

# User Manual

V1

# Contents

<b>Glossary of Terms</b>	<b>1</b>
<b>1 About Using the Species Detection Project</b>	<b>2</b>
1.1 How the files are organised	2
1.2 How to install and setup the system for use	3
1.2.1 Hardware Pre-requisites	3
1.2.2 Software Pre-requisites	3
1.3 Using Anaconda and Jupyter Notebook	3
1.3.1 Anaconda basics	3
1.3.2 Using a notebook in Jupyter	4
1.3.3 Using a Notebook	4
1.3.4 Setting up a Conda environment for the project	5
1.3.5 Checking the installation	5
<b>2 How to carry out detection of species on a raster</b>	<b>5</b>
2.1 Run the detection Notebook Detection	5
2.2 Interpreting the result	6
2.3 Inputting a results image into ArcGIS	6
<b>3 Training a model (simple version)</b>	<b>10</b>
3.1 Basics	10
3.2 Exporting an orthograph's rasters and labels	10
3.2.1 Troubleshooting exportation	10
3.2.2 Rejoining split tiles (if necessary)	11
4 Creating training tiles	11
5 Train from scratch or continuing training	12
6 Training	12
7 Producing a Confusion Map	13

## Glossary of Terms

Term	Simple Definition
Anaconda	The program on which all the components of the project run
Anaconda environments	Isolated environment where software libraries can be installed and used. This allows different environments to have software installed without interfering with each other, even though they are on the same machine
Anaconda Prompt	A command line tool to create and manage Anaconda environments, and use the software within.
Class	The AI model distinguishes between different classes. In the case of this project most classes correspond to a specific plant species.
Epoch	One session of training where an AI model goes over its entire training data and makes internal adjustments to learn the task. Training a model usually requires a large number of epochs.
Label	A man-made, pixel-based map showing the model which species are present at each pixel. Usually not easily readable by a person.
Species Map	A pixel-based map, generated by an AI model showing which species are likely present at each pixel according to said model. Each individual plant-type is represented by a unique colour.
Model/AI Model	For the scope of this project; a system can be trained, or is trained to recognise the presence of plant species.
Notebook	A file containing modifiable function or programs that work on the project's data; often using the AI model (for example: to detect the presence of species in a drone image).
Jupyter Notebook	A program to manage and use Notebooks. See: Notebook
Single Class Model	An AI model that is trained to detect only a single class (species). See Also: Class.
Tiles	An entire image is broken into small tiles in preparation for use in training. AI Models only train on small tiles. Each image tile will have a corresponding label tile.
Training	The process of adapting an AI Model for a task, by repeatedly exposing it to input data (an image) and the expected output for that input data (a label).
Train set	The data that is seen by the AI Model during training.
Validation set	Data not seen during the model's training but is essential to evaluate how well the model is learning a task.

# 1 About Using the Species Detection Project

This section contains general information that is essential to use the Species Detection software properly. The rest of this User Manual assumes knowledge contained under this heading. Instructions in subsequent sections will refer to specific parts of this section for further clarification.

## 1.1 How the files are organised

All files in the project are enclosed in the folder named IAS (short for Invasive Alien Species) which contains the necessary files to run the project and subfolders dedicated for specific use. The following table explains each location's purpose.

Folder Name	Description and Contents
IAS	Contains all files associated with the project, like: <ul style="list-style-type: none"><li>• Jupyter Notebook files to run the programs (.ipynb files)</li><li>• Files containing necessary Python code (.py files)</li><li>• Subfolders like Data, Models, Results...</li></ul>
IAS/Models	<p>This is where the trained Deep-Learning models are to be stored.</p> <p>Each subfolder in here contains saved checkpoints (trained models) for a specific trained model (.torch files) as well as some other files that are produced as a result of training (.optim .sched .csv...)</p> <p>When using the project, at least one of the checkpoints stored here will be used as the model that performs the detection of species, and will have to be referenced.</p>
IAS/Models / _usable	This folder is for model checkpoints that are deemed good to use for species detection
IAS/Models/ _usable_1_class	This folder is for model checkpoints to use for detecting one specific species.
IAS/Data	<p>This is where data that the AI models use is stored. These consist mostly of rasters and labels in .tif format that were exported from ARCGIS ortho-mosaics and shapefiles.</p> <p>This is also where the programs expect files to be stored by default. NOTE: The AI models only work with pixel data so anything placed here for the AI model to use must be exported in .tif format first.</p> <p>In general, this subfolder contains:</p> <ul style="list-style-type: none"><li>• Subfolders containing image data the models work on</li><li>• Some files used to analyse or manipulate the various image data contained herein</li></ul>
IAS/Data /source	<p>This is where full size rasters and corresponding labels are to be stored.</p> <p><b>IAS/Data/source/rasters</b> contains the full colour, raster image of the location as captured by the drone.</p> <p><b>IAS/Data/source/labels</b> contains the corresponding label map for each full colour raster. The label map identifies species and where they are present in the location. <u>These label maps are usually understood to be labelled manually by a person.</u></p> <p>Example: <b>IAS/Data/source/rasters/Ar1.tif</b> is a colour raster and has a corresponding grayscale bitmap at <b>IAS/Data/source/labels/Ar1.tif</b> to map the labelled species contained in the location.</p> <p>If the label is available, this raster can be used to:</p> <ol style="list-style-type: none"><li>1) Train new models</li><li>2) Compare the detection performed by a model against the labelling performed by a person</li></ol>
IAS/Data /traindata	<p>This subfolder contains images and labels in a format used to train models.</p> <p>These consist of 256 by 256 sized tiles (other sizes are possible) that can be derived from the larger images inside <b>IAS/Data/source</b>.</p>
IAS/Results	When models perform detection, resulting images are saved in this location

An up-to-date copy of the project's programs can be downloaded from this link:

[https://github.com/MarkMifsud/MITA-IAS-Invasive\\_Alien\\_Species/archive/refs/heads/main.zip](https://github.com/MarkMifsud/MITA-IAS-Invasive_Alien_Species/archive/refs/heads/main.zip)

## 1.2 How to install and setup the system for use

### 1.2.1 Hardware Pre-requisites

An internet-connected computer with a powerful GPU is essential. Ideally an Nvidia GPU with is required.

