Astroeco ML Guide

Login and setup

- This document is a short guide to the use of the astroecology ML software developed by Ross McWhirter during March 2020 and how to access data and resources on Mimir. This software can be used to automate much of the darknet machine learning training and evaluation.
- Users must login to the astroecology server 'Mimir' through the ARI external network.
- This is accomplished using the NoMachine software. Install this software and then create a connection to external.astro.ljmu.ac.uk using the username and password supplied by the ARI IT staff.
- Open a virtual desktop which will be the means to connect to Mimir. If you have an issue here (error message saying XFCE desktop cannot be created), please contact the ARI IT staff using ast-support@ljmu.ac.uk.
- Once connected to the virtual desktop, you must log into the Mimir server using the astroecology usernames provided.
- The next required step is to open a terminal window, this can be accomplished by clicking the black icon containing a dollar sign at the bottom of the screen. All actions will be conducted within the terminal window.
- The login to Mimir is accomplished by typing 'ssh 150.204.240.212 -l YOURUSERNAME -X'. You will be prompted for your password. If this is your first time connecting, it is strongly recommended you change your password once logged into Mimir using the 'passwd' command.
- Once you are logged in you may access the folders you are entitled to. You can type the command 'thunar' to open a file manager GUI for browsing.
- Standard users will have read only access to the /media/astroecology/raid drive which holds our data. The main astroecology data is held in the /media/astroecology/raid/ourdata folder.
- Users will have read and write access to the scratch disk at /media/astroecology/scratch and their own home folder at /home/USERNAME/. Both drives can be used to store datasets for ML which requires read/write access for the Astroeco ML python script.
- Your home directory is visible only to you whereas the scratch drive is visible to all users.
- There is also the 'astroecology' main account. This account has been setup with all the required software and full read/write access. Standard users have read only access to the /home/astroecology folder and subfolders EXCEPT for the darknet and yolov3 (ultralytics) folders required for machine learning.
- If full superuser access is required this must be done through the 'astroecology' user account. Login information can be provided if needed but make sure this is absolutely required as this account has read/write access to the full data.
- Standard XFCE GUI programs such as thunar and gedit may be launched from the command line by simply typing their names as the command. Both firefox and google chrome are installed as browsers and can be launched by typing 'firefox' or 'google-chrome'.
- If multiple programs are required, you can can open additional terminal windows or tabs of one window within the File dropdown at the corner of the window. Each additional terminal window will need to be logged into Mimir using the method described above.

Running the astroecology ML software

- First the correct conda environment must be activated. For this software it is simply just the (base) environment. This should automatically be the active environment. If not, the environment can be activated by typing 'conda activate base'. In the event that you must deactivate the environment, type 'conda deactivate'.
- Next, you must copy the astroeco_ml folder from the /home/astroecology/Documents folder into your own home directory using the command 'cp -r /home/astroecology/Documents/astroeco_ml /home/YOURUSERNAME/'.
- Use the cd command to navigate to the astroeco_ml folder by typing 'cd /home/YOURUSERNAME/astroeco_ml'.
- This directory contains the required python scripts as well as some required supplementary files. You must always launch the astroecology ML scripts from this directory.
- You can launch the python program by simply typing 'python astroeco_ml.py'.
- Executing the above command will pop up a list of accepted inputs for this program of which multiple are required to execute a number of possible tasks.
- This software is capable of training object detection models using the darknet library, evaluating detection models using darknet on validation and testing sets and running detection on a new folder of images.
- In the next two subsections, the inputs and outputs of this software (as shown when you ran the above command) are defined.
- A full copy of this software is stored on the cloud at: https://github.com/P-R-McWhirter/astroeco_ml/

Astroeco_ML software inputs

- The following options are input commands for the astroecology ML script and are input using a double hyphen followed by the name of the input (with no space) then a space and the value.
- For example: 'python astroeco_ml.py --gpu_id 1' which is the command to select GPU 1 (the second GPU) for training operations.
- If an argument is missing it may result in the program exiting during runtime. The error message will indicate which argument is missing so that the user may fix this.
- This software will NOT delete any files or folders it does not also create. If rerunning, for example, augmentation with new settings, it is recommended to delete the old augmentations folder inside the training data directory before proceeding.

- **INPUT COMMANDS :**
- --darknet : A path to the darknet folder. If running this code from Mimir, the correct path to the astroecology Documents directory version is automatically provided and is not required.
- --ultra : A path to the ultralytics (yolov3) folder. As with the darknet option, if running this code from Mimir, the correct path is provided and is not required.

