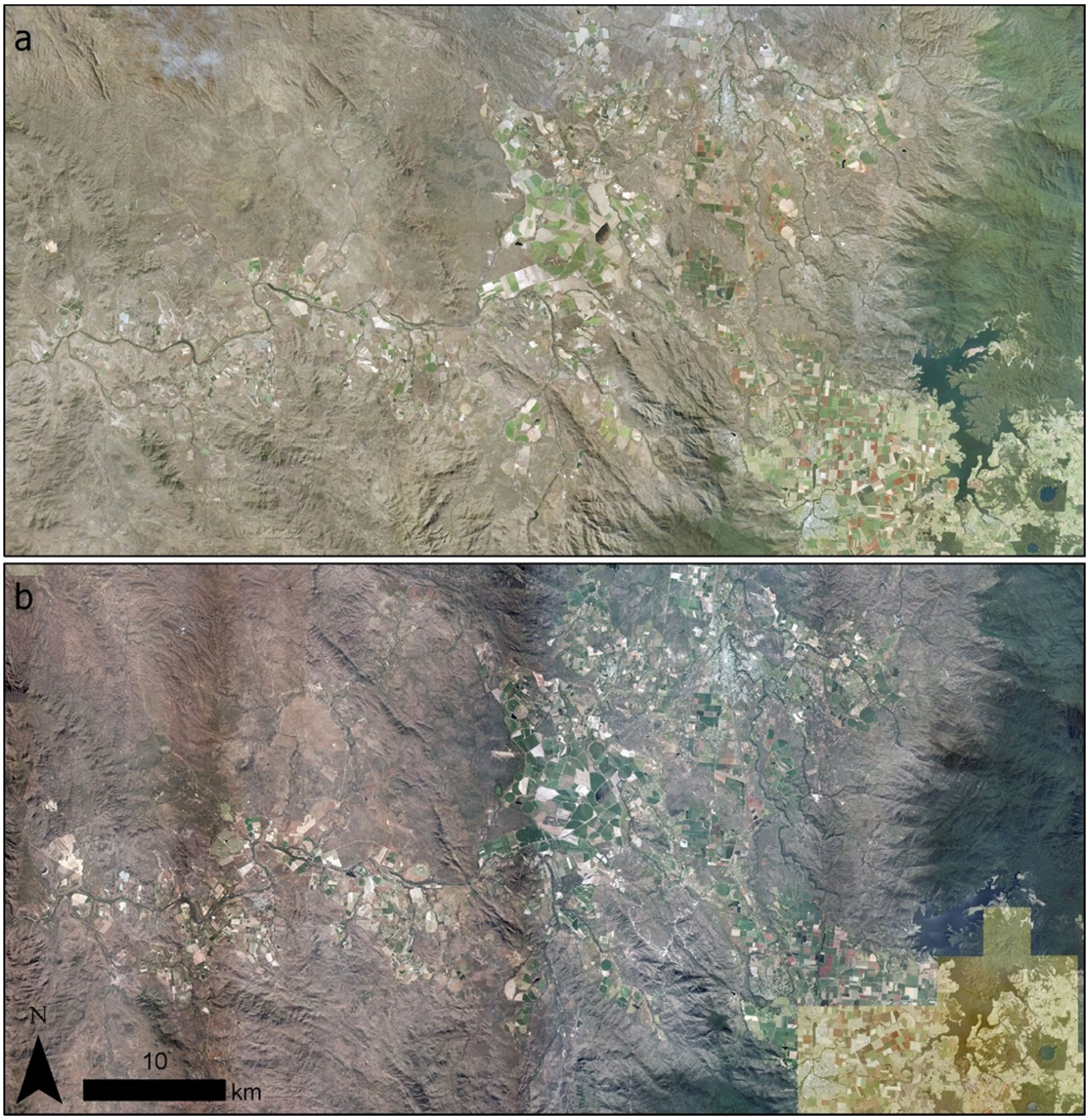


Fig. 2 Project orthorectified aerial imagery for 2015 (**a**) and 2018 (**b**). Data were supplied by the Queensland Government

Table 2 Class area and proportion of the project area for each class for 2015 and 2018

Class	2015 Area (ha)	2015 Area (%)	2018 Area (ha)	2018 Area (%)
Banana plantation	1898	0.63	1860	0.62
Berry crops	48	0.02	92	0.03
Forestry plantation	284,015	94.84	281,344	93.95
Sugarcane crop	875	0.29	981	0.33
Tea tree plantation	6102	2.04	7621	2.54
Tree crop—mature	133	0.04	188	0.06
Tree crop—young	5624	1.88	6249	2.09
Vineyards	655	0.22	988	0.33
Other	119	0.04	146	0.05
Total	299,469	100	299,469	100

Table 3 Distribution of training patches per class generated for the 2018 training data

Class	Area (ha)	Number of patches
Banana plantation	1860	2371
Berry crops	92	1876
Forestry plantation	981	2224
Sugarcane crop	7621	2716
Tea tree plantation	188	1957
Tree crop—mature	6249	3986
Tree crop—young	988	2311
Vineyards	146	1920
Other uses	281,344	3474
Total	299,471	22,835

This approach counteracts class imbalances, improving the overall classification accuracy particularly for under-represented classes of small area. For our project, we have chosen a patch size of 512 × 512 pixels. The patches were stored as polygons within an ESRI shapefile format.

Figure 3 illustrates the geographical distribution of the patches and demonstrates the effect of the stratified sampling approach, creating a more balanced distribution between larger and smaller area classes.

During the training process, a data generator uses the spatial extent of each patch to generate the training image chip and corresponding labels on demand. The labels are a rasterised version of the training data in a one-hot representation of the data where each band represents one of the nine classes. Within each band, 1 represents the presence of the class and 0 absence.

The data generator can optionally scale each batch of image chips between 0 and 255 and apply random augmentations to the data. Data scaling and augmentation is useful in capturing atmospheric, climatic, seasonal, and sensor variations present within earth observation data (Wieland et al. 2019), prevents overfitting of the data (Diakogiannis et al. 2020; Shorten and Khoshgoftaar 2019), and creates more generalised models (Kattenborn et al. 2021).

To apply the random augmentations to the patches, the python package imgaug v0.4.0 (Alexander Jung, 2020) was used. Augmentations were chosen based on their suitability to earth observation data. The augmentations included:

1. contrast and colourations manipulation (gamma, sigmoid, AllChannelsCLASHE, linear, multiply, allChannelsHistogramEqualization).
2. noise addition (salt and pepper, multiply element wise, additive Gaussian, additive poisson, multiply—different for each channel).
3. geometric and scale alterations (affine, elastic transformation, vertical and horizontal flips).
4. image blur and adding artificial clouds/fog/smoke.

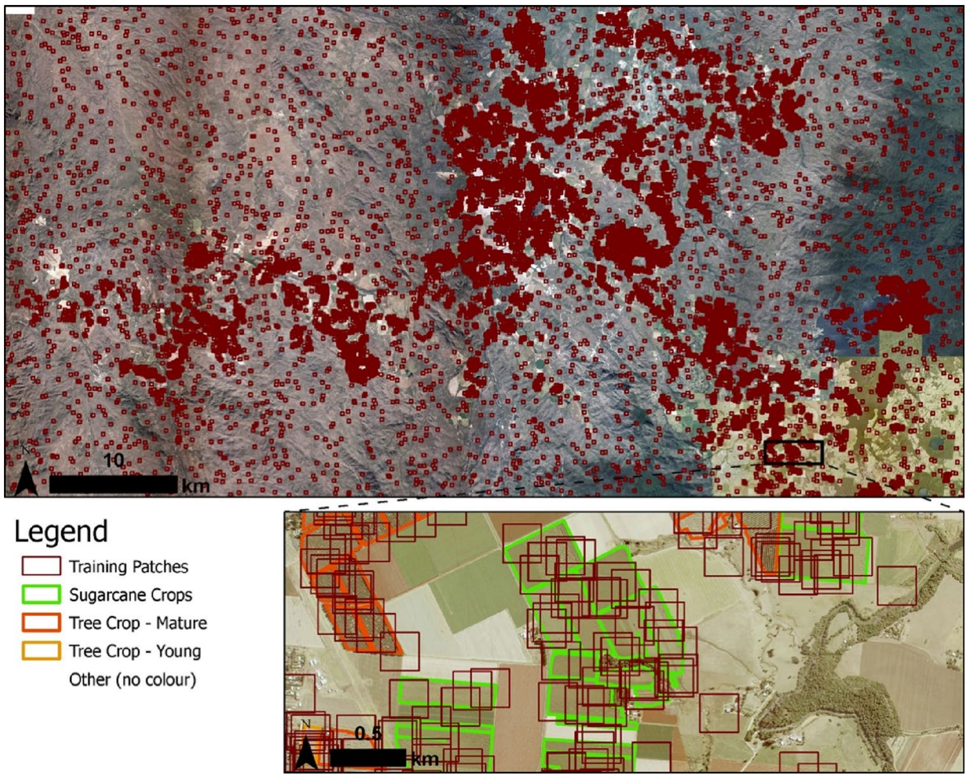


Fig. 3 Project training patches distribution for the 2018 training data generation

One of the contrast, noise and geometric distortions, were applied; each time, the image chip was created and 50% of the time either blur or artificial clouds/fog were applied.

Figure 4 shows examples of the augmentations applied to each patch. Note that the examples provided do not represent the patch size used in this study but are intended to demonstrate examples of the augmentations applied to each patch.

2.5 Training

The objective of the training was to produce a model labelling each pixel in the image as a class. To achieve this, the U-Net was chosen as the base architecture. The U-Net is a symmetrical encoder–decoder architecture and the resulting classification has the same resolution as the input data.

The training involved creating a model and iteratively providing it with batches of image chips and corresponding labels. This allowed the model to determine the relevant colour, texture, and contextual attributes of each class (Zhang et al. 2016) that maximised the overall accuracy of the classification.

Figure 5 is a graphical representation of the U-Net architecture from Clark and McKechnie (2020) showing the main components.

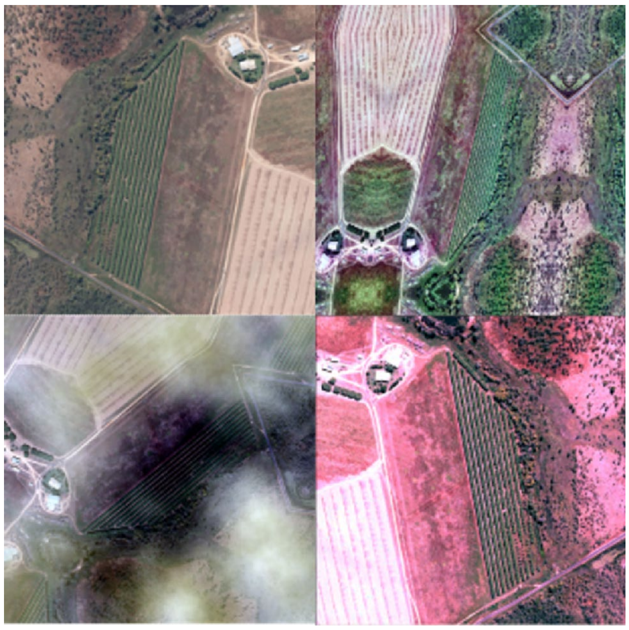


Fig. 4 Example of augmentations over an area consisting of a mature tree crop including the original image (top left) and three augmentations versions. Note that these examples do not represent the patch size used in this study but are for demonstration of the augmentations used for each patch

2.6 Prediction

During the prediction stage, the model was applied to the whole aerial image to produce pixel wise inference and classification. Similar to the input training data, the model attempted to classify each pixel as a 1 or 0 for each class. As the models are not perfect, the output predictions contain values between 0 and 1 and each class is represented by the raster bands. We simply used the class with the highest predictive value to classify each pixel resulting in a thematic classification with values between 1 and 9 representing each LULC class.