Disk space needs varies depending on how many rasters, labels and other maps one needs to store for detection. Considerably more disk space is required to train a model. Expect at least, an additional 110Gb to be required to train a model. This will include the data adapted for training and the checkpoints saved during training.

### 1.2.2 Software Pre-requisites

**Requirement 1)** This manual assumes you have ARCGIS Pro installed together with an Advanced licenses which includes Spatial Analyst license specifically.

**Requirement 2)** Anaconda with Jupyter notebook needs to be installed to run the AI models. If you have ARCGIS installed, Anaconda is usually installed by default. If it is not installed it can be downloaded for free from [www.anaconda.com](http://www.anaconda.com)

**Requirement 2)** You also need an internet browser.

**Requirement 3)** Once Anaconda is installed, please follow these instructions to complete the setup of a Conda environment so that the AI models can function properly (see section 1.3.3 *Setting up a Conda environment for the project*).

## 1.3 Using Anaconda and Jupyter Notebook

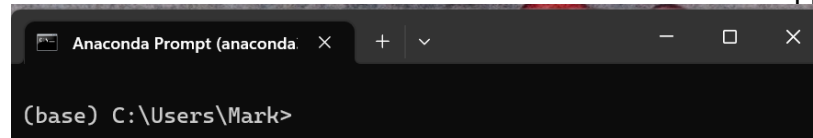
Anaconda (or simply Conda) is used to set up an independent environment that contains all the required software to run a project. Jupyter Notebook is one of the programs that is commonly available in a Conda environment.

### 1.3.1 Anaconda basics

To run Anaconda locate the Anaconda Prompt app from the Windows Start menu or by using the Search feature.

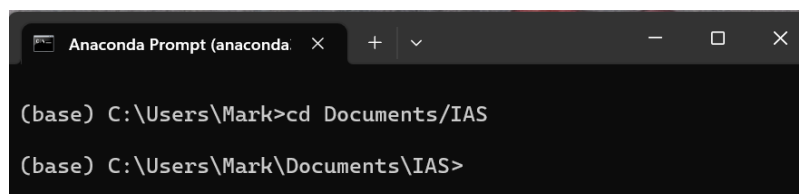


A black window with a command line similar to this one will appear.



The one in brackets is the name of the environment that is active. (base) is the default environment. Any software library that is installed from the Anaconda Prompt gets installed in the active environment without affecting (or conflicting with) the rest of Windows, or other environments. Creating other environments is possible.

The one following the environment name is the active folder. You can navigate to a different folder using the normal command line commands. **It is important to point at the folder containing the project.** In this example we use the cd command to navigate to the IAS folder.



Typing> **jupyter notebook** launches the jupyter app in your web browser.

If jupyter notebook does not start type the following command: **pip install notebook** then try launching jupyter notebook again.

**IMPORTANT:** Do not close the Anaconda Prompt while jupyter is running, or every running process will be ended abruptly.

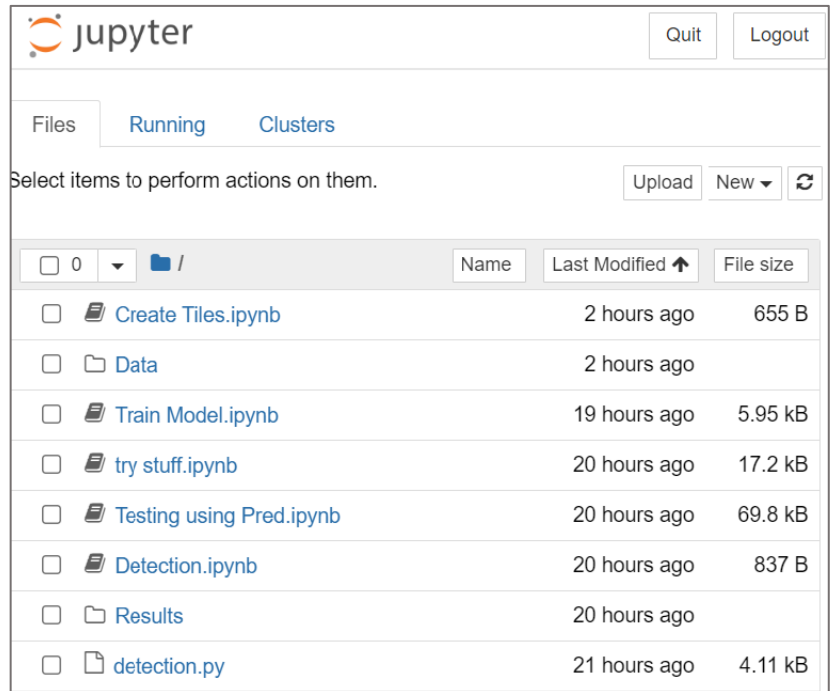
### 1.3.2 Using a notebook in Jupyter



Upon Jupyter appearing in the web browser you will see files and folders present in the folder active in the Anaconda Prompt.

If the content you see is not the same content present in the IAS project's folder you need to quit jupyter (button at the top right of the jupyter page) navigate to the IAS project folder in the Anaconda Prompt and launch jupyter notebook again.

Some icons represent folders. Clicking the folder's name shows its content.

