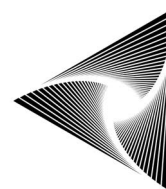




Version 1.1



ZRC SAZU



Anthony Corns, Dr Nejc Čož, Dr Susan Curran, Dr Dragi Kocev, Dr Žiga Kokalj, Dr Ana Kostovska
18/11/2025

Contents

1. Training Dataset Creation.....	11
1.1. Extent Digitisation	11
1.2. Extent Attribution	14
1.2.1. DFM Quality Rating Attribute.....	14
1.2.2. EDGE Attribute	16
1.2.3. STRUCTURE Attribute	18
1.2.4. TRUNCATED Attribute.....	19
1.2.5. Multivallate Attribute.....	20
1.3. Envelope testing.....	21
1.4. Masking.....	23
1.4.1. Object detection: labels based on DOTA (Dataset for Object Detection in Aerial Images) dataset format.....	24
1.4.2. Semantic segmentation: rasterized label masks	24
1.5. Division of training data.....	26
1.6. Visualisation production process.....	30
1.6.1. Visualisation coding.....	33
2. Machine Learning Modelling Development.....	34
2.1. Evaluation of FasterRCNN object detection model.....	37
2.1.1. The impact of the visualization type.....	37
2.1.2. The impact of the IOU threshold in non-maxima suppression	39
2.1.3. Removing patches annotated with DFM quality attribution of 4	40
2.1.4. Impact of sample size.....	42
2.1.5. Training predictive models for each object type separately	43
2.1.6. Impact of Augmenting the Training Data.....	44
2.1.7. Testing of MLM Architectures	45
2.2. Final Specifications of ADAF MLM	46
3. Evaluating the MLM	46
3.1. Domain Expert Evaluation	46
3.1.1. Known Archaeology Validation	47
3.1.2. New Positives Evaluation.....	47
3.1.3. Validated Segmentation Evaluation.....	49
3.1.4. Domain Expert Evaluation Summary	54
3.2. Incorporation of Evaluation in MLM ADAF	54
4. ADAF Tool Development	55
4.1. Background to Technology/Software Solution.....	55
4.1.1. Background technology	55

4.2.	ADAF Software Processing Steps.....	57
4.2.1.	Processing Stage 1: Input Data Options.....	57
4.2.2.	Processing Stage 2: ML Options	58
4.2.3.	Processing Stage 3: Post Processing Options	58
4.3.	GUI development.....	58
4.3.1.	Input Data Options.....	59
4.3.2.	Machine Learning Options	59
4.3.3.	Custom Machine Learning Options	60
4.3.4.	Post Processing Options.....	60
4.4.	Initial ADAF Tool Testing	61
5.	Appendix i: DFM processing.....	63
5.1.	Pre-processing	63
5.2.	Generating the elevation models	64
6.	Appendix ii: Processed data list.....	66
7.	Appendix iii: Preparation of ALS data for detection.....	73
8.	Appendix iv: Creating ML-ready data for (re)training.....	73
8.1.	Creating the DFM	73
8.2.	Creating the visualisation and patches from DFMs and vector files.....	74
8.2.1.	Creating visualisations.....	75
8.2.2.	Creating training data	75
8.2.3.	Python script for creating the visualisation.....	76
8.2.4.	Python script for creating patches.....	77
9.	Appendix v: Retraining of MLM.....	82
9.1.	Overview.....	82
9.2.	Getting Started	82
9.3.	Jupyter Notebook Structure	82
9.3.1.	Import Packages.....	82
9.3.2.	Loading train, validation and test data	82
9.3.3.	Model Creation	83
9.3.4.	Loading Pretrained ADAF Model (Optional).....	84
9.3.5.	Training the Model.....	84
9.3.6.	Model Evaluation	84
10.	Appendix vi: MLM & ADAF tool security	86
11.	Appendix vii: Suggestions for future improvements.....	87

Table of Figures

Figure 1: Bar chart illustrating the number of monument extent polygons created using the different source ALS datasets (note: SC_TII is a 2m resolution dataset from the TII N4 Carrick-on-Shannon to Dromod road scheme which may be used to test the performance of the model with lower resolution datasets).....	11
Figure 2: Elevation model extents of considered datasets. "CFRAM data" (dark yellow), and "DP data" and "N data" (red) with Noise Mapping data shown separately (orange).	12
Figure 3: Examples of overlapping and conjoined monuments within the provided ALS data.	13
Figure 4: UK Barrows and monument extent polygons (with associated DFM quality rating attribute).....	14
Figure 5: Examples of differing DFM quality within a selection of ringforts.....	15
Figure 6: Examples of differing DFM quality within a selection of enclosures.	15
Figure 7: Examples of differing DFM quality within a selection of barrows.....	16
Figure 8: Examples of monuments located at the edge of ALS data with the monument polygon extent marked in blue.	17
Figure 9: Examples of monument within the ALS data which have been attributed with having internal structural components.....	18
Figure 10: Examples of monument within the ALS data which display truncated form due to later activity.....	19
Figure 11: Examples of monument which display multivallate characteristics and the relevant DFM quality attribute applied.	20
Figure 12: Histogram displaying the distribution of the maximum diameter (m) of ringforts within the ALS data.	21
Figure 13: Histogram displaying the distribution of the maximum diameter (m) of enclosures within the ALS data.	22
Figure 14: : Histogram displaying the distribution of the maximum diameter (m) of barrows within the ALS data.	22
Figure 15: Comparing envelope tile size on ALS Noise mapping dataset (barrows - red, enclosures – green, and ringforts - blue) left 256 pixels (128m), right 512 pixels (256m).....	23
Figure 16: Comparing envelope tile size of enclosures from ISA-41_Wicklow ALS dataset. left 256 pixels (128m), right 512 pixels (256m).....	23
Figure 17: Example of a discarded data patch from ISA-15 Kilkee dataset. Bounding box over a fraction of an archaeology feature (highlighted in red) that did not contain any valid data.	25
Figure 18: Example of a discarded data patches from ISA-15 Kilkee dataset. Less than 1/3 of an archaeological feature has valid data.....	26

Figure 19: Ireland monument segmentations (grey dots barrows, blue dots enclosures, and red dots ringforts) with validation (green) and test (brown) zones.....	28
Figure 20: UK barrow segmentations (grey dots), ALS data used (yellow) and validation (green) and test (brown) zones.	29
Figure 21: Simple local relief model (SLRM) (a), VAT flat (slope, sky-view factor, and positive openness as a 3-band composite) (b), multi-scale topographic position enhanced version 2 (e ² MSTP) (c) and 3 (e ³ MSTP) (d).	31
Figure 22: Multi-scale topographic position enhanced version 4 (e ⁴ MSTP).....	31
Figure 23: Local dynamic range adjustment of a DEM with linear histogram stretch between minimum and maximum (a) and local dynamic range adjustment of a DEM with linear histogram stretch and 1% clipping of minimum and maximum values.	31
Figure 24: A map with data extents. It shows extents of the elevation model datasets and not the patches.	36
Figure 25: mAP scores of the predictive models using the different visualisations of the data calculated on the validation set across the different 20 learning epochs.....	37
Figure 26: mAP@50 scores of the predictive models using the different visualisations of the data calculated on the validation set across the different 20 learning epochs.....	38
Figure 27: mAP scores of the predictive models calculated on the validation set of the SLRM for the different IoU thresholds across the different 20 learning epochs.	39
Figure 28: mAP ₅₀ scores of the predictive models calculated on the validation set of the SLRM for the different IoU thresholds across the different 20 learning epochs.	40
Figure 29: mAP scores of the predictive models calculated on the validation set of the SLRM with 1, 2 and 4 and 1 and 2 level of SFM quality attribution across the different 20 learning epochs.	41
Figure 30: mAP ₅₀ scores of the predictive models calculated on the validation set of the SLRM with 1, 2 and 4 and 1 and 2 level of SFM quality attribution across the different 20 learning epochs.	41
Figure 31: mAP scores of the predictive models calculated on the validation set of the SLRM data with sample size of 256 and 512 pixels across the different 20 learning epochs.	42
Figure 32: mAP ₅₀ scores of the predictive models calculated on the validation set of the SLRM data with sample size of 256 and 512 pixels across the different 20 learning epochs.	43
Figure 33: Examples of strong positive (left) and weak positive (right) identified within the segmentation annotations (red).....	48
Figure 34: Example of a false positive in the segmentation annotations (red) caused by the identification of a putting green on a golf course.	48
Figure 35: Process diagram outlining the complete processing workflow and sequencing of the ADAF tool.....	57
Figure 36: ADAF GUI component for the selection of input data and options available.	59
Figure 37: ADAF GUI component for the selection of ML options.	59

Figure 38: ADAF GUI component for selection of ML options if a custom model is used.	60
Figure 39: ADAF GUI component for the post process filtering of segmentation polygons based on geometry.....	60

List of acronyms

ADAF-ML	<i>Automatic Detection of Archaeological Features using Machine Learning</i>
AI	<i>Artificial Intelligence</i>
AiTLAS	<i>Artificial Intelligence Toolbox for Earth Observation</i>
ALS	<i>Airborne laser scanning / Airborne laser system / Airborne LiDAR system</i>
API	<i>Application Programming Interface</i>
ASCII	<i>American Standard Code for Information Interchange</i>
BB	<i>Bounding Box</i>
CFRAM	<i>Catchment-based Flood Risk Assessment and Management</i>
COCO	<i>Common Objects in Context</i>
CRIM	<i>Colour Relief Image Map</i>
DEM	<i>Digital elevation model</i>
DFM	<i>Digital feature model</i>
DP	<i>Discovery Programme</i>
DTM	<i>Digital terrain model</i>
EO	<i>Earth Observation</i>
e ² MSTP	<i>Enhanced Multiscale Topographic Position version 2</i>
e ³ MSTP	<i>Enhanced Multiscale Topographic Position version 3</i>
e ⁴ MSTP	<i>Enhanced Multiscale Topographic Position version 4</i>
GDAL	<i>Geospatial Data Abstraction Library</i>
GSI	<i>Geological Survey Ireland</i>
GUI	<i>Graphical User Interface</i>
HRNet	<i>High-Resolution Net</i>
HTML	<i>HyperText Markup Language</i>
IoU	<i>Intersection over Union</i>
LD	<i>Local dominance</i>
LiDAR	<i>Light detection and ranging</i>
mAP	<i>Mean Average Precision</i>
mIoU	<i>Mean Intersection over Union</i>
ML	<i>Machine learning</i>

MLM	<i>Machine learning model(s)</i>
NFDTM	<i>National Framework Digital terrain model</i>
MSTP	<i>Multiscale topographic position</i>
NMS	<i>Non-maxima suppression</i>
OPW	<i>Office of Public Works</i>
PyPi	<i>Python Package Index</i>
R-CNN	<i>Region-based Convolutional Neural Network</i>
RGB	<i>Red-Green-Blue</i>
RRIM	<i>Red relief image map</i>
SD	<i>Standard Deviation</i>
SLRM	<i>Simple local relief model</i>
SMR	<i>Sites and Monuments Record</i>
SVF	<i>Sky-view factor</i>
TAR	<i>Tape Archive</i>
TII	<i>Transport Infrastructure Ireland</i>
TIN	<i>Triangulated irregular network</i>
VAT	<i>Visualisation for Archaeological Topography</i>
VRT	<i>Virtual Raster</i>

Executive Summary

The TII Automatic Detection of Archaeological Features Machine Learning (ADAF-ML) Research Project aimed to develop a user-friendly software tool capable of automatically detecting archaeological features from LiDAR data using machine learning, specifically deep learning algorithms. This included the development of a 'trained' machine learning model that can automatically detect three classes of earthwork monuments from LiDAR data. The software tool was initially designed for use by TII staff and archaeological consultants appointed to TII projects and has been made widely available as free and opensource software to provide opportunities for future collaboration and development.

The aims of the project as outlined by the TII project specifications included:

- Completion of a comprehensive literature review, review of available data and evaluation of potential solutions, including available market solutions, concluding with a recommendation of the preferred solution to achieve the Project Requirements
- Creation of appropriate 'training' data for input into the machine learning model
- Preparation of LiDAR for inputting to machine learning model
- Development of the machine learning model, including training, testing and validation
- Development of the software tool, including testing and validation, and preparation of a user guide
- Preparation of technical reports and publications.

This report sets out the technical developments achieved during the project and was informed by the recommendations outlined within the Research Review & Recommendations Report (Corns et al., 2022).

This report outlines the technical developments within Work Packages 3, 4 and 5.

Workpackage 3-Geospatial Data preparation:

- The creation of monument extents polygons.
- The development and application of a series of attributes to enable potential quality control filtering during the MLM process.
- The creation and testing of patches relative to the size of the monument training examples.
- The division of ALS data into training, validation and testing samples.
- Production of a range of ALS visualization training datasets for MLM evaluation and training.

Work Package 4 - Development of the machine learning model(MLM):

- Development of the Machine Learning Model and the implementation and evaluation of the FasterRCNN object detection model based upon several variables, including
 - ALS visualisation methodology
 - IOU threshold, variation

- training dataset quality adjustment
- sample size variation
- detection of different monument types of augmentation
- impact of augmenting the training data
- MLM architectures
- Domain expert evaluation of MLM results including testing against known archaeological record and validation samples,

Work Package 5 – ADAF Tool Development: Specification and development of ADAF tool and widgets within the Jupyter Notebook framework,

Additional appendices are provided which cover additional information required specifically for the retraining of the MLM and preparation of training data.

Key Findings

- The combined MLM (AO) consistently outperformed others, identifying the highest number of ringforts (711), enclosures (241), and barrows (55) and had the highest total number of monument annotations (1,307) and merged feature detections (1,002). It also identified the most potential new archaeological sites (259) but also had the highest number of false positives (169).
- During validation testing the combined MLM (AO) achieved the highest accuracy across all monument types: Ringforts: 82.56% correctly identified, Enclosures: 58.5% correctly identified, Barrows: 59.38% correctly identified.
- False positives, identified when using all MLM, were often due to vegetation, hedgerows, and modern landscape features like golf courses and farm structures. Filtering techniques were introduced to reduce these inaccuracies.
- Performance of the different MLMs developed was strongly correlated with the quality of ALS data (DFM ratings), with higher detection rates in DFM 1 and 2 datasets.
- Following the process of model refinement small performance gains were achieved with a slight improvement in enclosure segmentation process.

Recommendations

- AO MLM is recommended as the default model for the ADAF tool due to its superior detection and segmentation capabilities.
- Emphasis should be placed on acquiring high-quality ALS data (DFM 1 and 2) for future analyses.
- Continued refinement of filtering methods is necessary to reduce false positives from non-archaeological features.

1. Training Dataset Creation

1.1. Extent Digitisation

Digitisation of monument extent polygons utilised within the segmentation training was completed in QGIS in August 2022 and comprised a total of 10,718 recorded monuments (Table 1). 1,939 recorded barrows polygon extents from a UK dataset (Historic England Scheduled Monument GIS Data, 2022) were used to supplement the Irish dataset given the relatively low numbers of recorded barrows available for digitisation within the footprint of the Irish lidar datasets (see Appendix ii: Processed data list). Monuments with no visible surface trace on the Airborne Laser Scanning (ALS) derived Digital Feature Model (DFM) were not digitised; in some cases, the monuments had been excavated, in others, the remains are simply not visible. This happened across all monument types; however, it was particularly evident in the case of monuments classified as 'ring-ditch' on the Sites and Monuments record (contained within the 'Ringfort' monument extent polygons). Due to the often-ephemeral nature of the remains of this particular monument type, many ring-ditches were not visible on the lidar surface and therefore could not be digitised; a total of 34 monument extent polygons were created for this monument type.

Monument Classification	Number of Polygons
Ringfort	5,933
Enclosure	2,324
Barrow	522
UK Barrow	1,939

Table 1: Number of monument extent polygons created based on monument classification.

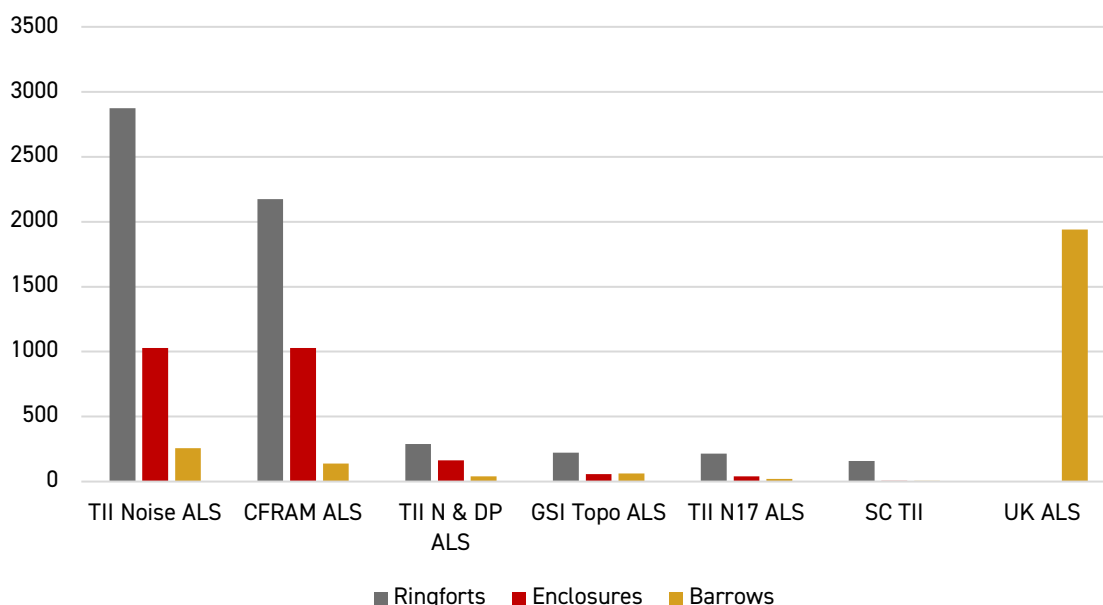


Figure 1: Bar chart illustrating the number of monument extent polygons created using the different source ALS datasets (note: TII N refers to National Road scheme ALS data, SC_TII is a 2m resolution dataset from the TII N4 Carrick-on-Shannon to Dromod road scheme which was used to test the performance of the model with lower resolution datasets).

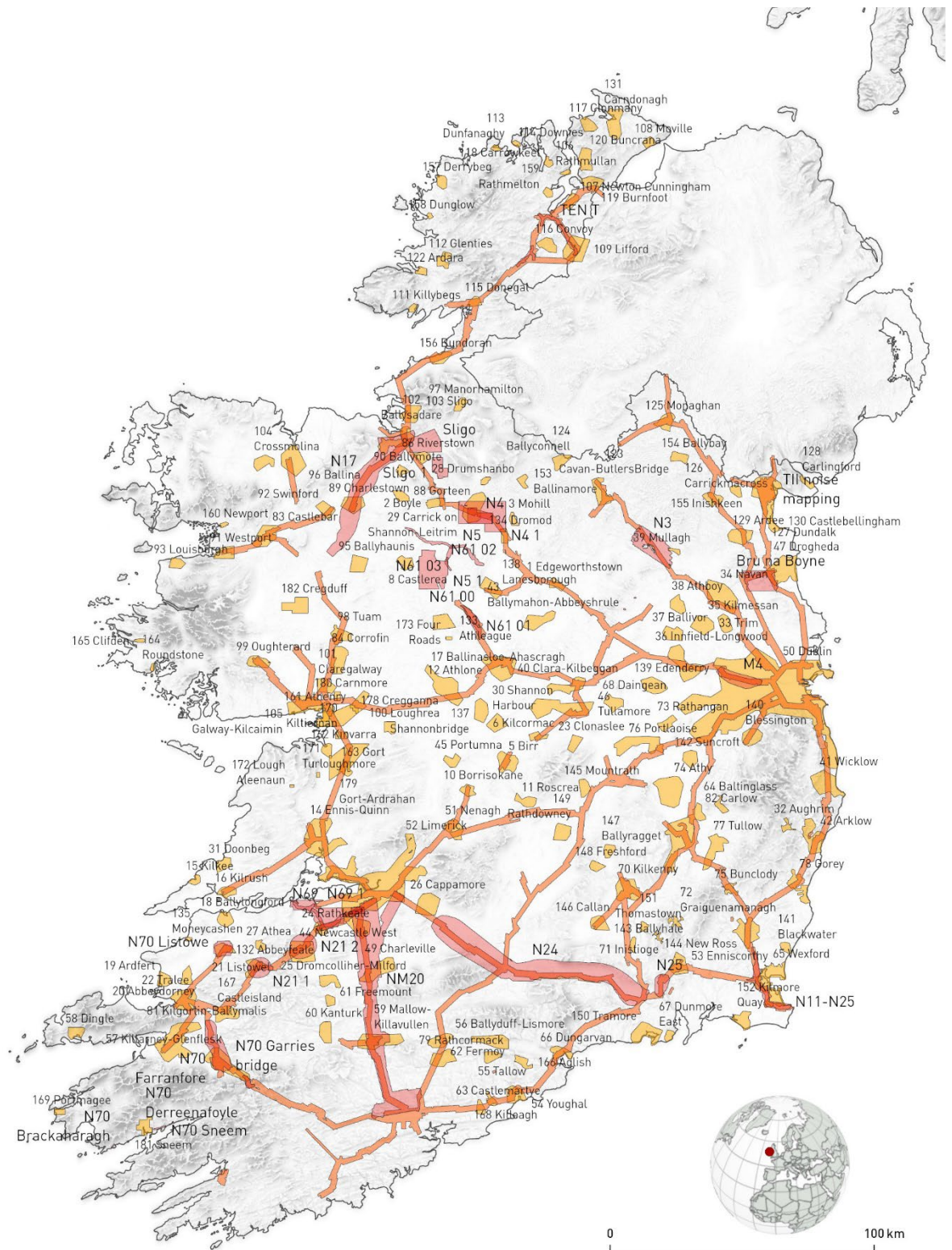


Figure 2: Elevation model extents of considered Irish datasets. "CFRAM data" (dark yellow), and "DP data" and "N data" (red) with Noise Mapping data shown separately (orange). Note: UK barrow ALS data utilised within the project can be observed in Figure 20.

A polygon was drawn around the perimeter of each monument, enclosing the entirety of its visible extent. An 'attribute table' was assigned to each segmentation polygon to record metadata for each monument digitised as part of this phase of the project. These attributes were identified with the aim of supporting and potentially facilitating the assessment of the testing of the Machine Learning Model (MLM) within the framework of various monument characteristics and are discussed further below.

In certain cases, monuments were overlapping, or one monument was contained within the footprint of another monument (Figure 3). Where the monuments were both recorded separately on the Sites and Monuments Record (SMR), two distinct monument extent polygons were created. Where only one recorded monument was represented (e.g., a figure-of-eight/conjoined enclosure), one polygon was created.

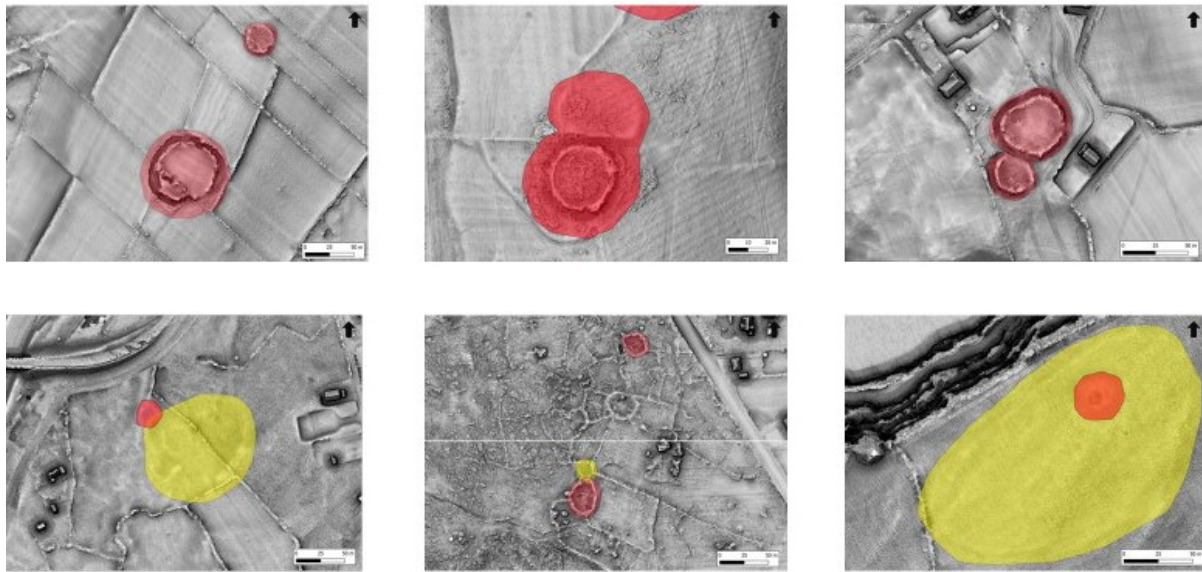


Figure 3: Examples of overlapping and conjoined monuments within the provided ALS data.

In the case of the UK barrows, the dataset already comprised a polygon encircling the recorded monument. However, in most cases, a single polygon was used to enclose a series or cluster of barrows. In these cases, the existing polygon was deleted, and separate polygons digitised for each recorded barrow (Figure 4).

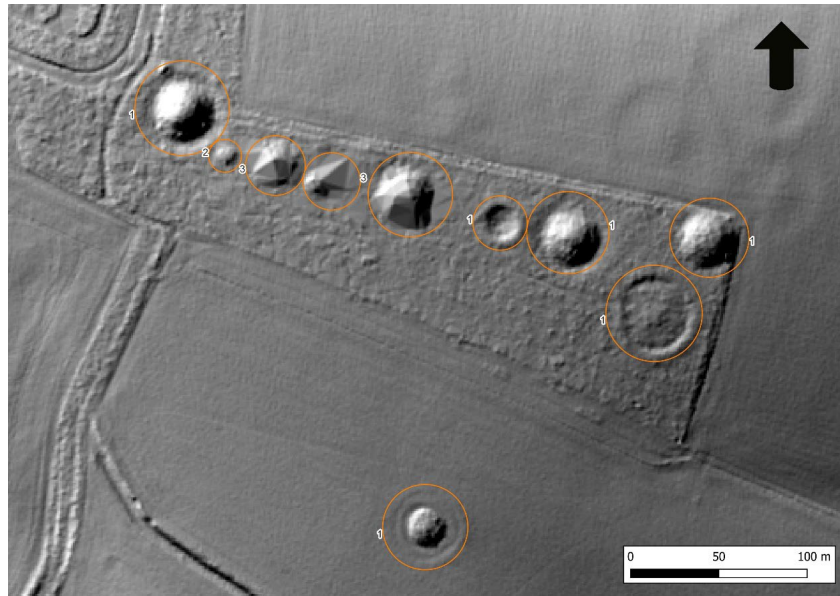


Figure 4: UK Barrows and monument extent polygons (with associated DFM quality rating attribute).

1.2. Extent Attribution

During the process of extent digitisation additional attribution information was recorded. This was carried out to enable the potential filtering of the ALS training data based on the attribute criteria if the quality of the MLM was affected by poor or erroneous monument examples. The attribute table comprises the following: DFM (quality rating), EDGE, STRUCTURE, TRUNCATED, MULTIVALLATE. Attributes were encoded within the table related to the relevant polygon shapefiles. All attributes were used in terms of the Irish dataset; however, only 'DFM' and 'EDGE' were utilised for the UK barrows dataset as the other criteria were not applicable here.

1.2.1. DFM Quality Rating Attribute

This records the quality of the DFM which features the monument; this can vary quite considerably due largely to the differing ability of the ALS footprint to penetrate vegetation coverage. Areas where vegetation penetration is poor, results in lower resolution data, which in turn creates in larger triangulation of results across the data gaps. It is important to note that this is not necessarily a reflection of a monument's survival and/or upstanding remains. DFM quality attribution values include:

1. Excellent (no triangulation of the model evident, enclosing features clearly identifiable)
2. Good (some triangulation of the model, enclosing features still discernible)
3. Poor (complete/almost complete triangulation, no distinguishable features)
4. Monument barely visible (little to no trace of the enclosing features visible)

All monument extent polygons were assigned a DFM rating. The majority of monuments returned a DFM quality rating of '1' (42%), followed by 31% of monuments rated as '2'. Only 13% of monuments were regarded as poorly defined (3). The statistics of this attribution classification are summarised in Table 2 and examples of attributed monuments are illustrated in Figures 5, 6 and 7

Monument Classification	DFM Quality Rating				Total
	1	2	3	4	
Ringfort	2,119	2,230	981	603	5,933
Enclosure	1,136	510	263	415	2,324
Barrow	285	113	56	68	522
UK Barrow	937	511	92	399	1,939

Table 2: Breakdown of the DFM quality attributes across the different monument classification types.