One of the limitations of a patch-based learning approach is that the edge of the patch had a lower accuracy than the centre (Sun et al. 2019). This created edge effects between the patches evident when the resulting classification was compiled. To counteract this issue, we employed a two-pass ensemble inference strategy. The strategy involved systematically iterating through the aerial image, extracting patches in a grid pattern. The inference was applied to the original patch and three rotated versions (90, 180, and 270 degrees) with the resulting classifications averaged. Once the model had been applied to the project area, we repeated the process with a half patch offset (256 pixels) to the grid, so the centre of each patch in the second pass was the join between four patches from the first. We applied a weighted average based on the distance from the centre to combine the classifications. Pixels classified towards the centre of the patch were given a higher weight than those generated at the edge. The resulting classification was a much smoother which can result in higher accuracy.

2.7 Deep Learning Trials

The objective of this study was to determine the optimal U-Net architecture configuration for earth observation applications. To achieve this objective, we performed a grid search on: the number of filters; kernel sizes; number of layers; kernel initialisations; convolution and class activations; loss functions; optimisation functions; and learning rates. There are two overall aspects in these trials which include altering the architecture (number of initial filters, filter kernel size, and number of layers) and algorithmic trials which are related to the training of the model (kernel initialisation, convolution activation function, loss function, loss optimiser, learning rate, and class activation).

Table 4 lists the different parameters, the test values, and the default value used when the parameter was not being tested. The only parameters altered within each test were the ones being trialled. All other parameters remained constant for the training. No data scaling or augmentations were applied to the parameter trials to minimise the random effects these would cause.

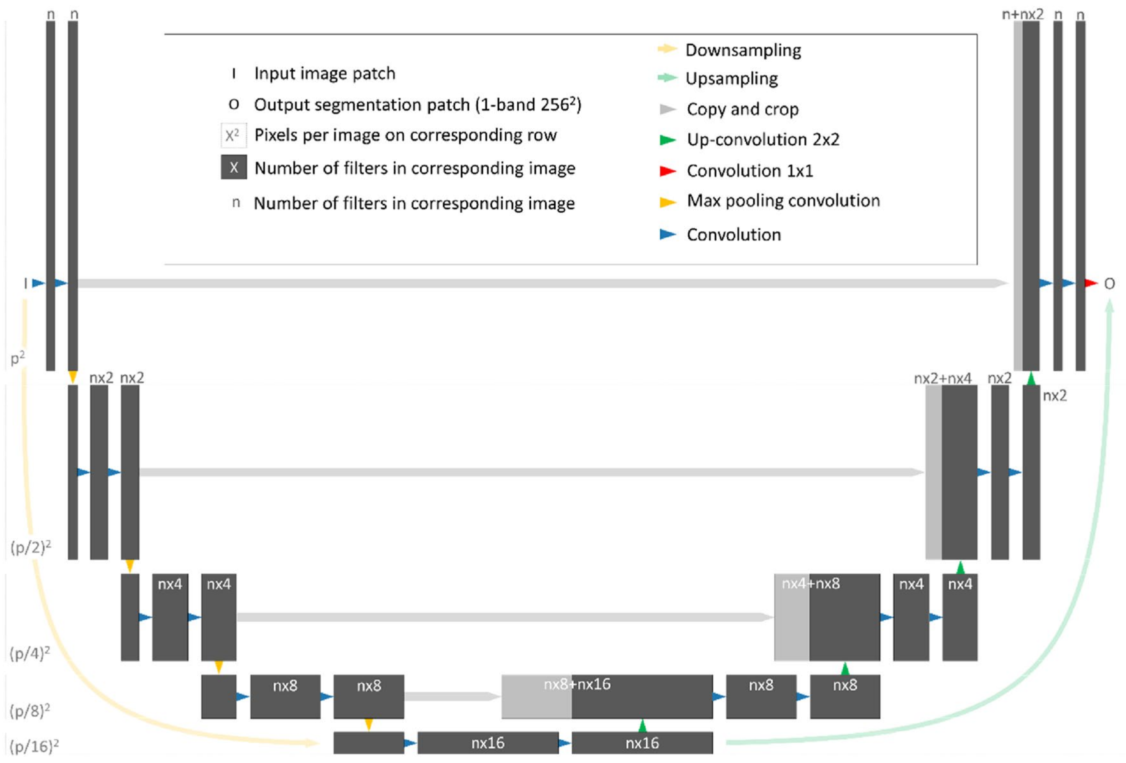


Fig. 5 The U-Net architecture (Clark & McKechnie 2020; Ronneberger et al. 2015)

Table 4 Parameter values and defaults for the deep learning trials

	Parameter	Test values	Default
Architecture trials	Number of filters	8, 16, 24, 32, 40, 48, 56, 64, 72, 80, 88, 96	32
	Kernel size	3, 5, 7, 9	3
	Number of layers	3, 4, 5, 6, 7	5
Algorithmic trials	Kernel initialiser	Glorot Normal, Glorot Uniform, He Normal, He Uniform, Lecun Normal, Lecun Uniform, Orthogonal, Random Normal, Random Uniform, Truncated Normal	Glorot Uniform
	Convolution activation function	ELU, Hard Sigmoid, Linear, RELU, SELU, Sigmoid, Softmax, Softplus, Softsign, Swish, Hyperbolic tangent	Rectified Linear Unit
	Loss function	BCE, CCE, CH, Hinge, Huber, Jaccard Loss, KLD, logcosh, MAE, MAPE, MSE, MSLE, Poisson, Squared Hinge	Categorical Cross-Entropy
	Optimiser	Adadelta, Adagrad, Adam, Adamax, Ftrl, Nadam, RMSprop, SGD	NAdam
	Learning rate	1e-1, 1e-2, 1e-3, 1e-4, 1e-5, 1e-6, 1e-7, 1e-8, 1e-9	1e-5
	Class activation	Hard Sigmoid, Sigmoid, Softmax	Softmax

The loss functions, activations, initialisers, and optimisers are part of the TensorFlow (Abadi et al. 2016) and the Jaccard Loss was adopted from (Eelbode et al. 2020). See Appendix 1 for a list of abbreviations

Within each parameter trial, five models were trained for 20 epochs for each test value resulting in 385 models. It was decided that repeating each test five times captured the random variance intrinsic to the training process. Twenty iterations of the training data were not sufficient to create fully trained models; however, this provides an indication of how well the parameter values fit the data while reducing resource requirements. Further to this, as data augmentations were not applied, the models were not generalised and cannot be transferred to the 2015 test image. At this stage of the project, we compared the learning performance on the training image as an indication of effectiveness of the parameter.

Applying the model on the test image occurred after the parameter trails results had been compiled and a model was fully trained using the optimised parameters.

For the deep learning trials, we produced the classification in a single pass without augmentations to avoid any data smoothing the two pass inference strategy causes. As with excluding the random augmentations in the training stage, we wanted to evaluate parameter performance without any post processing steps.

2.8 Accuracy Assessment

To evaluate the output classifications, we employed an independent accuracy assessment using 10,000 randomly generated points. Each point was assigned one of the nine classes for the 2018 training image and 2015 test image. A random sampling strategy was employed to simplify the sampling strategy, and to remove reliance on prior knowledge of class distribution.

Figure 6 shows the randomly generated points over the project area coloured according to the 2018 class.

The disadvantage of using a random sampling strategy in a landscape where classes have disproportionate areas was most of the points fell within classes with the largest area and few within those with small areas. As a result, 94% of our validation points fell within the 'other uses' class and some of the smaller classes consisted of 5 or 6 points. However, as the models are trained and assessed five times, this resulted in a minimum of 25 observations in each class for each parameter trial. Although it would be ideal to assess the accuracy using additional points, time and resource considerations limited this capacity and previous work has shown that with this number of sample points, a random sampling strategy showed no significant differences between sampling methods (Ramezan et al. 2019).

Using the random points, we assessed the accuracy of each parameter trial by calculating Cohen's kappa coefficient, Jaccard Index, user accuracy (precision), producer accuracy (recall), and F1-Score for each class. The results were ranked from one to the total number of parameter tests. This was done by ranking the average user and producer accuracies and the length of time it took to complete twenty epochs rounded to the nearest 15 min for each trial test. The parameter trials were scored by adding the user, producer, and time ranks with a final parameter rank based on this score.

2.9 Extended Training

Once the top performing parameters were determined, an optimised U-Net model was trained using these values. In addition, a model was trained using the original U-Net parameters for comparison to the optimised version. The original U-Net parameters are the same as the default parameters show in Table 4 with the exception of number of initial filters which was 64 to match the original U-Net architecture published in Ronneberger et al. (2015). The models were trained until the model improvement was < 0.05% over three epochs (63 epochs for the optimised model and 67 epochs for the original model). The prediction was applied to the 2015 test image to produce a classification on unseen data.

The accuracy of the optimised U-Net classification was compared to the original U-Net classification to determine if the performance of the U-Net for earth observation data

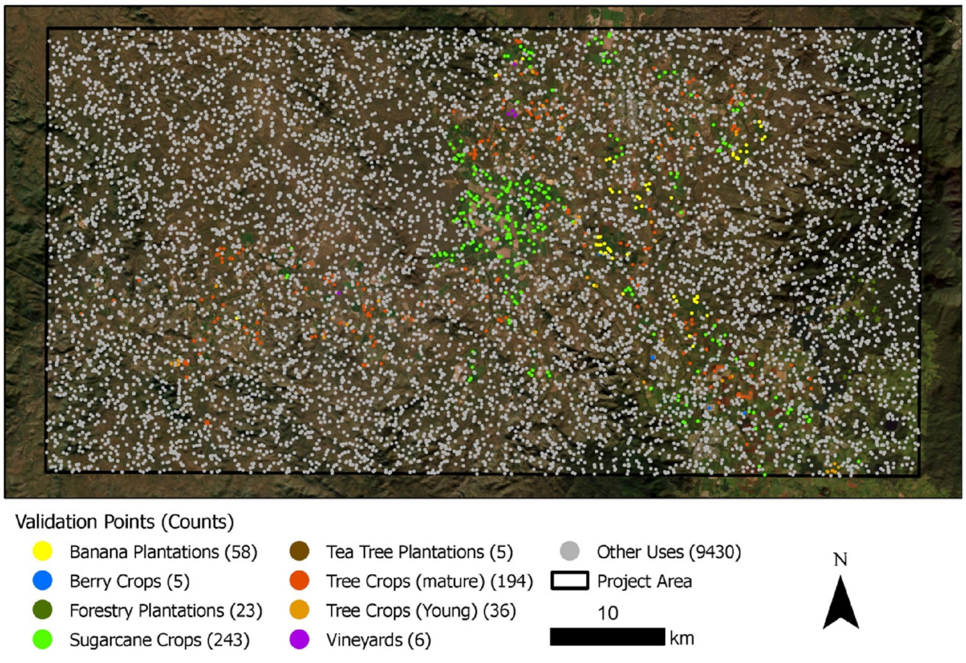

Validation Points (Counts)
● Banana Plantations (58)
● Berry Crops (5)
● Forestry Plantations (23)
● Sugarcane Crops (243)
● Tea Tree Plantations (5)
● Tree Crops (mature) (194)
● Tree Crops (Young) (36)
● Vineyards (6)
● Other Uses (9430)
▭ Project Area
10
km
N

Fig. 6 Validation points used to assess the accuracy of the output model classifications coloured according to the 2018 observation value

was increased. This was achieved through analysis of the kappa, Jaccard Index, user (precision), producer (recall), and F1-Score statistics along with confusion matrices.

2.10 Computing Infrastructure and Software

The training of the models was performed on Nvidia Tesla V100 GPUs. The processing of vector and image data was conducted using the Geospatial Data Abstraction Library (GDAL) version 3.1.0 (https://gdal.org/) and the deep learning part of the project used TensorFlow 2.1.0 (Abadi et al. 2016). Image augmentations were done using the python library imgaug 0.4.0 (https://imgaug.readthedocs.io/en/latest/).

3 Results and Discussion

Appendix 2 contains the full results and lists the trials, trial parameters, average training time, kappa, $F1$-Score, user (precision), producer (recall) rank, and corresponding confidence intervals. Table 5 shows a summary of the results and lists the top performing models for each of the parameter trials.

3.1 Architecture Trials

When using CNNs on earth observation data, we tried to learn the attributes of the target features such as colour and roughness of the class using convolutions or filters. These filters attempt to extract abstract data from the imagery through linear operations using kernels. The number of convolutional filters, kernel size, and the number of overall convolutions or network depth are defined when building the architecture.