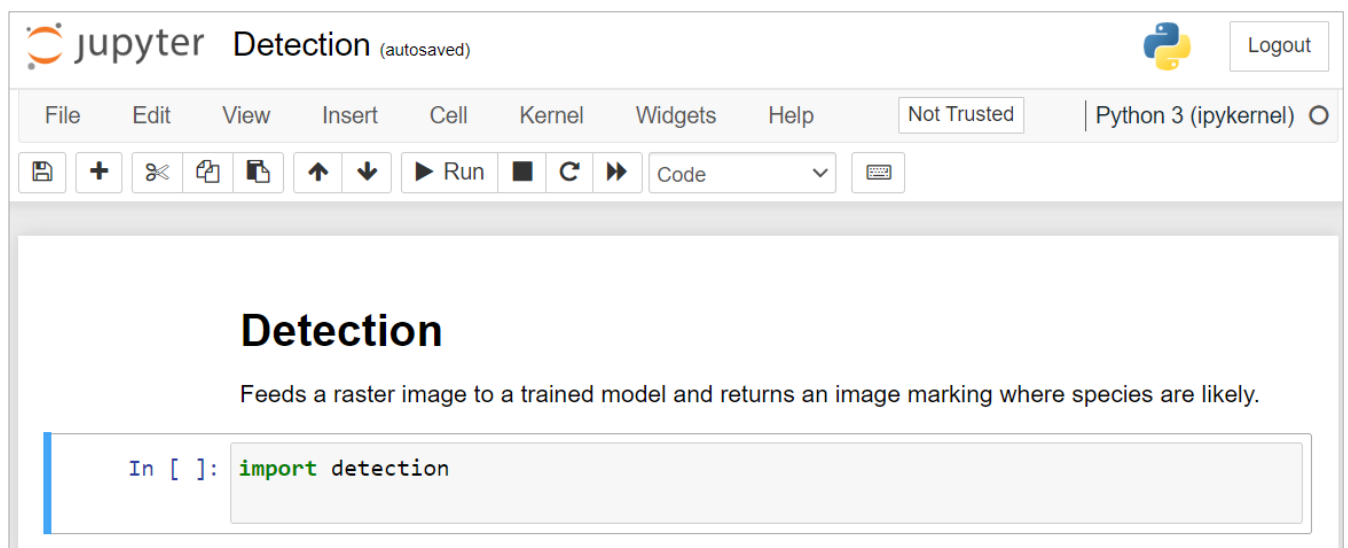
You can exit a folder by clicking the .. folder icon at the top of the icon list.





The .ipynb files are notebooks. Black notebooks are inactive.  Notebooks that are running have a green icon. 

### 1.3.3 Using a Notebook

Each notebook contains function to use the AI models or do something to the data. Double clicking a notebook's name opens that notebook in a separate browser tab.




The code inside a notebook is separated in different cells. For convenience, in most cases the functions have been compressed to one command inside a single cell (image above).

You can run all cells in sequence using the Run All button  or run each cell independently by selecting it (clicking on it) and clicking on the Run button . If you ever wish to stop a running process you can use the stop button.

**Note:** To stop a notebook from running always use the **Close and Halt** command from the **File** Menu. Simply closing the tab will not stop the notebook; it will keep running in the background.

### 1.3.4 Setting up a Conda environment for the project

The easiest way to install all the necessary tools for the project to work is to run the notebook called 'Setting up the environment.ipynb'.

1. First make sure that you are connected to the internet. The setup requires an internet connection.
2. Open the Anaconda Prompt
3. Enter the location of the folder where the notebooks are stored with the `cd` command.
4. (Optional) Activate the environment you wish to install the requirements in by typing **activate** followed by the name of the environment you wish to use. If you skip this step the default (base) environment will be used.
5. Type `jupyter notebook` to open jupyter
6. Click on the notebook named 'Setting up the environment.ipynb'. It will open in a separate browser tab.
7. Click the Run All button.  Confirm any prompts that may pop-up.

### 1.3.5 Checking the installation

Each cell should give an output. If the process halts before the last cell something in the process went wrong.


## 2 How to carry out detection of species on a raster

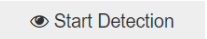
For an AI model to carry out detection of species on an orthograph, it must first be exported as a raster in .tif format. The rasters should be saved in location `Data/source/images`.

The following are necessary steps to perform detection...

### 2.1 Run the detection Notebook Detection

- 1) Decide if you wish to detect a single species or multiple species all at once<sup>1</sup>.
  - a) To Detect a single species, in Jupyter Notebook launch the notebook named [Detection-Single Class Models.ipynb](#). This uses models that specialise in detecting one specific plant class.
  - b) To detect a number of plant species at once, launch the [Detection.ipynb](#) notebook in Jupyter Notebook.

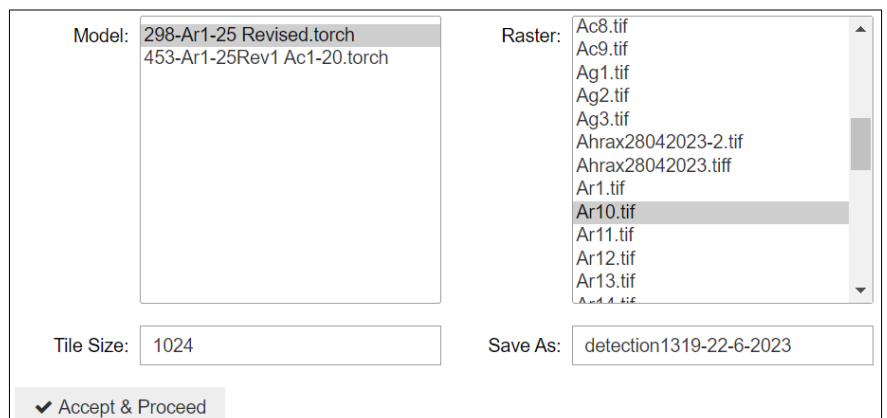
2) When the notebook opens up in your browser. Click the Run All  button

3) Click the  button

- 4) From the Model options window pick the trained model you want to use for detection. Note that these are the models stored in `IAS/Models/_usable` or `/_usable_1_class`.

- 5) From the Raster options window pick a raster to apply the detection model on. These are the rasters saved in `IAS/Data/source/rasters/`

- 6) (optional) Pick a tile size in the Tile Size box. This should be a multiple of 8. The ideal size when using a GPU that is limited in memory is 512. Sizes can go up to 3600 on machines with powerful GPUs



The screenshot shows a web interface for species detection. It has two main columns: 'Model' and 'Raster'. The 'Model' column has a dropdown menu with options: '298-Ar1-25 Revised.torch', '453-Ar1-25Rev1', and 'Ac1-20.torch'. The 'Raster' column has a list of files: 'Ac8.tif', 'Ac9.tif', 'Ag1.tif', 'Ag2.tif', 'Ag3.tif', 'Ahrax28042023-2.tif', 'Ahrax28042023.tiff', 'Ar1.tif', 'Ar10.tif' (which is highlighted), 'Ar11.tif', 'Ar12.tif', and 'Ar13.tif'. Below these columns, there is a 'Tile Size' input box with the value '1024' and a 'Save As' input box with the value 'detection1319-22-6-2023'. At the bottom, there is a button labeled 'Accept & Proceed' with a checkmark icon.