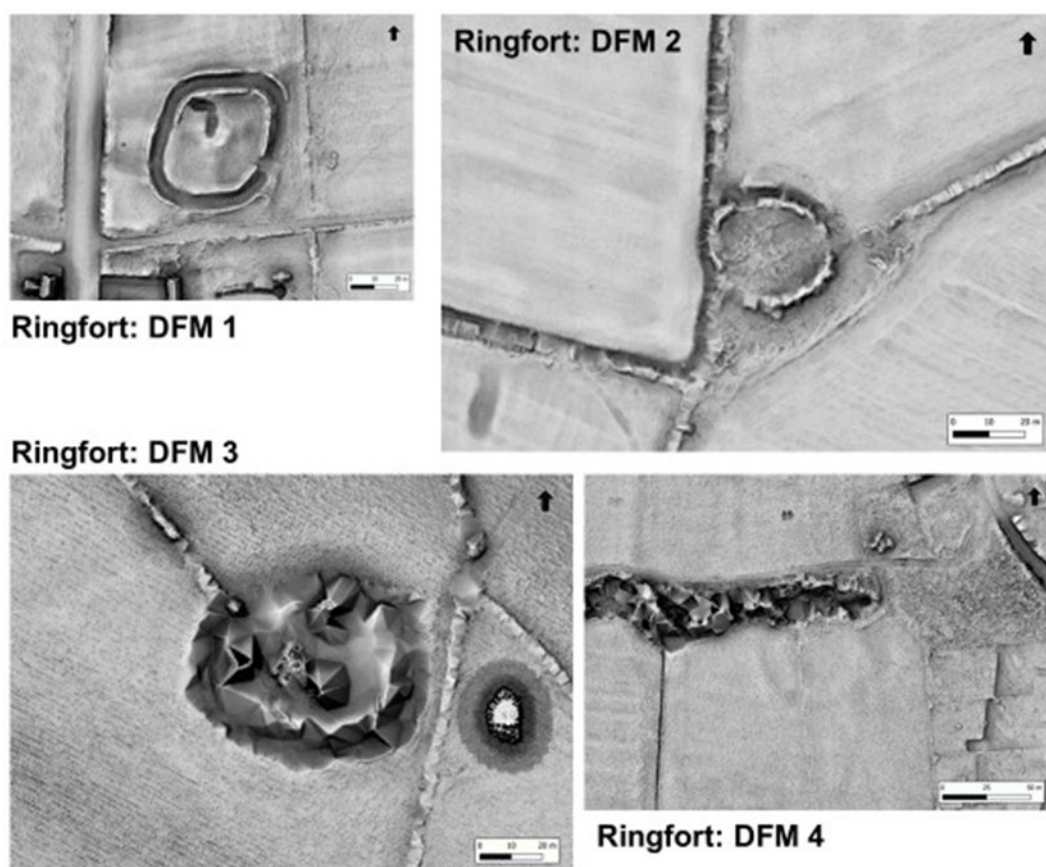


Figure 5: Examples of differing DFM quality within a selection of ringforts.

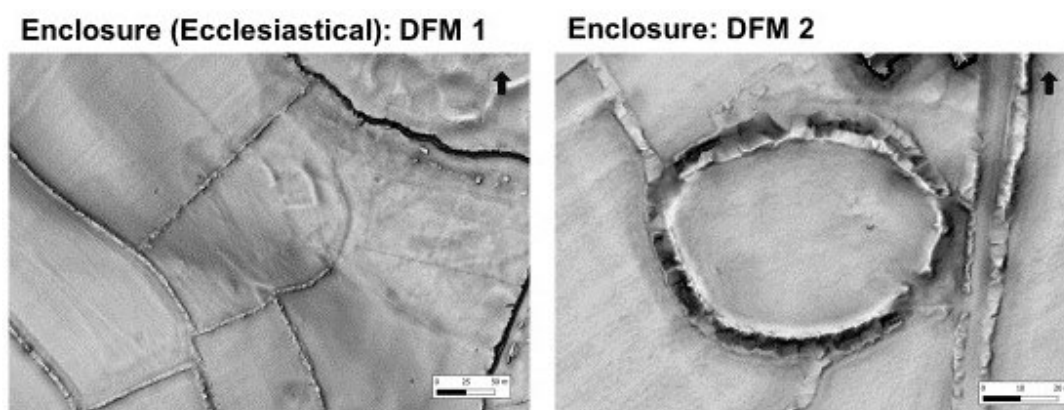
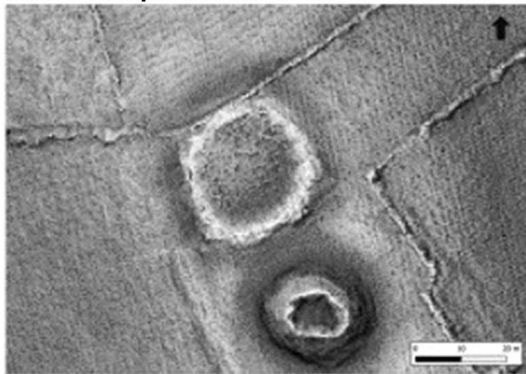


Figure 6: Examples of differing DFM quality within a selection of enclosures.

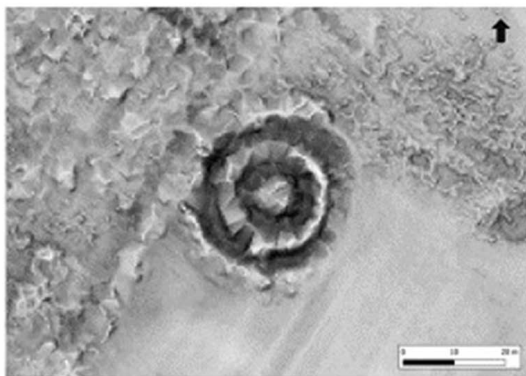
Barrow - pond barrow: DFM 1



Barrow - ring barrow: DFM 1



Barrow - multivalate: DFM 2



Barrow - mound barrow: DFM 3

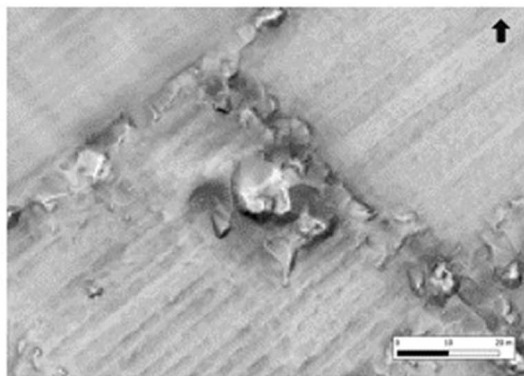


Figure 7: Examples of differing DFM quality within a selection of barrows.

1.2.2. EDGE Attribute

This identifies monuments that are situated on the periphery of an ALS dataset. This can have several implications including truncation of the monument (i.e., part located outside of the limits of the dataset) or deterioration of the dataset (e.g., triangulation) that can occur as part of the processing of dataset edges. Where the monument was located on the edge of a dataset but was complete and not impacted by the quality of the data, it was not marked as an 'edge' monument. Where the monument was truncated or affected by data quality, only the visible extent of the monument was digitised. The percentage of monuments affected was very low with only 2.3% of the overall total classified as '1' and 0.1% as '2'. Where no EDGE effects are present no attribution has been applied to the relevant polygons. EDGE attribution values include:

1. Monument located on edge of ALS dataset (i.e., limits of ALS survey)
2. Monument located on edge of ALS tile (adjacent tile available?)

A summary of the recording of this attribute is displayed in Table 3 and illustrated in Figure 8.

Monument Classification	EDGE Rating		Total
	1	2	
Ringfort	156	4	160
Enclosure	76	3	79
Barrow	11	-	11
UK Barrow	3	-	3

Table 3: Breakdown of the number of monuments located at the edge of ALS data based upon the monument classification types

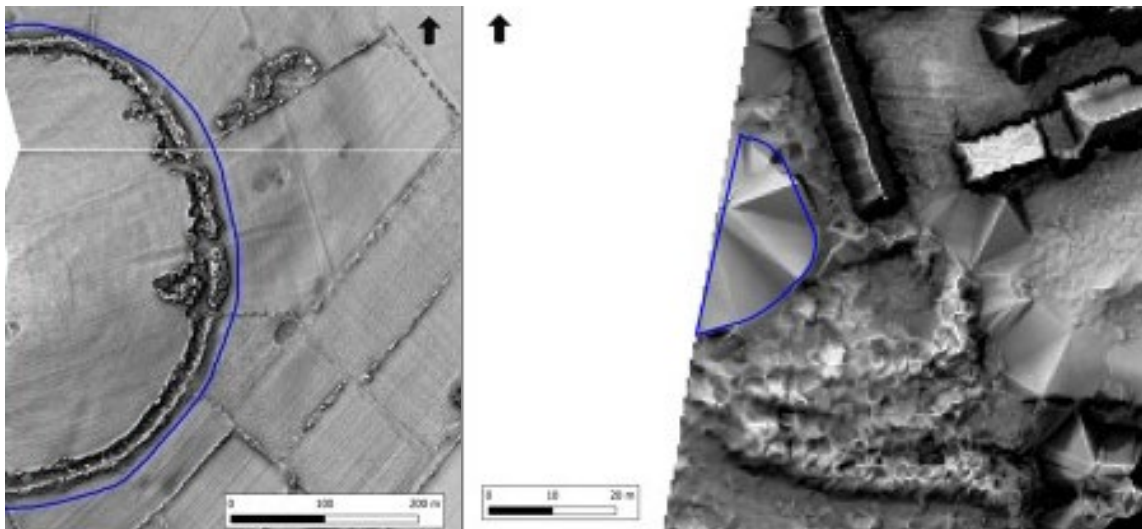


Figure 8: Examples of monuments located at the edge of ALS data with the monument polygon extent marked in blue.

1.2.3. STRUCTURE Attribute

The 'Structure' attribute relates to recorded monuments which have a visible structure (potentially archaeological and/or non-archaeological) contained within the external extent of the site. It is possible that a structure located within the confines of the monument may affect the machine learning model. 4.3% of the overall total contained a (possible) structure. Where no internal structure is present no STRUCTURE attribution was applied. A summary of the recording of this attribute is displayed in Table 4 and illustrated in Figure 9.

Monument Classification	STRUCTURE Rating
	1
Ringfort	241
Enclosure	217
Barrow	1
UK Barrow	0

Table 4: Breakdown of the number of monuments which contain internal structures based upon the monument classification types.

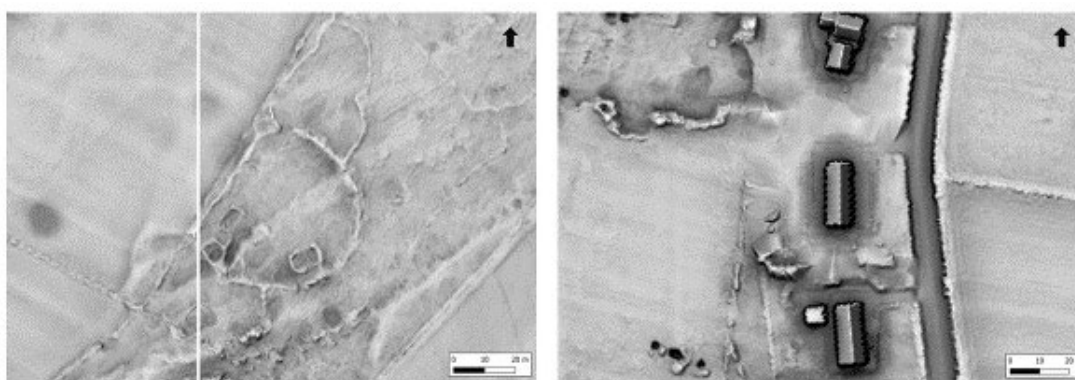


Figure 9: Examples of monuments within the ALS data which have been attributed with having internal structural components.

1.2.4. TRUNCATED Attribute

'Truncated' monuments comprise those monuments which have been partly removed, i.e., by being truncated by a later activity including the insertion of a field boundary, roadway, etc. In all cases, only the visible extent of the monument was digitised. Approximately 10% of all monuments of the selected class were truncated within the ALS data. Where no truncation is evident no TRUNCATED attribution was applied. A summary of the recording of this attribute is displayed in Table 5 and illustrated in Figure 10.

	TRUNCATION Rating
Monument Classification	1
Ringfort	693
Enclosure	347
Barrow	40
UK Barrow	0

Table 5: Breakdown of the number of monuments which have been truncated by later activity.

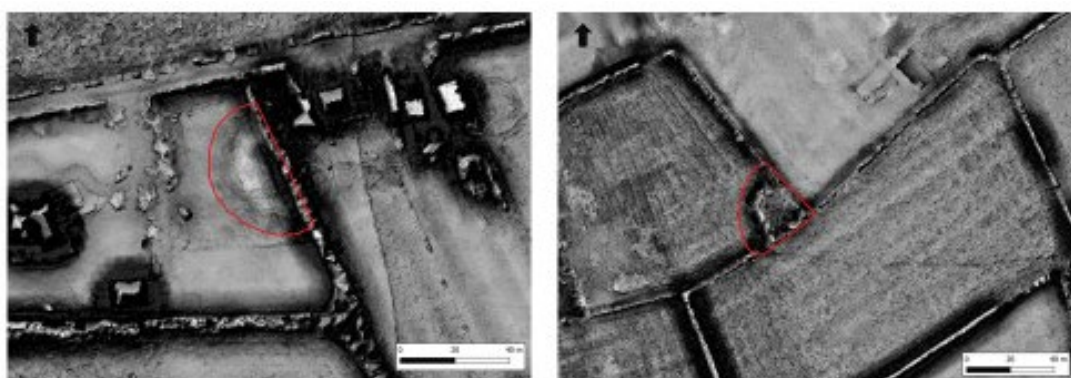


Figure 10: Examples of monuments within the ALS data which display truncated form due to later activity.

1.2.5. MULTIVALLATE Attribute

Multivallate monuments are those with more than one enclosing feature. The number of vallations was not recorded, this attribute was used to differentiate between univallate and multivallate monuments to assess whether this affects the machine learning model. Almost 5% of monuments digitised fell into this category, the majority within the 'Ringfort' classification. Where no evidence of multivallate form is identified no MULTIVALLATE attribution was applied. A summary of the recording of this attribute is displayed in Table 6 and illustrated in Figure 11.

	MULTIVALLATE Rating
Monument Classification	1
Ringfort	414
Enclosure	77
Barrow	36
UK Barrow	0

Table 6: Breakdown of the number of monuments which display multivallate characteristics.

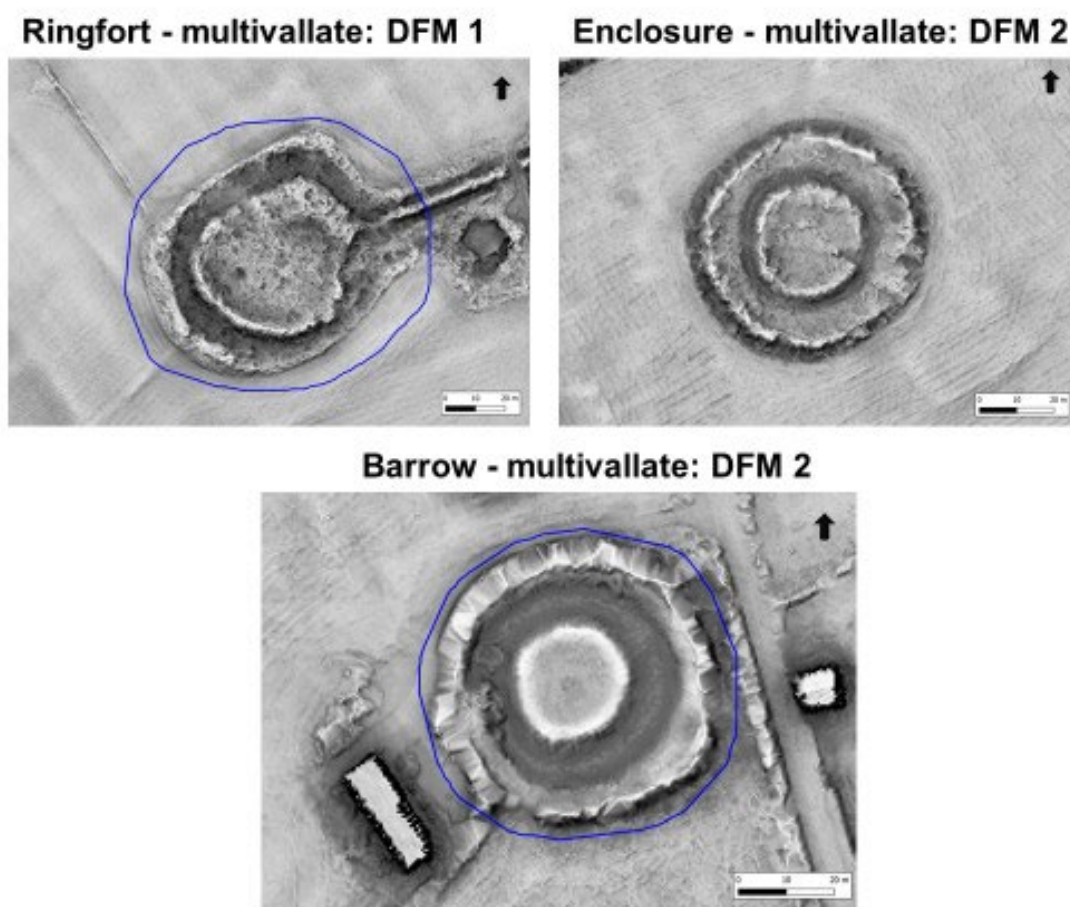


Figure 11: Examples of monuments which display multivallate characteristics and the relevant DFM quality attribute applied.

1.3. Envelope testing

Data patches for the MLM were created automatically. Each patch is represented by a labelled image tile. The patches are not centred on individual features but are filtered from a regular uniform grid created over the entire dataset. This inevitably results in patches that contain only partial objects, i.e., images where the edge of the patch cuts out part of the feature. To ensure that more patches contained full-size features, another set of tiles were created by shifting the grid by half the patch size. The advantage of this approach was that it almost doubled the number of training examples while giving us enough images with full and partial objects.

We have considered three different sizes of patches, including 128 pixels (64x64 m²), 256 pixels (128x128 m²) and 512 pixels (256x256 m²). To select the appropriate size, we have investigated how the size of monument extent polygons is distributed. The longest distance between two vertices was calculated for each feature and plotted the distribution in histograms for ringforts, enclosures and barrows respectively, as shown in Figures 12, 13 and 14. Enclosures are the largest features, with several instances larger than 200m, but with a median + 1SD value of 102 m, majority of them would still be able to fit inside a 256-pixel patch. Ringforts and barrows are relatively smaller, with median + 1SD of 77 and 50 m respectively (Table 7).

Monument Classification	Mean	Median + 1SD
Ringfort	61m	77m
Enclosure	70m	102m
Barrow	36m	50m

Table 7: Size of archaeological features. Size is defined as the longest distance between any two vertices of the convex hull.

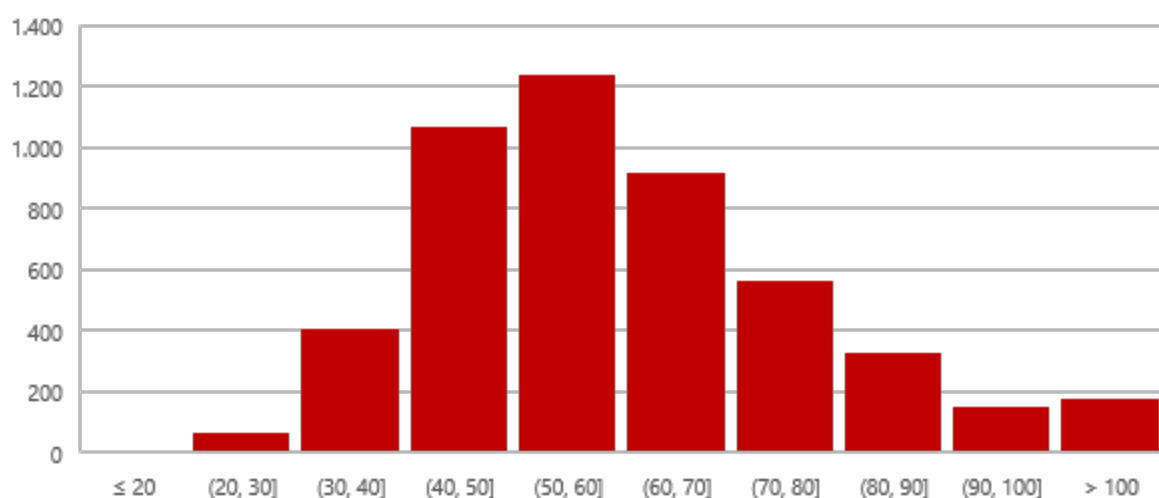


Figure 12: Histogram displaying the distribution of the maximum diameter (m) of ringfort polygon extents within the ALS data.

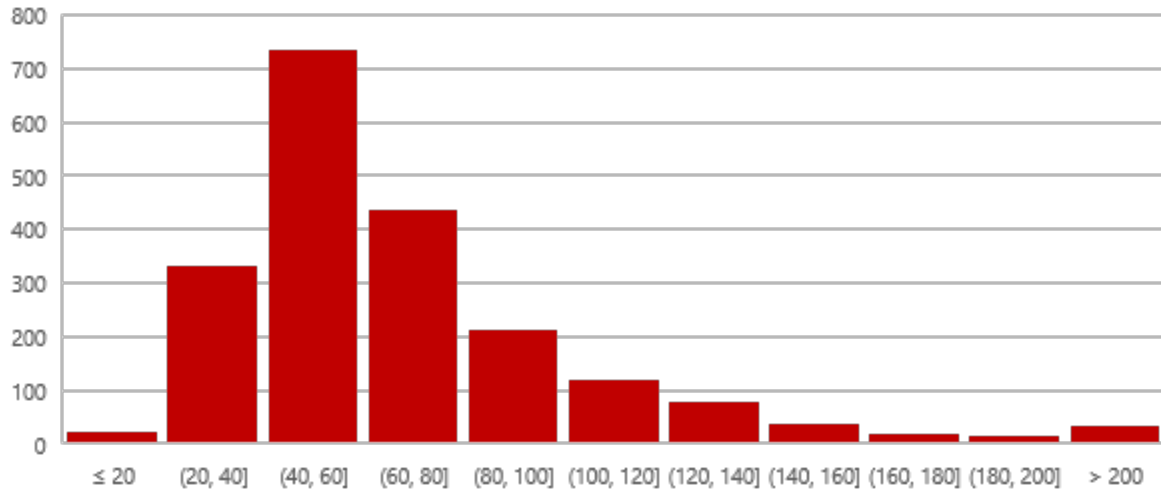


Figure 13: Histogram displaying the distribution of the maximum diameter (m) of enclosure polygon extents within the ALS data.

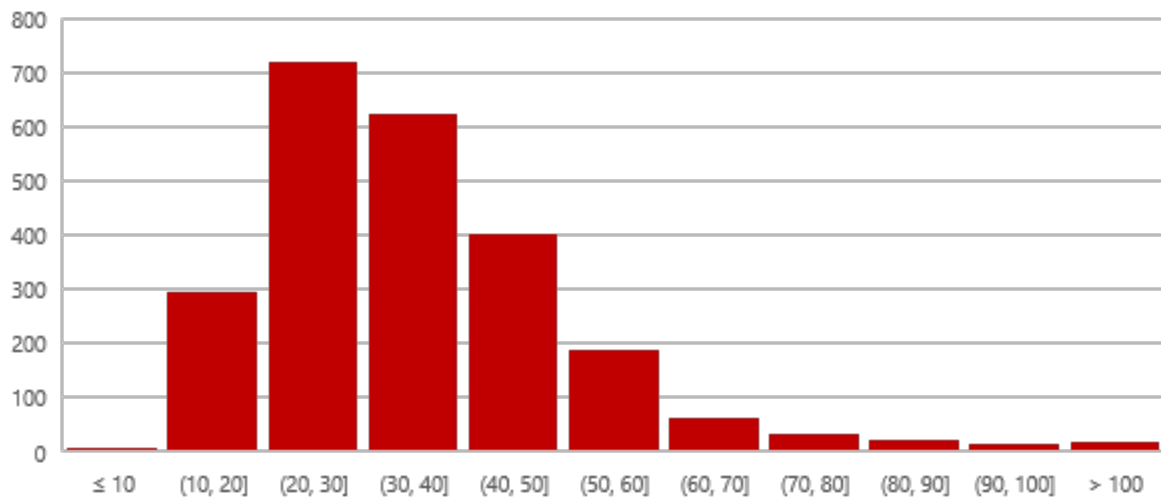


Figure 14: : Histogram displaying the distribution of the maximum diameter (m) of barrow polygon extents within the ALS data.

Based upon the size analysis, testing was carried out using two larger patch sizes of 256 and 512 pixels. Since a patch is supposed to contain a complete object with its surroundings, the smaller size would be sufficient for most barrows and ringforts, as the example in Figure 15 shows. For most enclosures and some larger ringforts, however, we would need a larger tile, as the second example in Figure 16 shows.

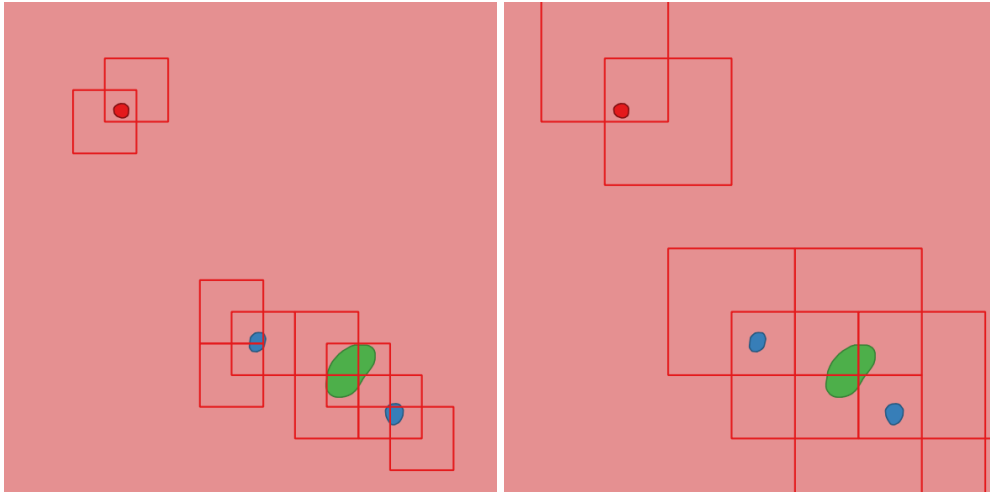


Figure 15: Comparing envelope tile size on ALS Noise mapping dataset (barrows - red, enclosures – green, and ringforts - blue), Left: 256 pixels (128m), Right: 512 pixels (256m).

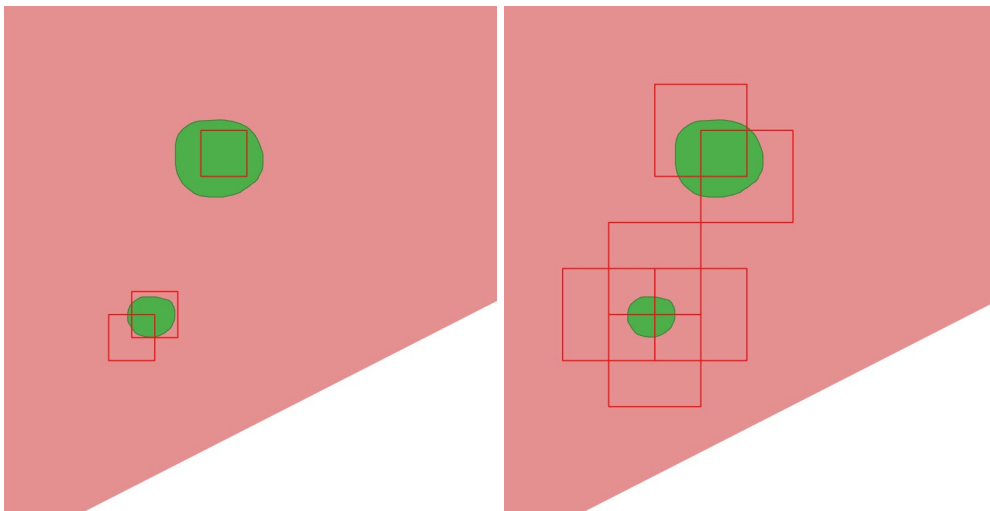


Figure 16: Comparing envelope tile size of enclosures from ISA-41_Wicklow ALS dataset. left 256 pixels (128m), right 512 pixels (256m).