Results revealed more complex models with more convolution filters, larger kernel size, and a deeper architecture improve the output classification accuracy, but they take longer to train. The optimal number of initial filters when taking training time into account (Appendix 2) was found to be 56 with a kappa score of 0.87 (Table 5). Although a larger number of initial filters slightly improved the model accuracy (Fig. 7), for example 80 filters with an average kappa score of 0.89, it dramatically increased the training time over 20 epochs (56 filters: 5.2 h; 80 filters: 8.7 h) (Appendix 2). The original U-Net architecture contained 64 filters which achieved a kappa score of 0.86 and took an average of 0.9 h longer to train over 20 epochs.

For our data, we found a kernel size of 5×5 to be optimal (Table 5) with a kappa score of 0.87. Increasing the complexity of the filters beyond 5×5 did not have a substantial effect on accuracy (Fig. 8) but substantially increased training time from 3.9 h for a 5×5 kernel to 9 h for a 9×9 kernel (Appendix 2). The original U-Net architecture contained a kernel size of 3×3 and for our data achieved a kappa score

Table 5 Top-performing results of the parameter trials based on the 2018 training image

	Parameter	Top ranking parameter	Kappa	Jaccard Index	$F1$-Score	User accuracy (precision)	Producer accuracy (recall)
Architecture trials	Number of initial filters	56	0.87 (0.84–0.89)	0.79 (73.69–85.15)	0.86 (0.81–0.91)	0.84 (0.79–0.89)	0.92 (0.9–0.94)
	Kernel size	5×5	0.83 (0.8–0.86)	0.72 (70.07–75.01)	0.81 (0.79–0.84)	0.76 (0.73–0.81)	0.91 (0.89–0.92)
	Number of layers	5	0.76 (0.62–0.84)	0.67 (64.02–70.5)	0.77 (0.73–0.79)	0.74 (0.69–0.79)	0.88 (0.85–0.89)
Algorithmic trials	Kernel matrix weights initialiser	Truncated normal	0.8 (0.77–0.83)	0.67 (65.4–69.06)	0.77 (0.75–0.78)	0.7 (0.68–0.73)	0.91 (0.9–0.91)
	Activation function	Exponential Linear Unit	0.79 (0.76–0.82)	0.64 (60.64–67.48)	0.75 (0.73–0.78)	0.68 (0.65–0.71)	0.89 (0.87–0.91)
	Loss function	Huber	0.8 (0.78–0.82)	0.65 (62.47–67.24)	0.76 (0.73–0.78)	0.7 (0.67–0.73)	0.88 (0.85–0.91)
	Optimiser	RMSprop	0.77 (0.65–0.86)	0.64 (52.03–73.07)	0.73 (0.6–0.82)	0.68 (0.55–0.79)	0.85 (0.73–0.9)
	Learning rate	1e-4	0.9 (0.88–0.92)	0.8 (76.12–84.99)	0.88 (0.85–0.91)	0.85 (0.79–0.89)	0.94 (0.91–0.96)
	Class activation function	Softmax	0.76 (0.74–0.78)	0.6 (55.53–64.35)	0.7 (0.67–0.74)	0.64 (0.59–0.69)	0.88 (0.83–0.9)

The number in brackets represents the 95% confidence intervals

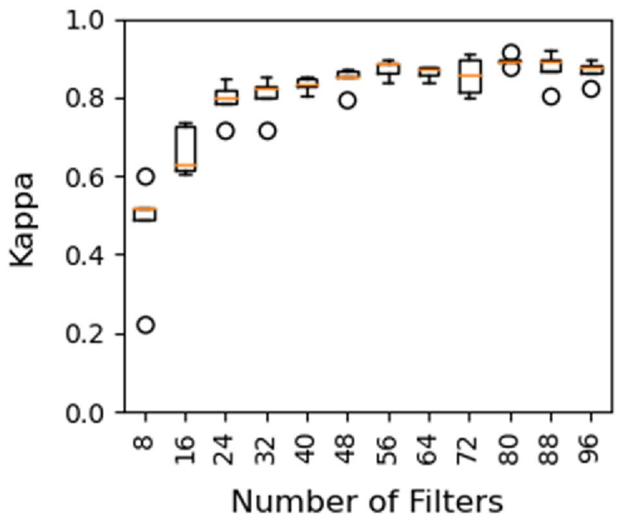


Fig. 7 Kappa statistic box and whisker plots for the initial number of filter trials. Each filter number includes the accuracy for each of the five trained models

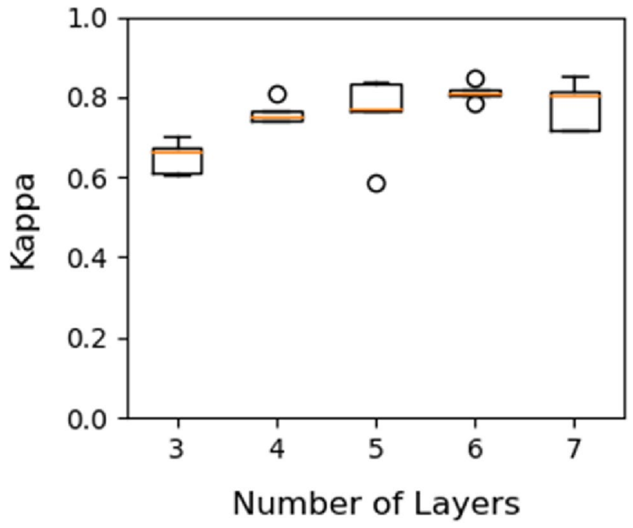


Fig. 9 Kappa statistic box and whisker plots for U-Net architecture depth trials. Each layer depth includes the accuracy for each of the five trained models

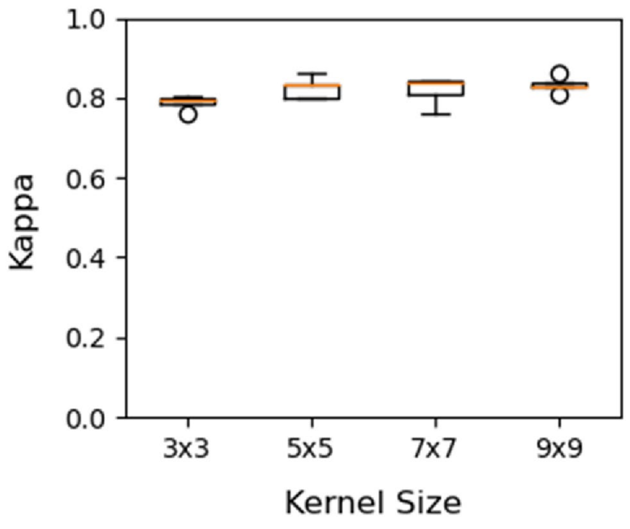


Fig. 8 Kappa statistic box and whisker plots for kernel size trials. Each kernel size includes the accuracy for each of the five trained models

of 0.79. As a result, we recommend the use of a 5×5 kernel size for earth observation data.

Experiments varying the number of layers or architecture depth indicated that the original U-Net architecture consisting of five layers was optimal. There were slight accuracy gains by increasing the depth of the architecture to six or seven layers (Fig. 9); however, these were not substantial (Appendix 2) as well as increasing training time.

3.2 Algorithmic Trials

Within CNNs, we are attempting to fit an algorithm to complex non-linear data by setting the optimal weight values to thousands or millions of neurons. The weight values determine if the neuron should activate and pass the data to the next, with the magnitude of the output calculated using an activation function. The output prediction was compared to the labelled data provided by measuring the classification error using a loss function. The optimisation function was used to alter the weight values through backward propagation with the learning rate determining the extent the weight values can be changed.

The initial values of the kernel weights, the core of the convolution filters, are determined by an initialiser function. For the kernel initialiser trials, our results showed the truncated normal function achieved a slightly higher accuracy score in all metrics (Appendix 2). However, many initialisers produced similar results with only he normal, he uniform, and random normal performing poorly (Fig. 10).

For the activation function trials, our results showed that there were numerous functions which achieved similar accuracies and, as a result, the same rank using the user, producer, and time-based score (Appendix 2). Analysing further using the kappa statistic, results shown graphically in Fig. 11 indicated the scaled exponential linear unit (selu) had the highest accuracy. However, this function was penalised as it had an average time slightly higher than the top ranked results (Appendix 2).

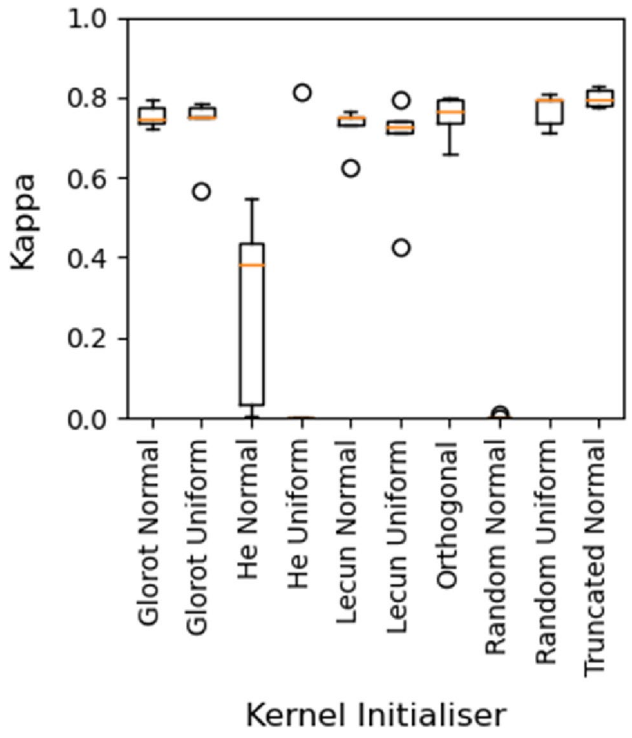


Fig. 10 Kappa statistic box and whisker plots for the kernel weights initialiser trials. Each kernel initialiser includes the accuracy for each of the five trained models

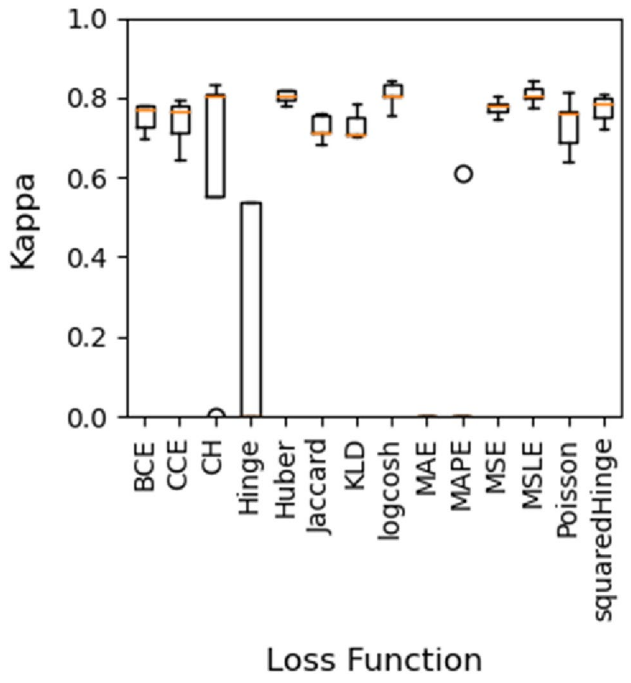


Fig. 12 Kappa statistic box and whisker plots for the loss function trials. Each loss function includes the accuracy for each of the five trained models

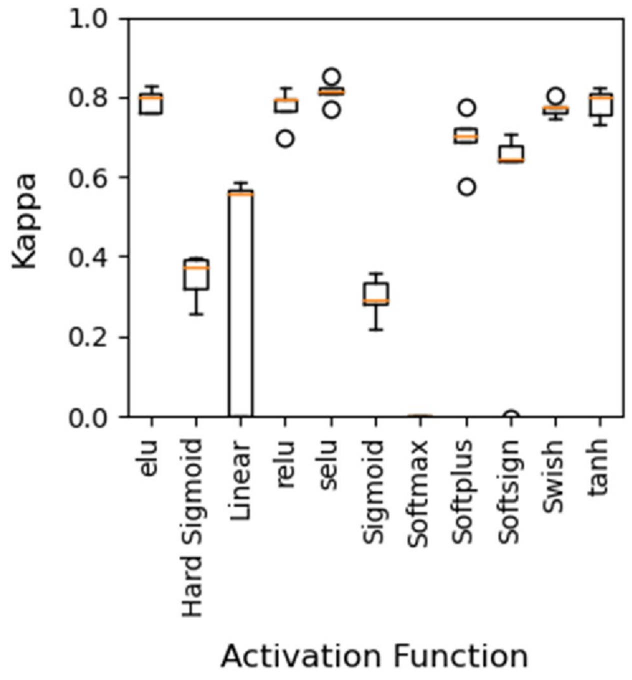


Fig. 11 Kappa statistic box and whisker plots for the activation function trials. Each activation function includes the accuracy for each of the five trained models

Based on these results, there are numerous activation functions which can successfully train a deep learning model for earth observation applications and, as time was quite similar between the top performing models, we recommend the use of selu.