<sup>1</sup> Refer to Section 1 for details on how to do this, if in doubt. Single species detection may be more accurate, while multi-species detection is faster on multiple species but may lack accuracy with some.

(24Gb or more). The larger the tile size the faster the detection will go, however, large sizes may cause the notebook to crash when using large rasters.

- 7) In the Save As box enter the name you want to give to the resulting species map. This box will contain a name by default but it is recommended that you give the map a meaningful name. As a good practice, it is recommended to include a combination of the raster name and the model used to name the result.
- 8) Click on the Accept and Proceed button.
- 9) Let the process finish. The results will be saved as a .tif file inside the Results folder (IAS/Results).
- 10 (Optional) Repeat steps 3 to 10 if you wish to run the detection process once again.

## 2.2 Interpreting the result

The model generates an image where each colour marks the species it detected. Black areas are where no species of interest were detected; each other colour matches a plant species as shown in the table on the right.

Class Name	Abbreviation	Class Value	Labelling Colour
Arundo_Class_1	Ar	10	Green
Opuntia_Class_1	Op	20	Red
Class3	3	30	Blue
Eucalyptus	Eu	40	Pink
Agave_Class_1	Ag	50	Purple
Acacia_Class_1	Ac	60	Yellow

The best way to view this result is by importing it in ARCGIS (next step).

## 2.3 Inputting a results image into ArcGIS

Step 1 – Open the orthomosaic into ArcGIS by adding the tiff file.

Step 2 – Next, a bounding box for the ortho will be created. Open the Analysis Tab and select tools.

Step 3 – Search for the create fishnet tool.

**Step 4 – Select the name for the output feature, you can call it the name of the ortho and bounding box for example. Select the dropdown near template extent and choose the ortho.**

**Step 5 – Input 1 and 1 into the number of rows and columns. Then choose polygon as geometry type. Press Run.**

**TIP:** If you wish to prevent the model from labelling certain regions, you can save a .tif file marking the regions you wish to exclude and save it in /IAS/DATA/source/exclusion/ folder. These maps can be

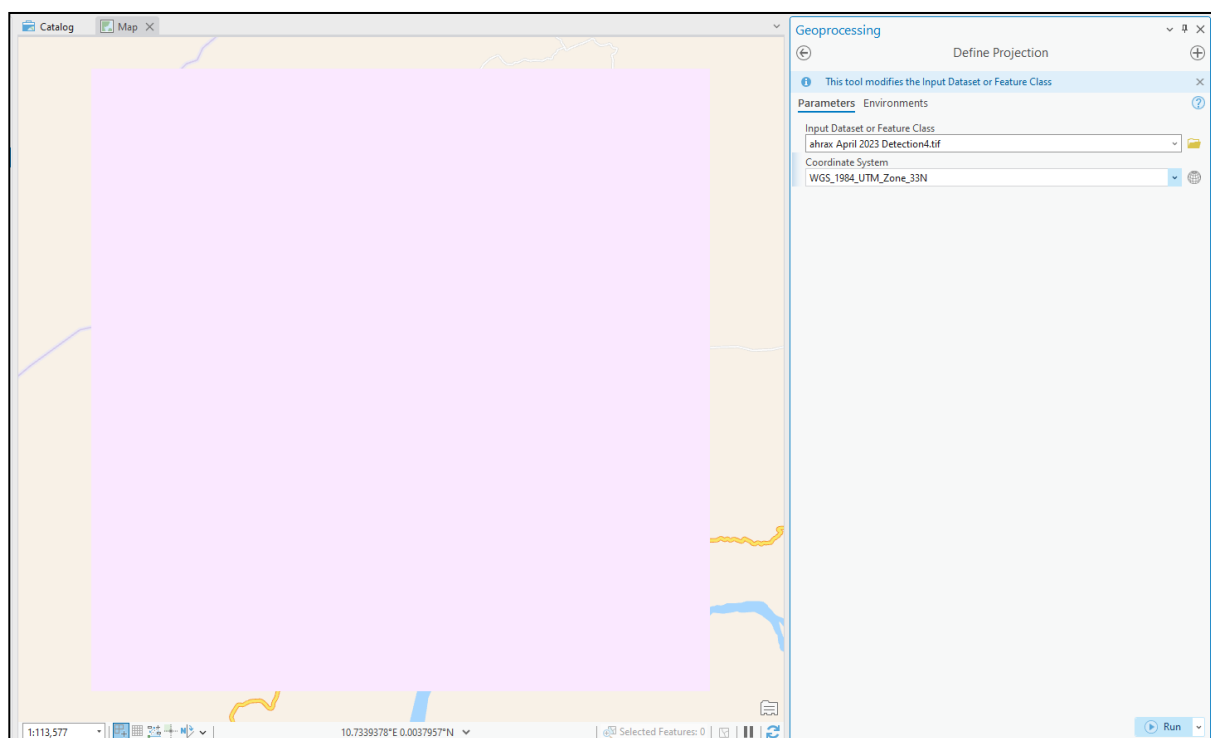
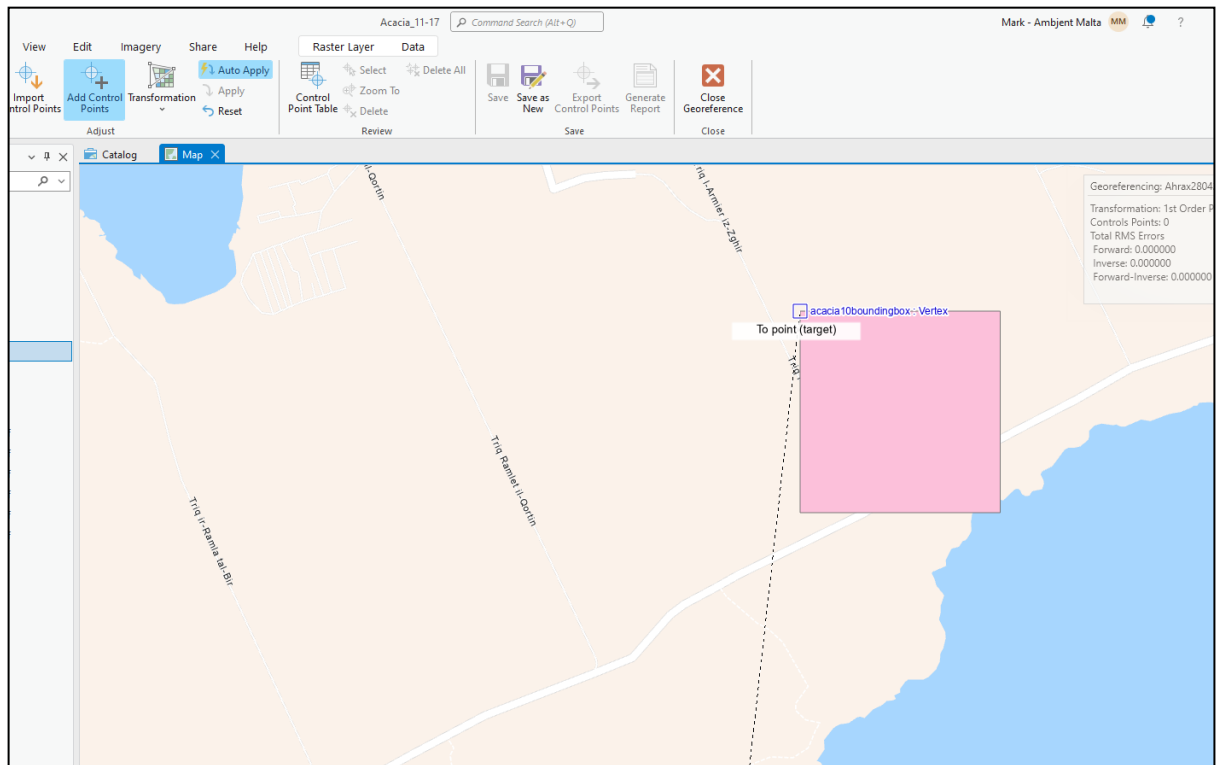