1.4. Masking

The labelled masks for MLM training, testing and validation were generated from vectorised segmentation masks created during the extent digitisation process. The data contains labels for three different archaeological features, namely Barrows, Enclosures and Ringforts. The files containing the labelled masks were automatically created along with the data patches. In doing so, the Python code checks each patch for potential outliers, e.g., truncated features with less than 1/3 of the feature area in the patch and discards them. Features which have been identified as "hard to detect" were assigned a DFM quality attribute value of 3 were also removed from this stage.

Two types of labelled masks were generated, one for object detection, and one for semantic segmentation. Both data formats are described below:

1.4.1. Object detection: labels based on DOTA (Dataset for Object Detection in Aerial Images) dataset format

The labels for object detection were created following the format of DOTA dataset, which was developed specially for object detection in aerial images.

<https://captain-whu.github.io/DOTA>

In the dataset, each object is annotated by an axis-aligned bounding box (BB), which is denoted as a sequence of x and y coordinates of the vertices of the BB. The coordinates are in pixels and are relative to each individual patch. Each instance is labelled with its respective feature class (i.e., barrow, enclosure, or ringfort), followed by four of the attributes provided during the digitisation process:

1. DFM: Quality of representation of the archaeological feature, as assessed by the Discovery Programme Project Archaeologist; Values 1 – 4; Category 3 excluded.
2. TRUN: Truncated objects (by road, building, etc.) [1-True, 0-False]
3. EDGE: On the edge of dataset [1-True, 0-False]
4. EDGE_P: On the edge of patch [1-True, 0-False]

Note STRUCTURE and MULTIVALATE attributes were not utilised in labelling as the number of identified sites was not significant enough for inclusion in the development of the MLM. The following is a template for a label of a single feature in a data patch:

x_1 y_1 x_2 y_2 x_3 y_3 x_4 y_4 <feature class> <DFM> <TRUN> <EDGE> <EDGE_P>

Labels for an individual patch are saved in a text file with the same file name as the patch and suffix "labelTxt". Each line in the text file represents an individual archaeological feature. Below are two examples of labelTxt.txt files:

File name: *90496_123776_Noise_Mapping_labelTxt.txt*

```
0 26 130 26 130 147 0 147 ringfort 1 0 0 0
199 13 256 13 256 109 199 109 ringfort 4 0 0 1
```

File name: *168192_150528_Noise_Mapping_labelTxt*

```
199 55 236 55 236 86 199 86 barrow 1 0 0 1
145 175 167 175 167 198 145 198 barrow 1 0 0 0
93 186 139 186 139 226 93 226 barrow 1 0 0 1
183 201 209 201 209 227 183 227 barrow 1 0 0 1
150 116 181 116 181 144 150 144 barrow 4 0 0 1
128 143 206 143 206 217 128 217 enclosure 1 0 0 0
```

1.4.2. Semantic segmentation: rasterized label masks

The labels for the semantic segmentations were created by rasterising the archaeological features from the vectorised segmentation masks. The same filtering as above was applied, i.e., features with areas less than 1/3 of the total patch and those with DFM = 3 were excluded. The label mask for each patch was saved as an unsigned integer 8-bit raster file with several bands, each band containing masks for the respective feature class in the (alphabetical) order barrow, enclosure, ringfort. Features were masked with a value of 255, while we used value 0

to denote background. Files follow the same naming convention, with the same base name of the file with an added "segmentation_mask" suffix.

Due to a large collection of ALS datasets, considerable effort was applied to identify the presence of outliers. The following are reports of two major outliers, that had to be taken into account in the Python code for creation of data patches:

1. Because we used a regular grid to create the data patches, some patches contained bounding boxes and/or segments that were too small to be included in the ML training set. Examples like the one in Figure 17 had to be discarded as they didn't contain any real information. Most archaeological features have a circular shape and are labelled with rectangular bounding boxes, which means that parts of the bounding boxes may not contain any archaeological data at all. The invalid patches were removed by adding a filter which discards all annotations that contain less than 1/3 of an area of the archaeological feature. This threshold was set in consultation with the Discovery Programme Project Archaeologist, Susan Curran, and Bias Variance Labs

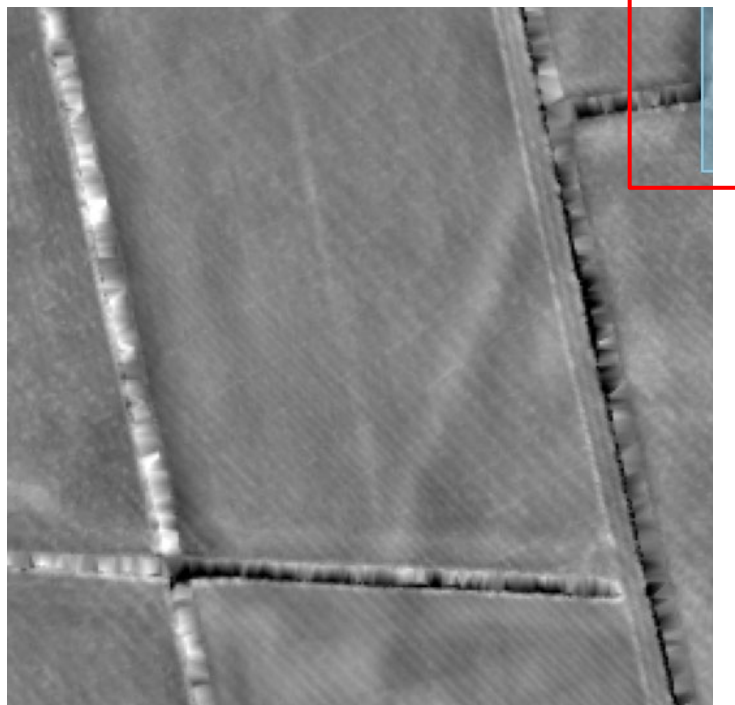


Figure 17: Example of a discarded data patch from ISA-15 Kilkee dataset. Bounding box over a fraction of an archaeological feature (highlighted in red) that did not contain any valid data.

A similar type of outlier was found in patches at the edges of the ALS datasets, as can be seen in Figure 18, where the boundary between valid data and no-data runs directly through a feature. The footprints of the ALS scans are arbitrary and cannot be fitted into a digital data format (raster array) without containing areas of no-data at the edges of an array. In some cases, archaeological features were located at the edge of the ALS scan, which meant that technically the entire feature could fit into a single patch but would contain no-data values. This usually happens because we use a global collection of archaeological features. Since individual datasets can overlap, there is usually another dataset that extends further south that contains data for a complete archaeological feature. In such cases, the patches from the first dataset are discarded and the patches from the second are kept.

To remove such features, we calculated the intersections of the segments with the footprint of the dataset and removed the labels with the same threshold of 1/3 of the object area as in the example above.

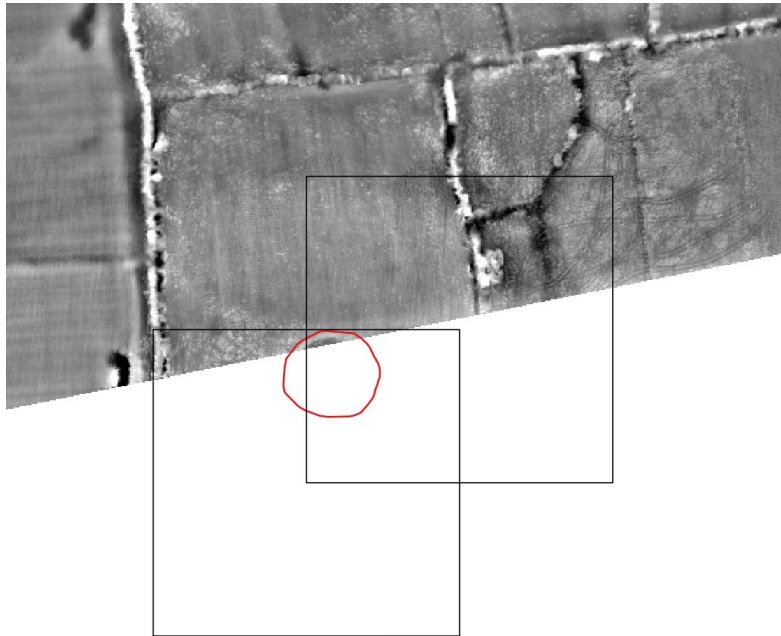


Figure 18: Example of a discarded data patches from ISA-15 Kilkee dataset. Less than 1/3 of an archaeological feature has valid data.

1.5. Division of training data

In developing the MLM, splitting the data into training, validation, and testing sets is essential for building models that generalise well to new, unseen data. The training set is used to teach the model by adjusting its internal parameters. The validation set is used during training to fine-tune parameters and make decisions about the model's architecture, helping to prevent overfitting by providing feedback on how well the model performs on data it hasn't seen during training. Finally, the test set is used only after the model has been fully trained and validated, to provide an unbiased evaluation of its performance.

Geographical zones for the validation and testing phases of the project were selected within the Irish and UK datasets. The full dataset was examined, and the zones selected to ensure that they covered a range of geographically and archaeologically distinct areas, i.e. a mixture of geology, monument type, and monument density. Moreover, they were selected to ensure that a wide range of monument extent polygons were available that reflected multiple monument types with a variety of different attributes. This means that while each testing and validation zone is not necessarily of comparable size, they do comprise a similar number of monuments. Twelve separate zones (6 x testing, 6 x validation) were identified within the Irish dataset (*Figure 19*), and six separate zones (3 x testing, 3 x validation) were identified within the UK dataset (*Figure 20*). In line with recommendations, the total number of monuments within the validation zone comprises 10-20% of the overall monument extent polygons and the testing zone encompasses c. 20% of all digitised monuments (*Table 8*)

Monument Classification	Validation	Testing
Ringfort	777 (16%)	990 (20%)
Enclosure	355 (17%)	414 (20%)
Barrow	85 (18%)	77 (17%)

Table 8: Breakdown of monument extent polygons for validation and testing.

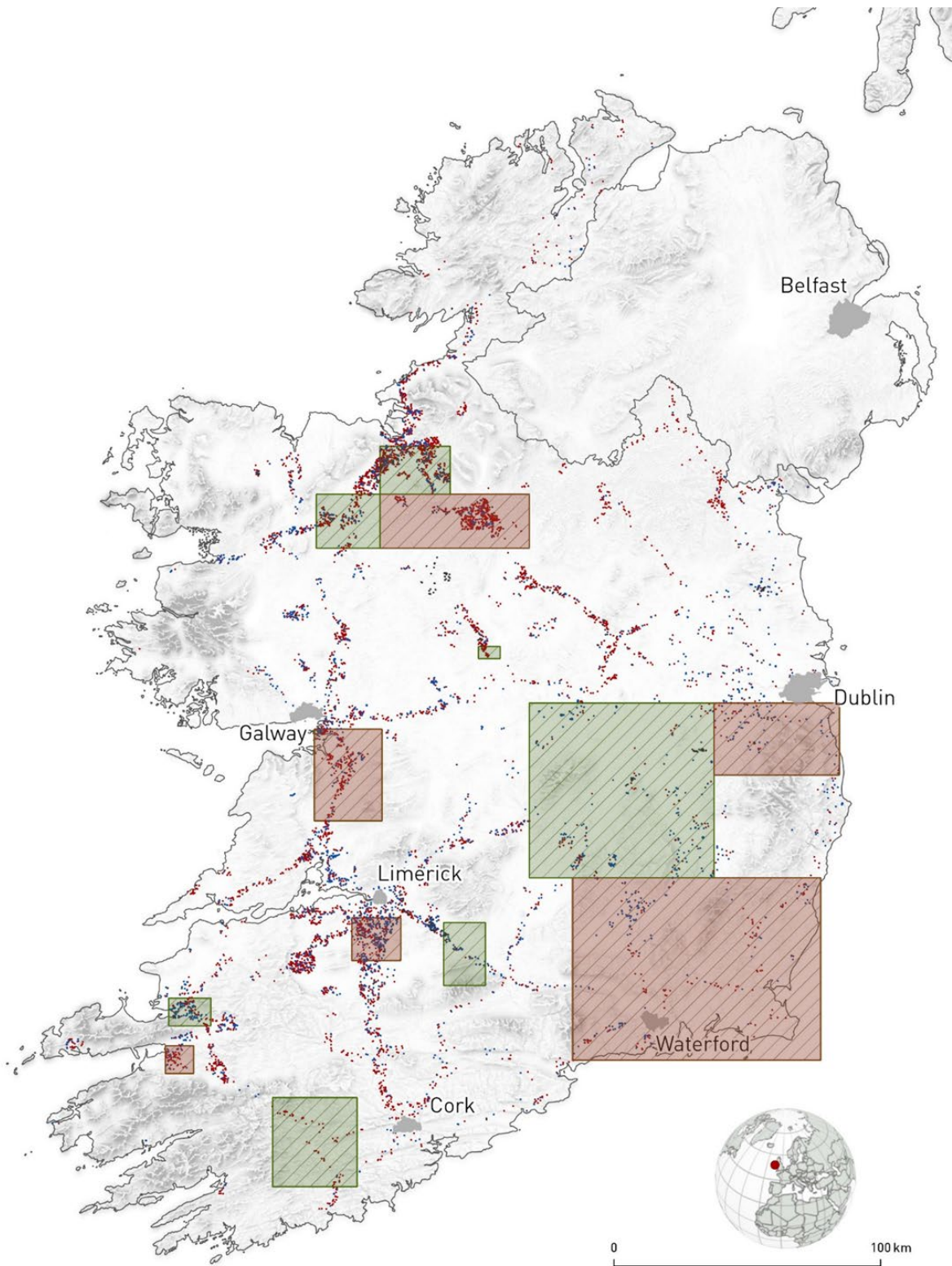


Figure 19: Ireland monument segmentations (grey dots barrows, blue dots enclosures, and red dots ringforts) with validation (green) and testing (brown) zones.

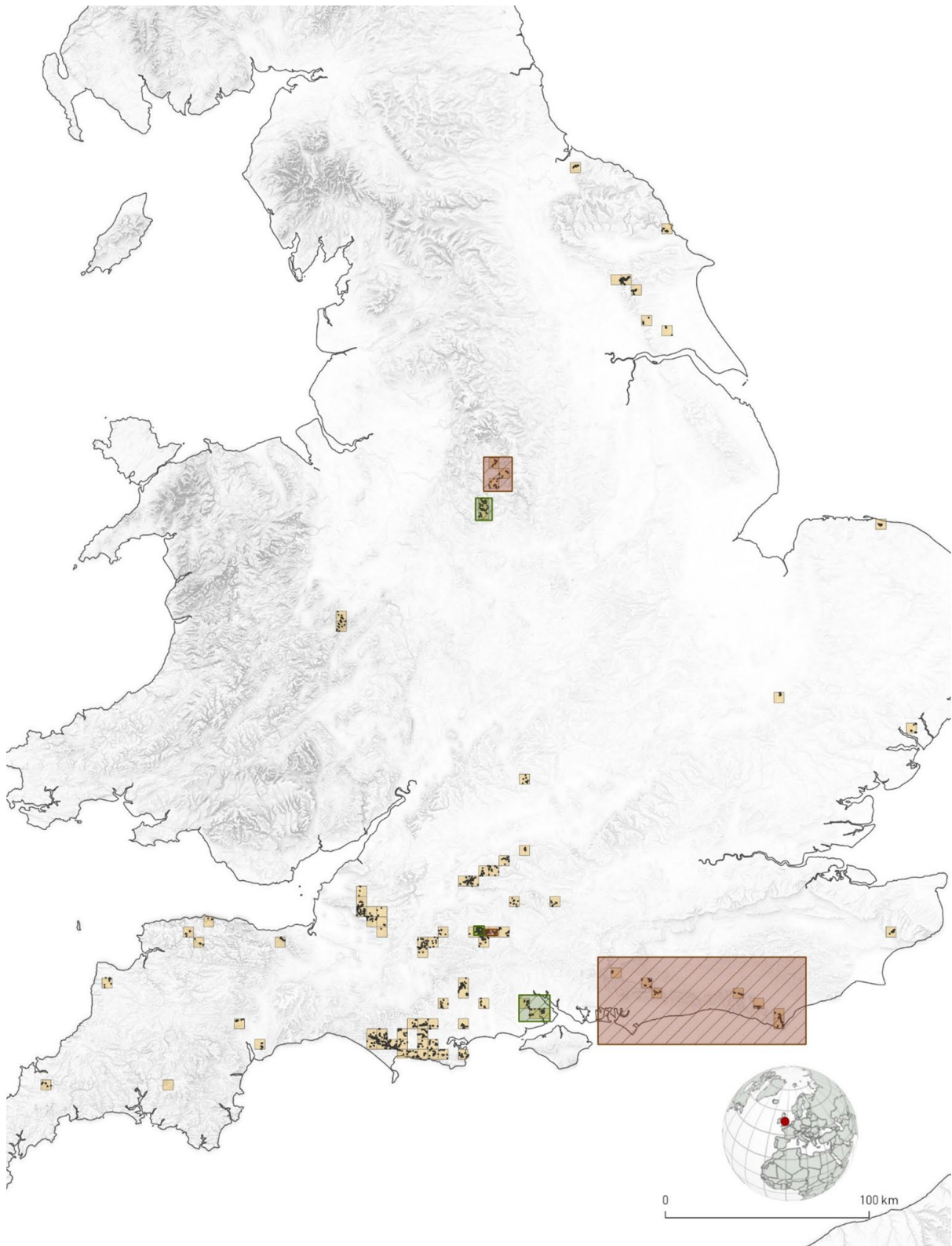


Figure 20: UK barrow segmentations (grey dots), ALS data used (yellow) and validation (green) and testing (brown) zones.

1.6. Visualisation production process

In researching the effectiveness of the final MLM in the detection of archaeological monuments within ALS data tests have been conducted with seven different representations of the elevation data: SLRM (simplified local relief model) (Figure 21a), VAT flat (slope, sky-view factor, and positive openness as a 3-band composite) (Figure 21b), three versions of enhanced multi-scale topographic position (e^2 MSTP (Figure 21c), e^3 MSTP (Figure 21d), e^4 MSTP (Figure 22), and two different histogram stretches of the elevation model (Figure 23). A short description of each is given below for reference with calculation and normalisation settings given in Table 9.

Simple Local Relief Model (SLRM) removes the large-scale morphological elements (hills, valleys...) from data so only small-scale features remain (e.g., archaeology) (Hesse 2010). In the simplified process a simple mean filter computes the trend and a trend removed model is produced directly by subtracting the filtered model from the original (Kokalj et al. 2022).

Visualisation for Archaeological Topography (VAT) is based on a fusion of four information layers into a single image. VAT can be used to enhance visibility of features of varied scale, height, orientation, and form that sit on terrain ranging from extremely flat to very steep (Kokalj and Somrak 2019, Maja et al. 2020). It works well for manual interpretation but is ill suited for data augmentation because it includes hillshading as a base layer. We thus only use slope, sky-view factor, and positive openness, calculated with settings for flat terrain, as a 3-band composite to train the machine learning models.

Enhanced multi-scale topographic position version 2 (e^2 MSTP) was designed by Guyot et al. (2021) and combines the multi-scale topographic position (MSTP) at micro, meso and macro scale ranges with a red relief image map (RRIM) (Chiba, Kaneta, and Suzuki 2008), which combines positive and negative openness and red-coloured slope, and SLRM.

Enhanced multi-scale topographic position version 3 (e^3 MSTP) mimics the e^2 MSTP but replaces the patented red relief image map with an orange–red colourmap for slope (coloured relief image map – CRIM).

Enhanced multi-scale topographic position version 4 (e^4 MSTP) was developed specifically for this project. We created a new, customised visualisation that highlights local topographic variations, provides multiscale information, and is directionally invariant. As such, it is suitable for a wide range of data augmentation methods. It combines multi-scale topographic position (MSTP), positive and negative openness, local dominance, sky view factor computed with settings for normal and flat terrain, and coloured slope.

Elevation models (DFM or DTM) are included with two local dynamic range adjustment (histogram stretch) methods. For each data patch the elevation values were stretched with a linear histogram stretch between minimum and maximum (DEM) (Figure 23a) and with linear histogram stretch and 1% clipping of minimum and maximum (DEM clipped) (Figure 23b).

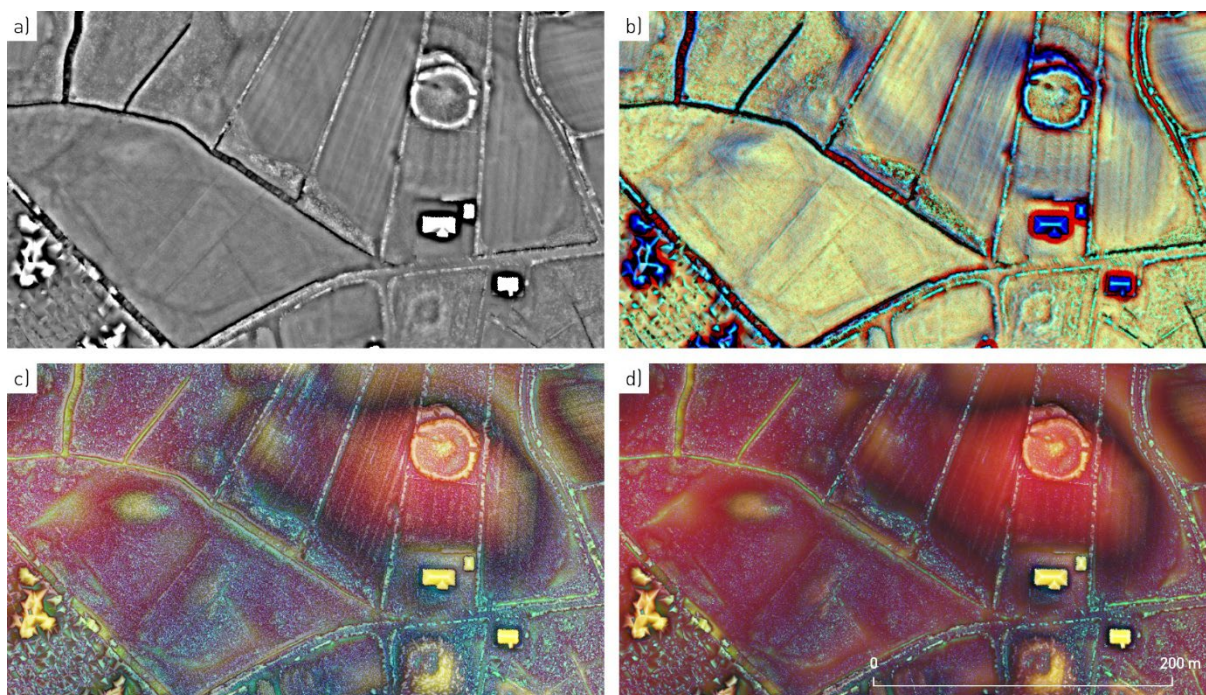


Figure 21: Simple local relief model (SLRM) (a), VAT flat (slope, sky-view factor, and positive openness as a 3-band composite) (b), multi-scale topographic position enhanced version 2 (e^2 MSTP) (c) and 3 (e^3 MSTP) (d).

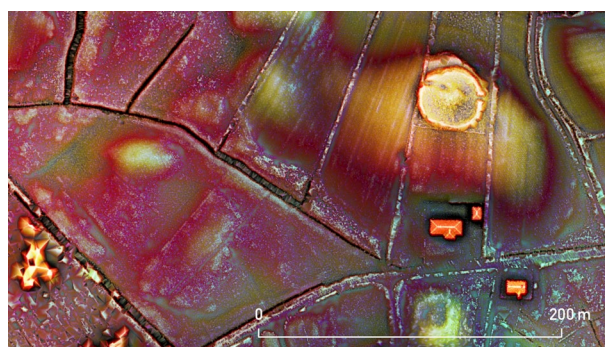


Figure 22: Multi-scale topographic position enhanced version 4 (e^4 MSTP).

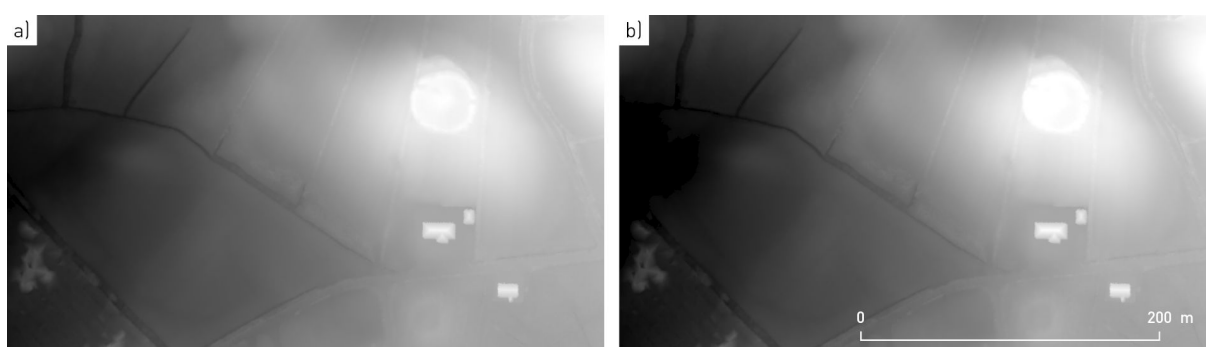


Figure 23: Local dynamic range adjustment of a DEM with linear histogram stretch between minimum and maximum (a) and local dynamic range adjustment of a DEM with linear histogram stretch and 1% clipping of minimum and maximum values (b).

	Description	Calculation Parameters	Visualisation Parameters
SLRM	simple local relief model	radius for trend assessment: 10 m	linear histogram stretch (LHS) between -0.5 and 0.5
slope	gradient of elevation	calculated in degrees	LHS between 0 and 45° for RRIM, CRIM, and e ⁴ MSTP; between 0 and 15° for VAT flat
SVF1	sky view factor	number of search directions: 16; search radius: 5 m	LHS between 0.7 and 1.0
SVF2	sky view factor	number of search directions: 16; search radius: 10 m; high noise remove	LHS between 0.9 and 1.0 for VAT flat;
OP1	positive openness	number of search directions: 16; search radius: 5 m	
OP2	positive openness	number of search directions: 16; search radius: 10 m; high noise remove	LHS between 85 and 93 for VAT flat;
ON1	negative openness	number of search directions: 16; search radius: 5 m	
LD	local dominance	minimum radius: 5 m; maximum radius: 10 m	LHS between 0.5 and 1.8
MSTP	multi-scale topographic position	micro scale (Blue): 0.5 to 2.5 m, step 0.5 m; meso scale (Green): 2.5 to 25 m, step 2.5 m; macro scale (Red): 25 to 250 m, step 25 m; lightness 1.2	LHS between 0 and 1
RRIM	red relief image map	source: slope (red-toned) and openness ((OP1-ON1) / 2)); blending: slope blended (normal blend mode, 50% opacity) with openness	see slope, OP1, openness: LHS between -25 and 25
CRIM	colour relief image map	source images: slope (orange-red-toned) and openness (OP1-ON1); blending: openness blended (overlay, 50% opacity) with openness, blended (luminosity, 50% opacity) with slope	see slope, openness: LHS between -28 and 28
e²MSTP	enhanced multi-scale topographic position version 2	source: MSTP, RRIM, SLRM; blending: SLRM blended (screen, 25% opacity) with RRIM, blended (softlight, 70% opacity) with MSTP	see MSTP, RRIM, SLRM
e³MSTP	enhanced multi-scale topographic position version 3	source: MSTP, CRIM, SLRM; blending: SLRM blended (screen, 25% opacity) with CRIM, blended (softlight, 70% opacity) with MSTP	see MSTP, CRIM, SLRM
e⁴MSTP	enhanced multi-scale topographic position version 4	source: slope (orange-red-toned), LD, openness (OP1-ON1), SVF1, SVF2, MSTP; blending: MSTP blended (overlay, opacity 90%) with SVF combined (SVF1 blended (normal, 50% opacity) with SVF2), blended (multiply, 25% opacity) with openness-LD (openness blended (normal, 50% opacity) with LD), blended (multiply, 100% opacity) with slope	openness (OP1-ON1): LHS between -15 and 15
VAT flat	3-band visualisation for archaeological topography	source: slope, SVF2, OP2; composite as RGB bands	see slope, SVF2, OP2

Table 9: ALS visualisation settings.