Similar to the kernel matrix weights initialiser, many loss functions performed well as shown in our data (Appendix 2). Our results found the Huber loss function (kappa: 0.8) marginally outperformed the other functions, but there are many potential options (Fig. 12). The functions which resulted in the lowest accuracy included hinge, categorical hinge, MAE, and MAPE.

The RMSprop optimiser was the top ranked and had an average kappa score of 0.77. Adam (kappa: 0.79) and NAdam (kappa: 0.77) also performed similarly to RMSProp (Fig. 13) but took slightly longer to train (Appendix 2). Any of these three optimisers could be used for earth observation applications, but we recommend the RMSprop optimiser.

The learning rate was a substantial factor in the model's ability to learn the training data (Fig. 14; Appendix 2). We found a learning rate of 1e-4 to be optimal and choosing an

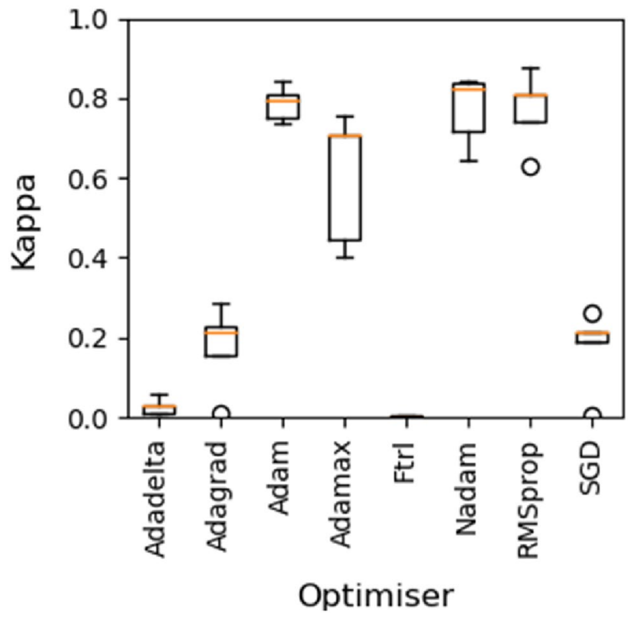


Fig. 13 Kappa statistic box and whisker plots for the loss optimiser trials. Each optimiser includes the accuracy for each of the five trained models

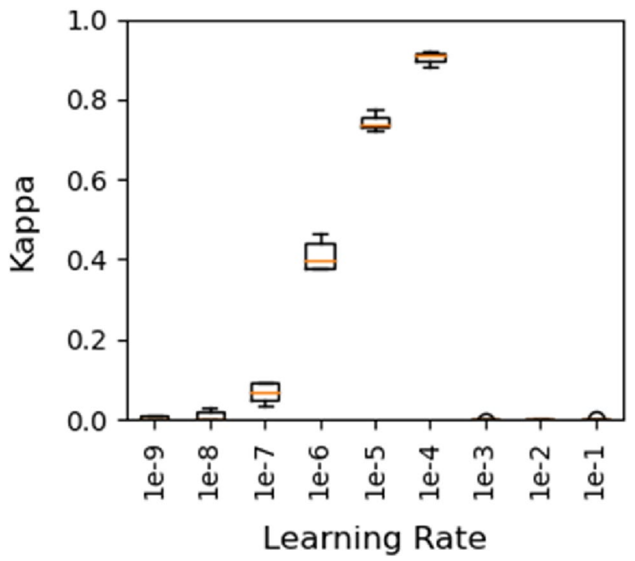


Fig. 14 Kappa statistic box and whisker plots for the learning rate trials. Each learning rate includes the accuracy for each of the five trained models

incorrect learning rate resulted in some trials failing to train. Learning rates larger than 1e-4 result in too large a change in the weights to find the optimal values, whereas reduced

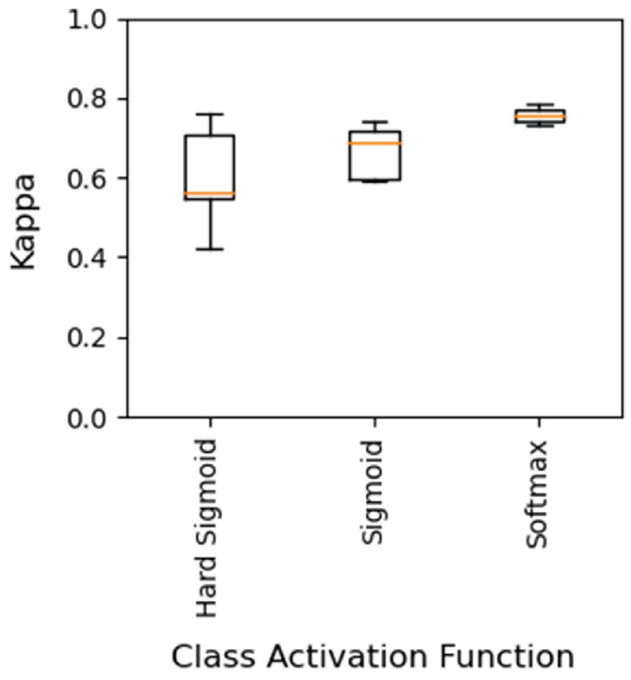


Fig. 15 Kappa statistic box and whisker plots for the class activation function trials. Each class activation includes the accuracy for each of the five trained models

learning rates resulted in slower learning and, as a result, a lower accuracy after 20 epochs of training.

As our data consisted of multiple classes, the softmax class activation function was best suited and recommended for similar earth observation applications (Fig. 15; Appendix 2). However, for binary classifications, future studies should consider sigmoid or hard sigmoid functions.

3.3 Optimised U-Net and Original U-Net

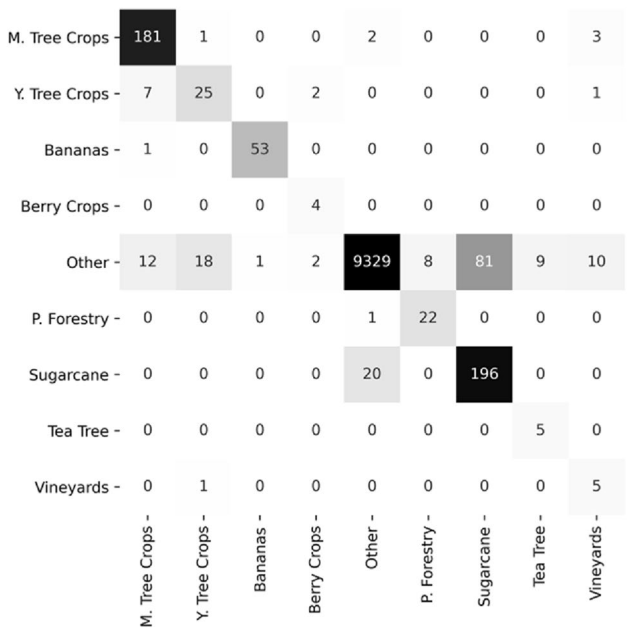
Using the top performing trials, we compared an optimised U-Net architecture against the original U-Net architecture, and found the optimised version achieved a higher accuracy (kappa: + 0.12; $F1$-Score: + 0.08; user: + 0.06; producer: + 0.04). Table 6 shows the overall and class results for the two models along with a human-derived classification for the same image. Tables 7 and 8 show the confusion matrix for the optimised model and original model, respectively.

For the optimised model, there were substantial improvements in some individual classes (Table 6) such as plantation forestry (user: + 0.63), berry crops (producer: + 0.25) and banana crops (producer: + 0.19). However, not all classes improved with user accuracy for vineyards and tea tree plantations decreasing by 0.12 and 0.06, respectively.

Table 6 Optimised model results for 2015 test image

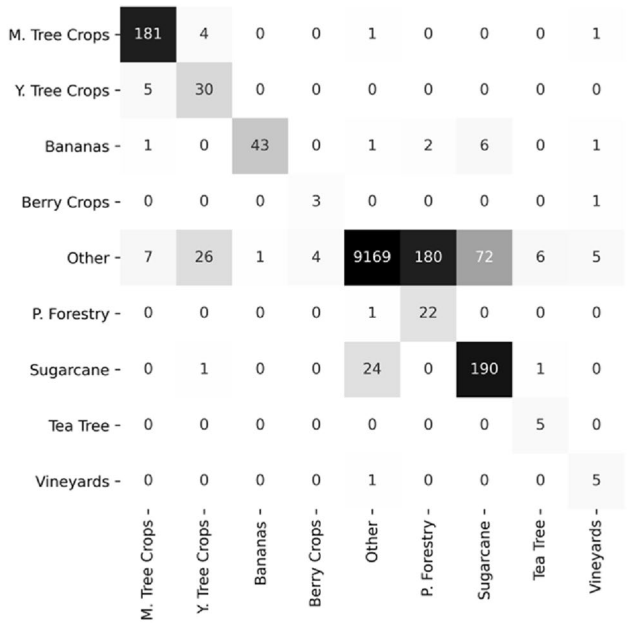
Class	Optimised U-Net				Original U-Net				Human			
	Jaccard Index	*F*1-Score	User (precision)	Recall (producer)	Jaccard Index	*F*1-Score	User (precision)	Producer (recall)	Jaccard Index	*F*1-Score	User (precision)	Producer (recall)
Banana plantations	0.964	0.981	0.981	0.981	0.782	0.878	0.977	0.796	0.982	0.991	0.982	1.000
Berry crops	0.500	0.667	0.500	1.000	0.375	0.545	0.429	0.750	0.750	0.857	1.000	0.750
Plantation forestry	0.710	0.830	0.733	0.957	0.107	0.194	0.108	0.957	0.783	0.878	1.000	0.783
sugarcane crops	0.660	0.795	0.708	0.907	0.646	0.785	0.709	0.880	0.853	0.920	0.995	0.857
Tea tree plantations	0.357	0.526	0.357	1.000	0.417	0.588	0.417	1.000	1.000	1.000	1.000	1.000
Mature tree fruits	0.874	0.933	0.900	0.968	0.905	0.950	0.933	0.968	0.887	0.940	0.914	0.968
Young tree fruits	0.455	0.625	0.556	0.714	0.455	0.625	0.492	0.857	0.526	0.690	0.870	0.571
Vineyards	0.250	0.400	0.263	0.833	0.357	0.526	0.385	0.833	1.000	1.000	1.000	1.000
Other	0.983	0.991	0.998	0.985	0.965	0.982	0.997	0.968	0.994	0.997	0.995	0.999
Total	0.639	0.750	0.666	0.927	0.560	0.675	0.605	0.890	0.864	0.919	0.973	0.881
Kappa	0.841				0.725				0.929			

Table 7 Optimised U-Net model confusion matrix



	M. Tree Crops	Y. Tree Crops	Bananas	Berry Crops	Other	P. Forestry	Sugarcane	Tea Tree	Vineyards
M. Tree Crops	181	1	0	0	2	0	0	0	3
Y. Tree Crops	7	25	0	2	0	0	0	0	1
Bananas	1	0	53	0	0	0	0	0	0
Berry Crops	0	0	0	4	0	0	0	0	0
Other	12	18	1	2	9329	8	81	9	10
P. Forestry	0	0	0	0	1	22	0	0	0
Sugarcane	0	0	0	0	20	0	196	0	0
Tea Tree	0	0	0	0	0	0	0	5	0
Vineyards	0	1	0	0	0	0	0	0	5