- --mode : A required input. This indicates which mode you are running the script in. Type '--mode' train if you wish to train a new model, eval to run the evaluation scripts on an existing model or detect to run inference on a folder of new images.
- The next commands are separated into which mode they are under however some of these inputs may be required for more than one mode.

- **TRAINING ARGUMENTS :**
- --train_folder : The directory path to the folder containing the training data in the darknet format. This folder contains either jpg or png images along with txt files for the labels. Deeplabel outputs this format when exporting using the darknet command (consult Deeplabel documentation).
- --val_folder : The directory path to the folder containing the validation data in the darknet format. See Deeplabel comments above.
- --model_type : The type of YOLO computer vision model you wish to train. There are four input options, 'yolov3' for a basic YOLOv3, 'yolov3-spp' for YOLOv3 with Spatial Pyramid Pooling, 'yolov3-tiny' for Tiny YOLOv3 and 'yolov3-tiny-3l' a special variant of Tiny YOLOv3 with three detection layers (for smaller objects).
- --gpu_id : Which GPU to use for training (starting from 0). There are only two in Mimir so this can only be 0 or 1. Evaluation and detection only use GPU 0. If left empty, will default to GPU 0.
- --subdivisions : Models are trained with a batch size of 64 but this often is still too big for the GPU memory. Darknet can subdivide the batch even more at a cost of slower speed. The lower the better, trial values of subdivisions starting at 1, then 2, 4, 8, etc. Must be a sub-multiple of 64. If ignored, will default to 16 which will work for most astroecology datasets, but may not be the fastest at training.
- --model_dim : The dimensions of the input 'image' for the training. The larger this value is the longer it will take to train. Defaults to 416×416 and must be a multiple of 32. It is recommended to use a value similar to the largest axis of the input images. The aspect ratio will be maintained and empty pixels will be padded with zeros.
- --num_of_epochs : An epoch is the amount of training iterations required to complete one full pass over the training data. The more epochs trained for, the longer the training takes but the better the opportunity for the model to converge. Recommended to start with 100 and increase if required.
- --data_folder : Required, a path to a directory (existing or not existing) in which runtime data can be written. This folder MUST have read/write access. Recommend using a directory inside either the user home directory or the scratch drive. Be careful not to over-write a previous run.
- --model_folder : Output folder for the trained models saved every 1000 iterations. As with the above directory, this folder MUST have read/write access as well as sufficient storage space for the models. Be careful not to over-write a previous run.
- --altaug : A flag (does not require a value after it) to select that you wish to augment the height of your training data. Requires the --init_height and --new_heights arguments. Can be run with --rotaug.
- --rotaug : Another flag to select that you wish to augment the rotation of your training data. Requires the --rot_angle argument. Can be run with --altaug.
- --init_height : Input for the initial height of the altitude augmentation (the height the data was collected at).

- --new_heights : A list of numbers (input like --new_heights 100 200 300 etc) which contain the new heights for the augmented data. BE CAREFUL: THIS CAN TAKE UP ALOT OF SPACE IF HEAVILY AUGMENTED.

- --rot_angle : The angle in which to incrementally rotate the data in degrees between 0 and 360. For example, 45 will turn one training image into 8 images at 0, 45, 90, 135, 180, 225, 270 and 315 degrees. BE SUPER EXTRA CAREFUL: THIS CAN TAKE UP AN INSANE AMOUNT OF SPACE IF HEAVILY AUGMENTED (small value input).

- **EVALUATION ARGUMENTS :**

- --test_folder : Directory path to the folder containing the testing set images and labels in darknet format. Note: --val_folder also required for evaluation despite listed under training arguments.

- --prefix : A string prefix to identify the models in a model folder for evaluation (in case multiple models using a different configuration are present). For example providing yolov3 will mean only .weights model files with yolov3 at the start are selected. Note: for string input it is not necessary to use single or double quotes.

- --iou_thresh : The IoU threshold to use for the calculation of the mAP statistic. Takes values between 0 and 1. The usual mAP values used are mAP-50 = 0.5 and mAP-75 = 0.75. This threshold determines what IoU (Intersection over Union) is required between ground truth bounding boxes and model predicted bounding boxes to consider the detection a correct one.

- **DETECTION ARGUMENTS :**

- --cfg : File path of the configuration file of the inference model. Will have a filetype of .cfg. If a model trained using this script, it will likely be in the data folder with a name based on the model type selected and a .cfg extension.