1.6.1. Visualisation coding

The processing of the visualisations is semi-automatic and based on the Python library of the Relief Visualisation Toolbox (<https://rvt-py.readthedocs.io>). For research purposes, we have divided the processing into two parts and saved all intermediate results. In the first step, all primary visualisations are prepared, which are then blended into final visualisations in the second step. This allowed us to change and test different parameters for the processing of the visualisations during the development of the code. The final version of the code will be streamlined and fully automated to output only the visualisations needed.

The code has been prepared to handle small and large DEMs in TIFF (GeoTiff) or VRT Raster (GDAL Virtual Format) format. The data is automatically divided into smaller tiles for more efficient processing. The tiling routine determines the maximum buffer required for seamless results and then trims the raster to the correct size before saving. The final results are saved in VRT format.

2. Machine Learning Modelling Development

The data preparation process yielded a large amount of data (ca. 430 GB) which cover different visualisation schemes and resolutions. More precisely, 7 visualisation schemes are considered (DEM, DEM clipped, e2MSTP v2, e3MSTP v2, e4MSTP, SLRM, and VAT flat 3B) and 2 image resolutions (256px and 512px). This means that 14 dataset variants in total were evaluated. As the images for the machine learning analysis are split in patches - each patch is the input data for the deep learning model. Furthermore, the training data contains patches that include objects of interest as well as the labels given as boundary boxes. The validation and the test sets also include patches without objects of interest in order to properly calibrate the trained models. The data format adopted in our analysis follows the widely known format of the COCO dataset for object detection (<https://cocodataset.org>). The extent of the available data is illustrated in Figure 24.

The project is utilising the AiTLAS toolbox (<http://aitlas.bvlabs.ai>) for implementing the needed methods and executing the experiments. The AiTLAS toolbox (Artificial Intelligence Toolbox for Earth Observation) includes state-of-the-art machine learning methods for exploratory and predictive analysis of satellite imagery as well as a repository of AI-ready Earth Observation (EO) datasets. It can be easily applied for a variety of Earth Observation tasks, such as land use and cover classification, crop type prediction, localization of specific objects (semantic segmentation), etc. The main goal of AiTLAS is to facilitate better usability and adoption of novel AI methods (and models) by EO experts, while offering easy access and standardised format of EO datasets to AI experts which allows benchmarking of various existing and novel AI methods tailored for EO data.

The following components of the MLM process have been implemented to date within the project. Code/scripts can be access using the hyperlinks for each stage

1. Enabled the toolbox to read and use data from the COCO format (and the format used in our study) -this is reflected in the following documents:
https://github.com/biasvariancelabs/aitlas/blob/master/aitlas/datasets/object_detection.py
2. Enabled the toolbox for object detection methods and data:
https://github.com/biasvariancelabs/aitlas/blob/master/aitlas/base/object_detection.py
3. Created the training and validation scripts for the object detection methods:
https://github.com/biasvariancelabs/aitlas/blob/master/configs/hrrsd/train_and_evaluate.json
4. Implementation and testing of the FasterRCNN object detection model:
<https://github.com/biasvariancelabs/aitlas/blob/master/aitlas/models/fasterrcnn.py>
5. Creation of a Jupyter notebook to illustrate the complete object detection process (this currently uses a publicly available dataset in order not to reveal the new data too early):
https://github.com/biasvariancelabs/aitlas/blob/master/examples/object_detection_example_hrrsd.ipynb
6. Implementation of additional object detection methods, including:

- a. DeepLabV3
(<https://github.com/biasvariancelabs/aitlas/blob/master/aitlas/models/deeplabv3.py>),
- b. DeepLabV3+
(<https://github.com/biasvariancelabs/aitlas/blob/master/aitlas/models/deeplabv3plus.py>),
- c. Swin Transformers
(https://github.com/biasvariancelabs/aitlas/blob/master/aitlas/models/swin_transformer.py).

Initially the MLM was trained and evaluated using FasterRCNN (using default parameters, and without data augmentations) on a small sample of the data to evaluate the consistency of the implementation and the prepared data. The process was repeated out, executing a larger parametrization of the FasterRCNN using all the available data. Following this a series of extensive experiments took place across the different visualisations and resolutions to compare the different performances including the predictive power and processing efficiency. The experiments for the Fast RCNN completed their first iterative testing cycle in early January 2023. These results informed the selection of optimal visualisation method and provides guidance to any additional experimentation using the additional recommended different deep models.

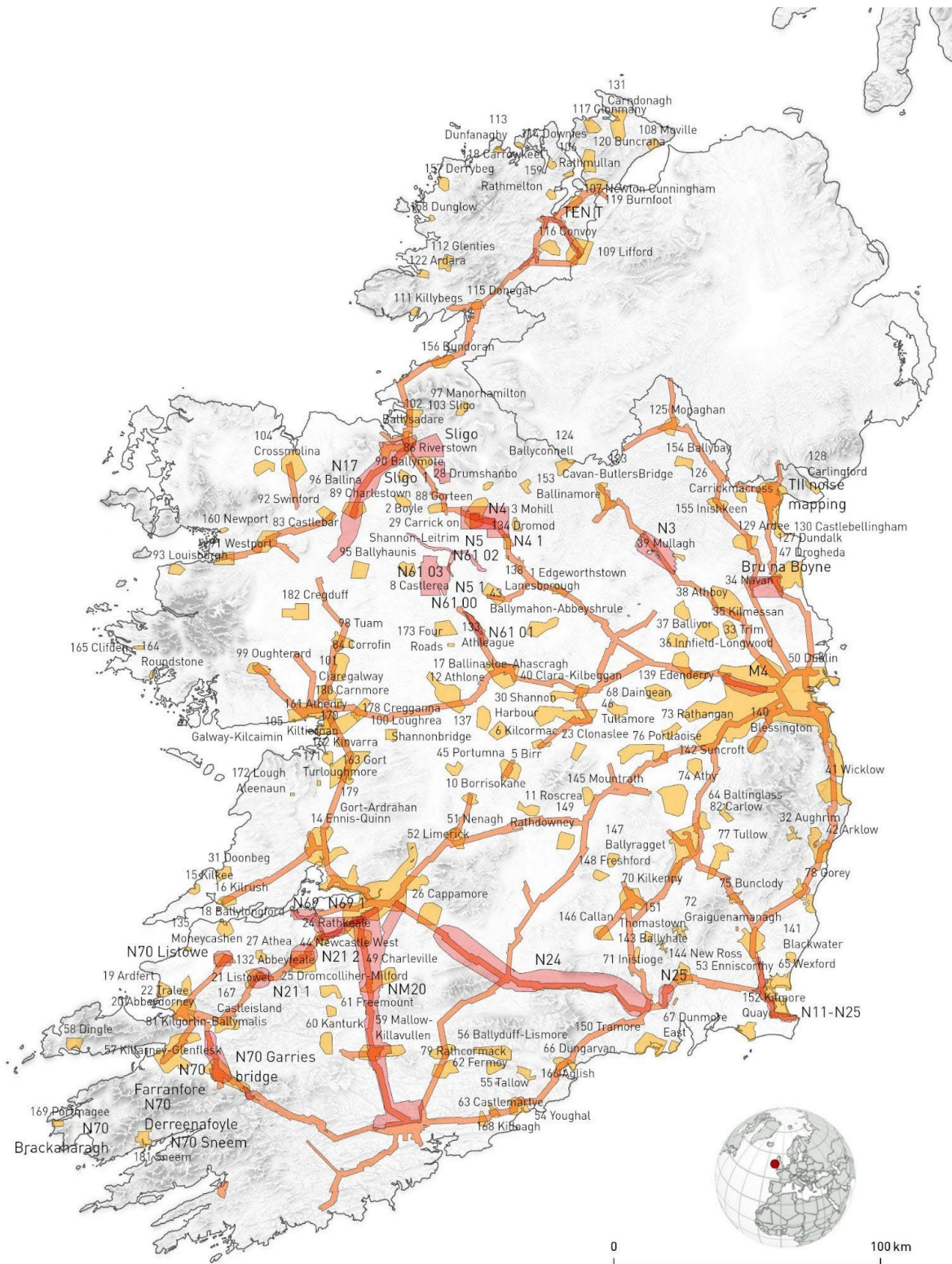


Figure 24: A map with data extents. It shows extents of the elevation model datasets and not the patches.

2.1. Evaluation of FasterRCNN object detection model

In the experiments and analysis undertaken, we have investigated the impact of a number of different dimensions on the performance of the predictive models. The experiments were executed using FasterRCNN.

2.1.1. The impact of the visualization type

Seven different predictive models were trained (one for each visualization type) on the patches with a size of 256 pixels and a DFM labelling 1, 2, and 4. Figures 25 and 26 depict the mAP (mean Average Precision) and mAP₅₀ scores for the predictive models across the learning epochs on the validation set. The models were trained over 20 learning epochs.

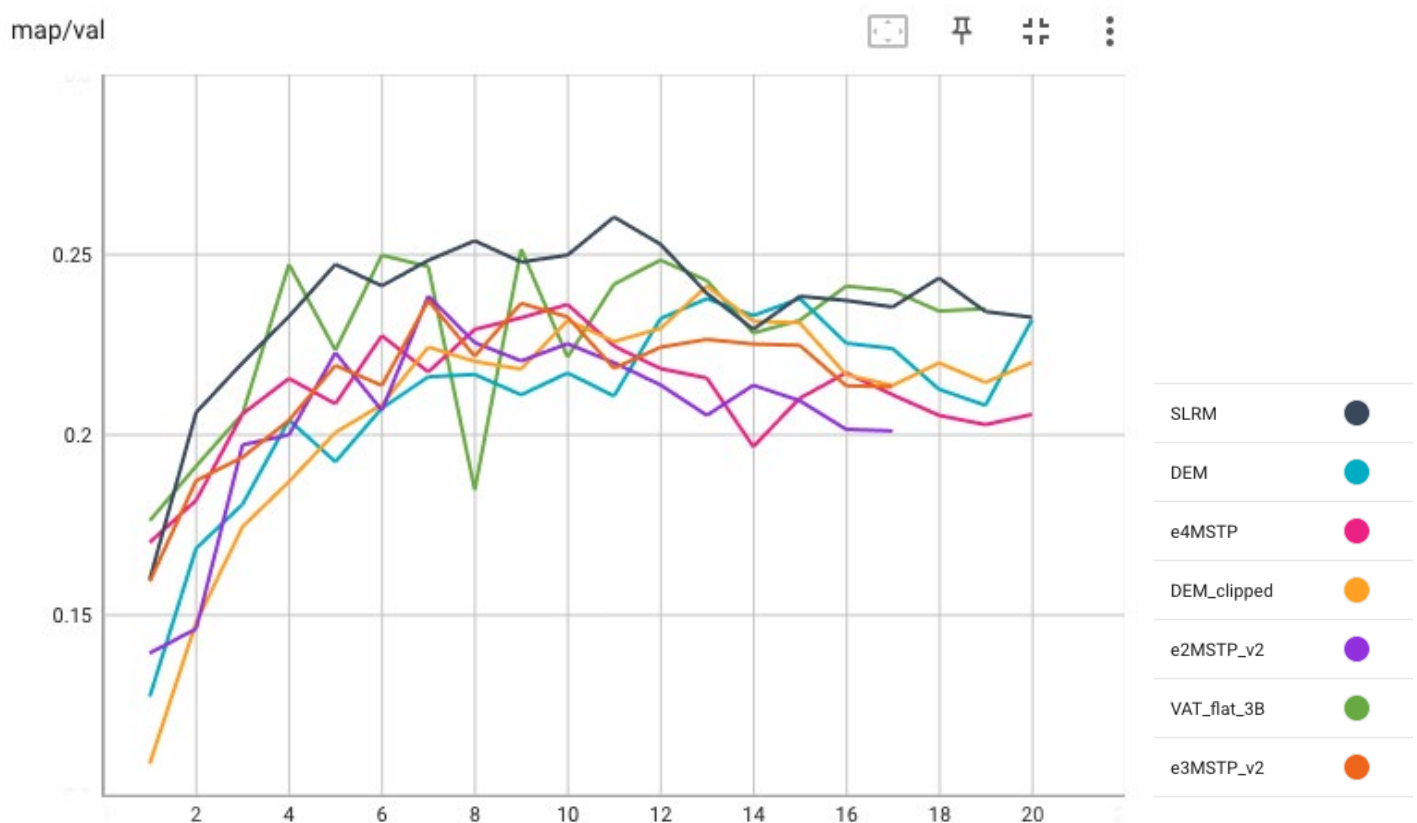


Figure 25: mAP scores of the predictive models using the different visualisations of the data calculated on the validation set across the different 20 learning epochs.

map_50/val

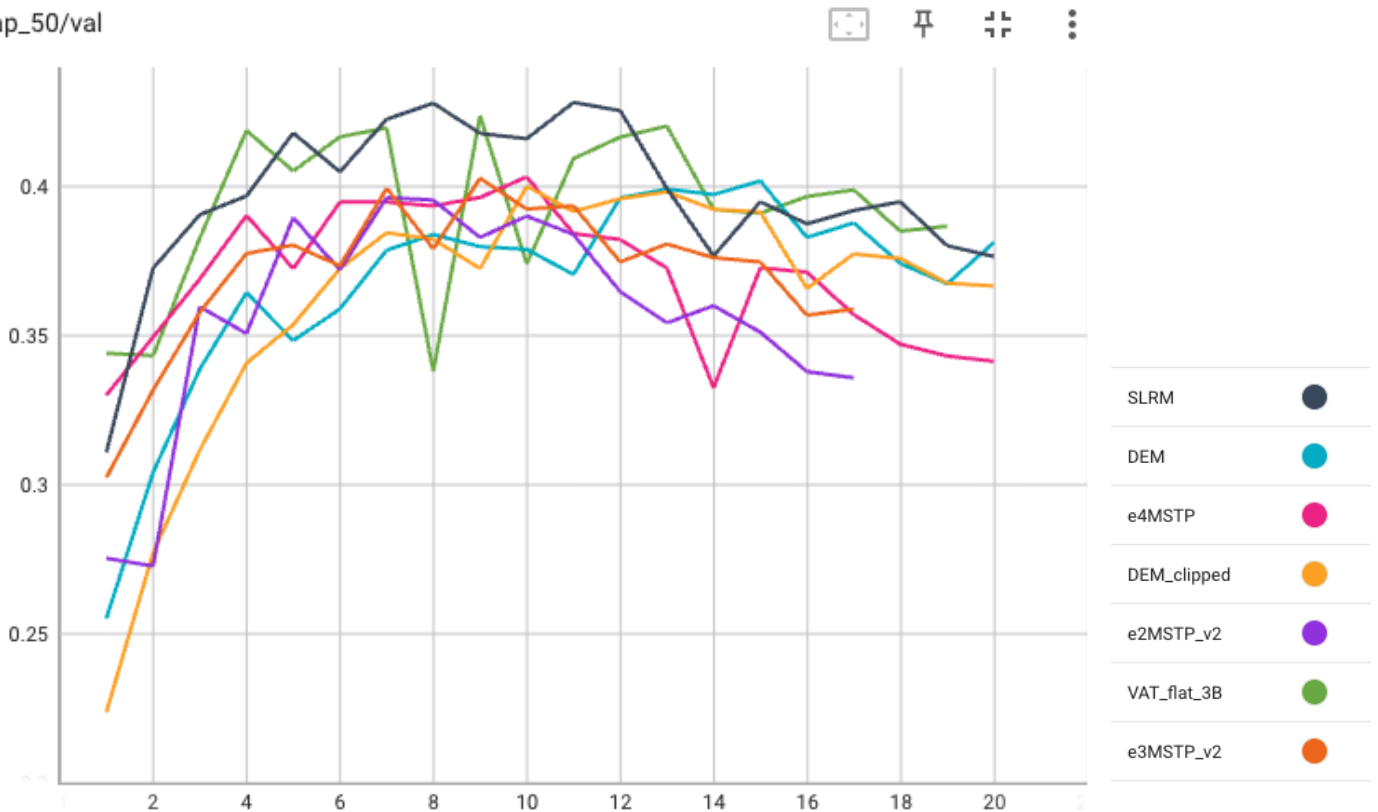


Figure 26: mAP@50 scores of the predictive models using the different visualisations of the data calculated on the validation set across the different 20 learning epochs.

Table 10 illustrates the mAP, mAP₅₀, and mAP₇₅ scores for the predictive models calculated on the test set. The results show that the SLRM visualization type performs the best, but only by a small margin. Therefore, it can be concluded that there is no significant difference in performance among the different visualization types.

Visualisation Method	mAP	mAP ₅₀	mAP ₇₅
SLRM	0.18989	0.34058	0.18785
e ⁴ MSTP	0.16410	0.29772	0.16575
DEM	0.18298	0.33138	0.17697
DEM_clipped	0.18789	0.32998	0.19264
e ² MSTP_v2	0.18449	0.33451	0.18341
e ³ MSTP_v2	0.17845	0.31945	0.18081
VAT_flat_3B	0.17836	0.31330	0.18138

Table 10: Scores of the predictive models across the different visualization types calculated on the test set based on 20 learning epochs.

2.1.2. The impact of the Intersection over Union (IOU) threshold in non-maxima suppression

In the prediction phase of the ML pipeline, a technique called non-maxima suppression (NMS) is employed to filter out multiple bounding boxes that have high overlap with each other. It is often used in object detection models to keep only the bounding boxes that are most likely to contain an object. To determine if two bounding boxes overlap, a IoU threshold is set. The amount of overlap (IoU) is measured in a value between 0 and 1, with 1 being a perfect overlap and 0 having no overlap. The IoU threshold represents the minimum IoU value that two boxes must have to be considered as overlapping. When NMS is applied, it will keep the box with the highest score and eliminate the boxes that have an IoU greater than the IoU threshold with the box that was kept.

In the AiTLAS system, the default value for the IoU threshold is set to 0.3. This threshold was used in the experiments described in the previous subsection. To investigate the impact of the IoU threshold on the performance of the machine learning model (MLM), we conducted experiments with two different threshold values: 0.5 and 0.7. The models were trained using the SLRM data, with a sample size of 256 pixels, over a total of 20 learning epochs. A comparison between the performance of the models under the different IoU threshold values to understand its impact was carried out.

Based on the results presented in Figures 27 and 28 and Table 11, it is concluded that the impact of the IoU threshold is not significant.

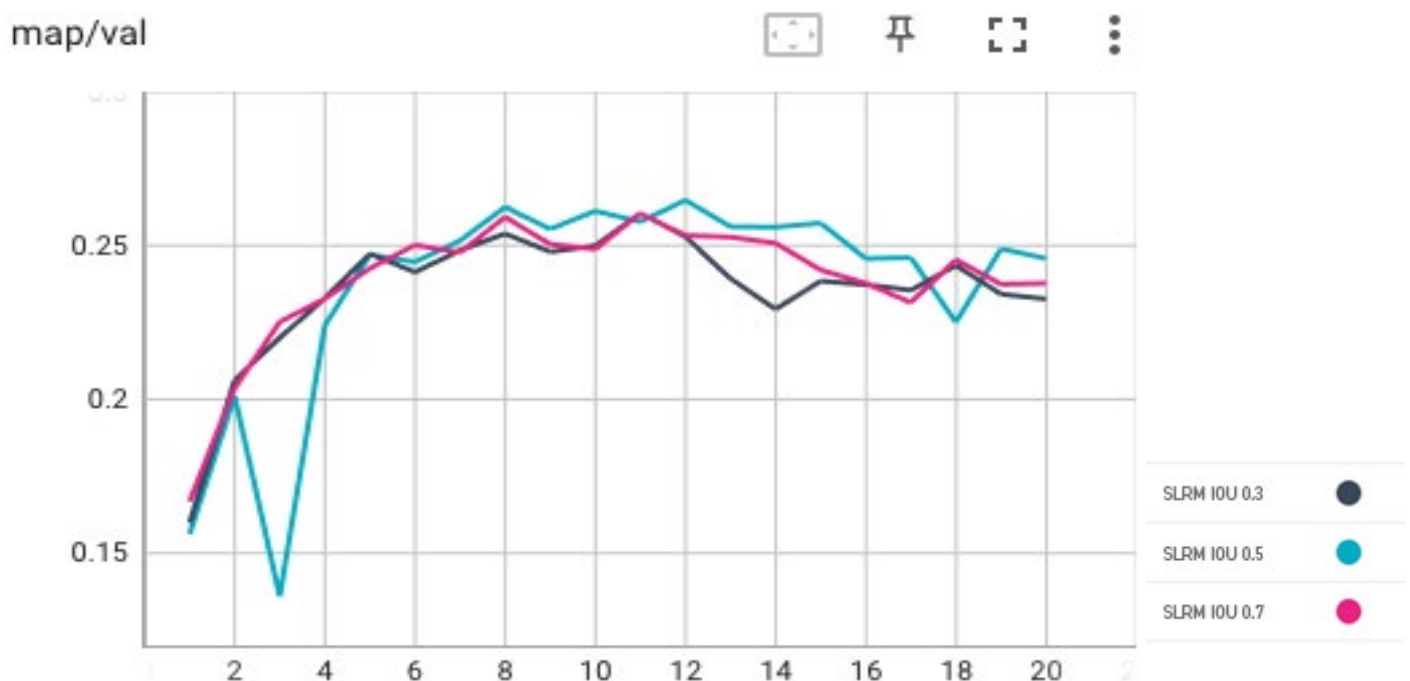


Figure 27: mAP scores of the predictive models calculated on the validation set of the SLRM for the different IoU thresholds across the different 20 learning epochs.

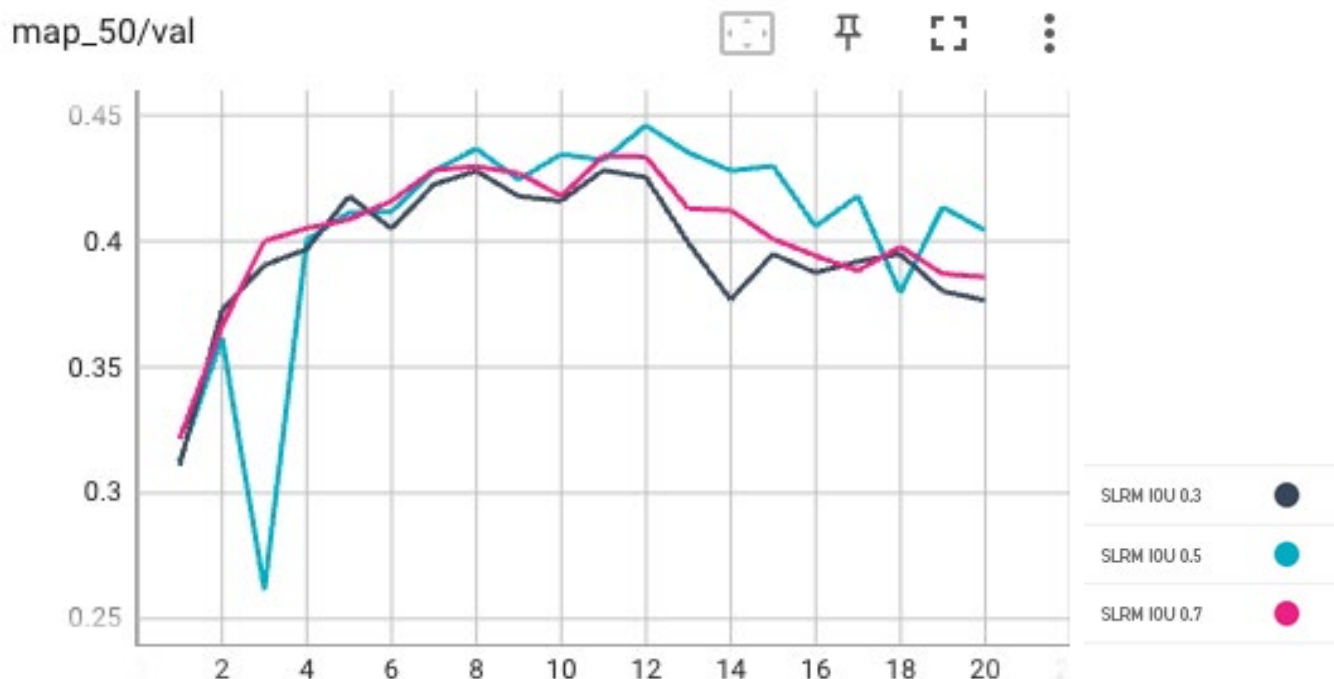


Figure 28: mAP₅₀ scores of the predictive models calculated on the validation set of the SLRM for the different IoU thresholds across the different 20 learning epochs.

	mAP	mAP ₅₀	mAP ₇₅
SLRM_IoU_0.3	0.18989	0.34058	0.18785
SLRM_IoU_0.5	0.18554	0.34004	0.18119
SLRM_IoU_0.7	0.19120	0.34154	0.18864

Table 11: mAP scores of the predictive models calculated on the test set of the SLRM for the different IoU thresholds.

2.1.3. Removing patches annotated with DFM quality attribution of 4

In the first two sets of experiments, all the data provided with sample size of 256 pixels was utilised. This includes all the patches which had been labelled with a DFM quality attribution of 1, 2 and 4. The patches with attribution value of 3 have been filtered out prior to the start of the ML part of the project.

In this set of experiments, we investigated if the exclusion of these patches that were the most challenging for annotation would improve the results. These patches (DFM attribution = 4) contain monument examples which are barely visible and may be contributing noise to the data.

Based on the results presented in Figure 29 and 30 and Table 12 we can observe that there is a boost in the performance.

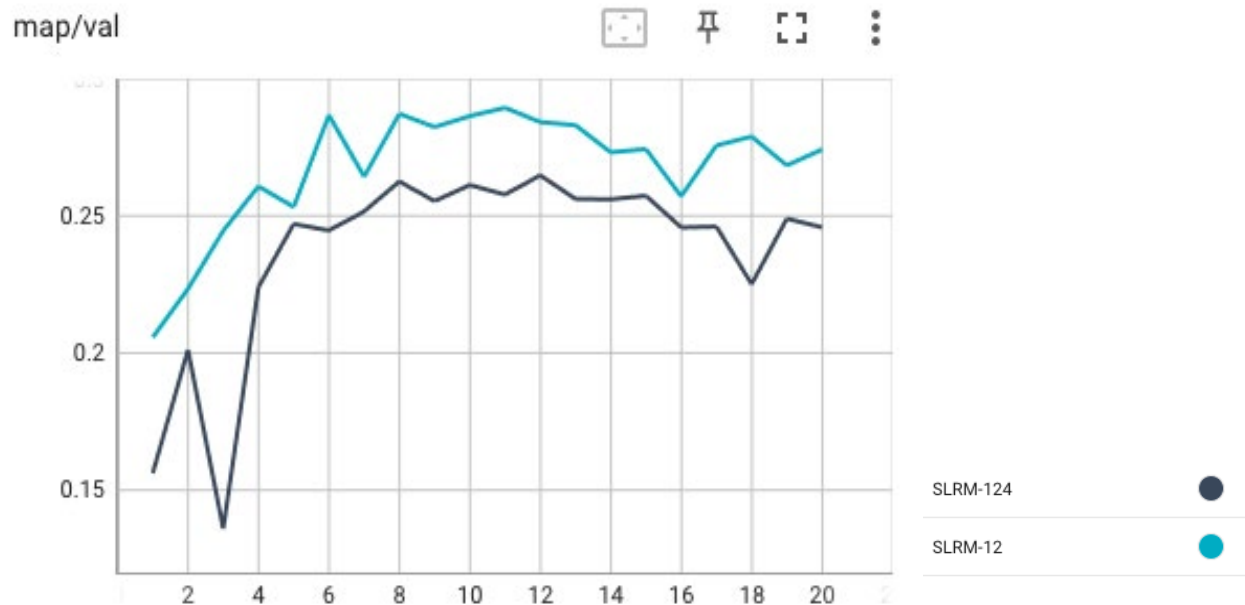


Figure 29: mAP scores of the predictive models calculated on the validation set of the SLRM with 1, 2 and 4 and 1 and 2 level of SFM quality attribution across the different 20 learning epochs.