Table 8 Original U-Net model confusion matrix



	M. Tree Crops	Y. Tree Crops	Bananas	Berry Crops	Other	P. Forestry	Sugarcane	Tea Tree	Vineyards
M. Tree Crops	181	4	0	0	1	0	0	0	1
Y. Tree Crops	5	30	0	0	0	0	0	0	0
Bananas	1	0	43	0	1	2	6	0	1
Berry Crops	0	0	0	3	0	0	0	0	1
Other	7	26	1	4	9169	180	72	6	5
P. Forestry	0	0	0	0	1	22	0	0	0
Sugarcane	0	1	0	0	24	0	190	1	0
Tea Tree	0	0	0	0	0	0	0	5	0
Vineyards	0	0	0	0	1	0	0	0	5

In some cases, the U-Net models were more accurate than the human-derived classification (mature tree crops) which indicates that CNN models can account for some level of error in the training data to a certain extent (Mnih 2013). Other studies such as (Burke et al. 2021) have noted that these errors can potentially result in the model result being

Table 9 Accuracy assessment for the 2018 human-derived training data used to train all models

Class	Jaccard Index	$F1$-Score	User (precision)	Producer (recall)
Banana plantations	1.000	1.000	1.000	1.000
Berry crops	1.000	1.000	1.000	1.000
Plantation forestry	0.724	0.840	0.955	0.750
Sugarcane crops	0.968	0.984	1.000	0.968
Tea tree plantation	1.000	1.000	1.000	1.000
Tree crop—mature	0.923	0.960	0.936	0.986
Tree crop—young	0.575	0.730	0.871	0.628
Vineyards	1.000	1.000	1.000	1.000
Other	0.996	0.998	0.997	0.999
Total		0.946	0.9732	0.9256
Kappa		0.962		

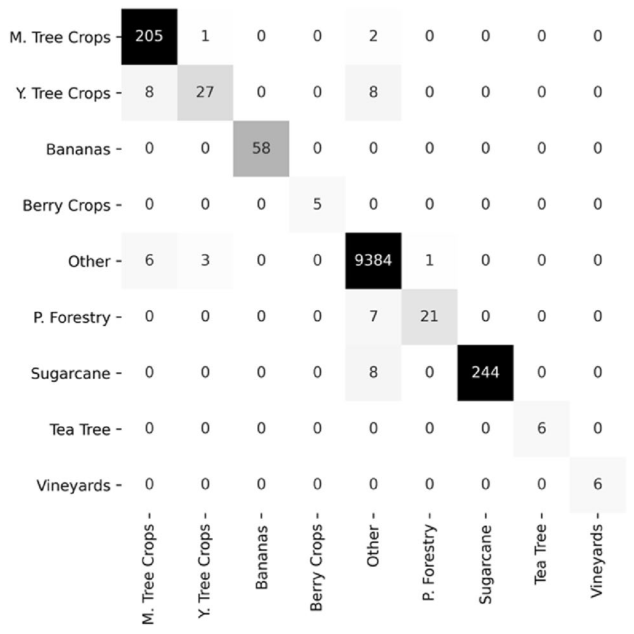
penalised for achieving a higher accuracy than data used to assess its performance.

Despite the changes in overall accuracies, the confusion matrices (Tables 7 and 8) showed similar confusion in the two models. The main confusion was between mature and young tree crops and misclassification of targeted classes with the other uses class, particularly for sugarcane and forestry plantation.

Although some studies found deep neural networks are quite tolerant of noise in the training data (Mnih 2013), we found confusion in the training data propagated through to the model's ability to classify classes. Table 9 shows the accuracy assessment for the 2018 human-derived training data and Table 10 shows the corresponding confusion matrix. The tables demonstrate the confusion between young tree and mature tree crops and with various classes and other uses classes, such as sugarcane crops, young tree crops, and plantation forestry. This demonstrates that collection of some LULC classes for both training and evaluation can be subjective, for example, the threshold between young and mature tree crops, and at what stage can a newly planted sugar crop be detected as such within the image resolution.

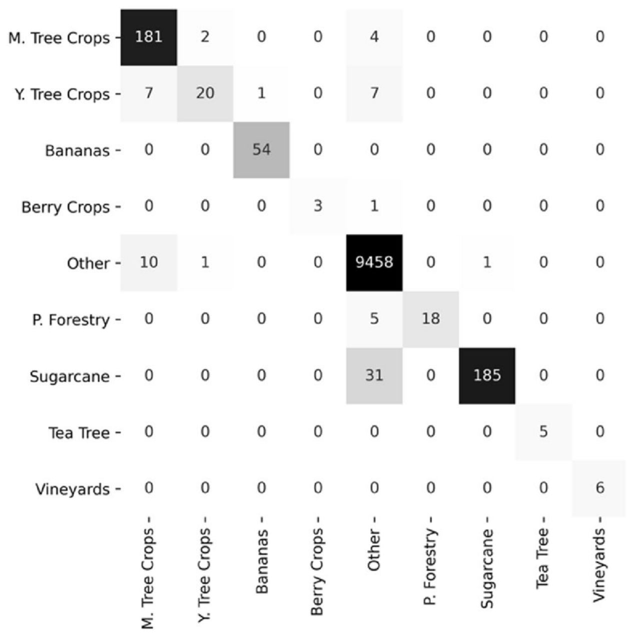
As a comparison, Table 11 presents the confusion matrix for the 2015 human-derived classification and demonstrated

Table 10 Confusion matrix for 2018 human-derived training data



	M. Tree Crops	Y. Tree Crops	Bananas	Berry Crops	Other	P. Forestry	Sugarcane	Tea Tree	Vineyards
M. Tree Crops	205	1	0	0	2	0	0	0	0
Y. Tree Crops	8	27	0	0	8	0	0	0	0
Bananas	0	0	58	0	0	0	0	0	0
Berry Crops	0	0	0	5	0	0	0	0	0
Other	6	3	0	0	9384	1	0	0	0
P. Forestry	0	0	0	0	7	21	0	0	0
Sugarcane	0	0	0	0	8	0	244	0	0
Tea Tree	0	0	0	0	0	0	0	6	0
Vineyards	0	0	0	0	0	0	0	0	6

Table 11 Confusion matrix for 2015 human classification



	M. Tree Crops	Y. Tree Crops	Bananas	Berry Crops	Other	P. Forestry	Sugarcane	Tea Tree	Vineyards
M. Tree Crops	181	2	0	0	4	0	0	0	0
Y. Tree Crops	7	20	1	0	7	0	0	0	0
Bananas	0	0	54	0	0	0	0	0	0
Berry Crops	0	0	0	3	1	0	0	0	0
Other	10	1	0	0	9458	0	1	0	0
P. Forestry	0	0	0	0	5	18	0	0	0
Sugarcane	0	0	0	0	31	0	185	0	0
Tea Tree	0	0	0	0	0	0	0	5	0
Vineyards	0	0	0	0	0	0	0	0	6

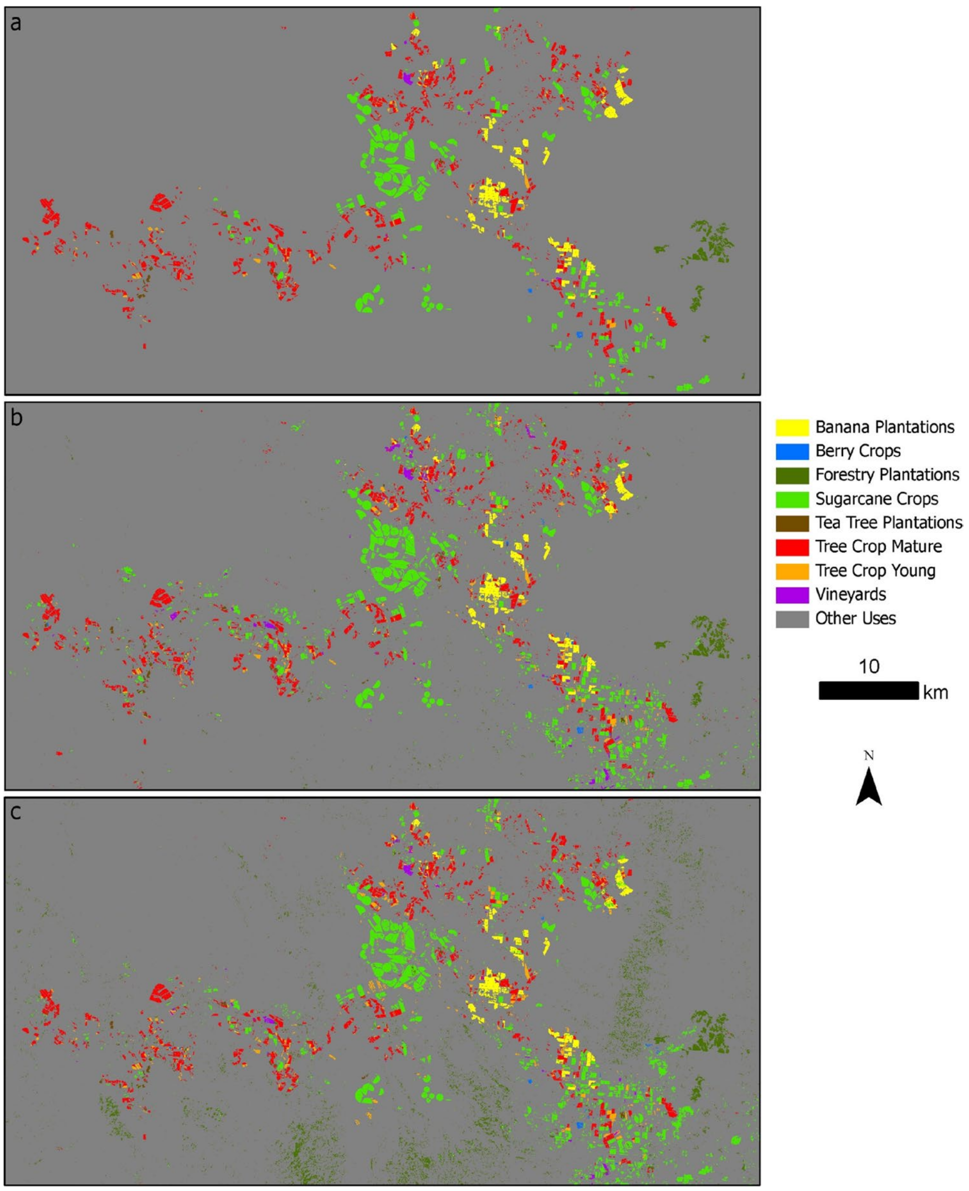


Fig. 16 Model classification comparison for the project area for human classification (**a**), optimised U-Net (**b**), and original U-Net (**c**)

the same confusion existed in the 2015 classification as in the 2018 training data.

Figure 16 shows the project area output classifications for the optimised U-Net (Fig. 16a) and original U-Net (Fig. 16b). At this scale, the confusion between sugarcane crops and other land uses is evident in both classifications. Whereas the confusion between plantation forestry and other land uses can be seen in the original U-Net classification.