made in either ArcGIS or a photo editor like GIMP or Photoshop. Make sure that the areas you wish to exclude are white and the areas to include in the detection are black. Name this Exclusion Map with the same name as the saved raster and corresponding label. If a detection process finds such a map, it will be used automatically.

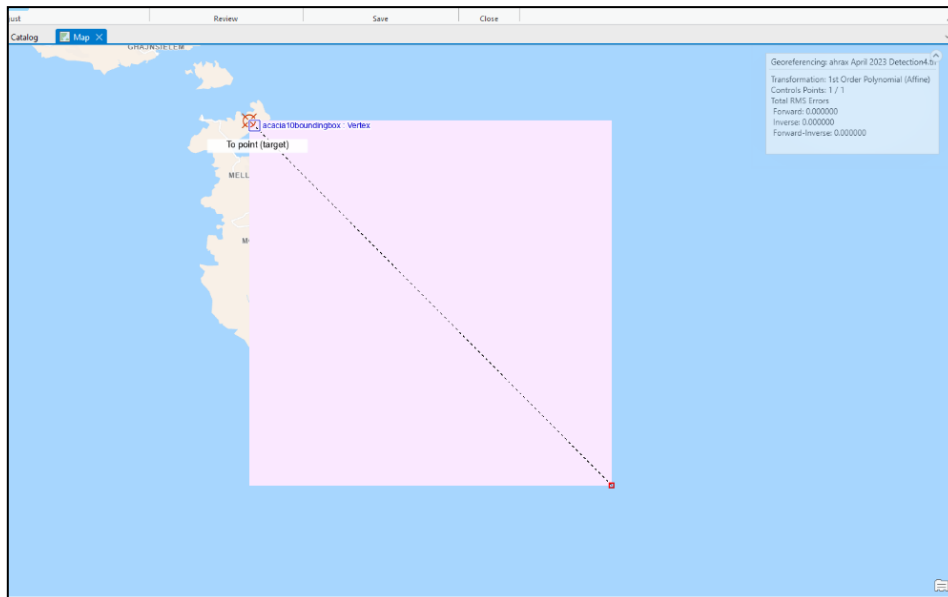
Step 6 – Open the results image into ArcGIS, it must now be georeferenced onto the orthomosaic.

Step 7 – Assign an SRS to the results image with the Define Projection tool. Ensure it is the same as the ortho.

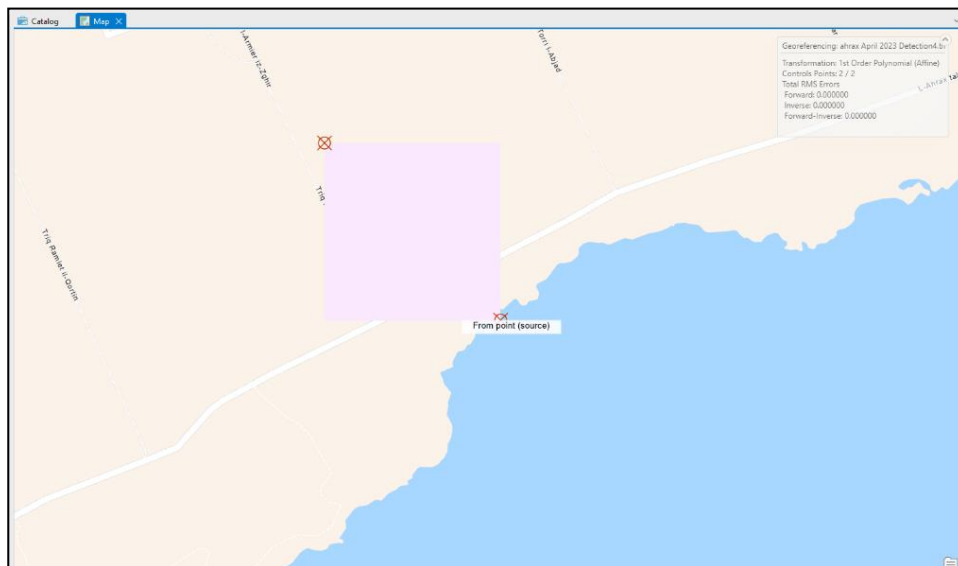
Step 8 – Open Georeference in the Imagery tab.