Figure 30: mAP₅₀ scores of the predictive models calculated on the validation set of the SLRM with 1, 2 and 4 and 1 and 2 level of SFM quality attribution across the different 20 learning epochs.

	mAP	mAP ₅₀	mAP ₇₅
SLRM_1_2_4	0.18554	0.34004	0.18119
SLRM_1_2	0.21747	0.39509	0.21549

Table 12: mAP scores of the predictive models calculated on the validation set of the SLRM with 1, 2 and 4 and 1 and 2 level of SFM quality attribution.

2.1.4. Impact of sample size

The data provided comes in two sample sizes, 256 and 512 pixels. Here, we investigate whether sample size affects the performance of machine learning models. All models are trained on SLRM data over the course of 20 learning epochs with an IoU threshold of 0.5.

Figure 31 and 32 and Table 13 illustrate the results on the validation and test set, respectively. It can be observed that data with a sample size of 256 pixels achieves slightly better performance.

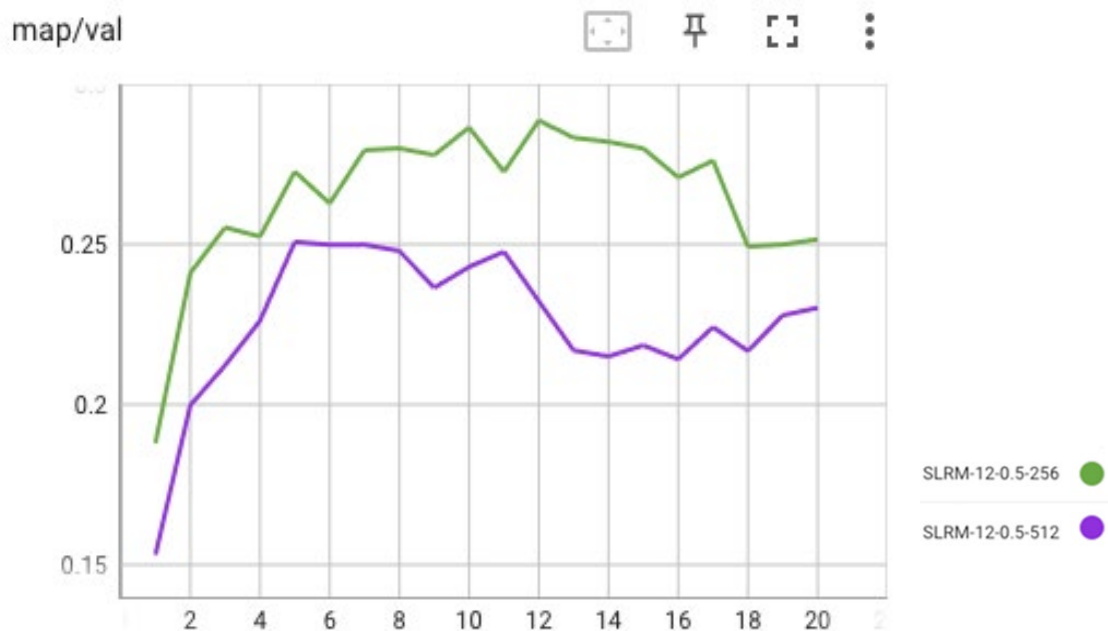


Figure 31: mAP scores of the predictive models calculated on the validation set of the SLRM data with sample size of 256 and 512 pixels across the different 20 learning epochs.



Figure 32: mAP_{50} scores of the predictive models calculated on the validation set of the SLRM data with sample size of 256 and 512 pixels across the different 20 learning epochs.

	mAP	mAP ₅₀	mAP ₇₅
SLRM_256px	0.21747	0.39509	0.21549
SLRM_512px	0.19751	0.35392	0.19508

Table 13: mAP scores of the predictive models calculated on the test set of the SLRM data with sample size of 256 and 512 pixels.

2.1.5. Training predictive models for each object type separately

In the next set of experiments, we aim to determine the efficacy of the MLM in predicting individual object types. To accomplish this, each MLM is trained, validated and tested using only the data that pertains to the specific object of interest. Patches that do not contain annotations of the object in question are discarded. All models are trained on the SLRM data, 256 pixels sample size, 20 learning epochs and 0.5 IoU threshold.

Table 14 shows the map scores for the three object types separately. In this scenario, we used the patches with DFM quality attribution of 1, 2 and 4. It is evident that the highest performance is achieved for ringforts, while the lowest for enclosures. These outcomes align with our expectations, as ringforts present in the ALS data are visually easier to detect compared to barrows and enclosures and have been trained with a significantly more examples.

Monument Classification	mAP	mAP ₅₀	mAP ₇₅
Enclosure	0.18057	0.36139	0.15871
Barrow	0.28129	0.59698	0.24799
Ringfort	0.44068	0.74744	0.47620

Table 14: Test results for the patches for the three chosen monument classifications with 1, 2 and 4 DFM quality attribution.

For the detection of barrows and ringforts, we have considered two additional scenarios: (i) use patches with labelling difficulty of 1 and 2 (results depicted in Table 15.), and (ii) use patches with labelling difficulty of 1 (results depicted in Table 16). As anticipated, the performance of the models improves as the quality of the data increases when the DFM rated 4 items are omitted in the case of barrow. However, for ringfort a slight decrease in detection is observed when DFM 2 rated monuments are removed, possibly reducing the exposure of the MLM to more variable forms of ringforts which are present in the DFM 2 rated data.

Monument Classification	mAP	mAP ₅₀	mAP ₇₅
Barrow	0.32509	0.68895	0.26076
Ringfort	0.48343	0.80029	0.52262

Table 15: Test results for the patches for the barrows and ringforts with 1 and 2 DFM quality attribution.

Monument Classification	mAP	mAP ₅₀	mAP ₇₅
Barrow	0.38630	0.75010	0.35033
Ringfort	0.45898	0.769948	0.48504

Table 16: Test results for the patches for the barrows and ringforts with 1 DFM quality attribution.

2.1.6. Impact of Augmenting the Training Data

We conducted an extensive analysis to assess the impact of employing transformations and augmentations on the performance of deep learning models in the context of segmentation tasks. These two techniques differ in their application: transformations are applied dynamically during the training process, altering images on-the-fly within each iteration of the deep learning algorithm, while augmentations involve creating physical copies of the training images. It's noteworthy that the use of transformations has become a standard practice due to its efficiency and effectiveness.

For transformations, we leveraged the capabilities of the AiTLAS toolbox's `aitlas.transforms` module, available at <https://aitlas.bvlabs.ai/>. Specifically, we employed two combinations of transformations: the first encompassed horizontal and vertical flips along with random rotations, while the second included additional colour transformations on top of the first set. Conversely, augmentations involved manipulating the data through horizontal and vertical flipping, rescaling, and rotations.

Our experiments focused on image segmentation, utilizing the best-performing method, HRNet, with SLRM visualization, a resolution of 256px, and starting from pre-trained weights. The results, as presented in Table 17, demonstrated that employing only the first combination of transformations yielded the most significant improvement in overall performance.

Augmentation Description	Barrow	Enclosure	Ringfort	mIoU
No-transformation	0.268	0.137	0.455	0.287
FlipHVRandomRotate	0.398	0.210	0.520	0.376
FlipHVRandomRotate+ColourTransformations	0.409	0.000	0.519	0.309

Table 17: Evaluation of two combinations of transformations on the training images.

In addition, we conducted a comparative analysis to evaluate the impact of applying transformations and/or augmentations on the predictive performance of the models. As shown in Table 18, employing only transformations not only delivered superior predictive performance but also proved to be more resource-efficient, underscoring its efficacy as a preferred approach in model training.

Transformation	Augmentation	Barrow	Enclosure	Ringfort	mIoU
TRUE	FALSE	0.5851	0.4733	0.3292	0.4877
FALSE	FALSE	0.5299	0.4082	0.2552	0.4520
TRUE	TRUE	0.5618	0.4733	0.2523	0.4701
FALSE	TRUE	0.4892	0.4304	0.2564	0.4500

Table 18: Evaluation of the performance of transformations and augmentations executed on 512px resolution images including the updated masks by the experts.

2.1.7. Testing of MLM Architectures

We conducted a thorough examination of four distinct architectures for image segmentation—DeepLabV3, U-Net, HRNet, and Mask2Former with Swin S, alongside one architecture for object detection, FasterRCNN. These selections were meticulously made based on their well-documented state-of-the-art performance across diverse image analysis domains.

Table 19 shows the performance of three segmentation methods DeepLabV3, U-Net, and HRNet using pre-trained weights, 50 epochs, 0.0001 learning rate, with SLRM visualization at 256px, whole range of DFM (annotation quality) and data transformation. The results indicate that HRNet yields the best performance.

MLM	Barrow	Enclosure	Ringfort	mIoU
DeepLabV3	0.344	0.168	0.509	0.340
U-Net	0.354	0.000	0.518	0.291
HRNet	0.398	0.210	0.520	0.376

Table 19: Performance comparison of DeepLabV3, UNet and HRNet.

We then tested the performance of the Mask2Former with Swin S architecture under similar conditions as the other methods, the only difference being the resolution – in this case we used the 512px patch size for improved efficiency during testing. Comparing the performance of HRNet and Mask2Former with Swin S we conclude that the performance is very similar, with the latter being a more complex and more resource demanding method. Considering all of this, we selected HRNet as the model to deploy in the ADAF toolbox.

MLM	Barrow	Enclosure	Ringfort
Mask2Former with Swin S	0.5939	0.5153	0.3614
HRNet	0.5753	0.5043	0.2817

Table 20: Comparison of performance between Mask2former with Swin S and HRNet at 512px resolution as measured with IoU.

2.2. Final Specifications of ADAF MLM

Following the testing of the different MLMs, eight different ADAF were finalised with the MLM split into two categories: four MLMs for the category of object detection and four MLMs for the category of semantic segmentation. Each type of archaeological site - barrows, enclosures, ringforts - has a dedicated MLM, plus there is one MLM that was trained on all the archaeological examples (AO) and treats the three classes as one.

For the semantic segmentation MLMs, the HRNet architecture available in the AiTLAS toolbox was selected for use. These models are trained on data with DFM quality levels 1 and 2 as indicated by the Project Archaeologist and on patches that contain only the specific objects the models are trained for. For example, when we train a model for barrows, the training data contains only data about barrows with DFM qualities 1 and 2. For validation and testing, we broaden the data to include DFM qualities 1, 2, and 4, and we use patches with various types of objects. The validation data for semantic segmentation integrates the domain expert feedback based on the analysis of the predictions of our previous models.

For the object detection MLMs, we use the FasterRCNN architecture available in the AiTLAS toolbox was selected for use. These models are trained on data with DFM quality levels 1 and 2 as indicated by the Project Archaeologist and on patches that contain only the specific objects the models are trained for. For example, when we train a model for barrows, the training data contains only data about barrows with DFM qualities 1 and 2. For validation we broadened the data to include DFM qualities 1, 2, and 4, and we use patches that contain only the specific objects the models are trained for. The validation data for object detection does not integrate domain expert feedback. For testing all classes of DFM attributes (1, 2, 3 and 4) were considered to make the results as close to those seen in reality (i.e. some DFM 3 data will exist in future ALS surveys)

Finally, in both learning scenarios (object detection and semantic segmentation) all our data is sourced from the SLRM visualisation, using a 256px patch size. Additional augmentation of the training data was also carried out in order to increase the number of training dataset examples utilising several data transformations including random horizontal and vertical flips and random rotations. The learning rate is set at 10^{-4} for optimal results. All ADAF MLMs are trained over the course of 50 epochs.

3. Evaluating the MLM

3.1. Domain Expert Evaluation

Following the completion of the experimentation stages of developing the MLM as outlined in Section 2, detailed annotations of predicted results were produced by each of the MLMs (Ringfort, Barrow, Enclosure, Any Archaeology (AO)). The resulting polygon segmentation annotations results were evaluated by domain experts using a three staged approach

3.1.1. Known Archaeology Validation

The first stage of expert validation was the comparison of the segmentation annotation polygons produced by each MLM against the known archaeology as represented by the SMR. For each SMR point a 5m buffer was generated representing a minimal monument footprint. Within QGIS overlay analysis was processed identifying annotations which intersected with SMR buffers. For each annotation intersected monument classifications were attributed. The results for each of the MLMs summarises the following details:

- Total number of ringforts identified.
- Total number of enclosures identified.
- Total number of barrows identified.
- Total number of either ringforts, barrows or enclosures identified (labelled as merged).
- Total number of other archaeology monument classes identified.

MLM	Total No. Annotations	Ringfort	Enclosure	Barrow	Merged	Other Class
Ringfort	1,054	658	159	26	843	34
Enclosures	906	605	151	18	774	32
Barrows	643	338	68	52	458	14
AO	1,307	711	241	55	1,007	38

Table 21: Summary of the comparison of segmentation prediction annotations against the occurrence of National Monuments SMR within the annotations

The results indicate that the AO MLM produces the greatest number of annotations, followed by ringforts, enclosures and barrows positively identify 1,045 known monuments, with 38 of these being outside the scope of this project.

3.1.2. New Positives Evaluation

For those segmentation annotations which are located on areas which are currently not identified as archaeological sites or monuments a process was carried out to distinguish which of these features are potential archaeology sites. Using QGIS, each of the segmentation annotations generated by each MLM was examined against a range of geospatial datasets (original ALS data, orthoimagery, historic mapping) and assigned an attribute:

- Potential archaeological sites/monuments (strong positives)
- Possible archaeological sites/monuments (weak positives)
- No archaeological sites/monuments evidence (false positives)

Examples of strong and weak positives identified within the ALS data are illustrated in Figure 33. The results of the evaluation are summarised in Table 22. One of the immediate observations from this stage of evaluations was the high number of false positives which are caused by either very small and long, thin features often observed as vegetation and scrub such as hedgerows. Filtering of these annotations was incorporated into the later stages of the ADAF tool. Following further inspection, the causes of some of the false positives within the annotations include the identification of several landscape structures including golf course

putting greens (see Figure 33), road roundabouts, quarried features and a range of farm structures such.

MLM	Total No. potential new sites	Strong positives	Weak positives	False positives
Ringfort	180	31	32	117
Enclosures	104	38	29	37
Barrows	172	19	41	112
AO	260	33	58	169

Table 22: Summary of the evaluation of potential new sites identified by each of the MLMs.

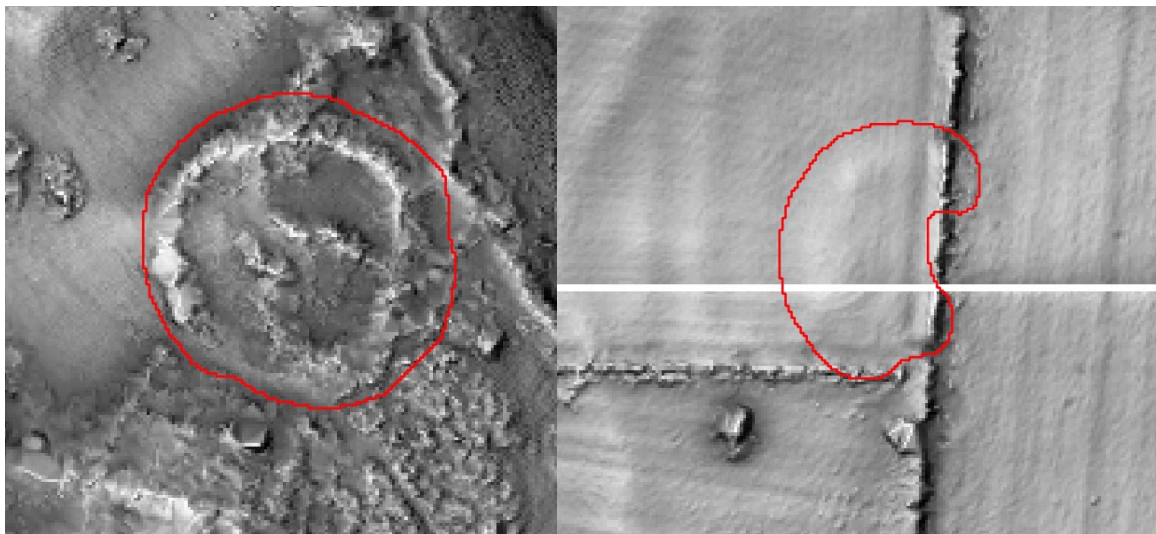


Figure 33: Examples of strong positive (left) and weak positive (right) identified within the segmentation annotations (red).



Figure 34: Example of a false positive in the segmentation annotations (red) caused by the identification of a putting green on a golf course.

Validated Segmentation Evaluation

The final stage of the domain expert evaluation was to compare the annotation results of each MLM against the remaining testing polygons (training data within the green areas of Figure 19). The result of this evaluation provides the overall effectiveness of each MLM but enables the opportunity to review the DFM rating of those sites identified and missed with the aim of informing potential improvements in the MLM. The results for each monument type and the full collection of merged validation polygons were calculated along with the associated DFM ratings. The results are summarised in Tables 23 to 34.

Analysis against 866 known ringforts	No.	Percentage
Number of Ringfort sites correctly identified by AO MLM	715	82.56%
Number of Ringfort sites missed by AO MLM	151	17.44%
Number of Ringfort sites correctly identified by RINGFORT MLM	665	76.79%
Number of Ringfort sites missed by RINGFORT MLM	201	23.21%
Number of Ringfort sites correctly identified by ENCLOSURE MLM	617	71.25%
Number of Ringfort sites missed by ENCLOSURE MLM	249	28.75%
Number of Ringfort sites correctly identified by BARROW MLM	329	37.99%
Number of Ringfort sites missed by BARROW MLM	537	62.01%

Table 23: Summary of the analysis of MLM segmentation annotation of ringforts against domain expert validation polygons for each of the MLMs.

MLM	DFM rating of ALS data				Analysis
	1	2	3	4	
AO	356	316	15	28	715
	41.11%	36.49%	1.73%	3.23%	82.56%
Ringforts	338	304	13	10	665
	39.03%	35.10%	1.50%	1.15%	76.79%
Enclosures	328	270	8	11	617
	37.88%	31.18%	0.92%	1.27%	71.25%
Barrows	188	135	3	3	329
	21.71%	15.59%	0.35%	0.35%	37.99%

Table 24: Summary of the analysis of DFM rating of ALS data for each of the MLMs for correctly identified ringforts.

MLM	DFM rating of ALS data				Analysis
	1	2	3	4	
AO	15	11	94	31	151
	1.73%	1.27%	10.85%	3.58%	17.44%
Ringforts	33	23	96	49	201
	3.81%	2.66%	11.09%	5.66%	23.21%
Enclosures	43	57	101	48	249
	4.97%	6.58%	11.66%	5.54%	28.75%
Barrows	183	192	106	56	537
	21.13%	22.17%	12.24%	6.47%	62.01%

Table 25: Summary of the analysis of DFM rating of ALS data for each of the MLMs for missed ringforts.

Analysis against 400 known enclosures	No.	Percentage
Number of Enclosure sites correctly identified by AO MLM	234	58.50%
Number of Enclosure sites missed by AO MLM	166	41.50%
Number of Enclosure sites correctly identified by RINGFORT MLM	156	39.00%
Number of Enclosure sites missed by RINGFORT MLM	244	61.00%
Number of Enclosure sites correctly identified by ENCLOSURE MLM	146	36.50%
Number of Enclosure sites missed by ENCLOSURE MLM	254	63.50%
Number of Enclosure sites correctly identified by BARROW MLM	65	16.25%
Number of Enclosure sites missed by BARROW MLM	335	83.75%

Table 26: Summary of the analysis of MLM segmentation annotation of enclosures against domain expert validation polygons for each of the MLMs.

MLM	DFM rating of ALS data				Analysis
	1	2	3	4	
AO	151	54	0	29	234
	37.75%	13.50%	0.00%	7.25%	58.50%
Ringforts	103	42	0	11	156
	25.75%	10.50%	0.00%	2.75%	39.00%
Enclosures	104	34	0	8	146
	26.00%	8.50%	0.00%	2.00%	36.50%
Barrows	46	14	0	5	65
	11.50%	3.50%	0.00%	1.25%	16.25%

Table 27: Summary of the analysis of DFM rating of ALS data for each of the MLMs for correctly identified enclosures.

MLM	DFM rating of ALS data				Analysis
	1	2	3	4	
AO	43	20	47	56	166
	10.75%	5.00%	11.75%	14.00%	41.50%
Ringforts	91	32	47	74	244
	22.75%	8.00%	11.75%	18.50%	61.00%
Enclosures	90	40	47	77	254
	22.50%	10.00%	11.75%	19.25%	63.50%
Barrows	148	60	47	80	335
	37.00%	15.00%	11.75%	20.00%	83.75%

Table 28: Summary of the analysis of DFM rating of ALS data for each of the MLMs for missed enclosures.

Analysis against 96 known barrows	No.	Percentage
Number of Barrow sites correctly identified by AO MLM	57	59.38%
Number of Barrow sites missed by AO MLM	39	40.63%
Number of Barrow sites correctly identified by RINGFORT MLM	25	26.04%
Number of Barrow sites missed by RINGFORT MLM	71	73.96%
Number of Barrow sites correctly identified by ENCLOSURE MLM	18	18.75%
Number of Barrow sites missed by ENCLOSURE MLM	78	81.25%
Number of Barrow sites correctly identified by BARROW MLM	54	56.25%
Number of Barrow sites missed by BARROW MLM	42	43.75%

Table 29: Summary of the analysis of MLM segmentation annotation of barrows against domain expert validation polygons for each of the MLMs.

MLM	DFM rating of ALS data				Analysis
	1	2	3	4	
AO	38	18	0	1	57
	39.58%	18.75%	0.00%	1.04%	59.38%
Ringforts	18	6	1	0	25
	18.75%	6.25%	1.04%	0.00%	26.04%
Enclosures	13	5	0	0	18
	13.54%	5.21%	0.00%	0.00%	18.75%
Barrows	40	12	1	1	54
	41.67%	12.50%	1.04%	1.04%	56.25%

Table 30: Summary of the analysis of DFM rating of ALS data for each of the MLMs for correctly identified barrows.

MLM	DFM rating of ALS data				Analysis
	1	2	3	4	
AO	11	8	11	9	39
	11.46%	8.33%	11.46%	9.38%	40.63%
Ringforts	31	20	10	10	71
	32.29%	20.83%	10.42%	10.42%	73.96%
Enclosures	36	21	11	10	78
	37.50%	21.88%	11.46%	10.42%	81.25%
Barrows	9	14	10	9	42
	9.38%	14.58%	10.42%	9.38%	43.75

Table 31: Summary of the analysis of DFM rating of ALS data for each of the MLMs for missed barrows.

Analysis against 1,362 merged features	No.	Percentage
Number of sites correctly identified by AO MLM	1006	73.22%
Number of sites missed by AO MLM	368	26.78%
Number of sites correctly identified by RINGFORT MLM	846	61.57%
Number of sites missed by RINGFORT MLM	528	38.43%
Number of sites correctly identified by ENCLOSURE MLM	781	56.84%
Number of sites missed by ENCLOSURE MLM	593	43.16%
Number of sites correctly identified by BARROW MLM	448	32.61%
Number of sites missed by BARROW MLM	926	67.39%

Table 32: Summary of the analysis of MLM segmentation annotation of ringforts, enclosures and barrows against domain expert validation polygons for each of the MLMs.

MLM	DFM rating of ALS data				Analysis
	1	2	3	4	
AO	545	388	15	58	1006
	40.01%	28.49%	1.10%	4.26%	73.86%
Ringforts	459	352	14	21	846
	33.70%	25.84%	1.03%	1.54%	62.11%
Enclosures	445	309	8	19	781
	32.67%	22.69%	0.59%	1.40%	57.34%
Barrows	274	161	4	9	448
	20.12%	11.82%	0.29%	0.66%	32.89%

Table 33: Summary of the analysis of DFM rating of ALS data for each of the MLMs for correctly identified ringforts, enclosures and barrows.

MLM	DFM rating of ALS data				Analysis
	1	2	3	4	
AO	76	44	152	96	368
	5.58%	3.23%	11.16%	7.05%	27.02%
Ringforts	162	80	153	133	528
	11.89%	5.87%	11.23%	9.77%	38.77%
Enclosures	176	123	159	135	593
	12.92%	9.03%	11.67%	9.91%	43.54%
Barrows	347	271	163	145	926
	25.48%	19.90%	11.97%	10.65%	67.99%

Table 34: Summary of the analysis of DFM rating of ALS data for each of the MLMs for missed ringforts, Enclosures and barrows.

3.1.3. Domain Expert Evaluation Summary

Based on the domain expert evaluation, several factors must be considered when applying the use of the ADAF tool

- The AO MLM is the most effective at identifying archaeological features in ALS data through the process of segmentation and therefore should be recommended to users as the default MLM to select. Form observations in the data, detection bounding boxes were created around each segmentation and therefore both should be used in the default setting by users.
- For those monuments which were not detected, the majority of them were represented in the ALS data as DFM 3 or DFM 4 datasets, highlighting the value on high quality ALS data in future analysis and effective employment of the ADAF tool.
- Small features in the ALS data often caused by vegetation clumps and hedgerows require filtering to reduce the number of false positives.

3.2. Incorporation of Evaluation in MLM ADAF

After the first set of experiments, we provided detailed annotations produced by the model for the validation part of the images to the domain experts. They examined the annotations and provided updates to the masks. We repeated the experiments under the same conditions for the model (HRNet) with the only difference being the updated annotations.

The results, as illustrated in Table 35, reveal that despite the incorporation of updated annotations, there were no discernible improvements in overall performance. Notably, while the model exhibited a marginal enhancement in performance for segmenting enclosures—an inherently challenging class to accurately delineate—it concurrently experienced a slight reduction in performance for identifying barrows and the AO class. However, it is imperative to note that these performance differentials were not of significant magnitude.

This iterative process underscores the importance of expert input in refining model outputs, yet it also highlights the nuanced challenges associated with achieving substantial performance gains. Despite the observed nuances in performance across specific classes, the overall impact on model efficacy was deemed minimal.

MLM	Ringfort	Barrow	Enclosure	AO
Original labels/masks	0.5850	0.4855	0.3216	0.5037
Updated labels/masks	0.5851	0.4733	0.3292	0.4855

Table 35: Comparison of performance of the model using expert feedback.

4. ADAF Tool Development

4.1. Background to Technology/Software Solution

The automatic detection of land features in ALS data using machine learning methods is a complex task. Several steps are required to develop a robust workflow that is both user-friendly and provides accurate and reliable results across an intricate domain of aerial archaeology. To create a state of the art product, many different software packages, processing methods and quality controls had to be integrated into a single workflow, including a carefully designed user interface that allows users without programming skills to access the technology. ADAF tool consists of two parts, a graphical user interface (GUI) in the form of a Jupyter Notebook widget and Python code in the background that handles all the processing. Below we explain why certain features were implemented, what benefits they bring and what problems they solve.

4.1.1. Background technology

Anaconda

Anaconda is a Python distribution for scientific computing that aims to simplify package management and deployment. ADAF leverages Anaconda's virtual environments to install and access various Python packages that are critical for machine learning and geospatial data processing.

<https://www.anaconda.com/>

Jupyter

The Jupyter Notebook is a web-based platform for interactive data processing. The notebook combines live code, equations, narrative text, visualisations, widgets, etc. It is popular for all kinds of data science tasks such as exploratory data analysis, data cleaning and transformation, data visualisation, statistical modelling, machine learning and deep learning. ADAF's user interface consists of Jupyter Widgets that provide interactive access to ADAF's processing functions. Jupyter Notebooks also allow you to run code on a cell-by-cell basis, which is excellent for exploring data and experimenting with different techniques. Python scripts, on the other hand, must be executed as a whole and are more difficult to modify. This kind of flexibility is particularly welcome when training machine learning models, which is also leveraged by ADAF. A number of Jupyter Notebooks are used to guide users in creating new training data from their own datasets, as well as training new and retraining ADAF's default machine learning models.