Figure 17 shows the outputs for the human-derived classification along with the optimised and original U-Net classifications and visually show the errors discussed. The 2018 training image was captured in August, early in the sugarcane harvest season when fields were either fully mature or in fallow. The 2015 test image was captured as late as October and, in some areas, fields harvested earlier in the year contained young sugarcane. Digitising the data for 2015 was consistent with the 2018 training data and only mature sugarcane was included. However, for the accuracy assessment points, we decided to call the validation point sugarcane if the canopy of the crop was predominantly

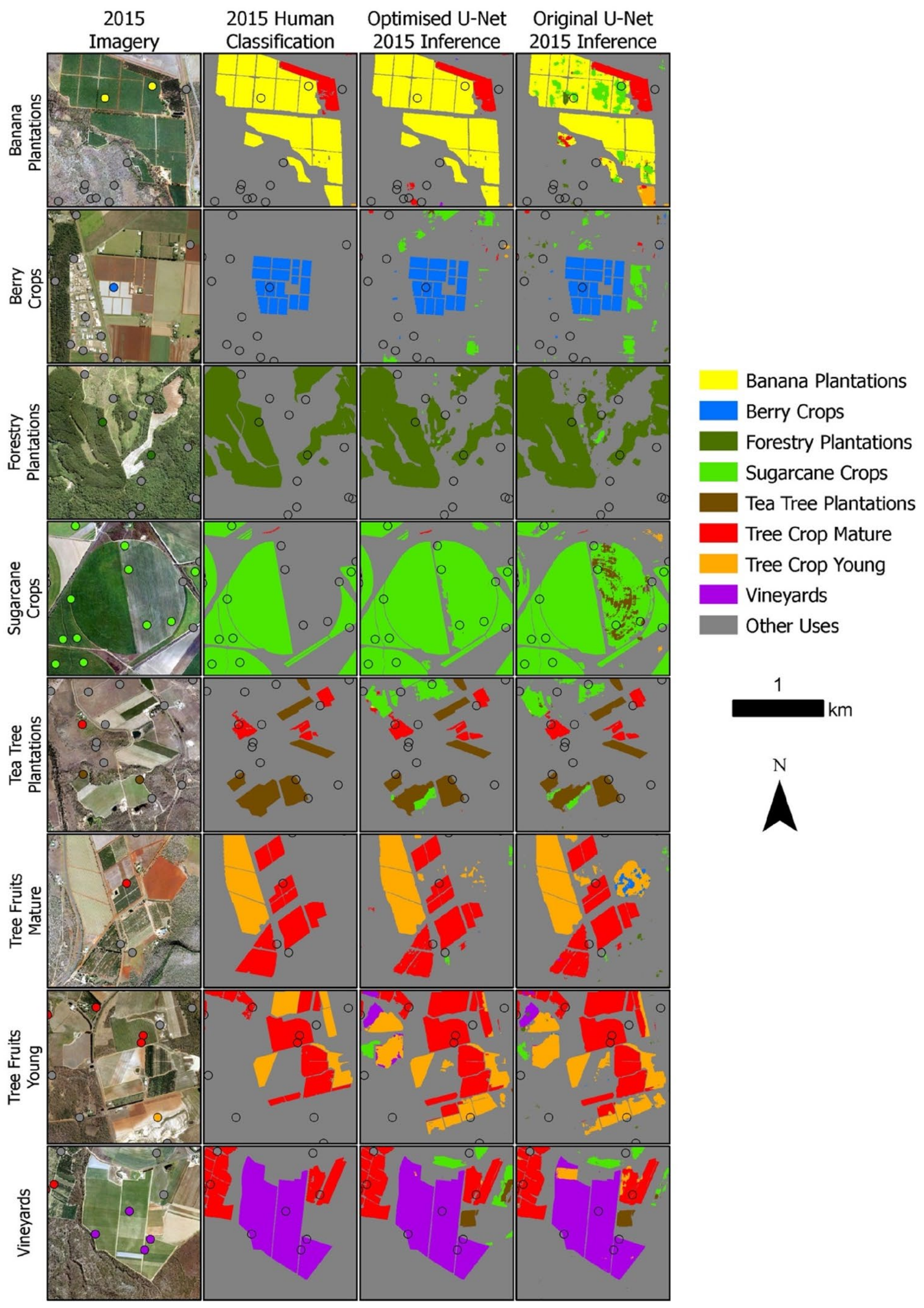


Fig. 17 I2015 test image, 2015 human-derived, optimised U-Net, and Original U-Net classifications. The circles indicate the location of the accuracy assessment points with the colour in the imagery column representing the class according to the legend

closed. Although no fields of young sugarcane were present in the training data, the models successfully classified these areas within the 2015 image (Fig. 17).

Further to this, due to human error, some tree crop features were missed in the human classification but were successfully identified by the models (Fig. 17).

3.4 Limitations and Future Research

All hyper-parameter trials were tested independently to identify the top performing architectural configuration and learning algorithms. However, we did not test for interdependencies between the parameters. Conducting this analysis would take a considerable amount of time and was beyond the scope of this study. Future research could build upon these findings and restrict such an analysis to a smaller sample of parameters. It is recommended to avoid a simple grid search and to use an optimisation framework such as KerasTuner (https://keras.io/keras_tuner/).

This study restricted the analysis to three band 50 cm imagery. These results may not be applicable for training a model for earth observation applications for different spatial and spectral resolutions. Furthermore, this study was conducted over the same geographic area with similar LULC features present in each image. Transferring the models to a different region will present challenges particularly where there are unseen LULC features which will likely confuse the model. Future research should focus on scaling this study up to include a broader geographical area and test the ability of the models to transfer to a different region with varying LULC features.

In addition to the trials, we have conducted in this study, and there are additional hyper-parameters and tests which could be considered for future research. These include (but not limited to) the inclusion of regularisation such as dropouts and batch normalisation layers in the U-Net architecture.

Additional research could investigate different approaches to producing the output classifications. In this study, we have used overlapping patches and a weighted mean to counteract patch edge effects when producing the output classification; however, there are alternatives such as trimming of edge pixels and ensuring the patches overlap by the same number of pixels as presented in Flood et al. (2019).

4 Conclusion

In this paper, we trialled the effectiveness of different parameters for defining and training deep learning models based on the U-Net architecture. We found more complex models containing a larger number of convolutional filters and kernel size achieved higher accuracies. Based on our results, we recommend the use of 56 convolutional filters with a kernel size of 5 × 5 while maintaining the original depth of the U-Net of five layers for earth observation LULC applications at this scale. We also recommend the use of truncated normal kernel weights initialiser, a selu activation function, the Huber loss function, RMSprop loss optimiser, and a learning rate of 1e-4. The combination of these parameters increased the original U-Net model kappa score from 0.73 to 0.84 for the optimised model. The equivalent human-derived classification obtained a kappa score of 0.93, and the subjectivity of classifying some LULC classes caused some confusion and resulted in the propagation of errors through to model classification.

Appendix 1

Abbreviation	Description
BCE	Binary cross entropy
CCE	Categorical cross entropy
CH	Categorical hinge
CNN	Convolutional neural network
DT	Decision tree
ELU	Exponential linear unit
GPU	Graphics processing unit
KLD	Kullback–Leibler divergence
logcosh	Log hyperbolic cosine
LULC	Land use and land cover
MAE	Mean absolute error
MAPE	Mean absolute percentage error
MSE	Mean squared error
MSLE	Mean squared logarithmic error
RELU	Rectified linear unit
RF	Random forest
SELU	Scaled exponential linear unit
SVM	Support vector machine
tanh	Hyperbolic tangent
Adadelta	Adaptive learning rate method
Adagrad	Adaptive gradient algorithm
Adam	Adaptive moment estimation
AdaMax	Adaptive moment estimation (infinity norm)
ftrl	Follow the regularised leader
Nadam	Nesterov-accelerated adaptive moment estimation
RMSprop	Root-mean-squared propagation
SGD	Stochastic gradient descent

Appendix 2

Trial	Parameter	Average training time (hours)	Kappa	Jaccard Index	*F*1-Score	User (precision)	Producer (recall)	Ranking
Number of initial filters	8	2.2	0.47 (0.28–0.58)	0.35 (23.14–41.29)	0.44 (0.29–0.51)	0.4 (0.27–0.46)	0.61 (0.38–0.7)	10
	16	2.1	0.66 (0.61–0.73)	0.51 (42.95–57.0)	0.62 (0.53–0.68)	0.54 (0.45–0.6)	0.87 (0.82–0.89)	8
	24	2.2	0.79 (0.73–0.84)	0.63 (52.75–70.13)	0.74 (0.64–0.79)	0.67 (0.55–0.75)	0.91 (0.89–0.92)	2
	32	2.7	0.8 (0.73–0.85)	0.71 (67.89–74.2)	0.8 (0.76–0.82)	0.76 (0.73–0.8)	0.9 (0.88–0.92)	4
	40	3.7	0.83 (0.81–0.85)	0.71 (66.84–73.56)	0.79 (0.74–0.82)	0.75 (0.69–0.8)	0.87 (0.84–0.92)	9
	48	4.5	0.85 (0.81–0.87)	0.74 (67.31–76.83)	0.83 (0.77–0.85)	0.78 (0.72–0.82)	0.91 (0.9–0.94)	4
	56	5.2	0.87 (0.84–0.89)	0.79 (73.69–85.15)	0.86 (0.81–0.91)	0.84 (0.79–0.89)	0.92 (0.9–0.94)	1
	64	6.1	0.86 (0.84–0.88)	0.75 (70.47–78.14)	0.83 (0.78–0.86)	0.8 (0.74–0.85)	0.9 (0.84–0.94)	10
	72	7.8	0.86 (0.8–0.91)	0.78 (71.19–82.01)	0.85 (0.8–0.89)	0.82 (0.75–0.87)	0.93 (0.91–0.95)	7
	80	8.7	0.89 (0.88–0.91)	0.8 (78.91–82.34)	0.88 (0.87–0.89)	0.84 (0.83–0.86)	0.93 (0.92–0.94)	3
	88	10	0.88 (0.82–0.92)	0.78 (72.35–82.75)	0.86 (0.82–0.89)	0.82 (0.77–0.88)	0.94 (0.93–0.94)	6
	96	11.5	0.87 (0.83–0.89)	0.78 (76.06–80.39)	0.86 (0.84–0.88)	0.82 (0.79–0.83)	0.93 (0.91–0.96)	10
Kernel size	3×3	2.8	0.79 (0.76–0.8)	0.6 (56.54–64.45)	0.7 (0.65–0.76)	0.66 (0.6–0.7)	0.83 (0.74–0.88)	2
	5×5	3.9	0.83 (0.8–0.86)	0.72 (70.07–75.01)	0.81 (0.79–0.84)	0.76 (0.73–0.81)	0.91 (0.89–0.92)	1
	7×7	6.6	0.82 (0.77–0.84)	0.71 (69.08–74.02)	0.81 (0.79–0.83)	0.76 (0.72–0.78)	0.91 (0.89–0.93)	4
	9×9	9	0.83 (0.81–0.86)	0.73 (70.01–76.41)	0.83 (0.8–0.85)	0.77 (0.73–0.8)	0.92 (0.91–0.92)	2
Number of layers	3	5	0.65 (0.61–0.69)	0.51 (48.02–54.22)	0.62 (0.59–0.65)	0.59 (0.53–0.64)	0.77 (0.76–0.79)	2
	4	5.3	0.76 (0.74–0.8)	0.62 (57.76–67.83)	0.72 (0.69–0.77)	0.7 (0.64–0.79)	0.83 (0.79–0.86)	5
	5	5.5	0.76 (0.62–0.84)	0.67 (64.02–70.5)	0.77 (0.73–0.79)	0.74 (0.69–0.79)	0.88 (0.85–0.89)	1
	6	5.7	0.81 (0.79–0.84)	0.67 (61.27–72.07)	0.77 (0.74–0.81)	0.74 (0.69–0.78)	0.85 (0.83–0.89)	3
	7	6.4	0.78 (0.72–0.84)	0.69 (59.25–74.8)	0.77 (0.68–0.83)	0.76 (0.65–0.83)	0.86 (0.84–0.88)	3
Kernel Matrix Weights Initialiser	Glorot Normal	2.7	0.76 (0.72–0.79)	0.61 (57.97–64.21)	0.72 (0.69–0.75)	0.64 (0.61–0.68)	0.89 (0.88–0.9)	3
	Glorot Uniform	2.8	0.73 (0.61–0.78)	0.6 (57.55–63.99)	0.71 (0.69–0.75)	0.64 (0.61–0.69)	0.89 (0.86–0.92)	5
	He Normal	2.7	0.28 (0.01–0.53)	0.24 (6.75–33.01)	0.28 (0.09–0.39)	0.25 (0.07–0.34)	0.43 (0.24–0.55)	8
	He Uniform	2.7	0.16 (0.0–0.65)	0.18 (10.42–39.78)	0.19 (0.11–0.45)	0.19 (0.1–0.43)	0.2 (0.11–0.48)	9
	Lecun Normal	2.7	0.72 (0.65–0.76)	0.6 (54.17–63.68)	0.7 (0.64–0.74)	0.64 (0.56–0.68)	0.87 (0.82–0.9)	4
	Lecun Uniform	2.7	0.68 (0.48–0.78)	0.49 (28.92–62.18)	0.59 (0.34–0.73)	0.52 (0.31–0.67)	0.75 (0.47–0.89)	7
	Orthogonal	2.7	0.75 (0.67–0.8)	0.58 (53.6–63.8)	0.68 (0.61–0.74)	0.61 (0.56–0.69)	0.82 (0.7–0.92)	6
	Random Normal	2.7	0.0 (-0.0–0.01)	0.1 (10.42–10.52)	0.11 (0.11–0.11)	0.11 (0.1–0.13)	0.11 (0.11–0.11)	10
	Random Uniform	2.7	0.77 (0.72–0.81)	0.62 (58.53–64.84)	0.72 (0.7–0.75)	0.65 (0.62–0.7)	0.9 (0.88–0.92)	2
	Truncated Normal	2.7	0.8 (0.77–0.83)	0.67 (65.4–69.06)	0.77 (0.75–0.78)	0.7 (0.68–0.73)	0.91 (0.9–0.91)	1