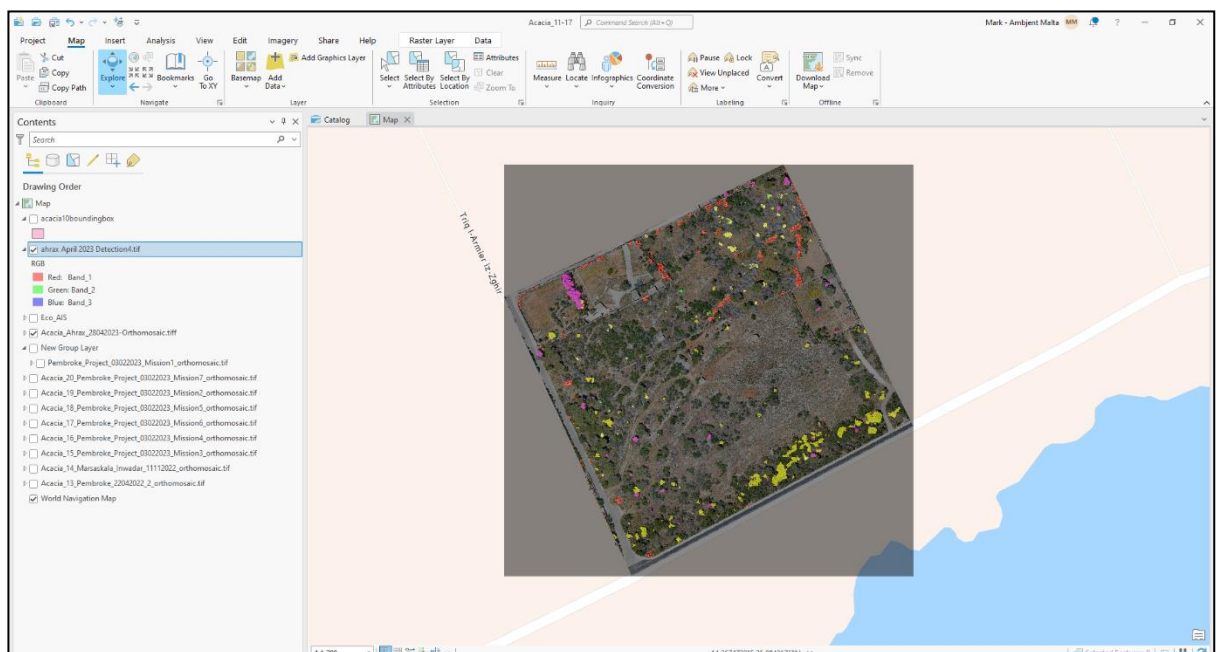
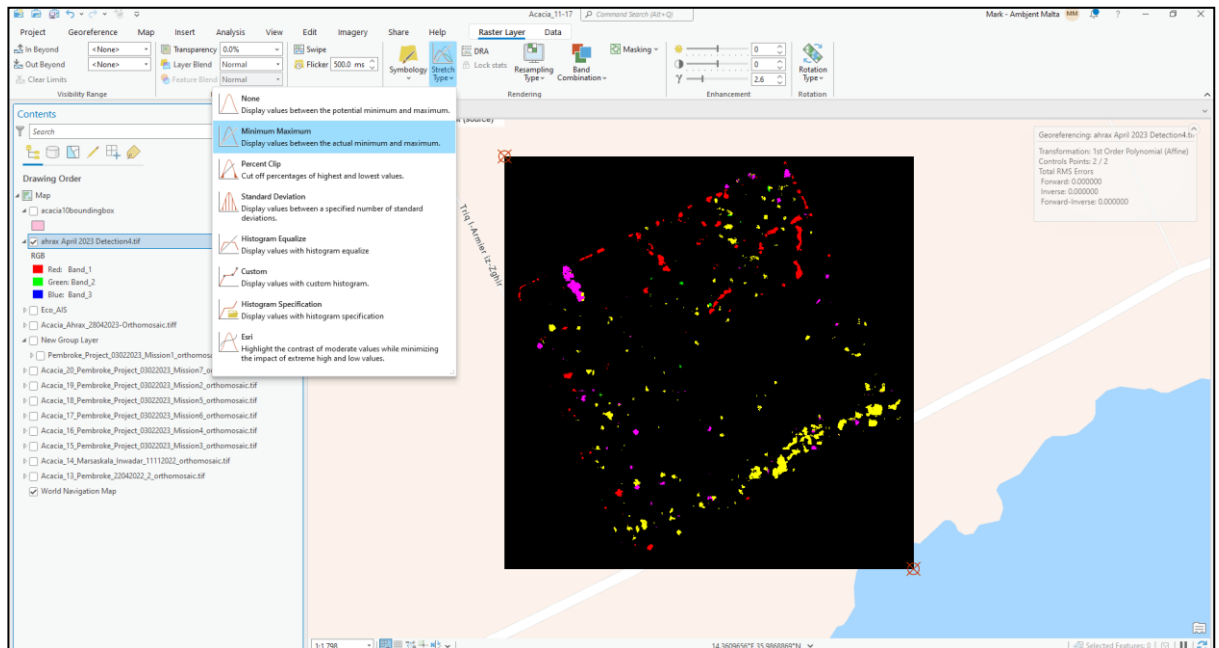
Choose add control points, select as close as you can to the top left vertex of the results image, and snap it to the corresponding vertex of the bounding box.



Repeat for the opposite bottom right vertexes and Save.



Step 9 – Edit any imagery values to ensure that you can correctly analyse the results. A good example is to enable Minimum-Maximum in Stretch in the Raster Layer Tab. Add transparency to analyse over the orthomosaic.



### 3 Training a model (simple version)

This section describes how to train one model (Unet++ with Resnet152 backbone), with many parameters fixed.

#### 3.1 Basics

The model trains for a number of epochs. It is not necessarily the case that the more training the better the model becomes. The performance of the model is evaluated at each epoch during its training and even after the training so that the best fit is found. In order to evaluate the model's training, it is custom to use 2 separate sets of data, one for training (training set) and one for validation (validation set). A 3rd set called the testing set is also commonly used after the training.