<https://jupyter.org/>

Relief Visualisation Toolbox

RVT is a free and open source software (Apache license v.2.0, <http://www.apache.org/licenses>) designed to help quickly visualise raster elevation model datasets with techniques that have proven to be effective for identification of small-scale terrain features. It is available as a standalone version, and open source Python library (rvt_py) that is available as a Python Package Index (PyPi) and Conda release. A QGIS Python plug-in and ArcGIS raster functions were developed to help scientists and practitioners with their workflows. The plugin and raster functions are a way to extend the base functionality of individual geographic information

systems using their separate Python APIs (Application Programming Interfaces). RVT provides a toolset for fast calculation of visualisations on large volumes of data, both for large individual datasets and for a large number of different datasets, which is also accessible and understandable for less technically experienced users. The RVT package is installed in the ADAF environment and is integrated into the ADAF workflow for machine learning data preparation. It can also be accessed via Jupyter Noetbooks and used to create custom visualisations for training machine learning models if required.

https://github.com/EarthObservation/RVT_py

AiTLAS

The machine learning backbone of the ADAF toolbox is based on AiTLAS (Artificial Intelligence Toolbox for Earth Observation). It includes state-of-the-art machine learning methods for exploratory and predictive analysis of satellite imagery as well as a repository of AI-ready EO datasets. It can be easily applied for a variety of EO tasks, such as land use and cover classification, crop type prediction, localization of specific objects (semantic segmentation), etc. The main goal of AiTLAS is to facilitate better usability and adoption of novel AI methods (and models) by EO experts, while offering easy access and standardised format of EO datasets to AI experts which allows benchmarking of various existing and novel AI methods tailored for EO data.

<https://github.com/biasvariancelabs/aitlas>

4.2. ADAF Software Processing Steps

The diagram below explains how the processing runs in the background. The processing sequence follows the same three steps into which the input options are divided.

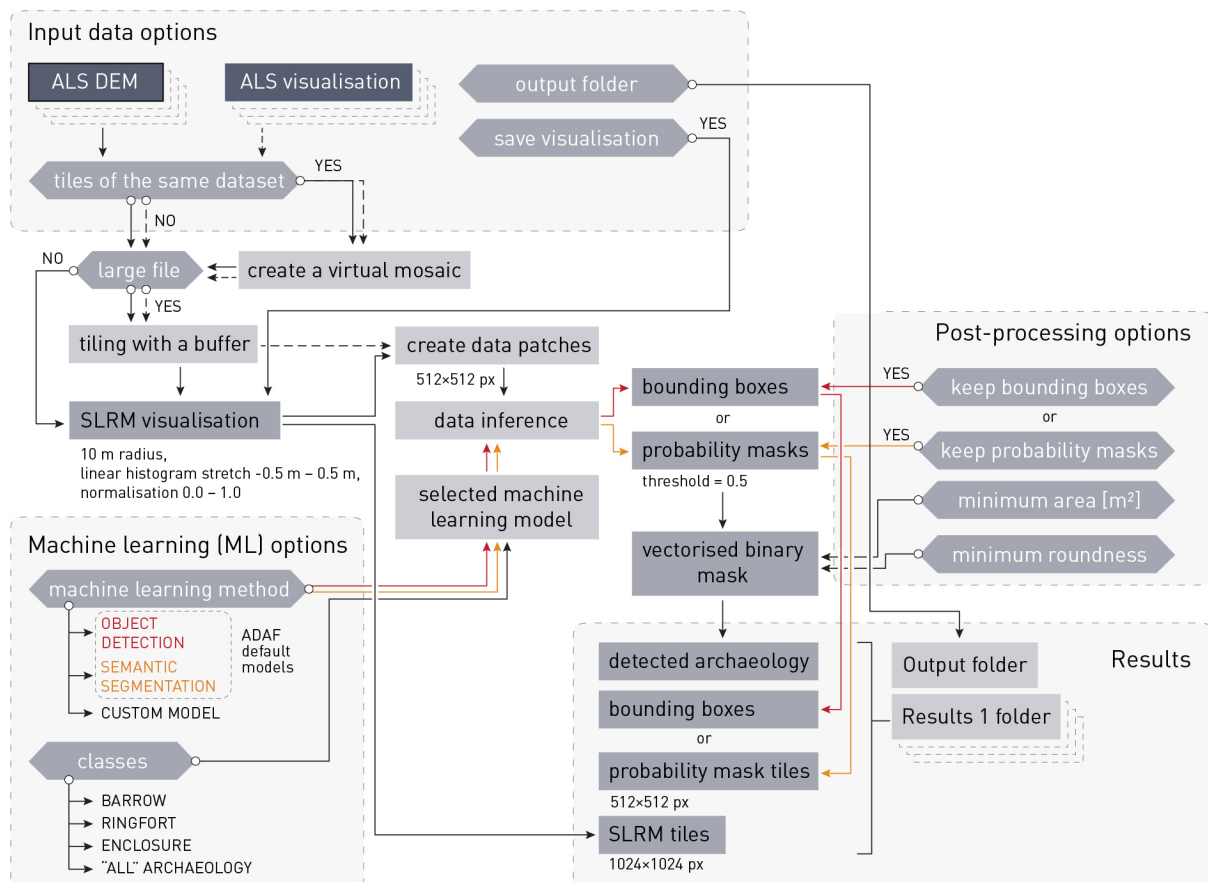


Figure 35: Process diagram outlining the complete processing workflow and sequencing of the ADAF tool.

4.2.1. Processing Stage 1: Input Data Options

The first step is to pre-process the data to ensure that the input data is ML-readable. Depending on the type of input data, different pre-processing steps are undertaken. ADAF accepts input files in three different formats: a single GeoTIFF file, a virtual mosaic (VRT) or a list of tiles from the same dataset converted on-the-fly into a virtual mosaic. If you specify a DEM as input, the required visualisation is calculated in the next step. This is the Simple Local Relief Model (SLRM) visualisation, which is calculated with `rvt_py` using a radius of 10 m, a linear stretch between -0.5 m and 0.5 m and a normalisation between the values 0 and 1. This step can be skipped if the visualisations have already been created externally. While the values for the radius and linear stretch are only recommendations based on our analysis, the normalisation values are a strict requirement on the input data for machine learning. The final step is to cut the data into square tiles of 512 pixels. ML processing can accept any tile size with a base of two (2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, etc.). However, it has been shown that a tile size of 512 pixels provides the best balance between memory constraints and processing speed.

4.2.2. Processing Stage 2: ML Options

The second step of processing performs inference with the selected machine learning model. The workflow is the same regardless of which machine learning model is used. However, the two different methods (i.e. semantic segmentation and object detection) yield results in two different formats. Semantic segmentation outputs probability masks where each pixel of the image is assigned an individual probability score. The threshold value of 0.5 is used to divide the pixels into positive results and background. Contiguous groups of pixels are used to extract polygons that are stored in an output vector file. Instead of a probability mask, object detection outputs text files with bounding boxes and associated probability scores. A threshold value of 0.5 is also used here and the bounding boxes are converted into polygons.

4.2.3. Processing Stage 3: Post Processing Options

The third step is important to reduce the number of false positive detections from the results. The vectors that do not meet the restrictions set by the user are removed before the final results are saved to disc. The minimum area is set to 40 m² by default. This means that segments smaller than 40 pixels (assuming 1 m raster) will be removed. However, the user can change this restriction depending on the size of the archaeological features expected in the area under investigation. The minimum area is applied to both ML methods, while the roundness parameter is only used in semantic segmentation, as object detection only ever outputs square bounding boxes. The roundness is expressed mathematically by dividing the area by the square of the perimeter of the polygon and normalising by multiplying the value by 4π . More rounded objects have a value closer to 1, while longer, thin, flat objects have a roundness value closer to 0. The default setting for this is 0.5, but if the results provide too many long, thin polygons that do not represent archaeological features, the user should increase the roundness threshold.

4.3. GUI development

The GUI has been adapted to the needs of the user with an intuitive interface design, significantly lowering the entry barrier to ML in archaeology for users without programming skills. The main task of the GUI is to collect all required input parameters and select the desired processing method. The GUI sends all the collected data to the backend and starts the processing. When processing is complete, the user receives a message at the bottom of the widget.

There are three groups of input parameters that control three main phases of automatic detection. The *Input data options* require the user to specify the ALS input data, the *Machine learning options* allow them to select the desired ML method and model, and the *Post-processing options* can be used to clean up the final results. Below you will find a list of all parameters for quick reference. A detailed description of the individual parameters can be found in the ADAF user manual.

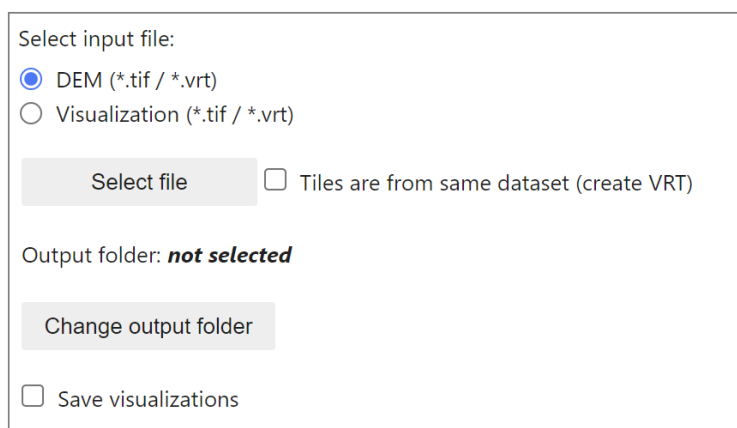
4.3.1. Input Data Options

Figure 36: ADAF GUI component for the selection of input data and options available.

Input data options:

- Select input file
 - Choose between **DEM** and **Visualisation** (GeoTIFF file(s) or VRT file)
 - The **Select file button** opens select file(s) dialog window
 - **Checkbox** that allows creation of VRT file on-the-fly if GeoTIFF tiles are given
- Output folder
 - Path to the default output folder is displayed here
 - The **Change output folder button** opens the select folder dialog window if

Input data options:



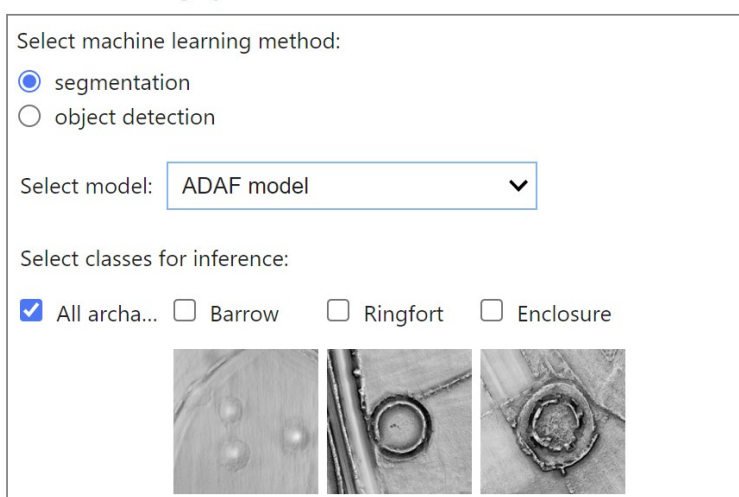
The screenshot shows a GUI panel titled "Input data options:". It contains the following elements: a label "Select input file:" followed by two radio buttons, "DEM (*.tif / *.vrt)" (which is selected) and "Visualization (*.tif / *.vrt)"; a "Select file" button; a checkbox labeled "Tiles are from same dataset (create VRT)"; a label "Output folder: **not selected**"; a "Change output folder" button; and a checkbox labeled "Save visualizations".

user wants to save results in different location

- The **Save visualisations checkbox** allows saving intermediate visualisation SLRM results

4.3.2. Machine Learning Options

Machine learning options:



The screenshot shows a GUI panel titled "Machine learning options:". It contains the following elements: a label "Select machine learning method:" followed by two radio buttons, "segmentation" (which is selected) and "object detection"; a label "Select model:" followed by a dropdown menu showing "ADAF model"; a label "Select classes for inference:" followed by four checkboxes, "All archa..." (which is checked), "Barrow", "Ringfort", and "Enclosure"; and three small grayscale images showing archaeological features.

Figure 37: ADAF GUI component for the selection of ML options.

Machine learning options:

- Select machine learning method.
 - Choose between **segmentation** and **object detection**
- Select mode
 - Dropdown menu where different models can be selected
 - Select **ADAF model** or **Custom model**
- If ADAF model is selected:
 - Use the default ADAF models
 - Select classes for inference, four **checkboxes** are available
 - All archaeology
 - Barrow
 - Ringfort
 - Enclosure

4.3.3. Custom Machine Learning Options

Machine learning options:

Figure 38: ADAF GUI component for selection of ML options if a custom model is used.

- If Custom model is selected:
 - User provides an external model (TAR file)
 - Type in the **Path to custom machine learning model [*tar]**

4.3.4. Post Processing Options

Post-processing options:

Figure 39: ADAF GUI component for the post process filtering of segmentation polygons based on geometry.

Post-processing options:

- **Select minimum area [m²]** - use slider to adjust the min area of polygons to keep
- **Select minimum roundness** - use slider to adjust the min roundness of polygons to keep

- **Checkbox** for saving intermediate results
 - If segmentation is selected:
 - **Keep probability masks** (raw machine learning results)
 - If object detection is selected:
 - **Keep bounding box text files** (raw object detection results)

4.4. Initial ADAF Tool Testing

Following user testing and consultation, several improvements were suggested and implemented:

- Added sliders for post-processing to help resolve an issue with false positives.
- Debugging and update of output log files with more information given:
 - log metadata of input file
 - log information on computed visualisations
 - log roundness and minimum area size for post-processing
- On-the-fly creation of a virtual mosaic if the input data set is tiled. The user has to ensure that all the tiles have the same spatial resolution and use the same projection.
- Results for multiple input datasets saved in separate folders.
- An option to change the parent folder for saving the results. The parent folder is set by default to be the same parent folder where the input file is located. We have added a button that opens a dialog window for users to select a desired output parent folder.
- Predictions for "all archaeology" class - all archaeological data (barrows, ringforts and enclosures) treated as a single class.
- GeoTIFF and VRT have become the de facto standard for geospatial raster files and are the only raster formats supported in the app.

References

- Corns, A., Davis, S., O'Keeffe, J., Koccev, D., Kokalj, Z. & Shaw, R.. (2022). ALS/MLM Project Research Review & Recommendations Report. Available at <https://discoveryprogramme.ie/assets/uploads/2025/11/M2-Research-review-and-recommendations-v1-3WEB.pdf>
- Sensing 12, no. 14: 2215. <https://doi.org/10.3390/rs12142215>.
- Štular B, Eichert S, Lozić E. Airborne LiDAR Point Cloud Processing for Archaeology. Pipeline and QGIS Toolbox. Remote Sensing. 2021; 13(16):3225. <https://doi.org/10.3390/rs13163225>
- Chiba, Tatsuro & Kaneta, Shin-Ichi & SUZUKI, Yusuke. (2008). Red relief image map: New visualization method for three dimensional data. Remote Sens. Spat. Inf. Sci.. 37.
- Historic England Scheduled Monument GIS Data (2022) In Available at: <https://www.data.gov.uk/dataset/640d5dab-4295-4b85-bc26-a8369abb73f6/scheduled-monuments-gis-data> (Accessed 20 December 2022)
- Hesse, R. (2010), LiDAR-derived Local Relief Models – a new tool for archaeological prospection. Archaeol. Prospect., 17: 67-72. <https://doi.org/10.1002/arp.374>
- Maja S., Džeroski S., and Kokalj Z. (2020). "Learning to Classify Structures in ALS-Derived Visualizations of Ancient Maya Settlements with CNN" Remote

Appendices

5. Appendix i: DFM processing

ZRC SAZU received four batches of data, delivered on three disks and via internet transfer, grouped by collection purpose:

- "N data" – collected for various road building and similar projects. The largest by far is the Noise Mapping dataset, provided by Transport Infrastructure Ireland.
- "CFRAM data" – collected for catchment-based flood risk assessment and management; provided by the Office of Public Works (OPW).
- "DP data" – collected for archaeological purposes, provided by Discovery Programme.

5.1. Pre-processing

In general, we created a common folder structure per scanning project and strived to homogenise the data with the properties described below.

Point clouds (when available):

- Format: LAZ (a dedicated compressed ALS point cloud format)
- Classification information:
 - ground (2),
 - low vegetation (between 0.5 and 2.0 m) (3),
 - medium vegetation (between 2.0 and 5.0 m) (4),
 - high vegetation (above 5.0 m) (5),
 - building (6),
 - low point (noise) (7),
 - model key point (if present in the original classification) (8), and
 - water (if present in the original classification) (9).
- Tiled data: preferably 1×1 km² but could be less if scanning density is very high. Tiles sometimes include a small buffer of 10–25 m. Very small datasets were merged into a single file.
- Extent shapefiles with holes filled created.

Rasters:

- Format: GeoTIFF
- Resolution: 0.5 m
- Projection information (either TM75 / Irish National Grid (EPSG 29903) or IRENET95 / Irish Transverse Mercator (EPSG 2157))
- A single file for smaller datasets or 1×1 km² tiles for larger ones, virtual mosaics generated (.vrt).

Converting all the point cloud data to LAZ and raster data to TIFF saves a lot of disk space and makes processing simpler and faster.

It has to be noted we did not deal with datasets not used in the creation of learning data. This is because the data either does not contain archaeology, covers very small areas, and/or is an organisational mess. The latter is especially true for the OPW NFDTM data.

In general, the best organised and uniform were the OPW CFRAM datasets. Nevertheless, we had to reclassify all the point clouds, as the originals only contained classes never classified (0)

and ground. We used the existing classification for most of the “N data”, with notable exceptions being Noise Mapping, N3, NM20, and TEN_T datasets.

Reclassification followed a five-step, archaeology-specific procedure described by Štular, Eichert and Lozić (2021). They recommend a mixture of averaging and segmentation techniques to process the data with a specific archaeological case in mind. The point cloud processing is based on averaging enhanced by sequential processing. First, the buildings are classified with settings optimized for building detection. Then, the entire point cloud without buildings is reprocessed with settings optimized for ground point detection, which favours the retention of archaeological features over vegetation removal. An additional step, targeting low-density data, is to classify all unclassified points that are within $\pm 0.2\text{m}$ of ground points as ground points.

We noticed the datasets are very diverse. Some point clouds were generated by Structure-from-Motion photogrammetry rather than acquired by ALS. They have an extremely high point density, but this does not necessarily mean the elevation models are of higher quality than the ones derived from lower density lidar point clouds, especially for archaeological purposes. This is for several reasons:

- mainly because vegetation penetration is much lower,
- buildings were often classified from vector data rather than automatically obtained from the point clouds,
- classification includes less used classes (e.g., roads, undefined) that are not assigned rigorously,
- files are much larger, also because RGB information from the images is often assigned to the points.

Other things we noted, and we recommend avoiding:

- duplicated or even triplicated files,
- password protected zip files,
- disordered classification – no points classified as class ground (2) or ground points classified as class 0,
- point clouds containing only ground points,
- very large percentage of points classified as overlap (12),
- rasters not cropped to data extent, but to project extent (large no-data areas),
- custom ASCII (text) formats,
- stray points – points that are very far from or high above the scanning area,
- non-square pixels.

5.2. Generating the elevation models

We created a digital feature model (DFM) that includes terrain and buildings for all the datasets with complete point cloud data. Point clouds were not delivered for 10 projects and their rasters thus represent terrain without buildings (DTM). The creation either followed an archaeology-specific processing or, to save time, a simple rasterization using existing classification information.

Data is seamless, which means there should be no visible border edge transitions between data tiles. Small no-data areas are interpolated, but large no-data areas may still be present, for example, because of major water bodies. Rasterising to 0.5 m resolution was not possible

due to missing point cloud data or was not meaningful due to low scanning density for six datasets (Sligo, Sligo_1, N4_1, N70/Listowel, N70/Farranfore, N5_1). These were converted to a higher resolution on-the-fly, before computing visualisations for generation of learning samples. We used Triangular Irregular Network (TIN) method to rasterise the point cloud data, because it performs well in the mostly gently sloping Irish landscape. It is also the preferred method of ALS data providers, which means the newly acquired datasets can have the same characteristics as the ones used for machine learning in this project.

6. Appendix ii: Processed data list

Original folder name	New name	Group	Projection	Copied LAZ for TII	Comment	Format of final DEM (DTM - terrain; DFM - terrain + buildings)	Location of final elevation raster
M4 DTM DSM Point Cloud - Chris Brady	M4	N_DATA	2157	yes	tiled (1x1km2), removed vegetation points, removed RGB	DFM, 0.5m	h:\N_data\M4\M4_dem_05m.tif
Virginia bypass	N3	N_DATA	2157	yes	tiled (1x1km2), buffered (10m), removed intensity >40,000	DFM, 0.5m	h:\N_data\N3\N3_dem_05m.tif
N5 Roscommon - Stephen Davis	N5	N_DATA	2157	no		DTM, 0.5m	h:\N_data\N5\N5_dtm_05m.tif
GSI_Roscommon	N5_1	N_DATA	2157	no		DTM, 1m	h:\N_data\N5_1\N5_1_dtm_1m.tif
	N17	N_DATA	2157	no		DTM, 0.5m	h:\N_data\N17\N17_dtm_05m.vrt
N21 NCW&ABF	N21	N_DATA	2157	yes	tiled (0.25x0.25km2)	DFM, 0.5m	h:\N_data\N21_1\N21_1_dem_05m.tif; h:\N_data\N21_2\N21_2_dem_05m.tif
no name	N22	N_DATA	2157	yes	tiled (1x1km2), removed overlap points, removed RGB	DFM, 0.5m	h:\N_data\N22\N22_dem_05m.tif
N24 Cahir_Wtrfd	N24	N_DATA	2157	yes		DFM, 0.5m	h:\N_data\N24\N24_dem_05m.vrt
N25 Waterford - Milan Gajic	N25	N_DATA	2157	yes		DFM, 0.5m	h:\N_data\N25\N25_dem_05m.tif
N61 - Alan Mulryan/01 N61 BKRP/00 Initial 2018	N61/00	N_DATA	2157	yes	combined with N61_01	combined with N61_01	combined with N61_01
N61 - Alan Mulryan/01 N61 BKRP/01 Additional 2020	N61/01	N_DATA	2157	yes	combined N61/00 and 01, removed RGB, removed vegetation points, tiled (1x1km2)	DFM, 0.5m	h:\N_data\N61_1\N61_1_dem_05m.tif
N61 - Alan Mulryan/02 N61 TCRP	N61/02	N_DATA	2157	yes	combined N61/02 and 03, removed RGB, removed vegetation points, tiled (1x1km2)	DFM, 0.5m	h:\N_data\N61_2\N61_2_dem_05m.tif
N61 - Alan Mulryan/03 N61 GTRP	N61/03	N_DATA	2157	yes	combined with N61_02	combined with N61_02	combined with N61_02
Foynes-Limerick	N69	N_DATA	2157	yes	removed overlap (12), remove undefined (15)	DFM, 0.5m	h:\N_data\N69\N69_dem_05m.tif
Foynes-Limerick_Phase_5	N69_1	N_DATA	2157	yes	removed overlap (12), remove undefined (15)	DFM, 0.5m	h:\N_data\N69_1\N69_1_dem_05m.tif
N70 Kerry - Mike Dineen/Brackaharagh_ITM	N70/Brackaharagh	N_DATA	2157	yes		DFM, 0.5m	h:\N_data\N70_Brackaharagh\N70_Brackaharagh_dem_05m.tif
N70 Kerry - Mike Dineen/Derreenafoyle_ITM	N70/Derreenafoyle	N_DATA	2157	yes	2 files, same area, almost identical poinclouds	DFM, 0.5m	h:\N_data\N70_Derreenafoyle\N70_Derreenafoyle_dem_05m.tif
N70 Kerry - Mike Dineen/Farranfore Killarney OSi LIDAR ITM 2020	N70/Farranfore	N_DATA	2157	no		DTM, 2m	h:\N_data\N70_Farranfore\N70_Farranfore_dtm_2m.tif
N70 Kerry - Mike Dineen/J50A - N22 Garries Bridge	N70/Garries_bridge	N_DATA	2157	yes	removed overlap (12), remove undefined (15)	DFM, 0.5m	h:\N_data\N70_Garries_bridge\N70_Garries_bridge_dem_05m.tif
N70 Kerry - Mike Dineen/J65 Tralee to Dingle LIDAR	N70/J65			yes	not used, organisational mess; cleaned a bit;		
N70 Kerry - Mike Dineen/Listowel ITM	N70/Listowel	N_DATA	2157	no		DTM, 1m	h:\N_data\N70_Listowel\N70_Listowel_dem_1m.tif
N70 Kerry - Mike Dineen/N70 Sneem to Blackwater	N70/Sneem	N_DATA	2157	yes		DFM, 0.5m	h:\N_data\N70_Sneem_to_Blackwater\N70_Sneem_to_Blackwater_dem_05m.tif
N-M20 Limerick - J Howard	NM20	N_DATA	2157	yes	used Steve Davis's rasters; cropped rasters to extent in name; removed RGB info (irrelevant)	DFM, 0.5m	h:\N_data\NM20\NM20_dem_05m.vrt
Noise_Mapping	Noise_Mapping	N_DATA	29903	yes	changed format to LAZ; tiled (1x1km2), buffered (10m); classified with archaeological settings (ground, vegetation, buildings); add projection information	DFM, 0.5m	h:\N_data\Noise_Mapping\Noise_mapping_dem_05m.vrt
O2RH Lidar Data	N11-N25			yes	not used	DTM, 0.5m	
QT_88493_TEN_T_Donegal	TEN_T	N_DATA	2157	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\N_data\TEN_T_1\TEN_T_1_dem_05m.tif; h:\N_data\TEN_T_2\TEN_T_2_dem_05m.tif
SC_TII_05m	N4	N_data	2157	no		DTM, 0.5m	h:\N_data\N4\N4_dtm_05m.tif
SC_TII_dem_188_204	N4_1	N_data	29903	no		DTM, 2m	h:\N_data\N4_1\N4_1_dtm_2m.tif

ISA-1 Edgeworthstown	ISA-1_Edgeworthstown	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-1_Edgeworthstown\ISA-1_Edgeworthstown_dem_05m.tif
ISA-2 Boyle	ISA-2_Boyle	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-2_Boyle\ISA-2_Boyle_dem_05m.tif
ISA-3 Mohill	ISA-3_Mohill	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-3_Mohill\ISA-3_Mohill_dem_05m.tif
ISA-4 Longford-Cloondara	ISA-4_Longford-Cloondara	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-4_Longford-Cloondara\ISA-4_Longford-Cloondara_dem_05m.tif
ISA-5 Birr	ISA-5_Birr	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-5_Birr\ISA-5_Birr_dem_05m.tif
ISA-6 Kilcormac	ISA-6_Kilcormac	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-6_Kilcormac\ISA-6_Kilcormac_dem_05m.tif
ISA-8 Castlerea	ISA-8_Castlerea	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-8_Castlerea\ISA-8_Castlerea_dem_05m.tif
ISA-9 Roscommon	ISA-9_Roscommon	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-9_Roscommon\ISA-9_Roscommon_dem_05m.tif
ISA-10 Borrisokane	ISA-10_Borrisokane	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-10_Borrisokane\ISA-10_Borrisokane_dem_05m.tif
ISA-11 Roscrea	ISA-11_Roscrea	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-11_Roscrea\ISA-11_Roscrea_dem_05m.tif
ISA-12 Athlone	ISA-12_Athlone	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-12_Athlone\ISA-12_Athlone_dem_05m.tif
ISA-13 Mullingar	ISA-13_Mullingar	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-13_Mullingar\ISA-13_Mullingar_dem_05m.tif
ISA-14 Ennis-Quinn	ISA-14_Ennis-Quinn	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-14_Ennis-Quinn\ISA-14_Ennis-Quinn_dem_05m.tif
ISA-15 Kilkee	ISA-15_Kilkee	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-15_Kilkee\ISA-15_Kilkee_dem_05m.tif
ISA-16 Kilrush	ISA-16_Kilrush	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-16_Kilrush\ISA-16_Kilrush_dem_05m.tif
ISA-17 Ballinasloe-Ahascragh	ISA-17_Ballinasloe-Ahascragh	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-17_Ballinasloe-Ahascragh\ISA-17_Ballinasloe-Ahascragh_dem_05m.tif
ISA-18 Ballylongford	ISA-18_Ballylongford	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-18_Ballylongford\ISA-18_Ballylongford_dem_05m.tif
ISA-19 Ardfer	ISA-19_Ardfer	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-19_Ardfer\ISA-19_Ardfer_dem_05m.tif
ISA-20 Abbeydorney	ISA-20_Abbeydorney	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-20_Abbeydorney\ISA-20_Abbeydorney_dem_05m.tif
ISA-21 Listowel	ISA-21_Listowel	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-21_Listowel\ISA-21_Listowel_dem_05m.tif
ISA-22 Tralee	ISA-22_Tralee	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-22_Tralee\ISA-22_Tralee_dem_05m.tif
ISA-23 Clonaslee	ISA-23_Clonaslee	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-23_Clonaslee\ISA-23_Clonaslee_dem_05m.tif
ISA-24 Rathkeale	ISA-24_Rathkeale	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-24_Rathkeale\ISA-24_Rathkeale_dem_05m.tif
ISA-25 Dromcolliher-Milford	ISA-25_Dromcolliher-Milford	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-25_Dromcolliher-Milford\ISA-25_Dromcolliher-Milford_dem_05m.tif
ISA-26 Cappamore	ISA-26_Cappamore	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-26_Cappamore\ISA-26_Cappamore_dem_05m.tif
ISA-27 Athea	ISA-27_Athea	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-27_Athea\ISA-27_Athea_dem_05m.tif
ISA-28 Drumshanbo	ISA-28_Drumshanbo	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-28_Drumshanbo\ISA-28_Drumshanbo_dem_05m.tif
ISA-29 Carrick on Shannon-Leitrim	ISA-29_Carrick_on_Shannon-Leitrim	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-29_Carrick_on_Shannon-Leitrim\ISA-29_Carrick_on_Shannon-Leitrim_dem_05m.tif
ISA-30 Shannon Harbour	ISA-30_Shannon_Harbour	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-30_Shannon_Harbour\ISA-30_Shannon_Harbour_dem_05m.tif