Trial	Parameter	Average training time (hours)	Kappa	Jaccard Index	$F1$-Score	User (precision)	Producer (recall)	Ranking
Activation function	Exponential Linear Unit	2.7	0.79 (0.76–0.82)	0.64 (60.64–67.48)	0.75 (0.73–0.78)	0.68 (0.65–0.71)	0.89 (0.87–0.91)	1
	Hard Sigmoid	3.1	0.35 (0.27–0.4)	0.15 (13.82–16.22)	0.18 (0.17–0.19)	0.17 (0.15–0.18)	0.22 (0.19–0.27)	10
	Linear	2.6	0.34 (0.0–0.58)	0.31 (10.44–46.78)	0.38 (0.11–0.58)	0.33 (0.1–0.5)	0.52 (0.11–0.85)	5
	Rectified Linear Unit	2.7	0.78 (0.71–0.82)	0.62 (55.71–67.29)	0.73 (0.68–0.78)	0.66 (0.59–0.71)	0.89 (0.87–0.9)	1
	Scaled Exponential Linear Unit	2.8	0.81 (0.78–0.85)	0.69 (65.92–70.94)	0.79 (0.76–0.81)	0.72 (0.69–0.74)	0.91 (0.89–0.93)	4
	Sigmoid	2.7	0.3 (0.23–0.35)	0.15 (12.04–16.18)	0.18 (0.14–0.2)	0.16 (0.14–0.18)	0.25 (0.21–0.29)	8
	Softmax	3.7	0.0 (0.0–0.0)	0.1 (10.44–10.44)	0.11 (0.11–0.11)	0.1 (0.1–0.1)	0.11 (0.11–0.11)	11
	Softplus	2.8	0.69 (0.6–0.77)	0.53 (39.85–63.64)	0.65 (0.51–0.75)	0.58 (0.43–0.7)	0.85 (0.77–0.88)	7
	Softsign	2.8	0.53 (0.13–0.7)	0.47 (17.55–60.89)	0.55 (0.21–0.71)	0.51 (0.2–0.66)	0.72 (0.27–0.89)	9
	Swish	3.1	0.77 (0.75–0.8)	0.62 (58.28–66.61)	0.73 (0.7–0.77)	0.65 (0.61–0.7)	0.89 (0.88–0.91)	6
	Hyperbolic tangent	2.7	0.79 (0.74–0.82)	0.61 (58.84–66.61)	0.72 (0.7–0.77)	0.65 (0.62–0.7)	0.9 (0.88–0.92)	1
Loss function	BCE	2.8	0.75 (0.7–0.78)	0.58 (53.42–60.73)	0.69 (0.65–0.72)	0.61 (0.57–0.64)	0.89 (0.87–0.91)	2
	CCE	2.7	0.74 (0.66–0.79)	0.59 (54.4–64.96)	0.7 (0.66–0.75)	0.63 (0.58–0.69)	0.88 (0.87–0.9)	5
	Categorical Hinge	2.8	0.6 (0.11–0.83)	0.41 (13.49–60.01)	0.48 (0.14–0.71)	0.44 (0.14–0.65)	0.59 (0.17–0.84)	11
	Hinge	2.7	0.22 (0.0–0.54)	0.16 (10.44–24.59)	0.17 (0.11–0.29)	0.17 (0.1–0.31)	0.21 (0.11–0.38)	12
	Huber	2.6	0.8 (0.78–0.82)	0.65 (62.47–67.24)	0.76 (0.73–0.78)	0.7 (0.67–0.73)	0.88 (0.85–0.91)	1
	Jaccard Loss	2.7	0.72 (0.69–0.76)	0.58 (53.57–61.66)	0.69 (0.65–0.73)	0.61 (0.56–0.66)	0.89 (0.89–0.9)	2
	KLD	2.8	0.73 (0.7–0.78)	0.58 (53.37–61.85)	0.69 (0.65–0.73)	0.61 (0.56–0.67)	0.88 (0.86–0.9)	5
	logcosh	2.7	0.81 (0.77–0.84)	0.6 (52.09–66.54)	0.7 (0.62–0.78)	0.64 (0.55–0.71)	0.85 (0.78–0.89)	7
	MAE	2.7	0.0 (0.0–0.0)	0.1 (10.44–10.44)	0.11 (0.11–0.11)	0.1 (0.1–0.1)	0.11 (0.11–0.11)	14
	MAPE	2.7	0.12 (0.0–0.49)	0.16 (10.44–32.95)	0.18 (0.11–0.38)	0.18 (0.1–0.4)	0.19 (0.11–0.43)	12
	MSE	2.8	0.78 (0.75–0.8)	0.55 (38.58–63.88)	0.66 (0.47–0.74)	0.62 (0.49–0.69)	0.82 (0.66–0.88)	8
	MSLE	2.8	0.81 (0.78–0.84)	0.61 (54.48–65.25)	0.71 (0.65–0.76)	0.65 (0.58–0.7)	0.86 (0.81–0.89)	2
	Poisson	2.8	0.73 (0.65–0.8)	0.56 (49.54–59.99)	0.67 (0.61–0.71)	0.59 (0.52–0.64)	0.88 (0.86–0.89)	8
	Squared Hinge	2.7	0.77 (0.73–0.81)	0.5 (42.49–57.42)	0.6 (0.53–0.68)	0.53 (0.45–0.61)	0.79 (0.7–0.87)	10
Optimiser	Adadelta	2	0.02 (-0.01–0.05)	0.07 (5.51–8.68)	0.09 (0.08–0.11)	0.11 (0.11–0.12)	0.12 (0.08–0.15)	7
	Adagrad	2	0.18 (0.04–0.27)	0.15 (3.81–20.77)	0.21 (0.05–0.28)	0.18 (0.05–0.24)	0.44 (0.29–0.54)	5
	Adam	2	0.79 (0.74–0.84)	0.63 (51.7–69.67)	0.73 (0.61–0.79)	0.67 (0.55–0.75)	0.86 (0.72–0.93)	2
	Adamax	2	0.6 (0.41–0.74)	0.44 (22.61–56.7)	0.53 (0.27–0.68)	0.47 (0.23–0.63)	0.68 (0.37–0.83)	4
	Ftrl	2	0.0 (-0.0–0.0)	0.08 (2.27–10.43)	0.09 (0.03–0.11)	0.09 (0.02–0.11)	0.12 (0.11–0.16)	8
	Nadam	2.1	0.77 (0.66–0.84)	0.64 (45.55–71.55)	0.72 (0.52–0.82)	0.68 (0.49–0.77)	0.82 (0.57–0.91)	3
	RMSprop	2	0.77 (0.65–0.86)	0.64 (52.03–73.07)	0.73 (0.6–0.82)	0.68 (0.55–0.79)	0.85 (0.73–0.9)	1
	SGD	2	0.18 (0.04–0.25)	0.13 (3.44–17.87)	0.18 (0.05–0.25)	0.16 (0.04–0.21)	0.46 (0.32–0.52)	5

Trial	Parameter	Average training time (hours)	Kappa	Jaccard Index	*F*1-Score	User (precision)	Producer (recall)	Ranking
Learning rate	1.e-01	2.7	0.0 (0.0–0.0)	0.04 (0.04–10.44)	0.04 (0.0–0.11)	0.04 (0.0–0.1)	0.11 (0.11–0.11)	6
	1.e-02	2.7	0.0 (0.0–0.0)	0.04 (0.02–10.44)	0.04 (0.0–0.11)	0.04 (0.0–0.1)	0.11 (0.11–0.11)	9
	1.e-03	2.7	-0.0 (-0.0–0.0)	0.1 (10.42–10.44)	0.11 (0.11–0.11)	0.1 (0.1–0.1)	0.11 (0.11–0.11)	6
	1.e-04	2.8	0.9 (0.88–0.92)	0.8 (76.12–84.99)	0.88 (0.85–0.91)	0.85 (0.79–0.89)	0.94 (0.91–0.96)	1
	1.e-05	2.8	0.74 (0.73–0.77)	0.6 (55.4–63.38)	0.71 (0.67–0.73)	0.64 (0.58–0.67)	0.88 (0.87–0.89)	2
	1.e-06	2.8	0.41 (0.38–0.46)	0.33 (27.88–37.74)	0.43 (0.36–0.48)	0.37 (0.32–0.42)	0.68 (0.6–0.77)	3
	1.e-07	2.7	0.07 (0.04–0.09)	0.09 (7.01–11.41)	0.13 (0.1–0.16)	0.14 (0.12–0.16)	0.2 (0.15–0.23)	4
	1.e-08	2.8	0.01 (-0.0–0.03)	0.05 (0.29–9.56)	0.06 (0.01–0.11)	0.11 (0.11–0.13)	0.11 (0.11–0.11)	5
	1.e-09	2.8	0.0 (-0.0–0.01)	0.03 (0.02–8.93)	0.03 (0.0–0.1)	0.11 (0.11–0.11)	0.11 (0.09–0.12)	6
Class activation function	Hard Sigmoid	2.7	0.6 (0.45–0.75)	0.44 (32.62–53.33)	0.53 (0.4–0.64)	0.51 (0.4–0.61)	0.65 (0.49–0.84)	3
	Sigmoid	2.7	0.67 (0.59–0.74)	0.56 (50.3–64.45)	0.66 (0.59–0.75)	0.6 (0.53–0.68)	0.84 (0.65–0.91)	2
	Softmax	2.7	0.76 (0.74–0.78)	0.6 (55.53–64.35)	0.7 (0.67–0.74)	0.64 (0.59–0.69)	0.88 (0.83–0.9)	1

Acknowledgements The authors would like to thank the Queensland Government for their support in this research especially for providing the aerial imagery and the computing infrastructure which was used to process the data.

Author Contributions Formal analysis, A.C.; investigation, A.C.; data curation, A.C.; writing–original draft preparation, A.C.; writing–review and editing, A.C., S.P., P.S.; visualization, A.C.; supervision, S.P. and P.S.; project administration, A.C.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions.

Availability of Data and Materials The aerial imagery data used to support the findings of this study were supplied by the Queensland Government under license and cannot be made freely available. Requests for access to these data should be made to SIProductDelivery@resources.qld.gov.au.

Declarations

Conflict of Interest The authors declare no conflict of interest.