If you have a manually labelled orthograph and wish to compare the model's detection to the manual one the label needs to be exported as well. Ideally the label should be saved in location Data/source/labels.

**IMPORTANT:** Make sure that the raster and any corresponding label, has the same name and are each in the correct folder.

#### 3.2 Exporting an orthograph's rasters and labels

While you only need an exported orthomosaic to detect plant species, it may be convenient to export labels and other data in a single process, if these are available and required. Exporting labels allows you to compare the model's detection against the labels.

The following steps assume that you have shapefile labels that are completed and checked. If you need assistance with this refer to Appendix 1

In ARCGIS, go to the Imagery Tab. From Classification Tools choose Label Objects for Deep Learning

Make sure that the Output No Feature Tiles checkbox is marked.

After inputting the location, the size of the tile and stride and other parameters as shown in the image to the right. Note that the x & y values change for each orthomosaic.

Click run to start the exportation process.

**Important:** Save the images and labels in the apposite folder IAS/Data/source .

The screenshot shows the 'Export Training Data' dialog box. The 'Output Folder' is set to 'I:\My Drive\Files for'. The 'Image Format' is 'TIFF Format'. The 'Tile Size X' and 'Tile Size Y' are both 22397 and 22538 respectively. The 'Stride X' and 'Stride Y' are also 22397 and 22538. The 'Rotation Angle' is 0. The 'Reference System' is 'Map space'. The 'Output No Feature Tiles' checkbox is checked. The 'Meta Data Format' is 'Classified Tiles'. The 'Additional Input Raster' is empty. The 'Run' button is at the bottom right.

##### 3.2.1 Troubleshooting exportation

In cases where the orthomosaic is very large, the exporting can fail due to memory or hardware limitations. In these cases, it may be possible to export the data in 4 (2 by 2) tiles and re-attaching them afterwards.

To export the images in 4 separate tiles, set the stride X & Y to be half of the image sizes X & Y respectively. In case of the ortho having either an odd x or y or both, divide the size by two, round to next whole number, and add one to the stride. Note table below for examples.

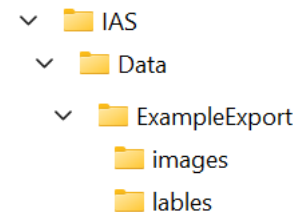
A table for example export size & stride values:

Ortho X & Y	Size X & Y	Stride X & Y	Result
20000x15000	20000x15000	20000x15000	Success, 1 image
30000x40000	30000x40000	30000x40000	Fail, error, so:
30000x40000	15000x20000	15000x20000	Success, 4 images
26497x34020	13249x17010	13250x17010	Success, 4 images
26497x34021	13249x17011	13250x17012	Success, delete all images & labels apart from first 4 (keep those ending 0 to 3, the rest are not needed)

**Note:** If there a tile fails to export because there is not label present in it, do not worry about it.

### 3.2.2 Rejoining split tiles (if necessary)

A Jupyter notebook that re-attaches the tiles is available in the IAS/Data/ folder. It requires that you store the tiles in the following manner:



Step 1) Inside IAS/Data/ place the exported folder containing the colour tiles in a folder named image and the label tiles in a folder named labels (example in image on the right). This is the default manner ARCGIS exports the data for classification.

Step2) Rename the exported folder to match the name you want to give to the resulting images once attached. The image and the label (if included in the export) will be saved using the name you give to this folder.

Step 3) Using Jupyter Notebook locate the notebook called ***stick split rasters.ipynb*** inside the IAS/Data folder. Launch it and click the Run All button.

Step 4) You will be asked to enter the name of the folder containing the images and labels subfolders. Enter the name and press Enter. Wait for the process to finish. The image and label will be automatically saved in the apposite folder IAS/Data/source.

## 4 Creating training tiles

The model is trained on tiles. The following section will describe how tiles can be created if not available and how to store them.

Tiles are recommended to be of size 256x256 or 512x512 pixels. Tiles can be created either by exporting them from ARCGIS or by generating them from the stored images in the IAS/Data/source folder. This section shows how to use a Jupyter notebook to create a set of data for training and testing.

Step 1) Make sure that the location you want to use for training:

- 1) Has its raster saved in IAS/Data/source/rasters in .tif format
- 2) Has a corresponding label with the same name saved in IAS/Data/source/labels in .tif format

Step 2) Open the ***Create Tiles.ipynb*** notebook located in the IAS folder and press the **Run All** button.

Step 3) The first prompt will ask you for the name of the raster. Type the name without including the .tif extension and press enter.

Step 4) You now need to confirm the tile size and the stride. The default tile size and stride are 256. Only change these numbers if you know what you're doing. Click the '**Accept & Proceed**' button to confirm these parameters.

Step 5) Wait for the tiling process to finish. The tiles are saved in the IAS/Data/dataset folder during this process.

Step 6) Repeat steps 1 to 5 for all the rasters you want to include in a data set (train set, valid set or test set). These will also be saved in the IAS/Data/dataset folder without overwriting.

Step 7) When all rasters you need have been tiled rename the dataset folder in IAS/Data/ to the name you wish. Once you rename the dataset folder, if you tile a new raster, a new dataset folder will be created from scratch. This is useful to create separate sets.

Step 8) The folder IAS/Data/trainData is a good place to store your data sets. It is recommended you move them there manually (but this is optional).

Note: Before training, please ensure that inside the folders containing training data (the ones named images and labels) there are ONLY tiles in .tif format. Any other files present are unnecessary and can interfere with, slow down or crash the training.

## 5 Train from scratch or continuing training

The notebook automatically decides whether to start training from scratch or continue training the last model from its last state. This is decided based on presence (or absence) of a log file in .csv format (can be opened in a spreadsheet program) in the main IAS folder.

The log file is normally named in the pattern: **Log for MC [Model type]-[Backbone]-[hour][minutes]-[day]-[month]-[year].csv**

Example: LOG for MC-UnetPlusPlus-ResNetEnc-756-25-2-2023.csv

If such a log is present the notebook will continue training from the last saved epoch; if not the training will start from scratch. This file can safely be deleted if you don't want to continue training, since a copy of it is saved in the same folder where the epochs are saved. Example: Models/UnetPlusPlus-ResNetEnc-756-25-2-2023/

## 6 Training

IM) Disable the computer's power saving features and pause all automatic updates (**important**).