ISA-31 Doonbeg	ISA-31_Doonbeg	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-31_Doonbeg\ISA-31_Doonbeg_dem_05m.tif
ISA-32 Aughrim	ISA-32_Aughrim	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-32_Aughrim\ISA-32_Aughrim_dem_05m.tif
ISA-33 Trim	ISA-33_Trim	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-33_Trim\ISA-33_Trim_dem_05m.tif
ISA-34 Navan	ISA-34_Navan	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-34_Navan\ISA-34_Navan_dem_05m.tif
ISA-35 Kilmessan	ISA-35_Kilmessan	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-35_Kilmessan\ISA-35_Kilmessan_dem_05m.tif
ISA-36 Innfield-Longwood	ISA-36_Innfield-Longwood	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-36_Innfield-Longwood\ISA-36_Innfield-Longwood_dem_05m.tif
ISA-37 Ballivor	ISA-37_Ballivor	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-37_Ballivor\ISA-37_Ballivor_dem_05m.tif
ISA-38 Athboy	ISA-38_Athboy	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-38_Athboy\ISA-38_Athboy_dem_05m.tif
ISA-39 Mullagh	ISA-39_Mullagh	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-39_Mullagh\ISA-39_Mullagh_dem_05m.tif
ISA-40 Clara-Kilbeggan	ISA-40_Clara-Kilbeggan	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-40_Clara-Kilbeggan\ISA-40_Clara-Kilbeggan_dem_05m.tif
ISA-41 Wicklow	ISA-41_Wicklow	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-41_Wicklow\ISA-41_Wicklow_dem_05m.tif
ISA-42 Arklow	ISA-42_Arklow	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-42_Arklow\ISA-42_Arklow_dem_05m.tif
ISA-43 Ballymahon-Abbeyshrute	ISA-43_Ballymahon-Abbeyshrute	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-43_Ballymahon-Abbeyshrute\ISA-43_Ballymahon-Abbeyshrute_dem_05m.tif
ISA-44 Newcastle West	ISA-44_Newcastle_West	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-44_Newcastle_West\ISA-44_Newcastle_West_dem_05m.tif
ISA-45 Portumna	ISA-45_Portumna	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-45_Portumna\ISA-45_Portumna_dem_05m.tif
ISA-46 Tullamore	ISA-46_Tullamore	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-46_Tullamore\ISA-46_Tullamore_dem_05m.tif
ISA-47 Drogheda	ISA-47_Drogheda	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-47_Drogheda\ISA-47_Drogheda_dem_05m.tif
ISA-48 Askeaton	ISA-48_Askeaton	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-48_Askeaton\ISA-48_Askeaton_dem_05m.tif
ISA-49 Charleville	ISA-49_Charleville	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-49_Charleville\ISA-49_Charleville_dem_05m.tif
ISA-50 Dublin	ISA-50_Dublin	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-50_Dublin\ISA-50_Dublin_dem_05m.vrt
ISA-51 Nenagh	ISA-51_Nenagh	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-51_Nenagh\ISA-51_Nenagh_dem_05m.tif
ISA-52 Limerick	ISA-52_Limerick	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-52_Limerick\ISA-52_Limerick_dem_05m.vrt
ISA-53 Enniscorthy	ISA-53_Enniscorthy	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-53_Enniscorthy\ISA-53_Enniscorthy_dem_05m.tif
ISA-54 Youghal	ISA-54_Youghal	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-54_Youghal\ISA-54_Youghal_dem_05m.tif
ISA-55 Tallow	ISA-55_Tallow	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-55_Tallow\ISA-55_Tallow_dem_05m.tif
ISA-56 Ballyduff-Lismore	ISA-56_Ballyduff-Lismore	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-56_Ballyduff-Lismore\ISA-56_Ballyduff-Lismore_dem_05m.tif
ISA-57 Killarney-Glenflesk	ISA-57_Killarney-Glenflesk	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-57_Killarney-Glenflesk\ISA-57_Killarney-Glenflesk_dem_05m.tif
ISA-58 Dingle	ISA-58_Dingle	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-58_Dingle\ISA-58_Dingle_dem_05m.tif
ISA-59 Mallow- Killavullen	ISA-59_Mallow-Killavullen	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-59_Mallow-Killavullen\ISA-59_Mallow-Killavullen_dem_05m.tif
ISA-60 Kanturk	ISA-60_Kanturk	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-60_Kanturk\ISA-60_Kanturk_dem_05m.tif

ISA-61 Freemount	ISA-61_Freemount	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-61_Freemount\ISA-61_Freemount_dem_05m.tif
ISA-62 Fermoy	ISA-62_Fermoy	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-62_Fermoy\ISA-62_Fermoy_dem_05m.tif
ISA-63 Castlemartye	ISA-63_Castlemartye	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-63_Castlemartye\ISA-63_Castlemartye_dem_05m.tif
ISA-64 Baltinglass	ISA-64_Baltinglass	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-64_Baltinglass\ISA-64_Baltinglass_dem_05m.tif
ISA-65 Wexford	ISA-65_Wexford	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-65_Wexford\ISA-65_Wexford_dem_05m.tif
ISA-66 Dungarvan	ISA-66_Dungarvan	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-66_Dungarvan\ISA-66_Dungarvan_dem_05m.tif
ISA-67 Dunmore East	ISA-67_Dunmore_East	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-67_Dunmore_East\ISA-67_Dunmore_East_dem_05m.tif
ISA-68 Daingean	ISA-68_Daingean	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-68_Daingean\ISA-68_Daingean_dem_05m.tif
ISA-70 Kilkenny	ISA-70_Kilkenny	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-70_Kilkenny\ISA-70_Kilkenny_dem_05m.tif
ISA-71 Inistioge	ISA-71_Inistioge	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-71_Inistioge\ISA-71_Inistioge_dem_05m.tif
ISA-72 Graigueanamanagh	ISA-72_Graigueanamanagh	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-72_Graigueanamanagh\ISA-72_Graigueanamanagh_dem_05m.tif
ISA-73 Rathangan	ISA-73_Rathangan	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-73_Rathangan\ISA-73_Rathangan_dem_05m.tif
ISA-74 Athy	ISA-74_Athy	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-74_Athy\ISA-74_Athy_dem_05m.tif
ISA-75 Bunclody	ISA-75_Bunclody	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-75_Bunclody\ISA-75_Bunclody_dem_05m.tif
ISA-76 Portlaoise	ISA-76_Portlaoise	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-76_Portlaoise\ISA-76_Portlaoise_dem_05m.tif
ISA-77 Tullow	ISA-77_Tullow	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-77_Tullow\ISA-77_Tullow_dem_05m.tif
ISA-78 Gorey	ISA-78_Gorey	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-78_Gorey\ISA-78_Gorey_dem_05m.tif
ISA-79 Rathcormack	ISA-79_Rathcormack	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-79_Rathcormack\ISA-79_Rathcormack_dem_05m.tif
ISA-81 Kilgorlin-Ballymalis	ISA-81_Kilgorlin-Ballymalis	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-81_Kilgorlin-Ballymalis\ISA-81_Kilgorlin-Ballymalis_dem_05m.vrt
ISA-82 Carlow	ISA-82_Carlow	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-82_Carlow\ISA-82_Carlow_dem_05m.vrt
ISA-83 Castlebar	ISA-83_Castlebar	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-83_Castlebar\ISA-83_Castlebar_dem_05m.tif
ISA-84 Corrofin	ISA-84_Corrofin	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-84_Corrofin\ISA-84_Corrofin_dem_05m.tif
ISA-86 Riverstown	ISA-86_Riverstown	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-86_Riverstown\ISA-86_Riverstown_dem_05m.tif
ISA-88 Gorteen	ISA-88_Gorteen	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-88_Gorteen\ISA-88_Gorteen_dem_05m.tif
ISA-89 Charlestown	ISA-89_Charlestown	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-89_Charlestown\ISA-89_Charlestown_dem_05m.tif
ISA-90 Ballymote	ISA-90_Ballymote	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-90_Ballymote\ISA-90_Ballymote_dem_05m.tif
ISA-91 Westport	ISA-91_Westport	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-91_Westport\ISA-91_Westport_dem_05m.tif
ISA-92 Swinford	ISA-92_Swinford	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-92_Swinford\ISA-92_Swinford_dem_05m.tif
ISA-93 Louisburgh	ISA-93_Louisburgh	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-93_Louisburgh\ISA-93_Louisburgh_dem_05m.tif
ISA-94 Foxford	ISA-94_Foxford	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-94_Foxford\ISA-94_Foxford_dem_05m.tif

ISA-95 Ballyhaunis	ISA-95_Ballyhaunis	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-95_Ballyhaunis\ISA-95_Ballyhaunis_dem_05m.tif
ISA-96 Ballina	ISA-96_Ballina	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-96_Ballina\ISA-96_Ballina_dem_05m.tif
ISA-97 Manorhamilton	ISA-97_Manorhamilton	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-97_Manorhamilton\ISA-97_Manorhamilton_dem_05m.tif
ISA-98 Tuam	ISA-98_Tuam	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-98_Tuam\ISA-98_Tuam_dem_05m.tif
ISA-99 Oughterard	ISA-99_Oughterard	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-99_Oughterard\ISA-99_Oughterard_dem_05m.tif
ISA-100 Loughrea	ISA-100_Loughrea	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-100_Loughrea\ISA-100_Loughrea_dem_05m.tif
ISA-101 Claregalway	ISA-101_Claregalway	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-101_Claregalway\ISA-101_Claregalway_dem_05m.tif
ISA-102 Ballysadare	ISA-102_Ballysadare	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-102_Ballysadare\ISA-102_Ballysadare_dem_05m.tif
ISA-103 Sligo	ISA-103_Sligo	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-103_Sligo\ISA-103_Sligo_dem_05m.tif
ISA-104 Crossmolina	ISA-104_Crossmolina	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-104_Crossmolina\ISA-104_Crossmolina_dem_05m.tif
ISA-105 Galway-Kilcaimin	ISA-105_Galway-Kilcaimin	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-105_Galway-Kilcaimin\ISA-105_Galway-Kilcaimin_dem_05m.tif
ISA-106 Rathmullan	ISA-106_Rathmullan	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-106_Rathmullan\ISA-106_Rathmullan_dem_05m.tif
ISA-107 Newton Cunningham	ISA-107_Newton_Cunningham	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-107_Newton_Cunningham\ISA-107_Newton_Cunningham_dem_05m.tif
ISA-108 Moville	ISA-108_Moville	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-108_Moville\ISA-108_Moville_dem_05m.tif
ISA-109 Lifford	ISA-109_Lifford	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-109_Lifford\ISA-109_Lifford_dem_05m.tif
ISA-111 Killybegs	ISA-111_Killybegs	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-111_Killybegs\ISA-111_Killybegs_dem_05m.tif
ISA-112 Glenties	ISA-112_Glenties	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-112_Glenties\ISA-112_Glenties_dem_05m.tif
ISA-113 Dunfanaghy	ISA-113_Dunfanaghy	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-113_Dunfanaghy\ISA-113_Dunfanaghy_dem_05m.tif
ISA-114 Downies	ISA-114_Downies	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-114_Downies\ISA-114_Downies_dem_05m.tif
ISA-115 Donegal	ISA-115_Donegal	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-115_Donegal\ISA-115_Donegal_dem_05m.tif
ISA-116 Convoy	ISA-116_Convoy	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-116_Convoy\ISA-116_Convoy_dem_05m.tif
ISA-117 Clonmany	ISA-117_Clonmany	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-117_Clonmany\ISA-117_Clonmany_dem_05m.tif
ISA-118 Carrowkeel	ISA-118_Carrowkeel	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-118_Carrowkeel\ISA-118_Carrowkeel_dem_05m.tif
ISA-119 Burnfoot	ISA-119_Burnfoot	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-119_Burnfoot\ISA-119_Burnfoot_dem_05m.tif
ISA-120 Buncrana	ISA-120_Buncrana	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-120_Buncrana\ISA-120_Buncrana_dem_05m.tif
ISA-122 Ardara	ISA-122_Ardara	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-122_Ardara\ISA-122_Ardara_dem_05m.tif
ISA-123 Cavan-ButlersBridge	ISA-123_Cavan-ButlersBridge	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-123_Cavan-ButlersBridge\ISA-123_Cavan-ButlersBridge_dem_05m.tif
ISA-124 Ballyconnell	ISA-124_Ballyconnell	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-124_Ballyconnell\ISA-124_Ballyconnell_dem_05m.tif
ISA-125 Monaghan	ISA-125_Monaghan	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-125_Monaghan\ISA-125_Monaghan_dem_05m.tif

ISA-126 Carrickmacross	ISA-126_Carrickmacross	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-126_Carrickmacross\ISA-126_Carrickmacross_dem_05m.tif
ISA-127 Dundalk	ISA-127_Dundalk	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-127_Dundalk\ISA-127_Dundalk_dem_05m.tif
ISA-128 Carlingford	ISA-128_Carlingford	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-128_Carlingford\ISA-128_Carlingford_dem_05m.tif
ISA-129 Ardee	ISA-129_Ardee	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-129_Ardee\ISA-129_Ardee_dem_05m.tif
ISA-130 Castlebellingham	ISA-130_Castlebellingham	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-130_Castlebellingham\ISA-130_Castlebellingham_dem_05m.tif
ISA-131 Carndonagh	ISA-131_Carndonagh	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-131_Carndonagh\ISA-131_Carndonagh_dem_05m.tif
ISA-132 Abbeyfeale	ISA-132_Abbeyfeale	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-132_Abbeyfeale\ISA-132_Abbeyfeale_dem_05m.tif
ISA-133 Athleague	ISA-133_Athleague	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-133_Athleague\ISA-133_Athleague_dem_05m.tif
ISA-134 Dromod	ISA-134_Dromod	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-134_Dromod\ISA-134_Dromod_dem_05m.tif
ISA-135 Moneycashen	ISA-135_Moneycashen	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-135_Moneycashen\ISA-135_Moneycashen_dem_05m.tif
ISA-136 Tarbert	ISA-136_Tarbert	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-136_Tarbert\ISA-136_Tarbert_dem_05m.tif
ISA-137 Shannonbridge	ISA-137_Shannonbridge	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-137_Shannonbridge\ISA-137_Shannonbridge_dem_05m.tif
ISA-138 Lanesborough	ISA-138_Lanesborough	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-138_Lanesborough\ISA-138_Lanesborough_dem_05m.tif
ISA-139 Edenderry	ISA-139_Edenderry	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-139_Edenderry\ISA-139_Edenderry_dem_05m.tif
ISA-140 Blessington	ISA-140_Blessington	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-140_Blessington\ISA-140_Blessington_dem_05m.tif
ISA-141 Blackwater	ISA-141_Blackwater	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-141_Blackwater\ISA-141_Blackwater_dem_05m.tif
ISA-142 Suncroft	ISA-142_Suncroft	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-142_Suncroft\ISA-142_Suncroft_dem_05m.tif
ISA-143 Ballyhale	ISA-143_Ballyhale	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-143_Ballyhale\ISA-143_Ballyhale_dem_05m.tif
ISA-144 New Ross	ISA-144_New_Ross	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-144_New_Ross\ISA-144_New_Ross_dem_05m.tif
ISA-145 Mountrath	ISA-145_Mountrath	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-145_Mountrath\ISA-145_Mountrath_dem_05m.tif
ISA-146 Callan	ISA-146_Callan	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-146_Callan\ISA-146_Callan_dem_05m.tif
ISA-147 Ballyragget	ISA-147_Ballyragget	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-147_Ballyragget\ISA-147_Ballyragget_dem_05m.tif
ISA-148 Freshford	ISA-148_Freshford	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-148_Freshford\ISA-148_Freshford_dem_05m.tif
ISA-149 Rathdowney	ISA-149_Rathdowney	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-149_Rathdowney\ISA-149_Rathdowney_dem_05m.tif
ISA-150 Tramore	ISA-150_Tramore	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-150_Tramore\ISA-150_Tramore_dem_05m.tif
ISA-151 Thomastown	ISA-151_Thomastown	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-151_Thomastown\ISA-151_Thomastown_dem_05m.tif
ISA-152 Kilmore Quay	ISA-152_Kilmore_Quay	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-152_Kilmore_Quay\ISA-152_Kilmore_Quay_dem_05m.tif
ISA-153 Ballinamore	ISA-153_Ballinamore	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-153_Ballinamore\ISA-153_Ballinamore_dem_05m.tif
ISA-154 Ballybay	ISA-154_Ballybay	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-154_Ballybay\ISA-154_Ballybay_dem_05m.tif
ISA-155 Inishkeen	ISA-155_Inishkeen	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-155_Inishkeen\ISA-155_Inishkeen_dem_05m.tif

ISA-156 Bundoran	ISA-156_Bundoran	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-156_Bundoran\ISA-156_Bundoran_dem_05m.tif
ISA-157 Derrybeg	ISA-157_Derrybeg	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-157_Derrybeg\ISA-157_Derrybeg_dem_05m.tif
ISA-158 Dunglow	ISA-158_Dunglow	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-158_Dunglow\ISA-158_Dunglow_dem_05m.tif
ISA-159 Rathmelton	ISA-159_Rathmelton	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-159_Rathmelton\ISA-159_Rathmelton_dem_05m.tif
ISA-160 Newport	ISA-160_Newport	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-160_Newport\ISA-160_Newport_dem_05m.tif
ISA-161 Athenry	ISA-161_Athenry	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-161_Athenry\ISA-161_Athenry_dem_05m.tif
ISA-162 Kinvarra	ISA-162_Kinvarra	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-162_Kinvarra\ISA-162_Kinvarra_dem_05m.tif
ISA-163 Gort	ISA-163_Gort	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-163_Gort\ISA-163_Gort_dem_05m.tif
ISA-164 Roundstone	ISA-164_Roundstone	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-164_Roundstone\ISA-164_Roundstone_dem_05m.tif
ISA-165 Clifden	ISA-165_Clifden	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-165_Clifden\ISA-165_Clifden_dem_05m.tif
ISA-166 Aglish	ISA-166_Aglish	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-166_Aglish\ISA-166_Aglish_dem_05m.tif
ISA-167 Castleisland	ISA-167_Castleisland	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-167_Castleisland\ISA-167_Castleisland_dem_05m.tif
ISA-168 Killeagh	ISA-168_Killeagh	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-168_Killeagh\ISA-168_Killeagh_dem_05m.tif
ISA-169 Portmagee	ISA-169_Portmagee	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-169_Portmagee\ISA-169_Portmagee_dem_05m.tif
ISA-170 Kiltiernan	ISA-170_Kiltiernan	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-170_Kiltiernan\ISA-170_Kiltiernan_dem_05m.tif
ISA-171 Turloughmore	ISA-171_Turloughmore	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-171_Turloughmore\ISA-171_Turloughmore_dem_05m.tif
ISA-172 Lough Aleenaun	ISA-172_Lough_Aleenaun	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-172_Lough_Aleenaun\ISA-172_Lough_Aleenaun_dem_05m.tif
ISA-173 Four Roads	ISA-173_Four_Roads	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-173_Four_Roads\ISA-173_Four_Roads_dem_05m.tif
ISA-178 Cregganna	ISA-178_Cregganna	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-178_Cregganna\ISA-178_Cregganna_dem_05m.tif
ISA-179 Gort-Ardrahan	ISA-179_Gort-Ardrahan	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-179_Gort-Ardrahan\ISA-179_Gort-Ardrahan_dem_05m.tif
ISA-180 Carnmore	ISA-180_Carnmore	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-180_Carnmore\ISA-180_Carnmore_dem_05m.tif
ISA-181 Sneem	ISA-181_Sneem	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-181_Sneem\ISA-181_Sneem_dem_05m.tif
ISA-182 Cregduff	ISA-182_Cregduff	CFRAM	29903	yes	classified with archaeological settings (ground, vegetation, buildings)	DFM, 0.5m	h:\OPW_data\CFRAM\ISA-182_Cregduff\ISA-182_Cregduff_dem_05m.tif
BOYNE VALLEY	Btu-na-boinne	DP	29903	no	No original laz/las files available, Tiled original dtm data	DTM, 1m	h:\DP_data\BRUNABOINNE\DP_brunaboinne_dem_1m.tif

7. Appendix iii: Preparation of ALS data for detection

The ADAF tool requires input data in a GeoTIFF raster format with a 0.5 m resolution and float (32-bit) data type. The input file should be either a digital elevation model (DEM) or a computed simple local relief model (SLRM) visualisation. ADAF can accept other resolutions, data types, and one-band visualisations, but the detection results may not be optimal as machine learning models were trained with these specific parameters.

Ideally, the DEM should represent the terrain with buildings (a digital feature model (DFM) as described in Appendix 3.1). However, a digital terrain model (DTM) can also be used.

When pre-calculating SLRM, use a radius of 10 m (20 px for 0.5 m resolution data). Apply a linear histogram stretch with a minimum of -0.5 m and a maximum of 0.5 m, and normalise the result to values between 0 and 1.

Because each input file is treated separately and consequently generates its set of results, use a virtual mosaic raster (.vrt) as the input for tiled datasets (e.g. in 1×1 km² tiles). ADAF supports on-the-fly generation of a virtual mosaic for an individual dataset. Ensure that all tiles have the same projection and resolution.

8. Appendix iv: Creating ML-ready data for (re)training

8.1. Creating the DFM

A digital feature model (DFM) is a raster elevation model representing terrain with buildings. We recommend classifying ALS data with a five-step, archaeology-specific procedure as described by (Štular, Eichert and Lozić 2021) who recommend a mixture of averaging and segmentation techniques. First, the buildings are classified with settings optimised for building detection. Then, the entire point cloud without buildings is reprocessed with settings optimised for ground point detection, which favours the retention of archaeological features over vegetation removal. An additional step, targeting low-density data, is to classify all unclassified points that are within ±0.2m of ground points as ground points. A python script for classifying point clouds with an archaeology-specific procedure with LAStools is copied below.

The rasterisation should include classes 2 (ground), 6 (buildings), and any points that may have been classified as model key points (7) in previous classifications. Preferred raster resolution is 0.5 m.

```
"""
Classify LAZ point cloud for archaeological interpretation with LAStools.
Created on 1 February 2022
@author: Ziga Kokalj, ZRC SAZU, Novi trg 2, 1000 Ljubljana, Slovenia

Licensed version of LAStools is required.
"""

import os
import subprocess
import shutil
import time

# Data and paths
# List of files to process (one file path per line)
src_LOF = r"e:\Noise_mapping\noise_mapping_LOF.txt"
# output folder
out_pth = r"e:\Noise_mapping\LAZ_classified"
# temporary folder
out_tmp = r"e:\tmp"
# Number of CPUs used
cores = 14

os.makedirs(out_pth, exist_ok=True)
tstart = time.time()

tmp_n = ["1_ground1", "2_height1", "3_class1", "4_ground2"]
tmp_pths = [os.path.join(out_tmp, pth) for pth in tmp_n]

for pth in tmp_pths:
    os.makedirs(pth, exist_ok=True)

# use switches -buffered and -remain_buffered for non-buffered tiles only!
n = 0
arg_list_LT_ground1 = ['Lasground',
                       '-lof', src_LOF,
                       '-city',
```

```

        '-ultra_fine',
        '-cores', str(cores),
        '-ignore_class', '7', '12',
        '-olaz',
        '-remain_buffered',
        '-odir', tmp_pths[n],
        '-odix', '_1',
        '-cpu64']
print(str(arg_list_LT_ground1))
subprocess.run(arg_list_LT_ground1)
n += 1

arg_list_LT_height1 = ['Lasheight',
                        '-i', f'{tmp_pths[n-1]}\\*.laz',
                        '-ignore_class 0 1 3 4 5 6 7 8 9 10 11 12"',
                        '-remain_buffered',
                        '-cores', str(cores),
                        '-olaz',
                        '-odir', tmp_pths[n],
                        '-odix', '_2',
                        '-cpu64']
print(str(arg_list_LT_height1))
subprocess.run(arg_list_LT_height1)
try:
    shutil.rmtree(tmp_pths[n-1])
except:
    pass
n += 1

arg_list_LT_classify1 = ['Lasclassify',
                        '-i', f'{tmp_pths[n-1]}\\*.laz',
                        '-remain_buffered',
                        '-small_trees',
                        '-cores', str(cores),
                        '-olaz',
                        '-odir', tmp_pths[n],
                        '-odix', '_3',
                        '-cpu64']
print(str(arg_list_LT_classify1))
subprocess.run(arg_list_LT_classify1)
try:
    shutil.rmtree(tmp_pths[n-1])
except:
    pass
n += 1

arg_list_LT_ground2 = ['Lasground',
                        '-i', f'{tmp_pths[n-1]}\\*.laz',
                        '-wilderness',
                        '-ultra_fine',
                        '-remain_buffered',
                        '-ignore_class', '6', '7',
                        '-cores', str(cores),
                        '-olaz',
                        '-odir', tmp_pths[n],
                        '-odix', '_4',
                        '-cpu64']
print(str(arg_list_LT_ground2))
subprocess.run(arg_list_LT_ground2)
try:
    shutil.rmtree(tmp_pths[n-1])
except:
    pass
n += 1

arg_list_LT_height2 = ['Lasheight',
                        '-i', f'{tmp_pths[n-1]}\\*.laz',
                        '-ignore_class', '6', '7',
                        '-classify_below', '-0.25', '7',
                        '-classify_between', '-0.2', '0.2', '2',
                        '-classify_between', '0.5', '2.0', '3',
                        '-classify_between', '2.0', '5.0', '4',
                        '-classify_above', '5.0', '5',
                        '-remain_buffered',
                        '-cores', str(cores),
                        '-olaz',
                        '-odir', out_pth,
                        '-odix', '_5',
                        '-cpu64']
print(str(arg_list_LT_height2))
subprocess.run(arg_list_LT_height2)
try:
    shutil.rmtree(tmp_pths[n])
except:
    pass

tend = time.time() - tstart

print()
print("DONE!")
print(f"Finished processing in {round(tend/60, ndigits=None)} min.")
print()

```

8.2. Creating the visualisation and patches from DFMs and vector files

A complete training dataset consists of so-called patches, i.e. tiled images that are provided with a labelled segmentation mask and/or labelled bounding boxes for semantic segmentation or object detection

respectively. The patches are systematically extracted from the DFM raster and only patches with positive samples (i.e. with archaeological sites) are retained. Labelled masks and bounding boxes are created from polygons of archaeological sites (i.e. vector files).

Patches are created from the DFM in two steps - creating visualisation from DFM and cutting DFM into ML-readable patches.

8.2.1. Creating visualisations

- INPUT: DFM in GeoTIFF or VRT format
- OUTPUT: visualisation of DFM in GeoTIFF or VRT format

The default visualisation for the ADAF model is SLRM (Simple Local Relief Model). The Irish dataset was processed with the SLRM radius of 10 metres.