References

Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado GS, Davis A, Dean J, Devin M, Ghemawat S, Goodfellow I, Harp A, Irving G, Isard M, Jia Y, Jozefowicz R, Kaiser L, Kudlur M, Zheng X (2016) TensorFlow: Large-scale machine learning on heterogeneous distributed systems. arxiv:1603.04467. http://arxiv.org/abs/1603.04467

Almabdy S, Elrefaei L (2019) Deep convolutional neural network-based approaches for face recognition. Appl Sci 9(20):4397. https://doi.org/10.3390/app9204397

Ball JE, Anderson DT, Chan CS (2017) Comprehensive survey of deep learning in remote sensing: theories, tools, and challenges for the community. J Appl Remote Sens 11(04):1. https://doi.org/10.1117/1.JRS.11.042609

Blaschke T, Kelly M, Merschdorf H (2015) Object-based image analysis: evolution, history, state of the art, and future vision. In: Ch M (ed) Remotely sensed data characterization, classification, and accuracies. UC Berkeley

Burke M, Driscoll A, Lobell DB, Ermon S (2021) Using satellite imagery to understand and promote sustainable development. Science 371(6535):eabe8628. https://doi.org/10.1126/science.abe8628

Castelluccio M, Poggi G, Sansone C, Verdoliva L (2015) Land use classification in remote sensing images by convolutional neural networks. ArXiv:1508.00092. http://arxiv.org/abs/1508.00092

Cheng G, Han J, Guo L, Liu Z, Bu S, Ren J (2015) Effective and efficient midlevel visual elements-oriented land-use classification using VHR remote sensing images. IEEE Trans Geosci Remote Sens 53(8):4238–4249. https://doi.org/10.1109/TGRS.2015.2393857

Clark A, McKechnie J (2020) Detecting banana plantations in the wet tropics, australia, using aerial photography and U-Net. Appl Sci 10(6):2017. https://doi.org/10.3390/app10062017

Deng L (2014) A tutorial survey of architectures, algorithms, and applications for deep learning. APSIPA Trans Signal Inf Process. https://doi.org/10.1017/atsip.2013.9

Diakogiannis FI, Waldner F, Caccetta P, Wu C (2020) ResUNet-a: a deep learning framework for semantic segmentation of remotely sensed data. ISPRS J Photogramm Remote Sens 162:94–114. https://doi.org/10.1016/j.isprsjprs.2020.01.013

DSITI (2017) Land Use Summary 1999–2015 for the Atherton Tablelands (p. 26). Department of Science, Information Technology and Innovation, Queensland Government. https://publications.qld.gov.au/dataset/land-use-summary-1999-2015/resource/d97bee40-5694-424a-9085-c7e4892475b8

Du Z, Yang J, Ou C, Zhang T (2019) Smallholder crop area mapped with a semantic segmentation deep learning method. Remote Sens 11(7):888. https://doi.org/10.3390/rs11070888

Eelbode T, Bertels J, Berman M, Vandermeulen D, Maes F, Bisschops R, Blaschko MB (2020) Optimization for medical image segmentation: theory and practice when evaluating with dice score or Jaccard Index. IEEE Trans Med Imaging 39(11):3679–3690. https://doi.org/10.1109/TMI.2020.3002417

Flood N, Watson F, Collett L (2019) Using a U-Net convolutional neural network to map woody vegetation extent from high resolution satellite imagery across Queensland, Australia. Int J Appl Earth Observ Geoinf 82:101897. https://doi.org/10.1016/j.jag.2019.101897

Freudenberg M, Nölke N, Agostini A, Urban K, Wörgötter F, Kleinn C (2019) Large scale palm tree detection in high resolution satellite images using U-Net. Remote Sens 11(3):312. https://doi.org/10.3390/rs11030312

Ghosh S, Das N, Das I, Maulik U (2020) Understanding deep learning techniques for image segmentation. ACM Comput Surv 52(4):1–35. https://doi.org/10.1145/3329784

Gurumurthy VA, Kestur R, Narasipura, O. (2019). Mango tree net–a fully convolutional network for semantic segmentation and individual crown detection of mango trees. ArXiv:1907.06915. http://arxiv.org/abs/1907.06915

He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. Proc IEEE Conf Comput vis Pattern Recognit. https://doi.org/10.48550/ARXIV.1512.03385

Hey T, Tansley S, Tolle K, Gray J (2009) The Fourth Paradigm: Data-Intensive Scientific Discovery. Microsoft Research. https://www.microsoft.com/en-us/research/publication/fourth-paradigm-data-intensive-scientificdiscovery/

Huang B, Zhao B, Song Y (2018) Urban land-use mapping using a deep convolutional neural network with high spatial resolution multispectral remote sensing imagery. Remote Sens Environ 214:73–86. https://doi.org/10.1016/j.rse.2018.04.050

Jensen JR (2005) Introductory digital image processing: a remote sensing perspective. Prentice Hall

Kattenborn T, Leitloff J, Schiefer F, Hinz S (2021) Review on Convolutional Neural Networks (CNN) in vegetation remote sensing. ISPRS J Photogramm Remote Sens 173:24–49. https://doi.org/10.1016/j.isprsjprs.2020.12.010

Kestur R, Meduri A, Narasipura O (2019) Mangonet: a deep semantic segmentation architecture for a method to detect and count mangoes in an open orchard. Eng Appl Artif Intell 77:59–69. https://doi.org/10.1016/j.engappai.2018.09.011

Khalel A, El-Saban M (2018) Automatic pixelwise object labeling for aerial imagery using stacked U-nets. ArXiv Preprint arXiv:1803.04953

Khryashchev V, Pavlov V, Priorov A, Kazina E (2018) Convolutional neural network for satellite imagery. In: Conference of Open Innovations Association, FRUCT (No. 22, pp. 344–347). FRUCT Oy

Kim JH, Lee H, Hong SJ, Kim S, Park J, Hwang JY, Choi JP (2019) Objects segmentation from high-resolution aerial images using U-Net with pyramid pooling layers. IEEE Geosci Remote Sens Lett 16(1):115–119. https://doi.org/10.1109/LGRS.2018.2868880

Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. Adv Neural Inf Process Syst 25(2):1097–1105

Kussul N, Lavreniuk M, Skakun S, Shelestov A (2017) Deep learning classification of land cover and crop types using remote sensing data. IEEE Geosci Remote Sens Lett 14(5):778–782. https://doi.org/10.1109/LGRS.2017.2681128

Litjens G, Kooi T, Bejnordi BE, Setio AAA, Ciompi F, Ghafoorian M, van der Laak JAWM, van Ginneken B, Sánchez CI (2017) A survey on deep learning in medical image analysis. Med Image Anal 42:60–88. https://doi.org/10.1016/j.media.2017.07.005

Liu S, Qi Z, Li X, Yeh A (2019) Integration of convolutional neural networks and object-based post-classification refinement for land use and land cover mapping with optical and SAR data. Remote Sens 11(6):690. https://doi.org/10.3390/rs11060690

Ma X, Wang H, Wang J (2016) Semisupervised classification for hyperspectral image based on multi-decision labeling and deep feature learning. ISPRS J Photogramm Remote Sens 120:99–107. https://doi.org/10.1016/j.isprsjprs.2016.09.001

Ma L, Liu Y, Zhang X, Ye Y, Yin G, Johnson BA (2019) Deep learning in remote sensing applications: a meta-analysis and review. ISPRS J Photogramm Remote Sens 152:166–177. https://doi.org/10.1016/j.isprsjprs.2019.04.015

Mahdianpari M, Salehi B, Rezaee M, Mohammadimanesh F, Zhang Y (2018) Very deep convolutional neural networks for complex land cover mapping using multispectral remote sensing imagery. Remote Sens 10(7):1119. https://doi.org/10.3390/rs10071119

Mnih V (2013) Machine learning for aerial image labeling. University of Toronto

Mondal M, Mondal P, Saha N, Chattopadhyay P (2017) Automatic number plate recognition using CNN based self synthesized feature learning. IEEE Calcutta Conf (CALCON). https://doi.org/10.1109/CALCON.2017.8280759

Neupane B, Horanont T, Aryal J (2021) Deep learning-based semantic segmentation of urban features in satellite images: a review and meta-analysis. Remote Sens 13(4):808. https://doi.org/10.3390/rs13040808

Othman E, Bazi Y, Alajlan N, Alhichri H, Melgani F (2016) Using convolutional features and a sparse autoencoder for land-use scene classification. Int J Remote Sens 37(10):2149–2167. https://doi.org/10.1080/01431161.2016.1171928

Pandey PC, Koutsias N, Petropoulos GP, Srivastava PK, Dor EB (2021) Land use/land cover in view of earth observation: data sources, input dimensions, and classifiers–a review of the state of the art. Geocarto Int 36(9):957–988. https://doi.org/10.1080/10106049.2019.1629647

Penatti OAB, Nogueira K, dos Santos JA (2015) Do deep features generalize from everyday objects to remote sensing and aerial scenes domains? IEEE Conf Comput vis Pattern Recognit Worksh (CVPRW). https://doi.org/10.1109/CVPRW.2015.7301382

Ramezan A, Warner A, Maxwell EA (2019) Evaluation of sampling and cross-validation tuning strategies for regional-scale machine learning classification. Remote Sens 11(2):185. https://doi.org/10.3390/rs11020185

Rao V, Rao H (2003) C++ neural networks and Fuzzy logic. BPB Publications

Ren Y, Yu Y, Guan H (2020) DA-CapsUNet: a dual-attention capsule U-Net for road extraction from remote sensing imagery. Remote Sens 12(18):2866. https://doi.org/10.3390/rs12182866

Romero A, Gatta C, Camps-Valls G (2016) Unsupervised deep feature extraction for remote sensing image classification. IEEE Trans Geosci Remote Sens 54(3):1349–1362. https://doi.org/10.1109/TGRS.2015.2478379

Ronneberger O, Fischer P, Brox T (2015) U-Net: convolutional networks for biomedical image segmentation. ArXiv:1505.04597 [Cs]. http://arxiv.org/abs/1505.04597

Schmidhuber J (2015) Deep learning in neural networks: an overview. Neural Netw 61:85–117. https://doi.org/10.1016/j.neunet.2014.09.003

Sharma A, Liu X, Yang X, Shi D (2017) A patch-based convolutional neural network for remote sensing image classification. Neural Netw 95:19–28. https://doi.org/10.1016/j.neunet.2017.07.017

Shorten C, Khoshgoftaar TM (2019) A survey on Image Data augmentation for deep learning. J Big Data 6(1):60. https://doi.org/10.1186/s40537-019-0197-0

Stoian A, Poulain V, Inglada J, Poughon V, Derksen D (2019) Land cover maps production with high resolution satellite image time series and convolutional neural networks: adaptations and limits for operational systems. Remote Sens 11(17):1986. https://doi.org/10.3390/rs11171986

Sun Y, Tian Y, Xu Y (2019) Problems of encoder-decoder frameworks for high-resolution remote sensing image segmentation: structural stereotype and insufficient learning. Neurocomputing 330:297–304. https://doi.org/10.1016/j.neucom.2018.11.051

Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 1–9

Wagner FH, Sanchez A, Tarabalka Y, Lotte RG, Ferreira MP, Aidar MPM, Gloor E, Phillips OL, Aragão LEOC (2019) Using the U-Net Convolutional Network to Map Forest Types and Disturbance in the Atlantic Rainforest with Very High Resolution Images. Remote Sens Ecol Conserv. https://doi.org/10.1002/rse2.111

Wei S, Zhang H, Wang C, Wang Y, Xu L (2019) Multi-temporal SAR data large-scale crop mapping based on U-Net model. Remote Sens 11(1):68. https://doi.org/10.3390/rs11010068

Wieland M, Li Y, Martinis S (2019) Multi-sensor cloud and cloud shadow segmentation with a convolutional neural network. Remote Sens Environ 230:1203. https://doi.org/10.1016/j.rse.2019.05.022

Wurm M, Stark T, Zhu XX, Weigand M, Taubenböck H (2019) Semantic segmentation of slums in satellite images using transfer learning on fully convolutional neural networks. ISPRS J Photogramm Remote Sens 150:59–69. https://doi.org/10.1016/j.isprsjprs.2019.02.006

Zhang L, Zhang L, Du B (2016) Deep learning for remote sensing data: a technical tutorial on the state of the art. IEEE Geosci Remote Sens Mag 4(2):22–40. https://doi.org/10.1109/MGRS.2016.2540798

Zhang M, Li W, Du Q (2018) Diverse Region-Based CNN for Hyperspectral Image Classification. IEEE Trans Image Process 27(6):2623–2634. https://doi.org/10.1109/TIP.2018.2809606

Zhu XX, Tuia D, Mou L, Xia G-S, Zhang L, Xu F, Fraundorfer F (2017) Deep learning in remote sensing: a review. IEEE Geosci Remote Sens Mag 5(4):8–36. https://doi.org/10.1109/MGRS.2017.2762307