Step 1) If you wish to train from scratch, check that there are no "LOG for MC-UnetPlusPlus-ResNetEnc " files in the IAS folder. Delete or move any that are present.

Step 2) Open the **Train Model.ipynb** notebook and click the Run All button.

Step 3) In the first line you may want to change the out channels to reflect how many classes you are interested in training for. If you wish to train a model for 7 species set the Out Channels to 8 (7 species + 1 for background). To train for a single species set the Output Channels to 2.

Step 4) In the lists that appears pick the Training and the Validation sets.

Step 5) Set the Batch size. The default value is calculated based on the available VRAM on your GPU. It assumes you are training with 256x256 tile, so lower that if you are

training on larger tiles. If the process crashes with a 'CUDA out of memory error' the batch size needs to be lowered. The larger the batch size the faster the training will go.

Step 6) Set the number of epochs to train for. If you are continuing training this is the additional amount of epochs that the model will be trained for.

In Channels:	3	Out Chann...	7
Train:	<div> 512-Ar1-25Rev1Ac1-20trainset  512-Ar111Ar21Rev1Ac10validset  A12346789ABm  A12346789ABm-512  A4m  A4m-512  A5Em  A5m  A5m-512  A9-COARSEGRAINED  A9Am  A9m-FINEGRAINED </div>	Validation:	<div> 512-Ar1-25Rev1Ac1-20trainset  512-Ar111Ar21Rev1Ac10validset  A12346789ABm  A12346789ABm-512  A4m  A4m-512  A5Em  A5m  A5m-512  A9-COARSEGRAINED  A9Am  A9m-FINEGRAINED </div>
Batch Size:	32	Epochs:	300
<input checked="" type="checkbox"/> Accept			

Step 7) Click Accept and let the process run.

Step 8) (**Single Class Only**) before training, a single class model you will be asked to specify the class value. This is always expected to be a multiple of 10, corresponding to the value assigned in ArcGIS.

## 7 Producing a Confusion Map

To generate a confusion map, you need to have both a trained model and a raster with a corresponding label. A confusion map is a visualisation done for each species. It shows:

- 1) where the model's detection matches the human labelling (true positive in green)
- 2) where the model suggests the species but the label does not (false positive in red)
- 3) where the model does not detect a species but the label does (false negative in blue)
- 4) where neither the label nor the model see a species (true negative in black)

This is useful to evaluate how well a model works.

Step 1) Open the Jupyter notebook named **Confusion Map.ipynb** and click the Run all button.

Step 2) Set the variable `output_classes` to equal the model's output channels (species+ 1 for background).

**If testing a single class model follow the instructions in the notebook carefully.**

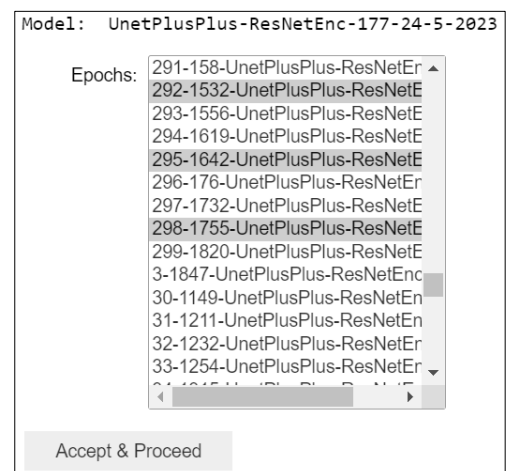
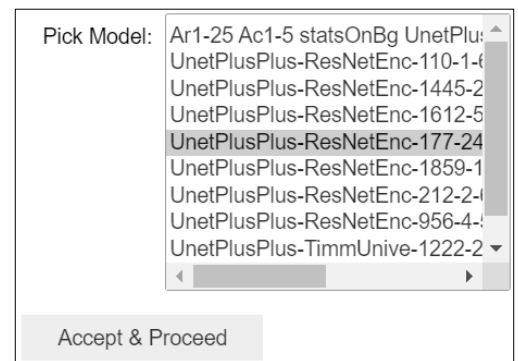
Step 3) In the interface that appears pick the trained model you want to test.

Step 4) Click the Accept & Proceed button

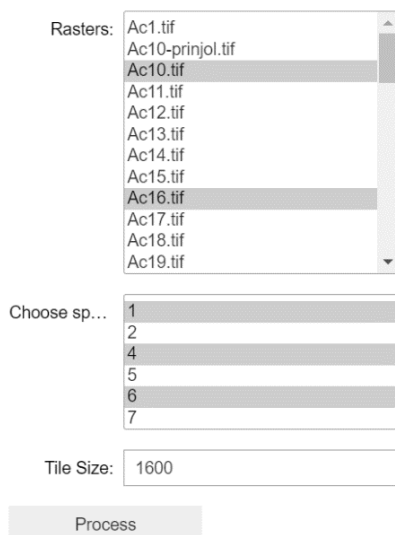
Note that the model you picked (as well as other settings) gets written out, so that you can verify that your last step was correct.

Step 5) In the interface that appears select which epochs to test.

You can select multiple epochs to test if you CTRL+click on more than one epoch. This makes it easy to test how multiple epochs perform. These should be selected after inspecting the aforementioned log. Some epochs are suggested next to this interface.



Step 6) Click the Accept & Proceed button and verify your epoch choices



Step 7 & 8) Pick the rasters to run the test on and which species to produce the Confusion Map for<sup>2</sup>.

This means that, each epoch will be tested with all rasters you choose and for each raster you will get a confusion map for each plant kind. For each raster an Excel file with a confusion matrix will also be produced.

Step 9) Pick the Tile Size. 512 is the lowest recommended but 256 can also be tried. The bigger the tile size the faster the process, however, a large size can result in a crash if the GPU VRAM resources are low.

On a 24Gb GPU the tile size can even go up to 3200. Yet, if the raster is very large it is best to keep the tile size moderate or low.

Step 10) Click the **Process** button and wait for the results. They will all be saved in the Results/[Model Name]/ folder.

<sup>2</sup> If you are testing a single-class model only the model's 1 class can be outputted and this will be set automatically.