- --weights : File path of the weights model file used for the inference. Will have a filetype of .weights and if trained using this script will be in the model folder.

- --obj_names : File path to the obj.names file for the model inference. This file is a text file with the names of the classes on new lines. This file is required for Deeplabel export and is likely not far from the training and validation folders.

- --input : Directory path to a folder containing images to be classified using the inference model. This folder does not need to contain labels (for obvious reasons) and will be labelled by the inference.

- --conf_thres : A probability between 0 and 1 defining the confidence required for a prediction before it will be drawn on the image. This can vary depending on the confidence of the model. If it is too high you will not get any labels, too low and you will get many false positives. Do not set it to zero, you will probably blow the program up. Starting with 0.5 is a good idea and adjust from there.

- --make_video : If the input folder contains a thermal flight extracted by flirpy in the form frame_000000.jpg, frame_000001.jpg and so on, you can supply this flag argument to trigger the production of a video from the resulting predicted images. This can be used to make nice videos for outreach but other than that is pretty pointless!

- --video_framerate : Value for the framerate of the output video, only needed if the --make_video flag is used. For our recent datasets on the FLIR Duo camera this is usually 30 for 30 fps.

- --output : Directory path to a folder (existing or not existing) which will contain the classified and labelled output images. Again, be careful and make sure there is sufficient file space for this output. Pretty easy to predict the output folder size as it will be very similar to the input folder size.

Astroeco_ML software outputs

- This software outputs all required information into three folders, all supplied by the user as arguments: The '--data_folder', 'the --model_folder' and the '--output' directories.

- **TRAINING OUTPUTS :**

- During the 'train' mode, the data folder will output text files containing training and validation file paths and .data files containing links to the model folder and the input training and validation file path text files. These are not needed by the user and can be ignored.

- The configuation file for the training will also be output into the data folder named after the model type selected with the file extension '.cfg'. This configuration file is important as it will be needed for both the 'eval' and 'detect' modes using the --cfg command.

- The model folder will also contain a set of .weights files. These are the models saved after 1000 iterations. Keep all these models, not just the final one as they will be important when used with the 'eval' mode to determine the model performance (and if later iterations are over-fitting).

- The final file in this folder is an image named 'chart' which contains a plot of the training loss against iteration number. This is a good indicator that the model is training if it is dropping over time however is not an indicator of model performance.

- **EVALUATION OUTPUTS :**

- The evaluation mode will place two files, map_output_val.csv and map_output_test.csv into the data folder. These files contain raw unformatted output from darknet and are most likely not useful to the user, but may contain additional information if required.

- The main outputs of this mode are put into a subdirectory in the data folder named eval_results. It is recommended that you clear this folder before every evaluation mode execution.

- In this folder there are $2(N+1)$ table files where N is the number of classes in the model. Each class has its own statistics file combined with an overall statistics file. This is applied to both the validation set and the testing set.

- Each class evaluation file contains as columns: Model file name, model file number (multiples of 1000 with a final and a last), The name of the class (shared by file name), the ID number of the class (starting at 0, contained in obj.names folder), the True Positive number, the False Positive number and the False Negative number and finally the accuracy percentage of that class from that model. The suffix of the file will be either _val or _test to indicate which dataset was used.

- The overall evaluation files contain as columns: Model file name, model file number (multiples of 1000 with a final and a last), overall Precision, overall Recall, overall F1-score, overall True Positive number, overall False Positive number and overall False Negative number, the average IoU of the predicted bounding boxes (in percentage) and finally the mAP value (in percentage) based on the IoU threshold provided (see --iou_thresh above).

- **DETECTION OUTPUTS :**
- The detection mode outputs all classified images into the --output folder.
- If the --make_video flag is used, a video will be produced with the name output.mp4 within the astroeco_ml folder. Be careful, this video will overwrite an existing output.mp4 in this folder.

Final remarks

- Following this guide and addressing error messages from the operation of this software should be sufficient to mitigate any major issues.
- Feel free to inspect any of the .py python scripts in the astroeco_ml folder but take care not to alter them or delete any of them as the software is dependent on their use.
- This software is limited in capability and designed to offer users access to some powerful computer vision machine learning but it is not designed to replace the users education!
- For further information on how to use darknet independently of this software please study closely the wise words of AlexeyAB on his github darknet repository (used extensively by this software) found at: https://github.com/AlexeyAB/darknet
- Take care and I hope this software is useful! Ross McWhirter, 1st April 2020.