The SLRM visualisation with default parameters can be prepared using the Python functions included in ADAF. The same visualisation can be created at your own discretion using external software such as RVT (rvt_py package, RVT plugin for QGIS, RVT desktop app) or other third-party software. To replicate the default visualisation, make sure you use :

- radius 10 m (NOTE: in RVT the radius is given in pixels, convert it accordingly depending on the spatial resolution)
- normalise to between 0 and 1 values
- set all nodata and nan values to 0
- save the visualisation in GeoTIFF or VRT file format

ADAF can also be trained on visualisations other than SLRM. In this case, make sure that the visualisation raster has:

- either 1 or 3 bands
- is normalised between 0 and 1
- contains no nan values

8.2.2. Creating training data

INPUT:

- visualisation of DFM in GeoTIFF or VRT format
- vector file(s) in SHP or GPKG format with labelled segments; one file per label

OUTPUT:

- a folder with visualisation tiles (GeoTIFF format),
- a folder with masks for semantic segmentations (GeoTIFF format),
- a folder with *labelTxt* for object detection (text file format).

The Python routines for creating patches are included in ADAF. The input visualisations must conform to the above guidelines and a separate vector file is required for each class (polygons with labelled archaeological features).

Users can set the size of the patches (i.e. the size of the image tiles in pixels) and the overlap of the tiles. If the "DFM" attribute is included in the vector file, this information is included in the segmentation masks and bounding boxes as well. The DFM value was used in the original vector data to indicate the quality of the archaeological features on the DFM and can be used to filter out data by quality during ML training.

Default values for patch creation parameters are:

- tile size of 512 pixels
- overlap of tiles by 0.5 tile (i.e. 256 pixels)

The ADAF is set up for detection of three default classes, namely barrows, ringforts and enclosures. The procedure for creating patches follows this format, which means that masks can contain up to 3 different classes (not necessarily using the default names of labels). At least one vector file must be specified. In this case, the segmentation mask still has 3 bands, but only the first band is filled with the labelled data.

NOTE: When training or retraining the models, only one band from the segmentation mask is used at a time. The user must specify the band ID and the corresponding labelling name.

Examples

For retraining of the existing models, three vector files are given for barrows, ringforts and enclosures. All three classes are saved in the same segmentation mask file. The user first trains the model for barrows by selecting band 1 and naming the class with the "barrow" label. Then he continues with the model for ringforts by selecting band 2 and naming the class "ringfort" and finally he follows the same pattern for enclosures.

Training a completely new class requires a single vector file with labelled data when creating patches. As only one vector file has been provided, only the first band of the segmentation mask file is filled with valid data, the other two bands contain all zeros. When training a new model, the user must specify band 1 and give the class a new name.

The training data must be divided geographically into training, validation and test subsets. This is not covered by any Python script and must be done at your own discretion. As the split is done geographically, it can be performed before or after processing, i.e.:

slice the DFM raster into appropriate regions and create patches for each group separately,
create patches for the entire DFM and split the resulting patches into 3 groups.

It is recommended to split the dataset in an approximate ratio of 60:20:20 (train:validation:test).

NOTE: If retraining the existing ADAF model, the visualisations have to be SLRM with 10 m radius and patch size of 512 px. If training a completely new ML model from scratch, any visualisation and tile size can be used. Refer to relevant literature or TII-ADAF technical report where we tested other visualisations and patch sizes.

8.2.3. Python script for creating the visualisation

NOTE: Run by calling the `run_visualization()` function.

```
"""
Create SLRM visualisation from DFM for ADAF ML training dataset.
Created on 26 May 2023
@author: Nejc Čož, ZRC SAZU, Novi trg 2, 1000 Ljubljana, Slovenia
Requires ADAF toolbox.
"""
from pathlib import Path
import adaf.grid_tools as gt
from adaf.adaf_vis import tiled_processing

def run_visualisations(dem_path, tile_size, save_dir, nr_processes=1):
    """Calculates visualisations from DEM and saves them into VRT (Geotiff) file.

    Uses RVT (see adaf_vis.py).
    Parameters
    -----
    dem_path : str or pathlib.Path()
        Can be any raster file (GeoTIFF and VRT supported).
    tile_size : int
        In pixels.
    save_dir : str
        Save directory.
    nr_processes : int
        Number of processes for parallel computing.
    Returns
    -----
    dict
        A python dictionary containing results from tiling, such as paths and processing time.
    """
    # Prepare paths
```

```

in_file = Path(dem_path)

# We need polygon covering valid data
valid_data_outline, _ = gt.poly_from_valid(in_file.as_posix())

# Create reference grid, filter it and save it to disk
tiles_extents = gt.bounding_grid(in_file.as_posix(), tile_size, tag=False)
tiles_extents = gt.filter_by_outline(tiles_extents, valid_data_outline)

# Run visualisations
out_paths = tiled_processing(
    input_raster_path=in_file.as_posix(),
    extents_list=tiles_extents,
    nr_processes=nr_processes,
    save_dir=Path(save_dir)
)

return out_paths

```

8.2.4. Python script for creating patches

NOTE: Run by calling `create_patches_main()` function as shown in the last line of the script.

```

"""
Create patches (training samples) for ML training from visualisations and labeled polygons.
Created on 26 May 2023
@author: Nejc Čož, ZRC SAZU, Novi trg 2, 1000 Ljubljana, Slovenia

Requires ADAF toolbox.
"""

import multiprocessing as mp
import warnings
from os import cpu_count
from pathlib import Path

import geopandas as gpd
import numpy as np
import pandas as pd
import rasterio
from rasterio.features import geometry_mask
from shapely.affinity import translate
from shapely.geometry import box

from adaf_utils import clip_tile

warnings.filterwarnings("ignore", category=np.VisibleDeprecationWarning)

def create_one_patch(one_tile, segments_gdf, dem_pth):
    """Creates ML patch for one tile. Takes tile as dictionary of GeoSeries, vector files with segments and path to
    the source image. Creates a tile of image (raster), segmentation mask of the image (raster) and LabelTxt (text file)
    and saves them to disk.

    Parameters
    -----
    one_tile : dict
        Dictionary containing all the data for one tile from GeoDataFrame.
    segments_gdf : list
        A list of GeoDataFrames containing all labeled segments.
    dem_pth : str or pathlib.Path()
        Path to source image, from which the patches are made.

    """
    # #####
    # 1 # Clip image tile
    # #####
    out_image = Path(one_tile["images_path"])
    out_image.parent.mkdir(exist_ok=True)

    image_path = clip_tile(
        list(one_tile["geometry"].bounds),
        out_image,
        dem_pth,
        out_nodata=0
    )

    # #####
    # 2 # Create segmentation mask
    # #####
    out_seg = Path(one_tile["segmentation_masks_path"])
    out_seg.parent.mkdir(parents=True, exist_ok=True)

    _ = create_segmentation_mask(
        one_tile["geometry"],
        out_seg,
        image_path,
        segments_gdf,
    )

    # #####
    # 3 # Create LabelTxt
    # #####
    label_pth = Path(one_tile["LabelTxt_path"])
    label_pth.parent.mkdir(parents=True, exist_ok=True)

    with open(label_pth, "w") as dst:
        dst.write(one_tile["LabelTxt"])

def uniform_grid(extents, crs, spacing_xy, stagger=None):
    """Creates uniform grid over the total extents.

```

```

Parameters
-----
extents : List
    A list of extents [x_min, y_min, x_max, y_max]
crs : rasterio.crs.CRS
    Projection as CRS instance.
spacing_xy : (int, int)
    Width and height of grid tile in meters.
stagger : float
    Distance to use for translating the cell for creation of staggered grid.
    Use None if grid should not be staggered.

Returns
-----
gpd.GeoDataFrame
    Grid in GeoDataFrame format.
"""
# total area for the grid [xmin, ymin, xmax, ymax]
x_min, y_min, x_max, y_max = extents # gdf.total_bounds
tile_w, tile_h = spacing_xy

# Target Aligned Pixels
x_min = np.floor(x_min / tile_w) * tile_w
# bottom = np.floor(extents.bottom / tile_h) * tile_h
# right = np.ceil(extents.right / tile_w) * tile_w
y_max = np.ceil(y_max / tile_h) * tile_h
_, bottom, right, _ = extents # ONLY TOP-LEFT NEEDS TO BE ROUNDED

grid_cells = []
for x0 in np.arange(x_min, x_max, tile_w):
    for y0 in np.arange(y_max, y_min, -tile_h):
        # bounds
        x1 = x0 + tile_w
        y1 = y0 - tile_h
        cell_1 = box(x0, y0, x1, y1)
        grid_cells.append(cell_1)
        if stagger:
            cell_2 = translate(cell_1, stagger, stagger)
            grid_cells.append(cell_2)

# Generate GeoDataFrame
grid = gpd.GeoDataFrame(grid_cells, columns=['geometry'], crs=crs)

return grid

def relative_bounds(polygon, patch, res):
    """Calculates coordinates of a BBOX relative to the image array.

    Parameters
    -----
    polygon : gpd.GeoSeries
        Segmentation polygon
    patch : gpd.GeoSeries
        Tile polygon (image extents)
    res
        Spatial resolution of image, needed to create array from image extents.

    Returns
    -----
    str
        A string containing relative coordinates in LabelTxt format.
    """
    # Get bounds of the intersected segment
    seg_minx, seg_miny, seg_maxx, seg_maxy = polygon.bounds

    # Arrays are indexed from the top-left corner, so we need minx and maxy
    x0 = patch.bounds.minx.iloc[0]
    y0 = patch.bounds.maxy.iloc[0]
    max0 = int((patch.bounds.maxx.iloc[0] - patch.bounds.minx.iloc[0]) / res)

    # Construct bbox coordinates
    x1 = max(int((seg_minx - x0) / res), 0)
    x2 = min(int((seg_maxx - x0) / res), max0)
    y1 = min(int((y0 - seg_miny) / res), max0)
    y2 = max(int((y0 - seg_maxy) / res), 0)

    label_txt = f"{x1} {y2} {x2} {y2} {x2} {y1} {x1} {y1}"

    return label_txt

def prepare_labeltxt(one_patch, gdf, res):
    patch = gpd.GeoSeries(one_patch, crs=gdf.crs)
    bbox = patch.bounds.iloc[0]

    # Segments that fall into this patch (there can be more than 1 segments in one patch)
    patch_segments = gdf.cx[bbox.minx:bbox.maxx, bbox.miny:bbox.maxy].reset_index(drop=True)

    # DISCARD SEGMENTS WITH LESS THAN 1/3 AREA ON THE PATCH
    # Calculate intersection of every segment with the patch extents
    patch_segments["geom_intersection"] = [segment.intersection(one_patch) for segment in patch_segments.geometry]
    # Calculate area of segments and fraction
    patch_segments["area1"] = patch_segments.area
    patch_segments["area2"] = [intersects.area for intersects in patch_segments["geom_intersection"]]
    # Only keep segments with more than 1/3 of the area over the patch
    patch_segments["area_fraction"] = patch_segments["area2"] / patch_segments["area1"]
    patch_segments = patch_segments.loc[patch_segments["area_fraction"] > 0.33].reset_index(drop=True)

    # Flag segments on the edge of a patch [1 ... is on the edge, 0 ... entire segment is present]
    patch_segments["edge_patch"] = np.where(patch_segments["area_fraction"] < 1, 1, 0)

    # Create LabelTxt (use empty string if there is no segments in this patch)
    if len(patch_segments) > 0:

```

```

# Prepare AABBB with relative coordinates for LabelTxt (Requires resolution!)
patch_segments["relative_bounds"] = [
    relative_bounds(seg, patch, res) for seg in patch_segments["geom_intersection"]
]

# If "DFM" attribute doesn't exist, set value to 1 everywhere
if "DFM" not in patch_segments:
    patch_segments["DFM"] = 1

# Other parameters for LabelTxt
patch_segments["LabelTxt"] = (
    patch_segments[["relative_bounds", "arch_type"]].agg(' '.join, axis=1)
    + " " + patch_segments["DFM"].astype(str)
)

label_txt = "\n".join(patch_segments["LabelTxt"].values.tolist())
else:
    label_txt = ""

return label_txt

def create_segmentation_mask(poly, file_path, image, segments_list):
    # Determine bounds of a patch
    bounds = poly.bounds

    # Determine bounds of a patch
    with rasterio.open(image) as src:
        out_image = src.read()
        out_profile = src.profile.copy()
        out_transform = src.transform

    # Prepare output array (all data set to 0)
    patch_shape = out_image.shape[1:]
    out_image = out_image.astype('uint8')
    out_image.fill(0)
    # The seg mask will always have 3 bands
    out_image = np.repeat(out_image, 3, axis=0)

    # Loop over all three classes (if less than 3, leave empty arrays)
    for i, seg in enumerate(segments_list):
        # Segments that fall into this patch (there can be more than 1 segments in one patch)
        patch_segments = seg.cx[bounds[0]:bounds[2], bounds[1]:bounds[3]].reset_index(drop=True)

        # # DISCARD SEGMENTS WITH LESS THAN 1/3 AREA ON THE PATCH
        # Calculate intersection of every segment with the patch extents
        patch_segments["geom_intersection"] = [segment.intersection(poly) for segment in patch_segments.geometry]
        # Calculate area of segments and fraction
        patch_segments["area1"] = patch_segments.area
        patch_segments["area2"] = [intersected.area for intersected in patch_segments["geom_intersection"]]
        # Only keep segments with more than 1/3 of the area over the patch
        patch_segments["area_fraction"] = patch_segments["area2"] / patch_segments["area1"]
        patch_segments = patch_segments.loc[patch_segments['area_fraction'] > 0.33].reset_index(drop=True)

        # If any polygons were found, add them to segmentation mask raster
        if len(patch_segments) > 0:
            # Write each polygon separately, to ensure correct DFM value is written to file
            for _, vector in patch_segments.iterrows():
                # Find pixels covered by polygon
                array = geometry_mask(
                    [(vector['geometry'], 1)],
                    patch_shape,
                    out_transform,
                    all_touched=True,
                    invert=True
                )
                # Populate pixels with DFM value
                out_image[i, array] = vector["DFM"]

    out_profile.update(
        dtype="uint8",
        count=3,
        driver="GTiff",
        compress="LZW",
        nodata=None
    )

    # !!! Don't use predictor 3 for LZW compression of uint8 array !!!
    with rasterio.open(file_path, "w", **out_profile, predictor=2) as dst:
        dst.write(out_image)

    return file_path

def patches_grid(ds, archaeology, patch_size, stagger):
    """Creates GeoDataFrame for patches with LabelTxt attribute.

    LabelTxt: <x1> <y1> <x2> <y2> <x3> <y3> <x4> <y4> <Label> <DFM>

    Calls:
    - uniform_grid
    - prepare_labeltxt
    - relative_bounds

    Parameters
    -----
    ds
    archaeology
    patch_size
    stagger

    Returns
    -----
    """

```

```

with rasterio.open(ds) as src:
    ds_crs = src.crs
    ds_res = src.res[0]
    ds_extents = list(src.bounds)

df_segments = archaeology

if len(df_segments) > 0:
    # Convert pixels to meters
    tile_size = (patch_size * ds_res, patch_size * ds_res)
    if stagger:
        stagger = stagger * ds_res

    # Initial grid
    grid = uniform_grid(ds_extents, ds_crs, tile_size, stagger)

    # Keep only patches with archaeology
    grid_filter = grid["geometry"].intersects(df_segments.dissolve().geometry[0])
    grid = grid[grid_filter].reset_index(drop=True)

    # Prepare LabelTxt for all tiles (empty string if there is less than 1/3 of object inside the patch)
    grid["LabelTxt"] = ""

    grid["LabelTxt"] = [prepare_labeltxt(one_tile, df_segments, ds_res) for one_tile in grid["geometry"]]

    # Filter (remove patches with empty LabelTxt, i.e. containing less than 1/3 of a segment)
    grid = grid[grid["LabelTxt"] != ""].reset_index(drop=True)

    # Create "filestem" that will be used for naming files, which is basically lower-left coordinates
    grid["filestem"] = grid.bounds[["minx", "miny"]].astype(int).astype(str).agg('_', axis=1)

    return grid

def create_patches_main(input_raster, seg_masks_dict, output_directory):
    """Main routine for creating patches.

    Parameters
    -----
    input_raster : str or pathlib.Path()
        Path to input raster.
    seg_masks_dict : dict
        Dictionary containing labels. E.g.: {"barrow": <path to barrow vectors file>}. At least one label, not more than
        three labels.
    output_directory : str or pathlib.Path()
        Path to directory where patches are saved.

    """
    # =====
    # PROCESS INPUTS:
    input_raster = Path(input_raster)

    nr_processes = cpu_count() - 2

    output_directory = Path(output_directory)
    output_directory.mkdir(exist_ok=True)

    # =====
    # PROCESSING STARTS HERE!

    # STEP 1: Join all DFs into one, add arch_type column containing Label name of each polygon
    crs_df = None
    df_list = []
    for key, value in seg_masks_dict.items():
        df = gpd.read_file(value)
        df["arch_type"] = key
        df_list.append(df)
    crs_df = df.crs
    df_grid = gpd.GeoDataFrame(pd.concat(df_list, ignore_index=True), crs=crs_df)

    # STEP 2: Create grid for patches
    df_patches = patches_grid(
        input_raster,
        df_grid,
        patch_size=512,
        stagger=256
    )

    # STEP 3: Generate paths for saving patches
    for data_type in ["images", "segmentation_masks", "LabelTxt"]:
        suff = ".txt" if data_type == "LabelTxt" else ".tif"
        ds_name = input_raster.stem
        df_patches[f"{data_type}_path"] = [
            Path(output_directory / data_type / f"{x}_{ds_name}_{data_type}{suff}")
            for x in df_patches["filestem"]
        ]

    # # Save GDF (for DEBUG)
    # df_patches.to_file("test-names.gpkg", driver="GPKG")

    # STEP 4: Multiprocessing run for create single patch
    input_process_list = []
    for _, in_tile in df_patches.iterrows():
        input_process_list.append(
            (
                in_tile.to_dict(),
                df_list,
                input_raster
            )
        )

    print("Start multiproc")

    with mp.Pool(nr_processes) as p:
        realist = [p.apply_async(create_one_patch, r) for r in input_process_list]

```

```

    for result in realist:
        pool_out = result.get()

# # Single run (FOR DEBUG)
# tiles_gpkg(*input_process_list[10])

if __name__ == "__main__":
    # Define paths to inputs and outputs
    input_image = r"../test_data/test_patches/my_DFM.vrt"
    output_dir = r"../test_data/training_samples"

    # The dictionary HAST TO BE!!! in this format - at least one label and max 3 labels (there is no check)
    # Key is name of label and value is path to vector file. Can use any label name, in example default names are used.
    # segmentation_masks = {
    #     "barrow": r"../test_data/test_patches/arch/barrow_segmentation_TM75.gpkg",
    #     "enclosure": r"../test_data/test_patches/arch/enclosure_segmentation_TM75.gpkg",
    #     "ringfort": r"../test_data/test_patches/arch/ringfort_segmentation_TM75.gpkg"
    # }
    segmentation_masks = {
        "barrow": r"../test_data/test_patches/arch/barrow_segmentation_TM75.gpkg",
        "enclosure": r"../test_data/test_patches/arch/enclosure_segmentation_TM75.gpkg",
        "ringfort": r"../test_data/test_patches/arch/ringfort_segmentation_TM75.gpkg"
    }

    create_patches_main(input_image, segmentation_masks, output_dir)

```

9. Appendix v: Retraining of MLM

9.1. Overview

We offer two specialized Jupyter notebooks designed for the tasks of retraining machine learning models (MLMs) for semantic segmentation and object recognition.

Each notebook serves a dual purpose:

- **Updating** an existing ADAF model: using an already trained MLM on the ADAF dataset for further training to improve its predictive power.
- **Training** a new ADAF model: training a new ADAF model without using an existing model. It's important to note that while this approach doesn't build on an existing ADAF model, the model isn't learned from scratch. By default, our methodology uses already trained weights available in the `aitlas` toolbox to have a head start in the training process.

9.2. Getting Started

When using the notebooks, it's crucial to ensure the correct setup to allow for a smooth and successful retraining. Start by selecting the `aitlas` conda environment as the kernel (similar to the inference task). This environment contains all the necessary dependencies and libraries required for the tasks at hand.

Next, customise the input parameters and paths to your specific data. This step is crucial to ensure that the model is trained on the correct data and that the paths for accessing the input data and storing the output models are set correctly.

Once the environment is set and the parameters are configured correctly, proceed to run each cell in the notebook in turn. This step-by-step execution is designed to guide you through the process, from data preparation and model initialization to training and evaluation, ensuring that each phase is completed without problems before moving on to the next.

9.3. Jupyter Notebook Structure

Both the notebook for semantic segmentation and the one for object detection have the same structure, each consisting of six different sections:

9.3.1. Import Packages

In this step, the required dataset classes and the machine learning models are loaded from the `aitlas` Toolbox. This includes:

`aitlas.datasets.TiiLIDARDatasetSegmentation` for semantic segmentation

`aitlas.datasets.TiiLIDARDatasetObjectDetection` for object detection

`aitlas.models.HRNet` model for semantic segmentation

`aitlas.models.FasterRCNN` model for object detection

This step also includes setting up the model configuration, which determines the number of classes, enables the download of pre-trained weights, sets the learning rate to 0.0001 and determines the evaluation metric (mean average precision for detection and intersection over union for segmentation).

9.3.2. Loading train, validation and test data

This part deals with the preparation of the training, validation and test data sets. All relevant input parameters and the data paths are defined and the data sets are loaded.

Most of the input parameters for this step are the same for both learning tasks (semantic segmentation and object detection). Specific details are listed below:

- **batch_size:** The number of samples that are processed before the model is updated. A larger batch size can speed up processing but requires more memory. A large batch size can lead to a runtime error if the computer does not have the required memory to load the data from the batch. The default setting for this parameter is 16, although adjustments to the specific system capabilities may be required.
- **num_workers:** The number of worker processes used to process data. Increasing the number of workers can significantly speed up data processing, but also increases memory and CPU/GPU consumption. The default value is 4.
- **object_class:** A string value indicating the type of archaeological object class you are interested in processing, e.g. 'AO', 'barrow', 'enclosure', 'ringfort'.
- **vizuelization_type:** The visualisation type used for the patches, e.g. 'SLRM'.
- **DFM_Quality:** List of DFM qualities that should be included in the processed data, e.g. '1,2'.
- **shuffle:** This parameter controls the randomization of the data sequence before processing. To facilitate learning and prevent the model from memorizing the order of samples, it is recommended to enable shuffling (set to True) for training data. For validation and test datasets, shuffling is normally disabled (set to False).
- **data_dir:** The directory path where the input image data is stored.
- **annotations_dir:** The directory path in which the annotations are stored. This includes segmentation masks for semantic segmentation tasks and bounding box labels for object detection.
- **transforms:** An optional parameter that contains the list of transformations that will be applied to the input image data during processing. Although customization is possible, we recommend using the default settings specified in the notebooks.
- **target_transforms:** An optional parameter that contains the list of transformations that will be applied to the target (annotations – either segmentation masks or bounding boxes) during processing. Although customization is possible, we recommend sticking to the default settings specified in the notebooks.
- **joint_transforms:** An optional parameter that contains the list of transformations that will be applied to both the input image data and the target.

Semantic segmentation datasets have two additional parameters:

- **object_class_band_id:** An integer parameter that specifies the band in which the annotations for a particular object class are located within the segmentation masks. The band ID depends on how the data was pre-processed. An incorrect band ID can lead to a runtime error or loads annotations for a different object class. As this is a parameter with a zero-index, the values 0, 1 or 2 would be permitted for a segmentation mask with three bands. If the `object_class` parameter is set to 'AO', the annotations are loaded from all available bands. In this case, `object_class_band_id` should be set to None.
- **keep_empty_patches:** A Boolean parameter that determines whether empty patches are kept. It should be set to False during training, as "empty" data cannot be used for training. For testing or validation, True can be used to check how the model handles empty patches.

9.3.3. Model Creation

Here, the model is initialized with the predefined model configuration and prepared for training by setting up the required configurations and preprocessing steps.

9.3.4. Loading Pretrained ADAF Model (Optional)

This step is optional and allows you to load an existing MLM that has been trained on ADAF data. It's intended for users who want to further train an existing ADAF model. The only required input is the file path of the model. When you perform this step, the weights of the model created in step 3 are replaced by the weights of the model stored in the specified model path.

9.3.5. Training the Model

In this step, the actual training of the MLM is performed with the loaded training data and its performance is evaluated using a separate validation dataset.

There are three input parameters that the user must specify:

- **epochs:** The total number of training cycles the model will undergo. Each epoch represents a complete run of the training data set through the model.
- **model_directory:** Path to the directory where the trained model and its test points are stored. This is used to save the model during and after training.
- **run_id:** Name of the subdirectory within the <model_directory> directory in which the results of the various runs are saved.

Once this step is completed, the now trained model is saved in the specified directory (i.e. <model_directory>/run_id). The file with the name `best_checkpoint_*.tar` contains the model weights that achieved the best performance during training.

9.3.6. Model Evaluation

The final step is to evaluate the performance of the trained model on unseen test data. The only user input required is the file path to the model trained in the previous step, which is stored in the local file system. Once the evaluation process is complete, the evaluation metrics calculated for the test data are displayed.

Example of the output of a semantic segmentation model in the evaluation step:

```
{
  'IOU mean': 0.863975617602004,
  'IOU per Class': array([0.98613997, 0.74181307])
}
```

Our primary focus is on the "IOU per Class" results, which provides specific Intersection over Union (IoU) scores for each class. The first value indicates the IoU for the Background, and the second is the IoU for the archaeological object class, as determined by the `object_class` parameter. The goal is to train models that have higher IoU scores for the selected object class (the second value).

Example output of an object detection model evaluation step:

```
{
  'map': 0.511692225393075,
  'map_50': 0.8819712400436401,
  'map_75': 0.5255807631868335,
  'map_small': -1.0,
  'map_medium': 0.4580733180064815,
  'map_large': 0.5440743028861223,
  'mar_1': 0.550572486406042,
}
```

```
'mar_10': 0.5869565010070801,  
'mar_small': -1.0,  
'mar_medium': 0.5909090904211426,  
'mar_large': 0.623404255032,  
'map_per_class': 0.511692225393075,  
'mar_100_per_class': 0.5869565010070801,  
'classes': 1.0  
}
```

The evaluation metrics presented here reflect the performance for the `object_class` only, without the background class. The mean average precision (mAP) is calculated for different IoU thresholds. It is advisable to optimize the `map_per_class` metric, which represents the mAP at an IoU threshold of 0.5, as this is a commonly used metric for object detection tasks.

10. Appendix vi: MLM & ADAF tool security

The developed toolbox is built upon the foundation of the Jupyter Notebook platform, leveraging its robust security model to ensure the integrity of user interactions and data handling. This security model adheres to stringent guidelines designed to safeguard against potential threats:

- **HTML Sanitization:** Any untrusted HTML content is rigorously sanitized, mitigating risks associated with malicious code injection.
- **Javascript Execution Control:** Untrusted Javascript code is strictly prohibited from execution within the notebook environment, reducing the possibility of malicious scripts compromising system integrity.
- **Markdown Cell Restrictions:** HTML and Javascript embedded within Markdown cells are treated as untrusted by default, reinforcing the principle of cautious handling for potentially hazardous content.
- **User-Generated Outputs:** Outputs generated directly by the user's code execution are inherently trusted, as they originate from the current user's actions within the notebook.
- **External Content Handling:** HTML or Javascript content originating from sources other than the current user, whether within Markdown cells or outputs generated by others, is consistently treated as untrusted to prevent unauthorized access or execution.

Central to this security paradigm is the fundamental question of trust: every action or output within the notebook environment is evaluated based on whether it can be attributed to the current user.

Furthermore, to maintain data integrity across sessions, each notebook's contents are cryptographically signed using a combination of its content digest and a secret key. This signature is then securely stored in a designated database, accessible only to the current user. By default, this database resides in platform-specific locations:

For Linux: `~/.local/share/jupyter/nbsignatures.db`

For macOS: `~/Library/Jupyter/nbsignatures.db`

For Windows: `%APPDATA%/jupyter/nbsignatures.db`

Each signature serves as a unique identifier for the series of outputs generated by code executed by the current user, thereby establishing a basis for trust.

Upon opening a notebook, the server automatically computes its signature and cross-references it with the database. If a match is found, HTML and Javascript outputs within the notebook are marked as trusted upon loading, ensuring a secure environment for subsequent interactions. Conversely, if no matching signature is found, such outputs are treated as untrusted as a precautionary measure.

Additionally, any outputs generated during interactive sessions are inherently trusted, providing seamless functionality without compromising security. This holistic approach to security empowers users to confidently engage with the notebook environment while mitigating potential risks associated with external content or code execution.

11. Appendix vii: Suggestions for future improvements

- Increased options for additional post processing
- Simplify the installation process
- Move processing to cloud
- Integration with external databases (such as land-use land-cover maps, archaeological records, etc.)